Adaptive Language Mapping

Version 7.40; April 10, 2019

Language Neuroscience Laboratory
Vanderbilt University Medical Center

1. Overview

Adaptive Language Mapping (ALM) is a procedure for identifying brain regions that are important for language processing, including in individuals with language deficits. A simple semantic matching task interacts with an adaptive staircase procedure that tailors item difficulty to individual performance, so that the same paradigm can be used to map language regions in patients with different degrees of language impairment, as well as healthy controls with normal language function.

The paradigm comprises a simple AB block design with language and control conditions. To avoid task-switching demands, which can be challenging for some people with impaired language function, a single task applies to both conditions: press a button to matching pairs.

In the language condition, pairs of words are presented, and a match is defined as a pair of words that are related in any way, e.g. *boy-girl*, *lizard-snake*, *grass-lawnmower*, but not *walnut-bicycle*.

In the control condition, pairs of false font strings are presented, and a match is defined as pair of strings that are identical (ΔΘδЂϟ-ΔΘδЂϟ, but not ΔΘδЂϟ-ϟΔƕƘΔ).

The language and control tasks are equivalent in terms of sensorimotor and executive components, yet make differential demands on language processing.

Critically, both the language task and the perceptual control task are independently adaptive to participant performance. Correct responses lead to the presentation of more difficult items, while incorrect responses lead to easier items.

In the language condition, item difficulty is manipulated via the salience of the semantic relationship between the matching pairs and by the lexical frequencies of the words. In the perceptual condition, item difficulty is manipulated via the degree of mismatch of the mismatching pairs (e.g. ΔΘδЂϟ-ΔΘƘδϟ is harder than ΔΘδЂϟ-ϟΔƕƘΔ). Inter-trial interval is also manipulated across both conditions.

The upshot of the adaptive staircase procedure is that all participants, including individuals with aphasia of different degrees of severity, as well as healthy controls, are always performing at a challenging yet mostly successful level in both conditions.

This document describes how to use the Adaptive Language Mapping software to implement the Adaptive Language Mapping protocol.

2. Requirements

ALM depends on MATLAB and the free open-source package Psychtoolbox-3.

ALM should run on any version of MATLAB that is compatible with Psychtoolbox-3. Officially, this is R2012a or more recent. However, both ALM and PTB-3 appear to work fine on R2011a, so older versions may work. On Mac OS X, Psychtoolbox-3 requires version 10.11 or later.

ALM has been tested on the following setups:

Xubuntu Linux 14.04, MATLAB R2011a, Psychtoolbox 3.0.12
Xubuntu Linux 16.10, MATLAB R2012a, Psychtoolbox 3.0.13
Mac OS X 10.11.6, MATLAB R2012a, Psychtoolbox 3.0.13
Mac OS X 10.12.2, MATLAB R2016a, Psychtoolbox 3.0.13
Windows 10, MATLAB R2015a, Psychtoolbox 3.0.13

3. Installation

(1) Install MATLAB. Officially, R2012a or later is required. No Mathworks toolboxes are needed.

(2) Install Psychtoolbox-3, by following the instructions on this website:
http://psychtoolbox.org/
If you are using Windows, it is important to install gstreamer, or text rendering will not work properly.

(3) Download *AdaptiveLanguageMapping.zip* from this website: https://aphasialab.org/alm

(4) Extract the *AdaptiveLanguageMapping* directory and place it in any convenient location, perhaps with your other MATLAB toolboxes. You should be aware that your log files and behavioral data will be saved to subdirectories within the *AdaptiveLanguageMapping* directory.

(5) Add the *AdaptiveLanguageMapping* directory to the MATLAB path and save the path. There is no reason to add the subfolders, and it is best not to.

4. Using the program

(1) Start MATLAB. It is recommended that MATLAB be run without its Java virtual machine, e.g. *MATLAB -nojvm*. On Windows, audio appears to work better in -nojvm mode. On other operating systems, no differences have been noted.

(2) Type *AdaptiveLanguageMapping* or *ALM* at the MATLAB command prompt.

(3) You should see the main menu, where the participant ID is entered, and experimental paradigms are selected. We will focus on the first three paradigms, which are the core of ALM. The others will be briefly discussed later.

(4) Start by entering a Participant ID; you won't be able to proceed until you do this. Only upper case letters, numbers, and the hyphen are permitted in the participant ID.

(5) Use the [Up] and [Down] arrows to choose paradigms to run (see below).

