Abstract

More and more researchers in linguistics use large-scale experiments to test hypotheses about the data they research, in addition to more traditional informant work. In this paper we describe a new set of free, open-source tools that allow linguists to post studies online, *turktools*. These tools allow for the creation of a wide range of linguistic tasks, including grammaticality surveys, sentence completion tasks, and picture-matching tasks, allowing for easily implemented large-scale linguistic studies. Our tools further help streamline the design of such experiments and assist in the extraction and analysis of the resulting data. Surveys created using the tools described in this paper can be posted on Amazon’s Mechanical Turk service, a popular crowdsourcing platform that mediates between ‘Requesters’ who can post surveys online and ‘Workers’ who complete them. This allows many linguistic surveys to be completed within hours or days and at relatively low costs. Alternatively, researchers can host these randomized experiments on their own servers using a supplied server-side component.

1. Introduction

In recent years, researchers in linguistics have begun to supplement traditional informant work with experimental methods. With crowdsourcing platforms such as Amazon’s Mechanical Turk (henceforth: AMT), researchers can develop and test hypotheses with many naïve speakers within days at very low cost. The purpose of this paper is to describe the process of creating an experiment, running it online, and obtaining results using originally-developed tools that allow researchers who have little programming experience to create linguistic surveys. These tools, as well as several example templates corresponding to different kinds of tasks, were written under a liberal, open-source license and are distributed for free online at [http://turktools.net](http://turktools.net).

The tools described here are designed for use with AMT, a popular online crowdsourcing website where companies or individuals (called *requesters*) can post small tasks that cannot

* For helpful comments and discussion of this paper and the associated tools, we would like to thank Martin Hackl, David Pesetsky, Coppe van Urk, and participants of our 2013 workshop at MIT on designing linguistic experiments for Mechanical Turk. The current paper has also greatly benefited from the feedback of four anonymous NLLT reviewers, as well as the editor Marcel den Dikken. Any and all errors are ours.
easily be automated, and therefore require human workers (called workers) for completion.\footnote{In AMT jargon, these tasks are called Human Intelligence Tasks, or HITs. The organization of linguistic surveys into HITs will be discussed in the appendix.} Most tasks on AMT are small in nature (such as identifying the contents of an image) and generally numerous—it is not unusual for requesters to post thousands of tasks at the same time. Requesters generally pay very little per task and retain the ability to Accept or Reject the results of each task before AMT sends payment to the worker. Tasks can be posted using an online interface (\url{http://mturk.com}), and their results are easily analyzable using mainstream statistical software (for more information, see Fort et al. 2011; Gibson et al. 2011; Sprouse 2011).

In the next section, we will discuss the value of experimental methods and, in particular, crowdsourcing for theoretical linguistics. In section 3, we discuss some suggestions for formulating linguistic questions that can be tested using experimental methods. In section 4, we present turktools: a set of tools that simplify the construction of online surveys for obtaining linguistic data, and a suggested workflow for their use. Details about the use of each of the tools and further suggestions can be found in the online supplement to this paper and in the documentation on the turktools website, at \url{http://turktools.net/use/}. Finally, in section 5 we provide a brief comparison with other existing experimental tools, in particular the turkolizer (Gibson et al. 2011) and Ibex (Drummond 2007).

While turktools were built with the AMT platform in mind, they by no means require the use of AMT. We have also developed the turkserver program that allows for AMT-style surveys to be hosted on the researchers’ own server, with independent recruitment of participants. This may be particularly useful for surveys conducted in languages that may not be well represented by AMT workers. For brevity, throughout the paper we describe the process of creating an experiment as if the researchers will upload the survey to AMT, but the majority of the process also applies to surveys hosted using turkserver.

2. The value of crowdsourcing in theoretical linguistics

2.1 Formal experiments and linguistic theory

Current research in theoretical linguistics relies heavily on direct informant work—obtaining linguistic data from one or several native-speaker consultants for a given language. This is often done by presenting a consultant with a sentence in a context or paired with a certain desired meaning and asking whether the sentence in appropriate in the given context (Chomsky, 1965; Schütze, 1996). This method has been criticized for often involving (a) relatively few participants, (b) a relatively small number of target stimuli, and (c) cognitive biases on the part of the researchers and participants (Schütze 1996; Cowart 1997; Edelman and Christiansen 2003; Featherston 2005; Ferreira 2005; Marantz 2005; Wasow and Arnold 2005; Myers 2009a,b; Sprouse 2009; Gibson and Fedorenko 2010).
Linguists have consistently offered strong arguments in response to these criticisms (e.g., Phillips and Lasnik, 2003; Marantz, 2005; Culicover and Jackendoff, 2010, Phillips 2010) and have reported formal experimental results, including those conducted on AMT, that corroborate many informal experimental results (Featherston, 2005; Phillips, 2009). Many judgments obtained through informal methods and presented in journals and textbooks have been experimentally replicated using diverse experimental methods and tasks, thus showing that many of the criticisms cited above are unwarranted (Munro et al. 2010; Cable and Harris 2011; Sprouse 2011; Sprouse and Almeida 2012, 2013; Sprouse, Schütze, and Almeida 2013).

Following this latter work, we believe that experimental methods are not necessary in order to substantiate every linguistic claim made in the literature. Nonetheless, such methods are often useful and helpful in the process of theory-building. Conducting an experiment allows the researcher to validate judgments obtained from consultants in the researcher’s vicinity with a larger population, and may allow for the detection of inter-speaker variation. Often, experimentation will lead to unexpected findings that help the researcher rethink and sometimes redefine the scope of the problem. Furthermore, experimentation makes it possible to quantify subtle patterns in the data. Using modern statistical tools, the researcher can model how complex inter-dependent factors might affect a linguistic phenomenon. Such statistical tools are also helpful in controlling for the effects of particular items and subjects tested in the experiment, to ensure that the observed effects are caused by the factors of interest and not by outliers that do not represent the general population.²

2.2 Online crowdsourcing for linguistic experiments

With crowdsourcing tools as AMT, researchers can develop and test hypotheses with many naïve speakers within days at relatively low cost. Therefore, although we do not advocate the use of these methods in all cases and at all costs, we believe that it is vital for theoretical linguists to develop the skills that would allow them to use such methods when necessary.

As the use of crowdsourcing websites has gained popularity in recent years, many studies have set out to compare experimental results of AMT workers and lab-based participants. These studies have found that crowdsourcing websites often allow for the recruitment of more diverse and representative participants than in many lab settings,³ and provide results that are as reliable as lab-based experiments.⁴ In addition to the replication of linguistic results, as noted above,

---

² The process of designing an experiment can itself be very valuable. As is often the case, expanding the scope of one’s investigation can lead to interesting findings about the factors that affect the phenomenon in question. Although this goal by itself can be achieved without experimentation, we believe that the exercise of turning a theoretical research question into a testable set of experimental predictions can inform one’s thinking about the problem.

