Journal of Computer Languages 73 (2022) 101170

Contents lists available at ScienceDirect

Journal of Computer Languages

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

Surveying communities of users of MATLAB and clone languages

Eduardo Reis [a], Catarina Gralha [a], Miguel P. Monteiro [a,b,*]

[a] NOVA LINCS, Portugal
[b] NOVA School of Science and Technology, Portugal

ARTICLE INFO

Keywords:
MATLAB
Surveys
Questionnaires
Modularity
Object-oriented programming

ABSTRACT

Context: *MATLAB* is a programming language vastly used in scientific and engineering domains by engineers, scientists, and researchers. Still, *MATLAB* seems to be perceived as being used mainly by non-professional programmers, not taking full advantage of *MATLAB*'s features (e.g., OOP-support). The current state of the art does not seem to verify these assumptions.
Objectives: Our goal was to fill the gap in the characteristics of the MATLAB community and its users, how proficient they are with the MATLAB, and what is their satisfaction level.
Methods: We performed a survey with 212 valid responses, aiming to characterize the community of *MATLAB* users and clone languages. The survey was published on several platforms, including Reddit, Linkedin, and MATLAB Central.
Results: There is a balanced distribution across different levels of experience in the community. (12.74%) of our sample uses MATLAB only through the command window. The more users expect other people to use their programs, the more effort they will put into making the code easy to understand, maintain and reuse. The use of OOP is not widespread (22%). The majority of MATLAB users are satisfied with its modularity support.
Conclusions: Our study provides insights into MATLAB's use patterns that are potentially useful for entities responsible for MATLAB's future evolution.

1. Introduction

MATLAB [1] is a programming language and computing environment used by many engineers, scientists, researchers, and organizations worldwide. It is extensively used in scientific and engineering domains [2], including but not limited to deep learning and machine learning, signal processing and communications, image and video processing, computational finance, and robotics.

Despite its wide-ranging uses, *MATLAB* seems to be perceived by many as more a specialized tool than a programmer's possible language of choice. Anecdotal evidence [3] suggests many users are people whose main professional activity is not programming and that professional programmers approach *MATLAB* mainly as a tool for specialized tasks, such as interactive matrix calculation and plotting. Though *MATLAB* includes object-oriented (OO) features [4] since 2008, it is generally thought that many *MATLAB* programmers do not use these features, even when they have experience with the OO paradigm. In the past, one of the authors of the present paper also made similar assumptions when working on the study of techniques for concern detection in *MATLAB* code [5–7]. Similar considerations apply to the communities of the so-called *MATLAB clones*, including *GNU Octave* [8], *Scilab* [9–11] and *Rlab* [12,13].

The current state of the art does not seem to include studies that confirm or refute the aforementioned assumptions. As far as we know, no previous work has characterized and categorized the different groups that make up the community of users of the *MATLAB* language and its clones. In principle, such groups can be characterized according to several factors, namely the purpose for which they program, their domain of application, and their levels of experience with each feature of these languages. It would also be desirable to know the relative size of each group in the context of the community of *MATLAB* users and their clones. This paper contributes to filling this gap in the state of the art.

This paper presents the results obtained from a survey carried out to obtain the information stated above. In October 2020, we published a questionnaire on various online platforms aimed at users of *MATLAB* and clone languages. The goal was to learn how users use *MATLAB* and its clones and, in particular, whether they use the *object-oriented programming* (OOP) [14] features provided by these languages. The survey also assesses user satisfaction with *MATLAB* in terms of support for *modularity* (see Section 3). The survey collected 215 complete responses, of which 212 were considered valid (see Section 3.2).

This section further includes the following subsections to characterize this study's motivation and aims. Section 1.1 characterizes some

* Corresponding author at: NOVA School of Science and Technology, Portugal.
E-mail address: mtpm@fct.unl.pt (M.P. Monteiro).
URL: http://ctp.di.fct.unl.pt/~mpm/ (M.P. Monteiro).

https://doi.org/10.1016/j.cola.2022.101170
Received 31 May 2022; Received in revised form 27 September 2022; Accepted 2 October 2022
Available online 12 October 2022

Table 1
Pairs of null (H_0) and alternative hypotheses (H_1), with corresponding RQ.

H_0	H_1	RQ
A user's level of experience with *MATLAB* is **not** correlated to the application domain in which they program.	A user's level of experience with *MATLAB* is correlated to the application domain in which they program.	1
A user's level of experience with *MATLAB* **does not** influence the usual size of their programs.	A user's level of experience with *MATLAB* influences the usual size of their programs.	1
The years of experience a user has with *MATLAB* is **not** correlated to the importance they give to their programs' reusability and maintainability.	The years of experience a user has with *MATLAB* is correlated to the importance they give to their programs' reusability and maintainability.	2
A user's effort to keep a program maintainable is **not** affected by their expectation of being the sole user of that program.	A user's effort to keep a program maintainable is affected by their expectation of being the sole user of that program.	2
A user's level of experience **does not** influence their opinion on *MATLAB*'s support to modularity.	A user's level of experience directly influences their opinion on *MATLAB*'s support to modularity	3
The importance a user gives to the program's maintainability **does not** influence their satisfaction with *MATLAB*'s support to modularity.	The importance a user gives to the program's maintainability directly influences their satisfaction with *MATLAB*'s support to modularity.	3

specialized terms, such as *concerns* and *modularity*, and Section 1.2 presents the research questions and hypotheses we analyze with our study.

1.1. Modularity and separation of concerns

In software engineering, a *concern* is any abstraction, concept, or consistent set of responsibilities whose code programmers would like to localize in its module. The term "concern" is perhaps best known in the popular term *separation of concerns* [15], which refers to the design principle that stipulates that computer programs should (ideally) be organized into several modules, with each module dealing with a single concern. The term *modularity* refers to the extent to which this principle is successfully followed.

Terms like *modularity* and *separation of concerns* should be avoided when addressing the communities of MATLAB and its clones since it is thought that the members of the MATLAB community are not familiar with these terms. A good workaround – employed in the present work context – is to refer to OOP, which is supported by MATLAB and its clones. Following this reasoning, our survey poses various questions regarding classes and objects (see 3.3).

1.2. Research questions and hypotheses

The research questions (RQ) addressed in this survey study are the following.

RQ1 How is the community of users of *MATLAB* and its clones structured and divided, according to their level of experience, and the application domain in which they program, among other factors?

RQ2 How proficient are the users of *MATLAB* and its clones?

RQ3 What is the level of users' satisfaction with *MATLAB*'s current support for modularity?

Table 1 defines the pairs of the null hypothesis (H_0) and the alternative hypothesis (H_1) formulated in the context of this paper. The third column indicates the corresponding research question.

Regarding RQ1, we wanted to stratify the community into different levels of expertise, domains, languages used, and other factors, if possible. The first two hypotheses enable us to do that, which is why they are related to RQ1.

Regarding RQ2, we wanted to better understand the level of proficiency of the users of MATLAB and its clones and derive an estimate of how much these users focus on their programs' maintainability. The third and fourth hypotheses provide some insights.

RQ3 is related to our goal of understanding how satisfied MATLAB users are with MATLAB's current support for modularity and what they consider the best alternatives to MATLAB. The fifth and sixth hypotheses reply to that.

1.3. Structure of the paper

The rest of this paper is structured as follows. Section 2 surveys related work. Section 3 describes the methodology used, including the rationale that determined the questionnaire's design and structure. It also presents an analysis of the responses obtained, including their validity. Section 4 describes the results obtained, Section 5 provides a discussion and Section 6 concludes the paper.

2. Related work

To our knowledge, there are very few research focused on probing the community of users of the MATLAB language or its clones. Initially, our review of related works was done ad hoc. At a certain point, we decided to complement these results with an approach that was lightweight but systematic and reproducible by others, and that would increase the level of confidence with which we make this claim.

The lightweight systematic search we carried out is described in Section 2.1. Sections 2.2 and 2.3 provide an overview that is a bit broader but still focused on surveys targeting communities of users of programming languages. Section 2.2 is focused on communities of users of *MATLAB* and Section 2.3 is focused on surveys targeting users of other languages.