(6) When you run a paradigm, first a red crosshair appears while the stimuli are loaded and prepared. This may be instantaneous, depending on which experiment you are running, and whether it has already been loaded. Once the stimuli are loaded and prepared, the crosshair turns yellow. At this point, the program is waiting for a trigger from the scanner, or the experimenter, to start the experiment. The default trigger is the grave/tilde [`~] key, but this can be changed by editing the *almPreferences.m* file. When the trigger key is pressed, the paradigm begins.

(7) If you need to quit any paradigm, you can press the [Q] or the [Esc] key at almost any time to return to the main menu. When a paradigm is complete, the program will also return to the main menu.

(8) When you have finished running paradigms, choose the *Quit* item on the main menu, or just press [Q] or [Esc] when you are on the main menu.

(9) Before you being to use ALM, you should choose the *Check fonts* menu item, second from the bottom, to confirm that all 20 symbols used in the control stimuli are included in your monospaced font, and to confirm visual similarity between your proportional and monospaced fonts. If there are problems, you can change your fonts by editing *almPreferences.m*. The 20 symbols should look like this:

ƋƏƜƱƪΦΔΘδϟƕŒƘƟƧΣΓΞΨЖ

5. Adaptive Language Mapping

There are three phases to the Adaptive Language Mapping paradigm: training, practice, and the functional scan itself. The first two phases are run before the participant enters the scanner. However, all three phases should be run on the same computer, because the participant's performance history outside the scanner is used to set the initial difficulty levels and influences the choice of items inside the scanner. If it is not possible to use the same computer, you should copy the relevant history file from one computer to the next (this is described in detail below).

5.1. Phase 1: Training

Choose the first paradigm: *Adaptive semantic matching -- visual -- training*, then once you see the yellow crosshair and you are ready, press [`~] (or whatever your trigger key is) to begin.

The training phase is used to explain the experiment to the participant. To that end, you can present the four different kinds of trials as many times as you need, in whatever order suits you.

Press [S] to present a language “match” trial. You can remember “[S] = same”. Two words that are semantically related will be presented, one above the other. The participant should press one of the defined “match” keys to pairs of words that go together. Then, a box will appear around the two words to visually indicate that they have been grouped together.

Press [D] to present a language “mismatch” trial. You can remember “[D] = different”. Two words that are not semantically related will be presented. The participant is supposed to do nothing when presented with words that do not go together. The experimenter can press [Z] to remove the words from the screen, simulating what would happen in the real experiment.

Press [F] to present a control match trial. Two identical false font strings will be presented. The participant should press one of the defined match keys to pairs of identical false font strings. Then a box will appear around the strings.

Press [G] to present a control mismatch trial. Two different false font strings will be presented. The participant should do nothing when presented with strings that are not identical. The experimenter can press [Z] to remove the strings from the screen.

You can control the difficulty of the training items. There are 7 difficulty levels. The top row of keys, from [W] thru [I] are used to set the difficulty level.

Once you are satisfied that the participant understands the task, you can press [Q] or [Esc] to quit.

5.2. Phase 2: Practice

Next, select the second paradigm: *Adaptive semantic matching -- visual -- practice*. The practice paradigm is very similar to the paradigm that will be used in the scanner. There are alternating blocks of language and control items. Each block contains 4 to 10 items. The participant has a certain amount of time to respond “match” to each item; no response means “mismatch”.

The practice paradigm serves two functions. First, it allows the participant to become familiar with the timing of the tasks. Second, the difficulty levels reached at the end of the practice period are used as the initial difficulty levels when the participant begins the task in the scanner.

There are some subtle differences between the training and scanning paradigms. In brief, whereas the scanning paradigms attempt to match the timing and the number of letters presented precisely between the language and control blocks, the training paradigm shows more flexibility in these respects.

You should run the training paradigm until the participant has completed at least four blocks (language-control-language-control). After that, you can stop it whenever you are confident that they are comfortable with the task, and have reached a stable level of performance. The paradigm will end after 12 blocks, but it is unlikely you would ever need to carry out that much training.

You may wish to run the practice paradigm again while the subject is getting situated in the scanner, e.g. during localizer or anatomical scans.

5.3. Phase 3: Scan

Select the third paradigm: *Adaptive semantic matching -- visual -- standard scan* when you are ready to start the functional scan. The standard scan has 10 blocks per condition, for 20 blocks total, and a total time of 6:40. There is also a “quick scan” option (the fourth paradigm) which has only 6 blocks per condition, for 12 blocks total, and a total time of 4:00.

The scanner is expected to send a trigger keypress at the beginning of data acquisition, which is [`~] by default but can be changed by editing *almPreferences.m*.