³ Participants in university lab settings often tend to be college students, and hence have a restricted distribution of age, education, and socio-economic status.

⁴ An anonymous reviewer asks whether there has been a comparison of AMT and lab data for tasks involving timing, for example for Self-Paced Reading. To the best of our knowledge, although there is ongoing work attempting to answer this question (see Tily and Gibson, ms), there are no published results.
other experimental results in the social sciences have also been replicated, for example the Stroop, Switching, Flanker, Simon, Posner Cuing, attentional blink, subliminal priming, and category learning tasks, classical experimental tasks drawn from the heuristics and biases literature, psychometric data, and even clinical findings (Gosling, Vazire, Srivastava, and John 2004; Ipeirotis, 2010; Ipeirotis, Provost, and Wang 2010; Paolacci, Chandler, and Ipeirotis 2010; Buhrmester, Kwang, and Gosling, 2011; Horton, Rand, and Zeckhauser 2011; Mason and Siddharth 2011; Berinsky, Huber, Lenz 2012; Germine, Nakayama, Duchaine, Chabris, Chatterjee, and Wilmer 2012; Crump, McDonnell, and Gureckis 2013; Shapiro, Chandler, and Mueller 2013; and references therein). We note that these studies have also found limitations on the use of crowdsourcing for experimental studies, in particular when an experiment is excessively long or when insufficient compensation is offered. See Reips (2002) for a useful general overview of advantages and potential pitfalls of the use of internet-based experiments.

The most recent demographic study of AMT workers in the literature was conducted in 2010 (Ipeirotis 2010; also Mason and Siddharth 2011). Ipeirotis (2010) surveyed 1,000 Turk workers, and received responses from participants from 66 different countries. Of those participants, however, 46.80% were from the US, 34% were from India, and 19.20% were from other countries. The authors of this paper know that turktools have been successfully used to recruit participants on AMT for experiments in English, German, and Indian languages such as Telugu and Hindi. To our knowledge, however, there is no existing survey of the languages spoken by AMT workers. We refer the reader to the worker map in Tamir (2011), which details the country of residence of 50,000 AMT workers.\(^5\)

We note that some concerns have been raised about the nature of the AMT participant pool in the context of linguistic experiments, in particular as to the proportion of workers who complete most of the tasks posted on AMT (Fort et al. 2011). From our own experience with AMT, however, we believe that such concerns are unfounded. In all the AMT experiments conducted at the Experimental Syntax-Semantic Lab at MIT between November 2010 and April 2013, a total of 4635 unique workers participated in at least one study; 643 (14%) participated in more than one study, 152 (3%) participated in more than two studies, and only 15 (0.3%) participated in more than five.\(^6\) Thus, we believe that the diversity of our data is not jeopardized by the tendencies described in Fort et al. (2011).\(^7\)

\(^5\) A screen capture of this map can be found at http://turktools.net/crowdsourcing/.

\(^6\) Data collected on April 24, 2013. The vast majority of experiments were on English and restricted IP addresses of workers to within the US. Our experiments request that workers participate in each experiment only once.

\(^7\) The only quantitative data cited by Fort et al. (2011) to motivate this concern comes from Little (2009) who reports that, over a 75 day period in their lab at MIT’s Computer Science and Artificial Intelligence Lab, 22% of their workers completed 80% of their the tasks that they posted on AMT. However, these tasks are not linguistic experiments that request that workers participate only once per experiment, unlike for the results we report above from the Experimental Syntax-Semantics Lab at MIT.
3. Formulating a research question and choosing an experimental design

Constructing an online experiment begins with choosing an experimental design which can test the hypothesis at hand. One common design pattern in linguistics and other experimental cognitive sciences is a factorial design. 2×2 designs, in particular, are common because of their simplicity and the relative ease of the analysis of their results. We leave it to the readers to scale this design up or down to suit their own needs. Additional references on experimental design follow below.

For concreteness, we will illustrate the use of factorial designs with a well-known example from the literature: there-constructions have been argued to show a definiteness effect, whereby a definite or quantified DP cannot appear in a there-construction (Milsark 1974, 1977 and much subsequent work):

(1) a. There is a boy in the room
    b. *There is the boy in the room
    c. *There is every boy in the room

In what follows we restrict our attention to the contrast between (1a) and (1b). We may formulate the hypothesis that definite DPs are ungrammatical in a there-construction. To test this, we will look at sentences that differ in whether they contain a definite or an indefinite DP—the first factor. We will cross this factor with whether or not a sentence contains a there-construction. A sample item set for a simple survey testing the grammaticality judgments reported in the literature for the there-construction is given in (2), with predicted grammaticality judgments indicated. This design provides a baseline for the factors of interest, in (2a), which will be useful for analysis purposes.

(2) a. A boy is in the room. –def, –there
    b. The boy is in the room. +def, –there
    c. There is a boy in the room. –def, +there
    d. *There is the boy in the room. +def, +there

In many cases, testing multiple similar items, differing only in the lexicalizations and otherwise systematically manipulating the factors of interest, may further the validity of the experimental results. Turktools automates the process of randomizing items for presentation, which we discuss below. We refer the reader to the literature on internal validity and external validity for suggestions on how to correctly define one’s research goals, how many items to construct, and how to construct the items (see e.g. Cowart 1997, 2012, Gelman and Hill 2007, Myers 2009a,
and references therein). See also Sprouse (to appear) for a recent survey of literature regarding such considerations.

Experimental paradigms similar to that illustrated above have led to the discovery of novel findings relevant for linguistic theory. One such case is Langendon et al. (1973), who investigated the claim made in the linguistic literature that the indirect object of double object constructions cannot be questioned:

(3) a. Who(m) did you give this book to?  
b. *Who(m) did you give this book?

Langendon et al. (1973) conducted two grammaticality surveys in New York City, whose results show two groups of speakers: 80% of the subjects could not form such questions, but 20% found them grammatical. This finding suggests a need for a theory that does not rule (3b) out wholesale, but rather is able to allow it under some parametric variation.

A more recent study by Kotek et al. (2011) studied the semantics of the proportional quantifier most in subject position, as in the sentence most of the dots are blue. Using acceptability judgment and picture-matching tasks on AMT, Kotek et al. identify a previously unnoticed reading of most in subject position which is difficult to detect in offline tasks, as well as inter-speaker variation whereby only half of the population has access to this reading. This, again, motivated the construction of a more fine-grained theory of most that is able to explain the existence of this reading.