2.1. Lightweight systematic search on Google scholar

We used the popular *Google Scholar* search engine[1] for our searches. Before the searches, we defined the following inclusion criteria:

- **Time interval.** The time interval starts in 2008, as it was the first year that support for MATLAB object-oriented programming was available. It ends in 2021, the last full year before the writing of this paper.
- **Additional settings.** The settings "include patents" and "include citations" were both deselected
- **Browser pages considered.** For each search, references from the first 3 *Google Scholar* pages were considered, corresponding to 30 references.
- **Research topic.** The research topic covered must be the community of programmers or users of MATLAB or one of its clone languages.

It would be expected that papers on certain topics would be represented in significant numbers, even if out of scope. Topics in such situations included the following:

- Publications dealing with specific systems, applications, libraries, tools, or techniques.

[1] https://scholar.google.pt/.

- Tutorials, including MATLAB tutorials and tutorials on specific aspects of MATLAB. Note that while a publication analyzing a particular feature of MATLAB is potentially interesting, such a publication does not provide information about the degree of adoption of that feature on the part of the MATLAB community, which is the focus of this research.
- Research studies, including survey studies, whose topic does not target or cover the community of programmers or users of MATLAB or one of its clones.

Our initial search attempts returned many publications that were out of scope. For this reason, certain words were explicitly excluded from the search strings so such references would not appear. For instance, "Matlab" is the name of a region in Bangladesh. Therefore we excluded the word "Bangladesh" from all searches. Similarly, some searches included the word "community", but "community detection" is an important research topic that appears quite often. For the same reason we excluded the word "detection" in all searches that included "community".

After this initial calibration phase, we use the following search strings:

1. `profiling the MATLAB programmer -Bangladesh`
2. `survey MATLAB community -detection -Bangladesh`
3. `survey MATLAB community -detection -algorithm -health -Bangladesh`
4. `survey profiling MATLAB programmer -Bangladesh`
5. `survey ''usage patterns'' MATLAB Object-Oriented -Bangladesh`

The procedure followed for each search consisted of the following. The first step involved reading the title of each reference to determine whether it should be considered or not. If yes (or when in doubt), the second step involved reading the publication's abstract to confirm that it should indeed be included. The publications passing both steps would be considered for the state of the art. Nevertheless, no publication in our searches passed both steps except for the Master thesis by Reis [16], which forms the basis for the present paper.

The exercise described in the present section was not intended as a basis for a survey of related work but as a complement. Still, it did reinforce our conviction that very few previous survey-based studies targeted the communities of MATLAB or their clones.

2.2. Surveys focused on the community of MATLAB users

One previous related work with which one of the authors was involved as supervisor, was the MSc thesis by Duarte [17]. It aimed to survey the *MATLAB* and *Octave* communities concerning the limitations in the support to modularity in those languages. The target population comprised *MATLAB* and Octave developers, from either the industry or academia, with recent experience in programming in projects of non-trivial size. The survey, using Qualtrics [18], collected 42 responses, 76.19% of which were complete, while the remainder contained only partial responses to the questionnaire. The experience gained from this study was incorporated into the present work. It turned out that the questionnaire was too ambitious for the chosen target population. It included specialized terms such as *modularity*, *cohesion*, *coupling* and *code/concern tangling*, which probably made it inaccessible to a significant number of participants. Initially, participants' selection method was to announce the questionnaire on a number of social networks such as *LinkedIn* [19] and *Facebook* [20], as well as at *MATLAB Central* [21]. The number of responses was considered low, and a second selection method was employed: collecting a list of papers based on a Google Scholar search and sending *emails* to the authors.

Cretchley et al. [22] conducted a study into the effects of scientific software, specifically *MATLAB*, on first-year university mathematics students. Their sample had 184 students who completed questionnaires at the beginning and the end of a semester. Data related to the participants' mathematical skills, feelings, attitudes, and beliefs were collected via several means, including a Likert-scale questionnaire administered at the beginning and end of the course. Data from the questionnaire was subjected to factor analysis, from which 6 different factors emerged. Afterward, the responses were analyzed by gender, first language, mode of study (external and on-campus), and degree. The authors found evidence suggesting the use of technology (e.g., *MATLAB*) had a strong impact on the learning strategies of particular students and that almost all students responded positively to using *MATLAB* for ease of computation and graphing.

Two studies involved questionnaires at the beginning and end of courses. The first study was conducted by Wallin et al. [23] to evaluate a *MATLAB* course that was part of a first-year introductory engineering course. The study involved questionnaires and interviews of 77 first-year engineering students, where the authors measured their experience and satisfaction with *MATLAB* and other tools during the course. The authors found that 43% of the students leaned towards a negative opinion on the *MATLAB* part of the course. Nevertheless, 70% said they deemed this part of the course meaningful; 90% believed their *MATLAB* knowledge would be useful during their coming studies; 70% believed their *MATLAB* knowledge would be useful during their coming careers. Later, a study was carried out by Hoole et al. [24] on the experience of the use of *MATLAB* gained through an initial two-week *MATLAB*-based module on matrix equation solution that was used in four courses at the Rensselaer Polytechnic Institute. The results of both studies helped to understand the students' satisfaction and experience with the different course materials available. Their focus was on a given course, and the survey was used to validate it. By contrast, our study used a questionnaire as its sole survey instrument, and the information obtained through the survey is the reason for it. Also, the sample of participants in our study is not limited to students and includes researchers and engineers.

2.3. Surveys focusing on other programming languages

During the fall of 2019, a *SciPy* [25] survey was conducted by Gwóźdź [26] via a *Google Forms* [27] questionnaire. The goal was to gather feedback that could be considered in the future development of *SciPy* and improve documentation. The author publicized the survey through several forums, including *Twitter* [28], the *SciPy* mailing list and website, several relevant university departments, blogs, and physical mailing lists. The questionnaire received a total of 185 responses. The results showed that most participants were satisfied with the documentation and 84.2% of them were able to quickly find the information they were looking for. The results also clarified which parts of the documentation were more commonly used and how the documentation was browsed.

The *SciPy* questionnaire included 11 questions, most of which comprised multiple choice and 5-point Likert scale questions. The last three were open-ended questions. The questionnaire was mainly focused on getting user feedback using the latter. By contrast, our questionnaire covers a wider range of topics across a higher number of questions, all relating to users' approaches to *MATLAB* and clone languages, as well as patterns of use. We intended the questionnaire to include as few open-ended questions as possible to minimize risks of misinterpretation, avoid exhausting participants and facilitate analysis. Still, the sampling strategy and the number of responses are similar.

In 2020, *JetBrains* and the *Python Software Foundation* (PSF) surveyed the *Python* community concerning a wide variety of topics to identify the latest trends and gather insight into what *Python* development looked like in that year [29]. They have been conducting this questionnaire every year to compare results to previous years, analyze changes and obtain an overview of the evolution of software development using *Python*. After filtering duplicate and unreliable responses, they gathered more than 28,000 responses. Participants

were contacted through the promotion of the questionnaire on numerous platforms, including *Python*'s official website, *PSF*'s blog, *Twitter*, *LinkedIn*, *Reddit*, and official mailing lists. To avoid bias, channels associated with specific products or services were not used. Topics covered by the questionnaire included: general language usage, reasons for using the language, language versions used, frameworks and libraries used, technologies and cloud platforms used, development tools used, employment, work, and age. With an extensive list of questions, they could gather extremely specific results when combining responses from different topics. For example, they could tell which version of *Python* was most used for web development or data analysis or how the version of *Python* correlates with the user's age. Using a lengthy questionnaire brought clear benefits in this case. Because the number of participants reached was relatively high, the mortality in the sample did not preclude reaching significant conclusions, something that risks occurring in smaller scale surveys.

Similarly, a yearly questionnaire is also conducted in which the above organizations analyze the state of the entire developer ecosystem to identify the latest trends in tools, languages, and technologies being used, among many other factors [30]. In the 2020 edition, they gathered responses from almost 20,000 individuals that were reached through *Facebook* ads, *Quora* [31], *Codefund* [32], among some other platforms, as well as through *JetBrains*' own communication channels. This questionnaire was made available in nine different languages to minimize potential bias and make it more accessible.