The tools provided in turktools support a variety of different types of experiments that are able to test a variety of linguistic phenomena and theoretical questions, some of which we describe below. For each type of experiment we describe here, we provide a skeleton HTML file for creating such an experiment. We will discuss skeletons in more detail in Section 4.2. For other types of experiments used in the psycholinguistic literature as well as recommendations for design and data analysis, see Schütze and Sprouse (2014) and citations therein.

3.1 Acceptability judgment task

Perhaps the most commonly used experimental task in syntax and semantics research is the acceptability judgment task where participants are asked to judge the acceptability or naturalness of a sentence or of a sentence-meaning pair. The dependent measure in this task is a rating on some scale, where the ends of the scale may correspond to ‘acceptable’-‘unacceptable,’ ‘natural’-‘unnatural,’ ‘grammatical’-‘ungrammatical,’ ‘good’-‘bad,’ etc. Alternatively, a fixed Likert scale, often with five or seven points, or a continuous scale (Chemla and Spector 2011) may be used. A similar magnitude estimation task requires participants to compare the

---

8 For example, Cowart 1997, 2012 gives practical suggestions for systematically constructing item paradigms in Excel. Myers 2009b presents MiniJudge, a tool designed to facilitating this process of constructing linguistic stimuli online.
acceptability of target sentences to some predetermined reference sentence (Bard et al. 1996). The acceptability judgment task is perhaps the easiest task to implement for online surveys. Much work has debated the usefulness of scales and magnitude estimation tasks for linguistic surveys as a way of gauging gradient acceptability of linguistic judgment. For comparisons of these tasks with binary ‘yes’-‘no’ tasks, see e.g. Keller 2000, Featherston 2005, Sprouse 2009, 2011, Fukuda et al. 2010, and Weskott and Fanselow 2011.

3.2 Forced-choice completion task

Another simple task is one in which a sentence or context is presented and the participant must choose which word or phrase better completes a sentence, or which whole sentence is better suited to describe a given situation. An example word completion task is given in Figure 1. The dependent measure is the rate of selection of each choice in comparison with its competitor.9

<table>
<thead>
<tr>
<th>There is _____ boy in the room.</th>
</tr>
</thead>
<tbody>
<tr>
<td>○ the ○ a</td>
</tr>
</tbody>
</table>

Figure 1. A sample word completion task

3.3 Picture-matching tasks

For tasks in which a visual scene must be presented, a picture-matching task is often used. In this task, participants are shown a picture and are asked to choose which among several options best describe the scene, or alternatively they are presented with a sentence and several pictures and must choose which picture best matches the description in the sentence. The dependent measure is the rate of choice of a given sentence or picture compared to its competitors. In another variant of the task, participants are presented with a picture and a sentence that describes it, and must decide whether or not the sentence is a good description of the picture. The picture-matching skeleton we provide also allows for an easy implementation of the ‘Covered Box’ technique, which is particularly suited for detecting less preferred interpretations of ambiguous sentences (for details, see Huang, Spelke and Snedeker, 2013; Pearson, Khan and Snedeker 2010).

3.4 Additional modifications

For the experiment types described above (and others that users of our tools might develop), several modifications can be easily made:10

- An audio file may be added.
- A context may be added before or after the target sentence.

---

9 Our supplied skeletons support choices introduced with buttons below the sentence, as in Figure 1, or with a drop-down menu.
10 Some of these modifications require custom JavaScript programming in the template. Our own templates utilize the jQuery JavaScript library (http://jquery.com/), and we recommend its use for such custom programming.
• A comprehension question may be added either before or after the target sentence.
• For cognitive load paradigms, the target task may be interrupted by a secondary task.
• Items may be presented one at a time instead of all together on the same page.
• The sequence in which items are shown can be controlled—for example, it is possible to first show a picture, then mask the picture and ask a question about the picture. This setup is also useful for cognitive load paradigms.
• Timing information may be logged for each item or parts of an item that have been viewed.

4. A suggested workflow for an AMT experiment using turktools

In this section we present a proposed workflow for the process of creating an experiment and uploading it to AMT, once an appropriate research question has been formulated. We introduce the different tools in turktools as they become relevant in the process. Details about the different tools and their use can be found in the appendices in the online supplement to this paper and in the documentation on the turktools website.\(^\text{11}\)

(4) Suggested workflow:

a. Formulate a research question and choose an appropriate experimental design. (Section 3)
b. Choose an appropriate skeleton to fit your design, and edit it for your needs.
c. Create an AMT-compatible template from the skeleton, using the Templater.
d. Create your items. Create an AMT-compatible randomized lists file, using the Lister.
e. Inspect your survey and make sure it is ready to be posted, using the Simulator.
f. Post your survey on AMT. Once you are done, decode your results file, using the Decoder.
g. Analyze your data using any statistical software of your choice.

For brevity, we describe the process of creating an experiment as if the researchers will upload the survey to AMT, but the majority of the process also applies to surveys hosted on the researchers’ own server using turkserver. We discuss turkserver at the end of this section.

4.1. Getting started: the structure of an AMT survey

An AMT survey consists of two parts that must be created by the researchers: (a) an HTML template, and (b) an items file. The template provides the overall structure of an experiment, and contains fields that specific strings will be plugged into. The items file contains the actual strings to be used, including the linguistic stimuli that will be presented to participants. AMT takes the template and items file, combines them, and presents them to workers for completion.

---
\(^{11}\) Turktools is an ongoing, open-source project. The documentation will be continuously updated as necessary, and we encourage contributions by other users. Details can be found at: http://turktools.net/use/.
Researchers who are using *turktools* for the first time are encouraged to begin the process by downloading *turktools* from [http://turktools.net/](http://turktools.net/).\(^\text{12}\) For ease of use, we recommend saving a copy of *turktools* in the same folder as all of the files associated with a particular study. If the study will be posted on AMT, we also encourage the researchers to create an account on AMT as a ‘Requester’ before continuing with the rest of the process.

### 4.2. Creating an AMT-compatible template: using the skeletons and Templater

Once a research question is formulated and an experiment type has been selected to test it, the presentation of the survey can be determined. Surveys are presented on AMT using an HTML template. Creating a template for use on AMT requires two steps: (a) choosing an appropriate skeleton out of the ones offered on as part of turktools and editing it to fit your study, and then (b) creating a template from the skeleton using the *Templater*.

*Skeletons* are recipes for different experiment types that abstract away from the number of items that will be presented in a given study. For each experiment type we surveyed in Section 3, we provide a corresponding skeleton. At the moment, 11 different skeletons are provided. These skeletons all share the same basic structure, illustrated in Figure 2 using the binary. skeleton.html skeleton. For details on the skeletons and this figure, see Appendix A.