The results underwent three weighting stages to further reduce the sampling bias and to display a more realistic picture of the worldwide developer population. In the first stage, they gathered the responses collected while targeting different countries and then applied estimations of the populations of developers in each country to those responses. In the second stage, they forced the proportion of student or unemployed participants to be 17% in every country. This 17% figure estimated the populations they had gathered from the previous year's questionnaire. This helped to maintain consistency concerning the previous year's methodology. For the third stage, a system of more than 30 equations was used, e.g., to compute the proportions of developers from each country for each of over 30 programming languages, as well as the proportions for those who made claims such as, e.g., "currently use *JetBrains* products" and "have never heard of *JetBrains* or its products". These proportions were then used as constants in the equations. This approach enabled a very elaborate and intricate analysis, covering a wide range of topics relating to the entire developer ecosystem. Despite efforts to the contrary, there was still some bias left since users of *JetBrains* products were more likely to complete the questionnaire than other developers. Nevertheless, this did not seem enough to call into question the validity of the approach.

In 2011, a survey was conducted by Prabhu et al. [33] on the prevalent programming practices of a community of researchers from diverse scientific disciplines relating to computing at a doctoral-granting university [33]. The questionnaire was publicized through e-mail to randomly selected researchers from the university's database. The 114 researchers who replied were interviewed by the authors. Results indicated, e.g., that new software tools and techniques were necessary to unlock the potential of high-performance computing and accelerate the pace of scientific advancements, as available tools did not meet the needs of computational science researchers. Similar to some of the works mentioned in Section 2.2, this survey differs from ours in that it is not focused on the users of a specific software tool or environment. Rather, the survey was used to characterize the scientific computing environment at Princeton University. The results were divided into three different topics. For each topic, the questionnaire included a group of questions from whose answers they derived patterns. For instance, it led to the conclusion that most scientists were unsatisfied with the speed of their programs and believed that performance improvements would significantly enhance the productivity of their research.

Table 2
Source of the responses - *Where did you hear about this survey?*.

Source	Number of participants	Percentage
Reddit	101	48%
LinkedIn	56	26%
ResearchGate	17	8%
MATLAB Central	13	6%
GNU Octave Discourse	9	4%
E-mail	10	5%
Word of mouth	6	3%
Total valid responses	212	100%

3. Research methodology

This section explains the rationale behind our decisions in the design of the survey instrument through which this research was carried out.

3.1. Target population and platforms used

The study aims at the population of users of *MATLAB* and any of its clone languages, regardless of the level of experience or background. The languages we considered *MATLAB* clones are *GNU Octave*, *Scilab* and *Rlab*. Note that the questionnaire does not include questions specific to any *MATLAB* clone.

When looking for representative online communities, we searched for communities primarily centered around the use of *MATLAB* or any of its clones. We favored platforms with a high number of members, whose profiles would range from beginners to experts, from students to researchers, and from people full-time professional programmers to even people who program as a hobby. The selected platforms are:*(i) MATLAB Central* forums; *(ii) MATLAB Central File Exchange*; *(iii) ResearchGate*; *(iv) Reddit* – *MATLAB*-related subreddits; and *(v) LinkedIn MATLAB*-related groups.

3.2. Survey sampling

The survey instrument was created using *Google Forms* [27]. We posted a question on each of the above platforms [34–37] (see Table 2), asking whether it would be a place suitable for publicizing the survey. The replies were positive. *Reddit* turned out to be the platform with the most responses. During this period, we also requested (and received) permission to post on a few *LinkedIn* groups [19]. Four *LinkedIn* groups were subsequently used: "*MATLAB* Users and Integrators" [38], "Matlab beginners and experts" [39], "*Scilab* Software" [40] and "*GNU Octave* users and developers" [41]. We started publicizing the survey in October 2020. The posts on *Reddit* were specifically on the "MATLAB" and "EngineeringStudents" subreddits.

Google Sheets enabled us to monitor the data by looking at responses while the questionnaire was still accepting replies. We started to notice a predominance of *MATLAB* users over other languages, so we decided to also post the questionnaire on a *GNU Octave Discourse* forum [42]. In total, we gathered 215 responses, of which 212 were considered valid (see Sections 3.2 and 4.1).

3.3. Structure of the questionnaire

The instrument used comprises a questionnaire whose structure reflects the fact that the community of users of *MATLAB* and clones widely differs in terms of the level of sophistication in the use of the language. For instance, it would not make sense to ask questions regarding modularity to participants that use the command window only. For this reason, one of its sections applies branch logic that divides participants into two groups, corresponding to two different paths along the questionnaire. One of the groups comprises those that stated they just use the command window, which are directed to the

questionnaire's closing section. The other group corresponds to those who use more than just the command window and therefore go through the entire questionnaire.

The instrument includes a core segment comprising three main sections, whose questions are associated with the RQs. It is framed by two smaller subsidiary sections, respectively at the beginning and end of the questionnaire:

Section 1 describes the questionnaire's purpose and clarifies issues regarding research ethics [43], e.g., explaining that participation is voluntary with the possibility to withdraw at any point and that the participant's data will be kept confidential. We also ask where the participant learned about the survey.

Section 2 opens the core segment of the questionnaire. It asks participants about their programming background, habits, and experience. It includes questions such as *"How many years of programming experience do you have?'* and *"What programming language do you use the most?"*. Questions from this section serve to stratify the community, i.e., categorize it into different segments, according to e.g., level of experience, application domain, and programming languages used. The last question of this section divides participants into two groups, depending on whether they use the command window only or make more sophisticated uses. We found it useful to also identify users that *never* use the command window (meaning they always use code files).

Section 3 is focused on the importance attached by participants to the maintenance and reusability of their *MATLAB* programs. It comprises questions relating to the size of programs and expectations on the part of developers, as regards reusability and maintainability.

Section 4 contains the last portion of the core segment of the questionnaire and includes questions concerning the participants' use of the languages. Some questions aim at finding out if and how the participants make use of *MATLAB*'s support to modularity and what is their level of satisfaction with that support. The questionnaire also asks participants if they use OOP features in other languages and which language they consider to be *MATLAB*'s strongest competitor relative to typical uses. This section aims to obtain a sense of how participants actually write their code, the extent to which they take advantage of *MATLAB*'s modularity capabilities, and if they are satisfied with them.

Section 5 closes the questionnaire. It thanks participants for their contribution and gives the option for them to leave their email address, in case they like to receive the aggregated results, and if they would be willing to participate in future research concerning this topic.

In Zenodo [44], **we share the entire questionnaire as presented to the participants, including its introductory text, questions, question descriptions, and the answer options to each question.**

3.4. Question types

Throughout its core segments (Sections 2, 3 and 4), the questionnaire presents questions of various different types:

- 10 **Likert scale questions**: 5-point Likert scale questions ranging between the values "Strongly disagree" and "Strongly agree".
- 6 **drop-down questions** in which the participant is presented with a list of different response options, and they pick exactly one of those options.
- 4 **checkbox questions** presenting a list of response options, from which participants can pick as many as they wish. It also provides an open text field allowing participants to insert their own responses.
- 3 **multiple choice questions** list of mutually exclusive options in which only one option may be selected; also enabling an open text field.
- 2 **open ended questions** in which participants have an open text field.

3.5. Questions for consistency validation of participants

In most surveys, there is a risk that some participants answer inconsistently. To detect such cases, we included a pair of questions, worded differently but measuring the same factor: the effort to obtain a satisfactory level of reusability and maintainability. Both are scaled as a 5-point Likert. The first question is at the start of Section 2 (Q16) and the second is in Section 3 (Q21). Their answers are used to measure the difference between both responses, where we discard participants that gave inconsistent answers beyond a given threshold. The questions are:

- *"When I develop a program in MATLAB I always try to make it easily reusable and maintainable"* (Q15);
- *"I try to find and minimize the use of duplicated code across the various m-files"* (Q21).