\(^\text{12}\) At the time of writing, these tools require the use of Python 2.6.x or 2.7.x, available at [http://python.org](http://python.org). However, the tools described here and their prerequisites and usage are subject to change. Please consult the latest information at [http://turktools.net](http://turktools.net) before using these tools.
PLEASE COMPLETE AT MOST ONE {{code}} SURVEY. YOU WILL NOT BE PAID FOR COMPLETING MORE THAN ONE SURVEY WITH THIS CODE.

Instructions: ...

Consent Statement: ...

If your browser has JavaScript turned on, a counter will be displayed at the bottom of the page indicating how many questions have been answered. It is highly recommended that you turn on JavaScript and use this tool before submitting to ensure that all questions have been answered and you can receive payment.

---

Demographic questions, questions about native language.

After submitting this HIT, do NOT submit another HIT with survey code {{code}}. You will not be paid for completing more than one survey with this code.

---

0 questions (out of {{total_number}} total) have been answered. If you submit now, you will not be paid.

Figure 2. An example skeleton file

After choosing a skeleton and editing it, a template must be created out of the skeleton using the Templatex Python script. Your template is now ready to be uploaded onto AMT. Details about the Templatex can be found in Appendix A of this paper.

4.3. Creating an AMT-compatible items file: using the Lister and Simulator

Once a template has been created, the next step is to construct the items, formatted to match the chosen skeleton. In our simple $2 \times 2$ experiment crossing two factors (here: definiteness, there-construction) with two levels each (here: +/- definite, +/- there-construction), a sample target item set for a sentence judgment task will look as in (5):
(5) A sample item set:

<table>
<thead>
<tr>
<th>Stimulus:</th>
<th>Condition name:</th>
<th>Factor values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. A boy is in the room.</td>
<td>indef-no.there</td>
<td>–definite, –there-condition</td>
</tr>
<tr>
<td>b. The boy is in the room.</td>
<td>def-no.there</td>
<td>+definite, –there-condition</td>
</tr>
<tr>
<td>c. There is a boy in the room.</td>
<td>indef-there</td>
<td>–definite, +there-condition</td>
</tr>
<tr>
<td>d. There is the boy in the room.</td>
<td>def-there</td>
<td>+definite, +there-condition</td>
</tr>
</tbody>
</table>

The raw items file can be created in any text editing software and saved as a plain text (.txt) file. Items must be formatted as in Figure 3 below. Details about the structure and format of the items file are given in Appendix B.

```plaintext
# target 1 indef-no.there
A boy is in the room.

# target 1 def-no.there
The boy is in the room.

# target 1 indef-there
There is a boy in the room.

# target 1 def-there
There in the boy in the room.
```

**Figure 3.** Sample item set with four conditions, in *Lister* format

Once the target and filler items for the survey have been constructed, the next step is to create multiple randomized lists of these items, which will help control for any possible effects of presentation order of your items. This step is done using the *Lister* tool. The *Lister* takes the raw items file and generates a file with multiple randomized lists of these items in comma-separated-value (.csv) format. The randomized lists file contains a row for each randomized list, including the actual text that the workers will see, with a header row at the top. AMT will substitute these strings into the HTML template for participants to see. Details about the *Lister* can be found in Appendix C.

Once the HTML template and randomized items file are created, you are ready to post your study on AMT. Before doing so, however, we recommend simulating your experiment using the *Simulator* tool. The simulator will take the template and items file and combine them to produce a simulation of an actual survey, as your participants will see it. We recommend that researchers complete their own study at least once. This can be beneficial for many reasons, including the detection of any problems or mistakes in the experiment. Details about the *Simulator* can be found in Appendix D.
Once the simulation is done, you are ready to post your study on AMT. To do so, see the instructions in Appendix E and at http://turktools.net/use/turk.html. Appendix E also discusses the importance of being a good requester, and about the Worker/Requester relationship.

4.3. Retrieving and analyzing the results of an AMT survey: using the Decoder and analysis.r

After your experiment has completed you can download the raw results file from AMT. We recommend saving this file in the same folder as the other documents produced in the course of preparing your survey. The Decoder script will create a decoded version of the file, which facilitates its analysis using statistical software. We have provided some basic R code in the analysis script analysis.r, which researchers can adapt for their own needs. Details about the Decoder and the analysis script can be found in Appendix F.

4.4. Using turktools with languages other than English

Turktools can also be used to construct experiments in other languages besides English. To do this, the researchers must (a) translate the skeleton that they wish to use into the target language, and (b) create items in that language. All other steps in the process remain as described above.

4.5. Hosting experiments without using AMT

Turktools were written specifically for use with AMT. However, in some cases researchers may choose not to use AMT for their study. This may be desirable, for example, in cases where the researchers would like to target a specific participant pool, or in cases when the target language of the survey is not well represented by AMT workers.

In such cases, researchers may choose to use the turkserver program we provide to host the experiment on their own server. If turkserver is used, participant recruitment and payment is left up to the researchers, who may choose any existing means for recruiting participants, e.g. mailing lists, popular websites in the target language, or recruiting of students through colleagues who work at universities in countries where the language is spoken. Turkserver is freely available at http://turktools.net/use/server.html under a liberal open-source license. Details about turkserver can be found in Appendix G.
5. Comparison with other experimental tools

A number of other software tools exist to aid in the construction of online linguistic surveys, but differ from turktools in their design, flexibility, and technical know-how required. The turktools we provide and describe here are inspired by the turmoilizer tool described in Gibson et al (2011). In comparison to the turmoilizer, turktools has been designed to be a general-purpose tool, with a very flexible template and item set structure, supporting a wider variety of different experimental tasks. (See Appendices A and B on the structure of turktools’ skeleton and item files, and in particular the Appendix’s footnote 23 which explicitly describes differences between the turmoilizer’s items format and turktools’.) Turktools also provides tools to automate more of the experiment construction and analysis process.13

Turktools and the turmoilizer are unique among tools for linguistic experiment generation in being explicitly designed with the AMT crowdsourcing platform in mind, allowing linguists to take advantage of the large, pre-existing participant pool on AMT. By providing the supplementary turkserver tool (Section 4.5), we go one step further in increasing the utility of our tools, so that the turktools we provide can also be used beyond the AMT participant pool.

Other tools offer the ability to implement a variety of complex experimental paradigms. Examples include WebExp (Keller et al. 1998, 2009) and Ibex (Drummond 2007). These platforms are designed to support a range of more sophisticated experimental paradigms, including those that control presentation timing and record response timing, such as self-paced reading (Just, Carpenter, and Woolley 1982) and speeded acceptability paradigms. AMT can be used to recruit participants for experiments on WebExp and Ibex, as well.14 For conducting timing-sensitive experiments, which turktools do not currently support, we recommend the use of these advanced tools.