4. Survey results

This section describes the steps taken to analyze the responses to the questionnaire and subsequently draw conclusions, in preparation for the final discussion presented in Section 5. All the statistical analysis was performed using IBM SPSS Statistics [45] on a mid-range laptop with Windows 10. The analysis, with the entry and exit data, can be found in Zenodo [44]. Section 4.1 describes the steps taken to ensure that analyzed data is consistent, using statistical tests adequate to our data set, Section 4.2 profiles the participants, and Section 4.3 reviews the hypotheses presented in Section 1.2 in light of the results.

4.1. Verifying the internal consistency

Both questions designed to verify the consistency of participants (Q15 and Q21) use a 5-point Likert scale. We used *Kendall's tau distance* [46], a metric to calculate the number of and the degree of disagreement between two lists, to measure the discrepancy between the answers provided by participants to the questions from this pair. A Kendall's tau distance of 0 means the participant provided the same answer to both questions, while a distance of 4 corresponds to the maximum inconsistency. In total, 121 participants had a 0 distance, 69 participants had a distance of 1, 22 participants had 2 and 3 participants had 3. No participants had 4. The mean distance is $\approx$0.57.

We also computed *Kendall's tau-b correlation coefficient* [47], a non-parametric measure of the strength and direction of an association between two variables. Results confirm a strong, positive correlation between the answers to both questions ($\tau_b = 0.45$, $p = 0$) [48]. Based on the mean *Kendall's tau distance* (0.57) and *Kendall's tau-b correlation coefficient* (0.45), we discarded the answers from the 3 participants whose Kendall's tau distance is 3. Thus, 212 answers out of 215 were considered valid.

We conducted a *Principal Component Analysis (PCA)* [49] to allow us to reduce the larger set of variables into a smaller set of "artificial variables" that account for most of the variance of the original variables. We started with the *correlation matrix* where we observe the correlation values between all the variables in the PCA and thus test the linearity between all variables. From the correlation matrix, we concluded that Q12 (*"The last time I programmed in MATLAB or a similar language was..."*) and Q19 (*"The m-files I deal with tend to have..."*) do not have a strong correlation with any of the other variables.

We then tested the sampling adequacy of the data by using the *Kaiser–Meyer–Olkin (KMO) measure* [50]. The KMO measure for our overall data set is 0.795. For KMO measures for individual variables, question 12 was the only question below the threshold of 0.5, with a KMO of 0.258. Since this variable also did not have a strong correlation in the *correlation matrix*, we excluded it and computed a new KMO. The overall KMO measure increased to 0.809. Results of the *Bartlett's test of sphericity* [51] show that the test is statistically significant,

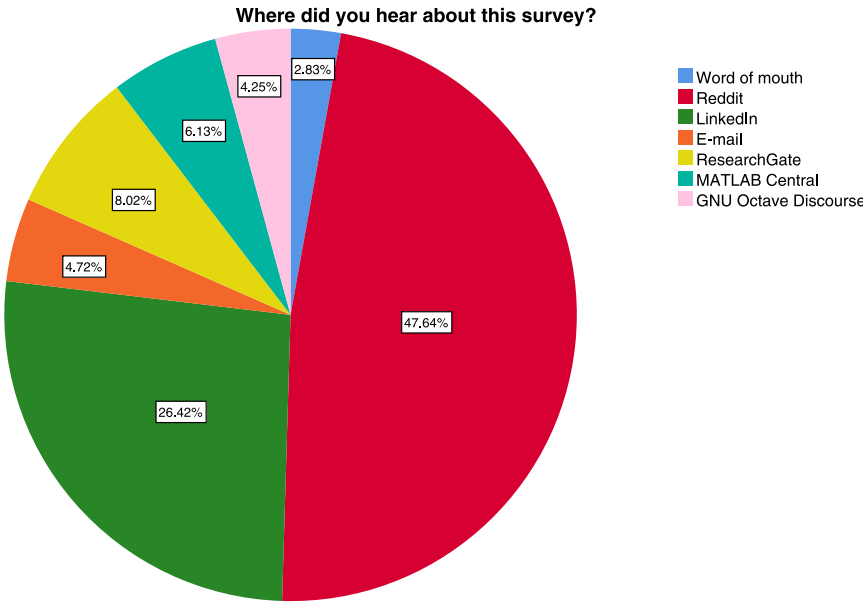

Fig. 1. *Where did you hear about this survey?* (Q1).

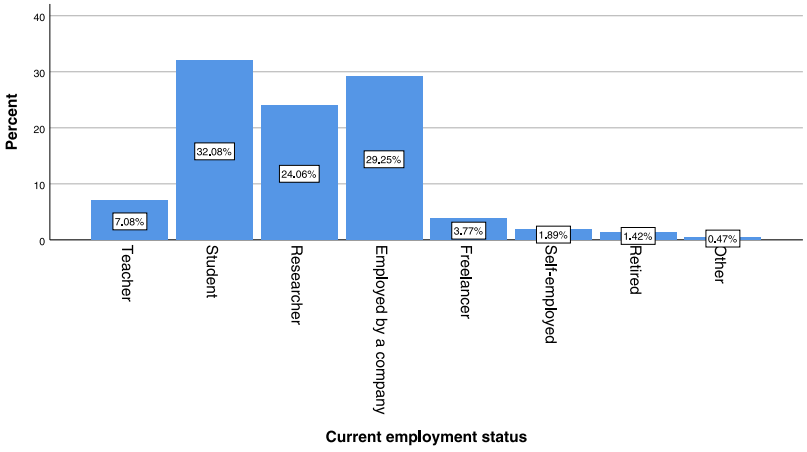

Fig. 2. Participants' employment status (Q3).

further suggesting that a PCA may be useful in this data, as it is likely "factorizable". We obtained 3 PCA components.

Finally, we used *Cronbach's Alpha (CA)* [52] to deduce how much the variables on each PCA component are measuring the same underlying construct or dimension. Since CA determines how well a set of questions are grouped together, we run multiple CA tests, one for each of the components resulting from the PCA. All 3 components have a CA value higher than 0.7, which is the usually recommended value [53, 54]. Thus, the components have a high level of internal consistency, as they accurately measure what is intended. A clear distinction between the components is reflected in their respective questions. The results are consistent with the intended design and structure of the survey.

4.2. Profiling the participants

Approximately 47.64% of the participants heard about the survey on *Reddit* and 26.42% on *LinkedIn*. The remainder heard about it on *MATLAB Central*, *ResearchGate*, *email*, *GNU Octave*, *Discourse* or word of mouth (see Fig. 1).

The participants comprised 68 students (32.08%), 15 teachers (7.08%), 51 researchers (24.06%), 74 programmers (34.91%, working on company, freelancers or self-employed) and 3 retirees (1.42%) (see Fig. 2).

Table 3
Application domain where participants use *MATLAB* and similar languages (Q11).

Domain	Percentage
Data Analytics	61.32%
Signal Processing	45.75%
Control Systems	35.38%
Image and Video Processing	28.77%
Machine Learning	26.42%
System Modeling	8.02%
Wireless Communications	5.66%
Simulations	5.66%
Computational Finance	5.19%
Computational Biology	5.19%

The questionnaire also asks participants the application domain for which they used *MATLAB* and similar languages, allowing participants to state multiple, non-exclusive different application domains. The majority of the participants (61.32%) reported that they use it for *Data Analytics* (see Table 3).

Approximately 94% of participants from our sample stated that *MATLAB* is one of the languages they use. 23.58% mentioned *Octave*, 10.38% mentioned *Scilab*, 3.77% mentioned *Rlab*, and 3.30% stated they use *Julia* (see Table 4). These options are non-exclusive. *Julia* was

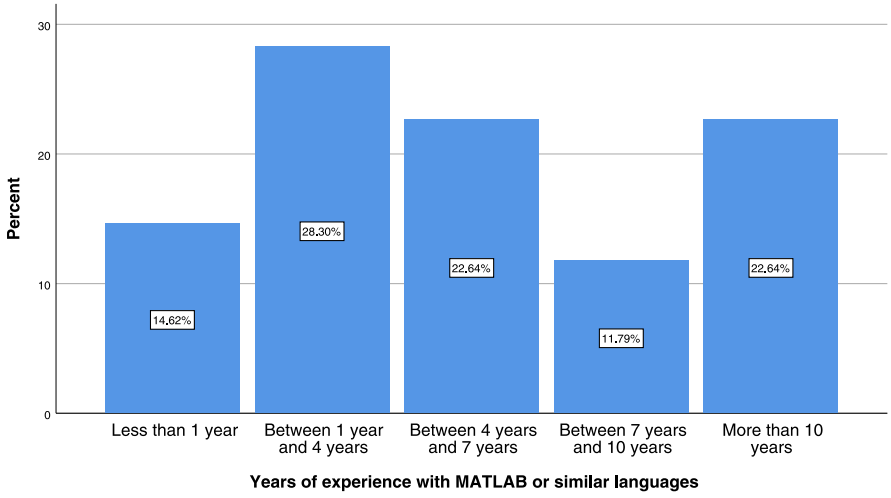

Fig. 3. Participants' years of experience (Q5).