6. Conclusion

In this paper we presented a set of tools we have developed that allow linguists to quickly and efficiently test diverse hypotheses stemming from their work using large-scale linguistic surveys. These tools further help streamline the design of such surveys and assist in the extraction and analysis of the resulting data. The tools were written with Amazon’s Mechanical Turk crowdsourcing platform in mind, and a supplied server-side component allows researchers to host these randomized surveys on their own servers without the need to use AMT, when

---

13 In the interest of space, we do not critically review the Gibson et al (2011) paper and turmoilizer tool here.
14 The strengths and increased flexibility of WebExp and Ibex come with a higher technical barrier to entry than turktools, both in terms of experiment creation and in the deployment of their experiments. Both are written as server-side software packages that are designed to run on the researchers’ own servers, configured in a particular way. To recruit participants for WebExp/Ibex experiments on AMT, a simple template is used in AMT to redirect participants to the externally-hosted survey. AMT provides a sample template, called “Survey Link,” for such purposes. An additional step of cross-referencing submissions between the AMT and WebExp/Ibex submission results is then necessary in order to verify experiment participation in order to pay participants on AMT.
appropriate. The ease and speed with which linguistic hypotheses can be tested on AMT makes this platform a useful tool for theoretical linguists, and we hope that more linguists will learn to take advantage of this resource. We hope that the tools we have developed will make experimental methods more accessible to researchers with less background in the use of such techniques and will thus make experimental methods more accessible to a wider range of working linguists.

**References**


Munro, R., Bethard, S., Kuperman V., Tzuyin Lai, V., Melnick, R., Potts, C., Schnoebelen, T., and Tily, H. 2010. Crowdsourcing and language studies: the new generation of linguistic data. Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk, Los Angeles, CA.


Myers, J. 2009b. The design and analysis of small-scale syntactic judgment experiments. Lingua 119. 425–44.


APPENDICES:

Posting a linguistic survey on AMT involves, in AMT jargon, the creation of a *Project*. Each Project has a corresponding HTML template with a fixed structure and number of items. The *Requester* uploads a *Batch* file, which we have referred to in this paper as a randomized lists file. Each row in the Batch file corresponds to one randomized list of experimental items and will be treated by AMT as a single Human Intelligence Task (HIT). In the AMT interface, the number of participants and predicted costs of a study are determined *per HIT*.

Below we present detailed information about the use of the different components of *turktools*, which were introduced in section 4 of the paper. These tools as presented in the order that they are used in the recommended workflow, described in the paper. Further ongoing documentation for the *turktools* project can be found online at: [http://turktools.net/use/](http://turktools.net/use/).

**A Skeletons and Templater: Creating an HTML template for use on Mechanical Turk**

Once a research question is formulated and an experiment type has been selected to test it, the presentation of the survey can be determined. Surveys are presented on AMT using an HTML template. We recommend creating a template in two steps: (a) choosing an appropriate *skeleton* out of the ones offered on as part of *turktools* and editing it to fit your study, and then (b) creating a template from the skeleton using the *Templater*.

*Skeletons* are recipes for different experiment types that abstract away from the number of items that will be presented in a given study. They are stripped-down versions of HTML templates which should be edited to fit each particular design or experiment. We recommend the use of a programming text editor to edit these files. For each experiment type we surveyed in Section 3 of the paper, we provide a corresponding skeleton:

(6) **Supplied skeleton files:**

   a. `binary.skeleton.html`  grammaticality judgment, yes-no task
   b. `slider.skeleton.html`  grammaticality judgment, gradient judgment task
   c. `likert.skeleton.html`  grammaticality judgment, Likert scale task
   d. `magnitude-estimation.skeleton.html`  grammaticality judgment, magnitude estimation task
   e. `constant-sum.skeleton.html`  grammaticality judgment, constant-sum scale task
   f. `image-choice.skeleton.html`  picture selection task
   g. `binary-image.skeleton.html`  picture judgment, yes-no task
   h. `binary-audio.skeleton.html`  audio stimuli, yes-no task
   i. `completion.skeleton.html`  word-completion task
   j. `sentence-choice.skeleton.html`  sentence-completion task

---

15 Good, free options include *Notepad++* for Windows and *TextWrangler* for Mac.
16 For technical details on the use of the audio template, see [http://turktools.net/use/audio.html](http://turktools.net/use/audio.html).
k. completion-menu.skeleton.html completion task with drop-down menu

A skeleton is itself an HTML file, but contains substitution tags which will be filled in by the Templater. These tags are all wrapped in double curly braces, i.e. {{...}}. The skeletons we provide all share the same basic structure, illustrated in Figure 4, using the completion.skeleton.html skeleton. The very top of the skeleton has a comment to remind users that a template must be created out of the skeleton before uploading to AMT.\textsuperscript{17} Next are the instructions for the survey, including any practice items, as well as a consent statement and contact information for the experimenters.\textsuperscript{18} Finally, there is text that requests workers to turn on JavaScript in their browser. This enables a counter that is included in all the skeletons which helps workers make sure that they have answered all the questions in the survey.\textsuperscript{19}

\textsuperscript{17} Comments, which will be completely ignored and removed by the Templater, are tags with an exclamation point, as in {{!...}}.

\textsuperscript{18} Approval for your proposed experiments from the Institutional Review Board (IRB) at your institution may be required before you run them on AMT. Please consult your institution’s IRB.

\textsuperscript{19} Researchers can make the use of JavaScript obligatory, for example if JavaScript is used for critical manipulations such as hiding and revealing stimuli or recording timing information. For most of the template skeletons that we provide, it is possible for participants to take part in surveys even if JavaScript is not enabled in their browser. The result is that they will not see the counter, but other functionality of the survey will be maintained.
Survey Code: {{code}}

PLEASEx20COMPLETE AT MOST ONE {{code}} SURVEY. YOU WILL NOT BE PAID FOR COMPLETING MORE THAN ONE SURVEY WITH THIS CODE.

Instructions
This questionnaire presents {{total_number}} English sentences. Each sentence contains a gap and there are two options below the sentence for what should go in that gap. Choose the option that sounds more natural to you. Here is an example:

There is ___ boy in the room.
○ the  ○ a

In this example...

Consent Statement: ...

If your browser has JavaScript turned on, a counter will be displayed at the bottom of the page indicating how many questions have been answered. It is highly recommended that you turn on JavaScript and use this tool before submitting to ensure that all questions have been answered and you can receive payment.

{{#items}}
{{number}}.
{{field_1}} ___ {{field_2}}
○ {{field_3}}  ○ {{field_4}}
{{/items}}

Demographic questions, questions about native language.

0 questions (out of {{total_number}} total) have been answered. If you submit now, you will not be paid.

After submitting this HIT, do NOT submit another HIT with survey code {{code}}. You will not be paid for completing more than one survey with this code.

Figure 4. An example skeleton file for a completions study, completion.skeleton.html

Notice the substitution tags in this skeleton. The Templater will replace {{code}} with the experiment’s unique code, and {{total_number}} with the number of items presented in the experiment.

The “survey code” at the top of the template is useful for record-keeping purposes on the experimenter’s end. It is also used to instruct workers to only complete one survey of a certain type. This is often desired in linguistic experiments, as collecting multiple judgments from the
same participant may result in skewed results of a study, if the researchers mistakenly treat these observations as independent. The same effect can be achieved, for example, by color-coding experiments instead of, or in addition to, displaying experiment codes, which will make it easier for participants to remember whether or not they have already participated in the study. Another way to ensure that workers only participate in a study once is to add JavaScript code to the template to check the current Worker’s ID against a list of previous participants’ IDs and to block the Worker from participating if they have already done so.\footnote{These alternative methods are not at the moment a default part of turktools. See the online documentation for how to implement these features in your skeleton, at \url{http://turktools.net/use/color.html} and \url{http://turktools.net/use/check.html}. We thank two anonymous reviewers for suggesting these methods.}

The main body of the skeleton is the items block, beginning with {{#items}} and ending with {{/items}}. The items block contains one sample item of the shape that all items in the survey will take. When a template is created out of the skeleton, this block will be duplicated as many times as there are items in your survey. The item contains the tag {{number}}, which will be replaced with the item number in the experiment, as well as a number of {{field_n}} tags. In this case, each item is made up of four fields: fields 1 and 2 are the parts of the sentence before and after the gap and fields 3 and 4 correspond to possible answer options. Each of these {{field_n}} tags will be replaced by a different “field” in the items file. The makeup of the items file will be described in the following section.

Below the items block are some demographic questions. We always ask participants to indicate their native language and any other languages they speak, although we clarify that payment is not contingent on their answers to these questions. Researchers may wish to add questions about gender, age, origin, etc. Finally, at the bottom of the skeleton is a counter to help workers ensure that they reply to all questions before submitting their survey. Below the counter is text warning participants not to accept a second survey with the same code after submitting their survey.

Note that all parts of the skeleton can be edited, moved around or removed to suit the needs of the researchers and the study. We recommend saving the changes to the skeleton in a dedicated folder that will contain all files pertaining to the same experiment. At least some changes will only have to be made once—for example, the consent statement should be the same in all surveys conducted under the same IRB approval.

After choosing a skeleton and editing it, a template must be created out of the skeleton using the Templater. The script templater.py should be saved in the same folder as the skeleton. The templater.py script and all other Python scripts described in this paper can be run in the following manner:

(7) Executing a Python script (here templater.py):
   a. UNIX shell: move (cd) to the directory that contains the script and execute python templater.py.
b. **Mac OS X**: Right-click on the file in the Finder and choose Open With... > Python Launcher.

c. **Windows**: Double-click on the file.²¹

The **Templater** will ask for three pieces of information:

(8) a. the skeleton file name: one of the skeletons provided here or others you create yourself.
b. the number of items in your survey: including all experimental and filler items but not practice items, which are coded in the skeleton.
c. a survey code: any letter-number combination you choose.

Once these are provided, a template file will be created. The template file will be located in the same folder as the skeleton file and the `templater.py` script.

**B Creating an items file**

Once a design and a template have been decided on for your survey, items need to be created. For a simple 2×2 experiment crossing two factors (here: *definiteness, there-construction*) with two levels each (here: +/- *definite, +/- there-construction*), each target item set will contain four different conditions. A sample target item set for a sentence judgment task will look as in (9).

(9) **A sample item set:**

<table>
<thead>
<tr>
<th>Stimulus</th>
<th>Condition name:</th>
<th>Factor values:</th>
</tr>
</thead>
<tbody>
<tr>
<td>a. A boy is in the room.</td>
<td>indef-no.there</td>
<td>–definite, –there-condition</td>
</tr>
<tr>
<td>b. The boy is in the room.</td>
<td>def-no.there</td>
<td>+definite, –there-condition</td>
</tr>
<tr>
<td>c. There is a boy in the room.</td>
<td>indef-there</td>
<td>–definite, +there-condition</td>
</tr>
<tr>
<td>d. There is the boy in the room.</td>
<td>def-there</td>
<td>+definite, +there-condition</td>
</tr>
</tbody>
</table>

**Turktools** uses a naming convention whereby the *condition name* specifies the chosen value for each of the factors, separated by hyphens (10). Names of this form will be assumed by the **decoder**, which we will discuss in Appendix F.²² For our *there*-construction experiment, this results in conditions names as in the middle column of example (9), where the settings of the two factors are shown in the third column and the first column shows the actual sentence that we would like participants to rate.

(10) **The structure of a condition name:**

<factor 1 value>--<factor 2 value>--<factor 3 value>--...

---

²¹ For Mac OS X and Windows, these instructions assume that Python was installed using the standard Python binary packages provided at [http://python.org](http://python.org). If not, use the UNIX shell instructions via the Terminal on Mac, and open using the `python` application on Windows. Please consult the latest information at [http://turktools.net](http://turktools.net) regarding supported Python versions and dependencies.

²² It is possible to use other naming schemes for condition names. In such a case, however, part of the decoding process, which is automated in the *Decoder*, will have to be done manually by the researchers. See footnote 30.
An experiment is composed of different sections, where normally at least one section will be used as fillers, and the others will be targets. This designation is made when using the Lister (Appendix C). Each item in the items file has two parts, each beginning on a new line.  

(11) a. The item header. The item header consists of four components, separated by spaces.
   (i) the symbol #;
   (ii) the name of the section (e.g., target in the example in Figure 3);
   (iii) the item set number within the section (we recommend 1, 2, 3, ...);
   (iv) the condition name (e.g., indef-no.there, def-no.there, indef-there, def-there in our example). The condition names are not constrained in any way, but it is most helpful to use labels that reflect the experimental design, and in particular to describe the setting of the factors of interest for that item.

b. The item body. The item body consists of fields, each on its own line. Line n corresponds to the text that will be substituted for the text {{field_n}} in the skeleton file. A field may specify a sentence to be judged, choices for completions, a picture, an audio file, a context, a comprehension question, etc.