Table 4
Which of the following programming languages (MATLAB and similar languages) do you use? (Q6).

MATLAB-like languages used	Percentage	Count
MATLAB	93.87%	199
Octave	23.58%	50
Scilab	10.38%	22
Rlab	3.77%	8
Julia	3.30%	7

Table 5
What programming language do you use the most? (Q8).

Most used language	Percentage	Count
MATLAB	62.74%	133
C	11.79%	25
Python	8.96%	19
C++	6.60%	14
R	2.36%	5
Octave	2.36%	5
C#	1.42%	3
Julia	1.42%	3
Scilab	1.42%	3

not anticipated and had not been included in the response list, but participants included it in the text field option.

Regarding the question of what language participants from our sample use the most, the majority (62.74%) mentioned *MATLAB*. In second place came *C*, mentioned by 11.79%. Next came *Python* with 8.96% and *C++* with 6.60%. Other languages also mentioned include *R*, *Octave*, *C#*, *Julia* and *Scilab* (see Table 5).

Regarding the experience with the language (see Fig. 3), 14.62% of participants from our sample have less than 1 year of experience with *MATLAB* or clone languages, 28.30% have between 1 and 4 years of experience, 22.64% have between 4 and 7 years and 11.79% have between 7 and 10 years of experience. The remaining 22.64% have more than 10 years of experience with *MATLAB* or a clone language. Unsurprisingly, we observe a correlation between years of experience with *MATLAB* and clones (see Fig. 4 and also Fig. 5 for the participants' self-assessed level of expertise with *MATLAB*).

Regarding which operating systems the participants use the most during development, the majority (84.91%) use *Windows*, 33.96% use Linux and 20.75% use macOS. Note these options are non-exclusive.

Approximately 87% (185) of participants stated they do not exclusively use the command window when working with *MATLAB*. These participants typically write their *MATLAB* code in m-files and either never use the command window, or use it just to solve small problems, or to complement their coding (e.g., to inspect variables and/or test functions). The remaining 27 participants were directed to the final section of the questionnaire.

We also wanted to know the percentage of participants using OOP, by analyzing the usage of both Classes and Objects in their programs (Q20). Curiously, around 61% of participants stated they use OOP in other programming languages, but only 22% use OOP in *MATLAB*.

Regarding typical uses of *MATLAB* (Q25), participants considered *MATLAB*'s strongest competitors to be: *Python*, mentioned by approximately 60% of participants; *Octave*, with approximately 8% of responses and *R* with approximately 6%. Other languages mentioned include *Scilab*, *Julia*, *Wolfram Mathematica*, and *C++*. Each of these accounted for approximately 5% of valid responses or less.

4.3. Hypotheses testing

Table 6 provides a summary making it easier to follow how each hypothesis was tested, what was the result of the test, and the resulting overall evaluation of the research question.

Hypothesis 1. *A user's level of experience is not correlated to the application domain in which they program.* To test this hypothesis, we measure the correlation between the users' level of experience with *MATLAB* and the application domain in which they program, for each of the domains in the data. We calculate the One-Sample Proportion Test (OSPT) [55], a non-parametric test used to assess whether a proportion of a population is different than its hypothesized proportion in the population from which the sample data are drawn, in Q7 (*"Rate your level of experience with MATLAB"*) and Q11 (*"For what do you use MATLAB or similar languages?"*). There was **not** a statistical significant correlation between the level of experience with *MATLAB* and whether participants use *MATLAB* and its clone languages for *Data Analytics*, *System Modeling*, *Simulations*, *Computational Finance*, *Computational Biology*, and other purposes ($p \geq 0.050$). However, there was a statistically significant correlation detected for the domains of *Signal Processing* ($\rho = 0.171$), *Control Systems* ($\rho = 0.173$), *Image and Video Processing* ($\rho = 0.207$), *Machine Learning* ($\rho = 0.180$), and *Wireless Communications* ($\rho = 0.192$). We conclude there is a **weak to moderate positive correlation** between the level of experience in *MATLAB* and whether the participants use it for these 4 domains.

Hypothesis 2. *A user's level of experience with MATLAB does not influence the usual size of their programs.* To test this hypothesis, we measure the correlation between the users' level of experience with *MATLAB* and the number of m-files their programs tend to have. We perform

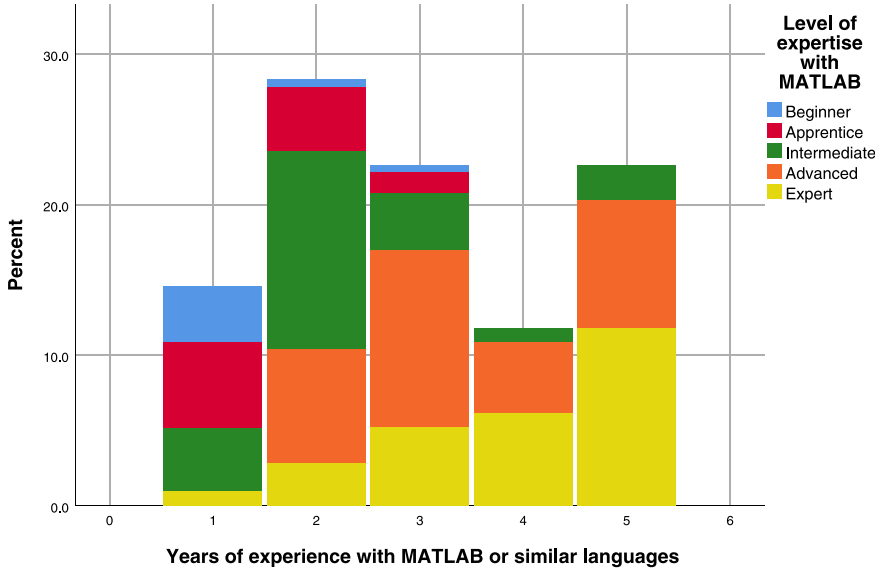

Fig. 4. Years of experience with *MATLAB* (Q5) vs level of expertise with *MATLAB* (Q7).

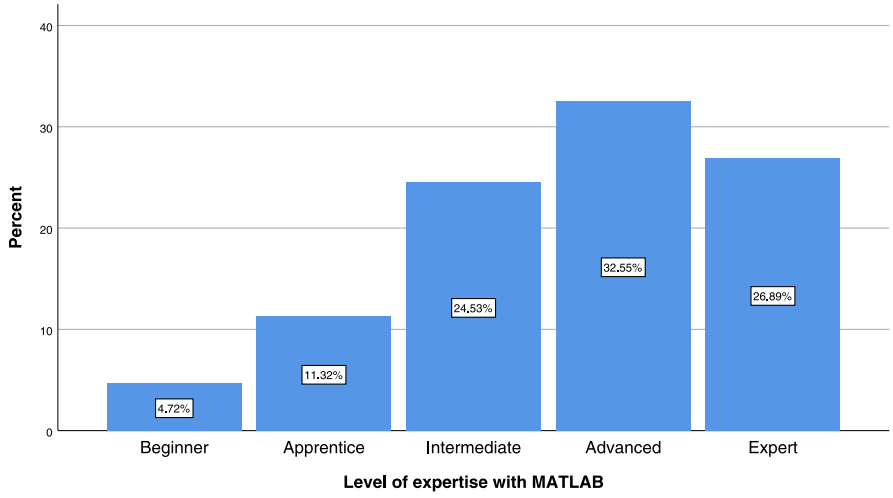

Fig. 5. Participants' level of expertise (Q9).

Table 6
Summary of Hypotheses testing procedures and summary of conclusions.

Hypothesis	RQ	Test used	Test result	Conclusion
1. A user's level of experience with *MATLAB* is **not** correlated to the application domain in which they program	1	OSPT (One-Sample Proportion Test) in Q7 and Q11	$p \geq 0.050$ for 5 domains	Weak to moderate positive correlation
2. A user's level of experience with *MATLAB* does **not** influence the usual size of their programs	1	SC test (Spearman's Correlation) between Q7 and Q18	$\rho = 0.365$ $p = 0.000$	Moderate to positive correlation ($\rho = 0.365$)
3. The years of experience a user has with *MATLAB* is **not** correlated to the importance they give to their programs' reusability and maintainability.	2	SC test (Spearman's Correlation) between Q5 and Q15	$\rho = 0.06$ $p = 0.414$	No statistically significant correlation ($p \geq 0.05$)
4. A user's effort to keep a program maintainable is **not** affected by their expectation of being the sole user of that program	2	Somers' d test between Q15 and Q16	$d = -0.226$ $p = 0.000$	Negative correlation ($d = -0.226$)
5. A user's level of experience **does not** influence their opinion on *MATLAB*'s support to modularity	3	Somers' d test between Q7 and Q23	$d = 0.154$ $p = 0.014$	Weak positive correlation ($d \leq 0.2$)
6. The importance a user gives to the program's maintainability **does not** influence their satisfaction with *MATLAB*'s support to modularity	3	Somers' d test between Q15 and Q23	$d = 0.230$ $p = 0.000$	Positive correlation ($d = 0.230$)

the Spearman's Correlation (SC) test, a non-parametric measure of the strength and direction of association between two continuous or ordinal variables [56], between Q7 (*"Rate your level of experience with MATLAB".*) and Q18 (*"My MATLAB programs tend to have..."*). There **was** a statistically significant correlation between Q7 and Q18 ($p = 0.000$). We conclude there is a **moderate positive correlation** ($\rho = 0.365$) between a user's level of experience with *MATLAB* and the number of m-files of their programs.

Hypothesis 3. *The years of experience a user has with MATLAB is not correlated to the importance they give to their programs' reusability and maintainability.* To test this hypothesis, we measure the correlation between years of experience and the importance of reusability and maintainability. We perform the SC test between Q5 (*"How many years of experience do you have with MATLAB or a similar language?"*) and Q15 (*"When I develop a program in MATLAB I always try to make it easily reusable and maintainable".*). There was **not** a statistically significant correlation between the years of experience a user has with *MATLAB* and the importance they attach to their programs' maintainability and reusability ($p = 0.414$).

Hypothesis 4. *A user's effort to keep a program maintainable is not affected by their expectation of being the sole user of that program.* To test this hypothesis, we perform the Somers' d test, a non-parametric measure of association between an ordinal dependent variable and an ordinal independent variable [57,58], between Q15 (*"When I develop a program in MATLAB I always try to make it easily reusable and maintainable".*) and Q16 (*"I expect to be the sole user of my MATLAB programs".*) Before running the test, we made sure we had a dependent variable and an independent variable on an ordinal scale, and that there is a monotonic relationship between the variables. There **was** a statistically significant correlation between Q15 and Q16 ($p = 0.000$). We conclude there is a **negative correlation** $d = -0.226$ between a user's effort to keep a problem maintainable and their expectation of being the sole user of their programs.

Hypothesis 5. *A user's level of experience does not influence their opinion on MATLAB's support to modularity.* To test this hypothesis, we performed Somers' d test between Q7 (*"Rate your level of experience with MATLAB".*) and Q23 (*"I am satisfied with MATLAB's current support to modularity".*) Before running the test, we made sure we had a dependent variable and an independent variable on an ordinal scale, and that there is a monotonic relationship between the variables. There **was** a statistically significant correlation between Q7 and Q23 ($p = 0.014$). We conclude there is a **weak positive correlation** ($d = 0.154$) between a user's level of experience with *MATLAB* and their satisfaction with *MATLAB*'s support to modularity.

Hypothesis 6. *The importance a user gives to the program's maintainability does not influence their satisfaction with MATLAB's support to modularity.* To test this hypothesis, we perform the Somers' d test between Q15 (*"When I develop a program in MATLAB I always try to make it easily reusable and maintainable".*) and Q23 (*"I am satisfied with MATLAB's current support to modularity".*) Before running the test, we made sure we had a dependent variable and an independent variable on an ordinal scale, and that there is a monotonic relationship between the variables. There **was** a statistically significant correlation between Q15 and Q23 ($p = 0.000$). We conclude there is a **positive correlation** ($d = 0.230$) between the importance a user gives to their programs' maintainability and their satisfaction with *MATLAB*'s current support to modularity.

5. Discussion

This section provides a discussion of the results obtained. Section 5.1 provides answers to the research questions presented in Section 1.2. Section 5.2 discusses threats to validity.

5.1. Answering the RQs

RQ1 How is the community of users of *MATLAB* and its clones structured and divided, according to their level of experience, the application domain in which they program, among other factors?

There is a healthy mix of all levels of experience in the *MATLAB* community, with almost all levels representing between 10 to 20% of users. The majority (28.3%) of users have between 1 and 4 years of experience, but overall the distribution is balanced. Therefore, as new users are joining the community, more experienced users remain in the community as well. This means the community is able to retain its members while still attracting new people.

From our sample, we concluded that a *MATLAB* user's level of experience is directly correlated with the size of their programs. The more experienced the users are, the more likely they are to work with programs involving a larger number of m-files, with a mean of 6 to 10 m-files. Users with above-average experience tend to deal with programs larger than 10 m-files. This suggests that more experienced users have a better grasp of the techniques that allow for a better organization of a *MATLAB* program, allowing them to build and maintain larger and more scalable programs. In turn, this suggests that *MATLAB* provides the necessary means and support (namely strong support for modularity) to build a large-scale project, which the more experienced *MATLAB* users take advantage of.

We stratify the *MATLAB* community according to application domain (see Section 4.2). A *MATLAB* user's level of experience is directly correlated with their application's domain. The more experienced users are, the more likely they are to be working with *MATLAB* or its clone languages for domains such as *Signal Processing*, *Control Systems*, *Image and Video Processing*, *Machine Learning* and *Wireless Communications*. This suggests these domains may have a steeper learning curve than others, as they may require more complex programming techniques. Beginners, on the other hand, are more likely to be working with *MATLAB* for domains such as *Data Analytics*, *System Modeling*, *Simulations*, *Computational Finance* and *Computational Biology*.

MATLAB is the most used programming language used by the participants in our study (62.74%), which was somewhat expected, given our target population. *MATLAB* followed by C (11.79%), Python (8.96%) and C++ (6.6%). The other programming languages do not have a significant representation.

Conslusion: From our sample, there is a balanced distribution across different levels of experience in the community. A user's level of experience is directly correlated with their application domain. The community uses *MATLAB* the most, followed by *C*, *Python*, and *C++*.

RQ2 How proficient are the users of *MATLAB* and its clones?

Contrary to what we expected, students are not the only group that just uses the command window of *MATLAB*, as opposed to, e.g., writing in m-files. From the 27 participants (12.74%) who only use the command window, approximately 30% were students. These users are most likely to use *MATLAB* for simpler purposes and are not focused on creating complex programs.

In general, users try to make their programs easily reusable and maintainable. However, there is an inverse correlation between *MATLAB* users' effort put into the maintainability of their programs and their expectation of being the sole user of those programs. In other words, the more users expect other people to use their programs, the more effort they will put into making sure the code is easily understandable, maintainable, and reusable. This suggests *MATLAB* users are more worried about how well others perceive and understand their code than how they themselves will understand their own code in the future. Users working solo do not seem to care as much about reusability and maintainability. Perhaps they are confident that they

will easily understand their own code when they later revisit it. On the other hand, users who work with colleagues may feel more strongly about their duty to produce code that is clear, easy to understand, and maintain.