Note that the choice of experimental design and skeleton will dictate the format of your items. That is, each item must have at least as many fields as there are fields in the sample item in the skeleton that you have chosen. It is possible to add hidden fields to the body of an item by adding extra lines that do not have corresponding substitution tags in the HTML template. This is a way to specify additional information for a given item that is useful for the researchers but should not be available to the participants. For example, if a skeleton refers to {{field_1}} through {{field_4}} but not {{field_5}}, then anything that occurs in the 5th line (and onward) of an item’s body will not be displayed on the screen and participants will not have access to this information. However, the data will still be part of the results file created by AMT after the experiment is completed and can be used as part of the data analysis. For example, the correct answers to comprehension questions or expected answers to filler items may be specified in this way.

---

23 This structure is very similar to that used for items files in Linger, written by Doug Rohde (http://tedlab.mit.edu/~dr/Linger), and Gibson et al.’s (2011) Turkolizer. Note that our terminology differs slightly: Gibson et al refers to a group of trials together forming an item, which corresponds to our notions of individual items which are grouped together into item sets.

Furthermore, the structure of the item body in our Lister format is much more flexible, allowing for an arbitrary number of fields. Unlike in Linger and Gibson et al.’s Turkolizer, comprehension questions are not given a special status but are instead treated like any other field that could be added to an item. If a comprehension question is given or if some filler items are expected to have correct answers, these values should be specified as hidden fields in the items file; see discussion following (11).

24 If a line break is required within a single text field, use the HTML code <br/>

25 If some items have more than one hidden field, it is important to make sure that each field is exclusively used for one purpose (e.g. field 5 for the correct answer to the item and field 6 for the expected answer to a comprehension question). It is possible to leave a field empty if a given item does not have a value associated with some field (e.g.,
The format for a sample item set in a there-construction study is given in Figure 5. For illustration, the experiment here pairs each sentence with a picture. Each item consists of two fields, the first specifying the sentence and the second, a picture. This corresponds to two substitution tags in the binary-image.skeleton.html skeleton: {{field_1}} stands in for the sentence, and {{field_2}} for the URL in an <img> tag, specifying the picture to be shown.

```
# target 1 indef-no.there
A boy is in the room.
http://DOMAIN/experimentpictures/definiteness-effect/picture1.png

# target 1 def-no.there
The boy is in the room.
http://DOMAIN/experimentpictures/definiteness-effect/picture1.png

# target 1 indef-there
There is a boy in the room
http://DOMAIN/experimentpictures/definiteness-effect/picture1.png

# target 1 def-there
There in the boy in the room
http://DOMAIN/experimentpictures/definiteness-effect/picture1.png
```

Figure 5. Sample item set with four conditions in Lister format

C Lister: Creating randomized lists of items to be uploaded on Mechanical Turk

The Lister takes an items file as input and returns a comma-separated-value (.csv) format file with randomized lists of items that can be uploaded to AMT. AMT will combine this item lists file with the template created by the Templater to create the final survey that will be viewed by participants. When you run lister.py, the program will maximally ask for the following information:

(12) a. the name of your items file (described in the previous section)
   b. which sections should be treated as fillers
   c. how many filler items you would like placed between each target item

no expected correct answer to an item, but there is an expected correct answer to the comprehension question—leave field 5 empty, as a blank line, and enter the expected correct answer to the comprehension question in field 6).

26 Resources such as images and audio or video files must be hosted separately, rather than included with the AMT experiment. We recommend hosting such resources on your own web space, for example in a hidden directory on your website, and including links to those resources in the items.
d. how many filler items you would like placed at the beginning and end of the experiment

 e. how many lists you would like to create

 f. whether or not you would like the reverse of each list to also be created, to help reduce any ordering effects that may occur

After this information is given, the Lister will create as many lists as requested, using a Latin Square design. That is, for each item set in a given section, an item for only one of the specified conditions for that item set will be included in a list. If the number of conditions does not match for all item sets in a section, an error will be given. Note that if all the item sets in a particular section have only one item, they will all be included, regardless of the condition names given.  

It is possible to dictate a minimum number of filler items to be placed between each two target items and at the beginning and end of each list, if there are enough filler items. The algorithm creates counterbalanced and randomized lists for each section individually (including the fillers), and then spreads fillers between target items to minimally comply with the constraints given to the Lister. If additional filler items are left after this process, they are randomly distributed across the list. If there are not enough filler items to comply with the constraints given to the Lister, an error message will appear. Note that it is possible to have less filler items than target items, but in that case it is not possible to impose restrictions on their position relative to target items. It is also possible to have no sections designated as fillers. In this case, all the items in the file will be randomized first within their sections and then across other sections, but no other conditions can be imposed on the ordering.

If some items have more fields than others (for example, if some items have expected correct answers but others don’t), the Lister will produce a message notifying the user of the discrepancy. The user may choose to go back and check that the file is correct or to continue with the current file. Assuming no such issues, the Lister will output a randomized lists file in CSV format (xxxxxxxx.turk.csv) that can be uploaded onto AMT. For example, if the input to the Lister is the items file definiteness-effect.txt, the output will be named definiteness-effect.turk.csv. The randomized lists file contains a row for each randomized list, indicating the actual text that the workers will see, with a header line at the top. AMT will substitute these strings into the HTML template for the participants to see.

**D Simulator: Verifying that the survey works properly**

With your items randomized into lists and your HTML template prepared, you are in principle ready to upload your survey onto AMT. However, before doing so, we recommend pre-testing

---

27 In a filler section where each item set is made up of one item each, it may be helpful to use different condition names even though all items will be displayed. This may help make the analysis script easier to create. For example, some fillers may be designated as ‘catch’ items. Accuracy can then be calculated for those items only and then used to discard participants from the analysis.
your study using the Simulator script provided here. The Simulator takes as input a randomized
lists file (xxxxxxx.turk.csv) and a template and creates a version of the survey based on one
of the lists in the item lists file (you can choose which list to simulate). The Simulator allows
researchers to bypass the rather cumbersome process of uploading a survey onto AMT, or using
AMT’s Sandbox to simulate an experiment. The output of the Simulator is an HTML file that
can be opened in any web browser.

It is beneficial for researchers to complete their own study, for several reasons. Doing so helps
spot any typos or errors in the rendering of the survey. If your items file includes hidden fields,
you should make sure that all hidden fields are not displayed and all other fields are displayed as
expected. Additionally, it is important to get a feeling for how hard the experiment is and how
long it takes to complete it. This will help you determine payment for your survey. Furthermore,
because of the existence of satisficing behavior on AMT—the behavior of some workers to put
in minimal effort and still get paid—it is useful to make sure that there are no easy strategies to
complete the study without doing the linguistic task of interest. Strategies for dealing with such
behavior include the addition of comprehension questions to items, the inclusion of ‘catch’ filler
items that should have clear correct answers, and the use of diverse types of fillers that cannot all
be answered using the same strategy.