Approximately 22% of the participants use OOP with *MATLAB*. However, 61% of the participants stated they use OOP with other programming languages. These results suggest that, when they wish to use OOP, the majority of the participants opt to use a programming language other than *MATLAB*. This could be because they view it as a better environment for an OOP approach, or due to a lack of awareness of the current state of *MATLAB*'s OOP capabilities. Further studies are needed to understand the cause of the apparently low usage of OOP in *MATLAB* when compared to other programming languages.

Conslusion: A significant part of the users in our sample (12.74%) use MATLAB only through the command window and that is observable across all levels of experience. The more users expect other people to use their programs, the more effort they will put into making the code easy to understand, maintain and reuse. The use of OOP in *MATLAB* is not widespread, even among users knowledgeable about OOP.

RQ3 What is the level of users' satisfaction with *MATLAB*'s current support for modularity?

We identified a direct correlation between the experience level of the *MATLAB* users in our sample and their opinion about the current modularity support of *MATLAB*. More experienced users claimed to be more satisfied with *MATLAB*'s support for modularity than less experienced users. We argue that less experienced users might also be less comfortable with the concept of modularity itself, hence the *MATLAB*'s support for modularity might not be properly understood and appreciated.

In addition, we identified a direct correlation between the importance *MATLAB* users give their programs' maintainability and their satisfaction with *MATLAB*'s support for modularity. Users who give more importance and put more effort into maintainability and reusability are also the ones most satisfied with *MATLAB*'s current support for modularity. From our sample, we conclude that more experience and practice with *MATLAB* leads to greater satisfaction with *MATLAB*'s current support for modularity. However, the low levels of use of OOP in *MATLAB* suggest that OOP in this language is not particularly beginner-friendly. Beginners and less dedicated users may be more inclined to pick alternatives to *MATLAB*.

Approximately 69% of the participants that use more than the command window when working with *MATLAB* consider *MATLAB*'s strongest competitor to be *Python*. However, 41% of these participants have stated that they do not use OOP in other languages. This suggests that although many view *Python* as a good alternative to *MATLAB*, this is not due to better OOP capabilities. We conjecture that other reasons could include differences in price and accessibility, or differences in available support tools, as some participants highlighted in comments to posts in which the survey was announced.

Participants whose programs tend to have between 2 and 10 m-files are also the ones who have the most trouble understanding the code or its structure when maintaining a *MATLAB* program. This means that participants whose programs tend to have only 1 m-file or more than 10 m-files are more likely to quickly understand the code and how it is structured. This suggests that smaller *MATLAB* programs, with just 1 m-file, are easily understood and that *MATLAB* programs are efficiently scalable for the most part, as evidenced by the fact that programs with more than 10 m-files are reported to be better understood than programs with 2 to 10 m-files.

Conslusion: From our sample, the majority of *MATLAB* users are satisfied with its support for modularity. *Python* is largely considered to be *MATLAB*'s strongest competitor. *MATLAB* is considered to be efficiently scalable but also easily understood when containing only 1 m-file.

5.2. Threats to validity

For the threats to validity, we are following Wohlin et al.'s guidelines [43].

Conclusion Validity. Interpretations from a single or few individuals are always subject to a potential bias, which is the case with this study. There may be different interpretations to be drawn from the results obtained that are not thought of or presented here.

Internal Validity. Due to a less than ideal questionnaire questions' wording or layout of the different sections, the questions could be perceived as ambiguous by the participants. However, the questions were validated by all the authors, and we tested the questionnaire's internal consistency.

Construct Validity. The study's constructs may not be properly and clearly defined before they were translated into measures in the construction of the questionnaire. This means that the theory and intention behind the study may be incoherent, and the validity of the results may be affected. However, the questionnaire and the measures were validated by all the authors. Finally, it is a human tendency to try to look better when being evaluated. During the questionnaire, the participants may have felt like they were being evaluated, and thus, they may have provided false information in order to seem better (e.g. saying they have more experience than they actually have). However, we stated that the purpose of the questionnaire was to collect feedback on *MATLAB* and that the participants were not being evaluated.

External Validity. If the subject population is different than the population that was initially planned to generalize the results to, the validity of the results is threatened. For instance, programmers that have never dealt with *MATLAB* or any similar language may have decided to respond to the questionnaire. Given the participants we were able to obtain, some of the results obtained are only extendable to a portion of the population (e.g. the participants who use *MATLAB* beyond just the command window).

6. Conclusions

We conducted a survey aiming to characterize the community of users of *MATLAB* and clone languages. The survey was published at *MATLAB Central*, *Reddit* and *LinkedIn*, and we analyzed a total of 212 valid responses. Based on this sample, we use the results to derive an in-depth stratification and demographic analysis of the community of users of *MATLAB* and its clone languages. We derived relevant insights for our sample population, related to the distribution of users across levels of experience, the correlation between the level of experience and application domain, what languages are used the most, and what language is perceived to be *MATLAB*'s strongest competitor, the proportion of users interacting with *MATLAB* through the command window only, the correlation between the expectation that others will use the program and effort put into maintainability and reuse, the proportion of users who use *MATLAB*'s object-oriented features, and the proportion of users who are satisfied with *MATLAB*'s support for modularity.

The initial motivation for the present work was to obtain reasonably solid support for the claim that a significant part of the MATLAB community does not use the object-oriented parts of MATLAB in their tasks. Some of our previous work was based on this assumption [5–7], but we felt a lack of citable references to support it. The present study represents a step towards filling that gap. The present study also provides diverse insights into MATLAB's use patterns in our sample population, which are potentially useful for entities responsible for the language's future evolution.

In future work, we plan to deepen the characterization of the community of users and programmers *MATLAB* and its clones and better understand the usage patterns of this group of languages. For instance, concerning the use of OOP in *MATLAB*, participants revealed different methods that are worthy of further analysis. Although we were capable

of analyzing which of these modules *MATLAB* programmers use, we do not know the frequency or the purpose with which they use each module. This aspect could be further analyzed, allowing us to better understand how the community perceives the strengths and weaknesses of each of these modules. A more focused analysis of the users' opinions on *MATLAB* and its competitors is also interesting. While we can measure user satisfaction and analyze which other languages the community deems as strong competitors, we were not able to explore the rationale behind those answers. Further studies hold the promise of leading to additional insights on the perceived limitations *MATLAB* and point to possible roads for improvements.

CRediT authorship contribution statement

Eduardo Reis: Investigation, Data curation, Formal analysis (statistical), Writing (original thesis document), Visualization (figures and tables). **Catarina Gralha:** Methodology, Supervision, Formal analysis (statistical), Writing (paper) – Reviewing and Editing. **Miguel P. Monteiro:** Conceptualization, Supervision, Writing (paper) – Reviewing and Editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by NOVA LINCS, Portugal (UIDB/04516/2020).

References

[1] MathWorks, Math. Graphics. Programming, 2022, URL https://www.mathworks.com/products/matlab.html?s_tid=hp_products_matlab (Last access: August 2022).
[2] MathWorks, MathWorks - about us, 2022, URL https://www.mathworks.com/company/aboutus.html (Last access: August 2022).
[3] Stackoverflow, What is MATLAB good for? Why is it so used by universities? When is it better than Python?, 2022, URL https://stackoverflow.com/questions/179904/what-is-matlab-good-for-why-is-it-so-used-by-universities-when-is-it-better-th (Last access: August 2022).
[4] C. Moler, J. Little, A history of MATLAB, Proc. ACM Program. Lang. 4 (HOPL) (2020) URL https://doi.org/10.1145/3386331.
[5] M.P. Monteiro, J. Cardoso, S. Posea, Identification and characterization of crosscutting concerns in MATLAB systems, in: Conference on Compilers, Programming Languages, Related Technologies and Applications (CoRTA 2010), Braga, Portugal, 2010, pp. 1–12.
[6] M.P. Monteiro, N.C. Marques, B. Silva, B. Palma, J. Cardoso, Toward a token-based approach to concern detection in matlab sources, in: EPIA Conference on Artificial Intelligence, Springer, 2017, pp. 573–584.
[7] N.C. Marques, M.P. Monteiro, B. Silva, Analysis of a token density metric for concern detection in matlab sources using UbiSOM, Expert Syst. 35 (4) (2018) URL https://doi.org/10.1111/exsy.12306.
[8] GNU octave - about, 2022, URL https://www.gnu.org/software/octave/about.html (Last access: August 2022).
[9] Scilab, Scilab - About, 2022, URL https://www.scilab.org/about/scilab-open-source-software (Last access: August 2022).
[10] Scilab, Scilab - company, 2022, URL https://www.scilab.org/about/company (Last access: August 2022).
[11] ESI, ESI group - leading innovator in virtual prototyping solutions, 2022, URL https://www.esi-group.com/ (Last access: August 2022).
[12] I. Searle, Rlab web site, 2022, URL http://rlab.sourceforge.net/ (Last access: August 2022).
[13] M. Kostrun, Rlabplus, 2022, URL http://rlabplus.sourceforge.net/ (Last access: August 2022).
[14] B. Meyer, Object-Oriented Software Construction, second ed., Prentice hall Englewood Cliffs, 1997.
[15] E. Rietsch, Separation of concerns, 2022, URL https://en.wikipedia.org/wiki/Separation_of_concerns (Last access: August 2022).
[16] E. Reis, Surveying Communities of Users of MATLAB and Clone Languages, Universidade NOVA de Lisboa, Portugal, 2021.
[17] K. Duarte, Limitations in the Support to Modularity in MATLAB: a Survey-Based Empirical Study (Master's thesis), Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, 2017.
[18] Qualtrics, Online survey software - qualtrics, 2022, URL https://www.qualtrics.com/uk/core-xm/survey-software/ (Last access: August 2022).
[19] LinkedIn, LinkedIn, 2022, URL https://linkedin.com/ (Last access: August 2022).
[20] Facebook, Facebook, 2022, URL https://www.facebook.com/ (Last access: August 2022).
[21] MathWorks, MATLAB central - about, 2022, URL https://www.mathworks.com/matlabcentral/about.html?s_tid=gn_mlc_about (Last access: August 2022).
[22] P. Cretchley, C. Harman, N. Ellerton, G. Fogarty, MATLAB in early undergraduate mathematics: An investigation into the effects of scientific software on learning, Math. Educ. Res. J. 12 (3) (2000) 219–233.
[23] H.P. Wallin, U. Carlsson, U. Ross, K.E. Gaidi, Learning MATLAB: Evaluation of methods and materials for first-year engineering students, Int. J. Eng. Educ. 21 (4) (2005) 692–701.
[24] S.R.H. Hoole, Programming skills in graduate engineering classes: Students from disparate disciplines and eras, Int. J. Eng. Educ. 26 (3) (2010).
[25] SciPy developers, SciPy, 2022, URL https://www.scipy.org/ (Last access: August 2022).
[26] M. Gwóźdź, Github – scipy user survey results, 2022, URL https://github.com/mkg33/GSoD/blob/master/user_survey_summary.pdf (Last access: August 2022).
[27] Google Forms, Google forms: Free online surveys for personal use, 2022, URL https://www.google.com/forms/about/ (Last access: August 2022).
[28] Twitter, Twitter, 2022, URL https://twitter.com/ (Last access: August 2022).
[29] JetBrains, Python developers survey 2020 results - JetBrains, 2022, URL https://www.jetbrains.com/lp/python-developers-survey-2020/ (Last access: August 2022).
[30] JetBrains, Methodology - the state of developer ecosystem 2020, 2022, URL https://www.jetbrains.com/lp/devecosystem-2020/methodology/ (Last access: August 2022).
[31] Quora Inc., Quora, 2022, URL https://quora.com/ (Last access: August 2022).
[32] EthicalAds, Privacy-preserving ad network for developers - EthicalAds, 2022, URL https://www.ethicalads.io (Last access: August 2022).
[33] P. Prabhu, H. Kim, T. Oh, T.B. Jablin, N.P. Johnson, M. Zoufaly, A. Raman, F. Liu, D. Walker, Y. Zhang, et al., A survey of the practice of computational science, in: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE, 2011, pp. 1–12.
[34] MathWorks, Surveying the MATLAB community, 2022, URL https://www.mathworks.com/matlabcentral/answers/541361-surveying-the-matlab-community (Last access: August 2022).
[35] ResearchGate, Surveying the MATLAB community, 2022, URL https://www.researchgate.net/post/Surveying_the_MATLAB_community (Last access: August 2022).
[36] Reddit - MATLAB, Surveying the MATLAB community, 2022, URL https://www.reddit.com/r/matlab/comments/hj91dx/surveying_the_matlab_community/ (Last access: August 2022).
[37] R.E. Students, Surveying the MATLAB community, 2022, URL https://www.reddit.com/r/EngineeringStudents/comments/hj9n6t/surveying_the_matlab_community/ (Last access: August 2022).
[38] LinkedIn, MATLAB users and integrators, 2022, URL https://www.linkedin.com/groups/134533/ (Last access: August 2022).
[39] LinkedIn, MATLAB for beginners and experts, 2022, URL https://www.linkedin.com/groups/1843503/ (Last access: August 2022).
[40] LinkedIn, Scilab software, 2022, URL https://www.linkedin.com/groups/3688414/ (Last access: August 2022).
[41] LinkedIn, GNU octave users and developers, 2022, URL https://www.linkedin.com/groups/4044339/ (Last access: August 2022).
[42] Discourse, A study on the users of MATLAB and similar languages, 2022, URL https://octave.discourse.group/t/a-study-on-the-users-of-matlab-and-similar-languages/467 (Last access: August 2022).
[43] C. Wohlin, P. Runeson, M. Höst, M. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Springer Science & Business Media, 2012.
[44] E. Reis, C. Gralha, M. P. Monteiro, Surveying communities of users of MATLAB and clone languages: Companion site, 2022, http://dx.doi.org/10.5281/zenodo.7100751, Last access: August 2022.
[45] IBM, SPSS statistics, 2022, URL https://www.ibm.com/products/spss-statistics (Last access: August 2022).
[46] M.G. Kendall, A new measure of rank correlation, Biometrika 30 (1/2) (1938) 81–93.
[47] Laerd Statistics, Kendall's Tau-b using SPSS statistics, 2022, URL https://statistics.laerd.com/spss-tutorials/kendalls-tau-b-using-spss-statistics.php (Last access: August 2022).
[48] D.A. Walker, JMASM9: Converting Kendall's tau for correlational or meta-analytic analyses, J. Modern Appl. Statist. Methods 2 (2) (2003) 525–530, http://dx.doi.org/10.22237/jmasm/1067646360.
[49] Laerd Statistics, Principal components analysis (PCA) using SPSS statistics, 2022, URL https://statistics.laerd.com/spss-tutorials/principal-components-analysis-pca-using-spss-statistics.php (Last access: August 2022).

[50] H.F. Kaiser, An index of factorial simplicity, Psychometrika 39 (1) (1974) 31–36.
[51] M.S. Barlett, The effect of standardization on a Chi-square approximation in factor analysis, Biometrika 38 (3/4) (1951) 337–344.
[52] L.J. Cronbach, Coefficient alpha and the internal structure of tests, Psychometrika 16 (3) (1951) 297–334.
[53] R.B. Kline, Principles and Practice of Structural Equation Modeling, Guilford Publications, 2015.
[54] R.F. DeVellis, Scale Development: Theory and Applications, Vol. 26, Sage Publications, 2016.
[55] V. Nijs, Compare a single proportion to the population proportion, 2022, URL https://radiant-rstats.github.io/docs/basics/single_prop.html (Last access: August 2022).
[56] G.V. Glass, A ranking variable analogue of biserial correlation: Implications for short-cut item analysis, J. Educ. Meas. 2 (1) (1965) 91–95.
[57] R.H. Somers, A new asymmetric measure of association for ordinal variables, Am. Sociol. Rev. (1962) 799–811.
[58] R. Newson, Confidence intervals for rank statistics: Somers' d and extensions, Stata J. 6 (3) (2006) 309–334.