E Uploading surveys onto Mechanical Turk and downloading results files

After verifying that the survey is ready, it can be uploaded onto AMT. We refer the reader to our
online instructions on uploading surveys onto AMT: http://turktools.net/use/turk.html. In some
cases it may be necessary to reject submissions from workers who have completed more than
one survey. It may also be desirable to reject workers with low accuracy on comprehension
questions and ‘catch’ filler items and those workers who failed to complete at least a certain
proportion of the study, e.g. 80%. To quickly identify workers who completed more than one
survey, open the file in software such as Excel and sort by ‘WorkerId.’ Some suggestions and
sample code for additional exclusion criteria and how to identify workers who meet them using
R can be found in the analysis script analysis.r which is provided with turktools and can be
modified by researchers to meet their own needs. In order to maintain a positive reputation as a
requester on AMT, it is advisable to approve submissions in a timely fashion and to maintain a
low rejection rate.

A note on being a good requester: We believe it is important to maintain a positive reputation as
a requester and to treat workers fairly. Note that workers expect their HIT results to be accepted
unless something has gone seriously wrong, for example if they did not complete a large portion

28 Amazon Mechanical Turk’s Sandbox allows researchers to upload “dummy” versions of their experiment, much
like using the Simulator. However, to use the Sandbox it is necessary to set up the experiment independently on the
Sandbox. This entire process must be repeated every time a mistake is discovered. Then one must go through all the
same steps once again in order to create the actual survey on AMT; there is no way to transfer a survey from one
platform to the other.
of the experiment or exhibited obvious guessing behavior. Workers strive to maintain a low rejection rate. This is very important as it allows them to work on tasks that restrict workers’ acceptance rates. As a result, many workers will avoid HITs by requesters who reject many HIT submissions, and will be very upset at being rejected if they believe that they completed the task in good faith. To maintain a good relationship with workers, we suggest that new requesters make sure to write clear instructions for their surveys which include specific rejection. We also encourage requesters to respond to email inquiries from workers promptly.29

As a general good practice, we recommend accepting all submissions unless a worker clearly did not follow the instructions. We have provided code that requesters can use to restrict the access of workers who they have identified as bad experimental participants (see http://turktools.net/use/check.html). It is also possible to restrict tasks to workers who have a certain acceptance rate and a certain number of tasks completed, using AMT’s “worker requirements” field, when creating a template online. We recommend using these criteria rather than using Masters, since Amazon's commission fee is lower for non-Masters and as far as we can tell, Masters are not better linguistic consultants than other AMT workers.

F Decoder: Preparing the results for analysis

After your survey is completed, download a final copy of the raw results file from the AMT Manage tab. We recommend re-naming this file, e.g. definiteness-effect.results.csv (to continue using the same example as above), and saving it in the same folder as the other documents produced in the course of preparing your survey. Use the Decoder script, which should be in the same folder. The Decoder will ask for the name of the raw results file (here: definiteness-effect.results.csv) and return a decoded file, here named definiteness-effect.results.decoded.csv.

Unlike the raw results file, the decoded results file contains one row for each item in each list in the survey. For each item, the decoded file specifies information about the Section, Condition, Item, List, PresentationOrder, and all fields and participant responses associated with that item, including hidden fields. In addition, the decoder adds a number of Factor columns, corresponding to different factor values. These factor values are constructed by simply splitting the Condition value up by hyphens.30 For example, in our example where an item may have Condition value ‘indef-no.there,’ Factor1 would be set to ‘indef’ and Factor2 would be set to ‘no.there.’ This processing of condition names using the schema in (10) facilitates the analysis of the data later.

29 AMT workers maintain websites to keep track of requesters’ reputations. Among other things, they collect information about payment, rejection rates, overall fairness, and promptness in responding to inquiries. See for example http://turkopticon.ucsd.edu/.
30 If condition names were not constructed using factor names and hyphens, as suggested in (10), custom code may be needed in the analysis script to recover factor values from the value of the Condition column.
Metadata about a submission, including the AssignmentId (unique for each worker-survey pair), SubmissionStatus, WorkerId, WorkTimeInSeconds (for the entire survey) and answers to demographic questions, is duplicated for each item for a given worker in the file. Therefore, for each participant who worked on your survey there will be as many rows in the file as there were items in the survey, and the meta-information about the submission will be the same in all rows from that submission.

Once decoded in this way, the results are ready to be analyzed by any standard statistics software. We have provided some basic code in the analysis script analysis.r, but researchers should adapt the script for their own needs.

G Hosting AMT-style experiments on your own server with turkserver

In addition to posting an experiment on AMT, the tools we have provided with this paper can be used to generate surveys to be hosted on the researchers’ own server. In order to do so, we have developed a program called turkserver, freely available at http://turktools.net/use/server.html under a liberal open-source license. Turkserver requires a server capable of serving PHP scripts, and is designed for Apache web servers. See the online documentation for detailed instructions on installation and usage.

To use this option, construct the experiment as described in Appendix A-D. An experiment is set up with turkserver by uploading the template and item lists file (.turk.csv). A URL for the experiment is generated, and this URL can then be shared with potential participants. Turkserver handles the random assignment of participants to different lists, displaying the survey using the chosen list of items, and recording the results in a format compatible with AMT’s results files.

There are some differences between running an experiment on AMT and on turkserver. AMT comes with many potential participants registered on the system, and therefore acts as a subject recruitment tool as well. On the other hand, the potential participant pool is limited to those people who have chosen to sign up to be a worker on AMT. With turkserver it is up to the experimenter to recruit subjects and send them to the experiment URL. Similarly, unlike AMT, turkserver does not handle payment to participants. Turkserver does, however, attempt to produce a unique AssignmentId for each submission, as AMT does, and to produce a WorkerId for each participant that is stable across experiments.

Running AMT-style surveys on turkserver can be useful for situations where the AMT subject pool does not include many speakers of a particular language, but such speakers can be recruited

---

31 But note that if a worker completed two surveys, for example, there will be twice as many rows. All the rows corresponding to the first survey completed by this worker will share metadata pertaining to the first submission and all rows corresponding to the second survey will share metadata pertaining to the second submission. The AssignmentIds will be distinct, but the WorkerId will be the same across these rows. This allows us to easily identify workers who completed multiple surveys. At the time of writing, AMT cannot itself enforce that each worker only complete one survey per logical experiment.

32 See also some analysis suggestions on the turktools website at http://turktools.net/use/analysis.html.
separately. It is also ideal for when the experimenter wishes to recruit participants from a particular group, such as a vetted subject pool, a class, or among colleagues. Furthermore, the ability to generate experiments that run both with and without AMT offers great flexibility. For example, one could use the exact same experimental materials (template and item lists file) and conduct the experiment with the general public on AMT, and also on recruited non-naïve (working linguist) participants via turkserver, and compare their behavior.