Note that dummy prescans should be collected before the trigger keypress is sent.

The scan paradigm is very similar to the practice paradigm, except that the allowed response time is fixed for each pair of blocks, so that it matches between the two conditions. After every pair of blocks, the allowed response time is reset based on the current difficulty levels of the language and control conditions. Moreover, the total number of letters presented is matched in each pair of blocks in order to match visual processing.

6. Log files and history files

Log files are saved in the directory *AdaptiveLanguageMapping/logs*. The log files contain information about configuration, and detailed timing of the presentation of stimuli and responses received. The log files are generally intended for troubleshooting, and most users should not need to look at log files.

There is one log file for each time the *AdaptiveLanguageMapping* script is run. It is named based on the date and time that the script started, as well as the participant ID. In principle, it is possible to changed the participant ID and run multiple participants without quitting the program. In this case, the log file would be named based on the final participant run before quitting.

History files are saved in the directory *AdaptiveLanguageMapping/history*. There is one history file per participant ID. If the same participant is run in a separate session, even on a separate day, the same history file is used. This allows the difficulty levels reached at the end of one session to serve as the initial difficulty levels for the following session, and it also allows the algorithm to attempt not to reuse items as far as possible.

If you use one computer for practice outside the scanner, and a different computer to present stimuli in the scanner, then you should copy the history file for the participant you are running from the practice computer to the scan computer, so that the information recording during practice can be taken into account during the scan. If you plan on subsequent sessions (e.g. in a longitudinal study), you should copy the file back after the scan.

7. Auditory paradigms

Auditory paradigms are provided that can be used with participants with reading impairments, or children. The words are presented in auditory instead of written form. A "ding" indicates a response instead of a box around the words. The control stimuli consist or pairs of sequences of musical notes, which are either identical or not. The timing is a little slower than the visual version.

The auditory version of the task has not been piloted extensively. Initial testing suggests that it somewhat less effective than the written version.

8. Adaptive phonological paradigms

Adaptive rhyming and syllable matching tasks are provided in which participants must determine whether pairs of pseudowords rhyme or have the same number of syllables.

These paradigms activate phonological encoding regions in the ventral precentral gyrus and supramarginal gyrus that are not activated by the semantic task.

9. Other paradigms

Narrative comprehension, picture naming, and breath holding paradigms are also provided. The adaptive semantic matching paradigm has been directly compared to the narrative comprehension and picture naming paradigms; a paper describing this study is in preparation. The breath holding paradigm can be used to calculate differential voxelwise hemodynamic responses.

10. Credits

Adaptive Language Mapping was developed by Stephen M. Wilson, Melodie Yen and Dana Eriksson. If you use ALM, please cite:

Wilson SM, Yen M, Eriksson DK. An adaptive semantic matching paradigm for reliable and valid language mapping in individuals with aphasia. *Hum Brain Mapp* 2018; 39: 3285-307.

Thanks to Andrew DeMarco, Pagie Beeson and Jeff Binder for helpful discussions.

Fabi Hirsch, Pagie Beeson, Kindle Rising and Janet Hawley assisted with recruitment of individuals with aphasia who served as pilot participants.

Stefanie Lauderdale assisted with collection of pilot data.

Andrew DeMarco designed the pseudoword syllable matching task and generated an initial set of pseudowords.

Leslie Baxter applied the semantic matching paradigm in presurgical contexts and provided pilot data.

Paul Fillmore, Kaleb Vinehout and Jonas Kaplan helped by testing ALM on different setups.

ALM is built on Psychtoolbox-3, by Mario Kleiner, David Brainard, Denis Pelli, Chris Broussard, Tobias Wolf, Diederick Niehorster:
http://psychtoolbox.org

Semantic matching items are based on semantic relatedness data from the University of South Florida Free Association Norms:
http://w3.usf.edu/FreeAssociation

ALM uses the readwav function and some dependent functions from voicebox version 2016-12-16, Copyright (c) 1998-2014 Mike Brookes, downloaded here:
http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html
Equivalent MATLAB functions are available, but the voicebox function is faster.

Excerpts from the following two audiobooks are used in the narrative comprehension paradigms:

Who Was Albert Einstein? Audible – Unabridged
Jess Brallier (Author), Kevin Pariseau (Narrator), Penguin Group USA and Audible (Publisher)
https://www.amazon.com/Who-Was-Albert-Einstein/dp/B0021249SO

Who Were the Beatles? Audible – Unabridged
Geoff Edgers (Author), Kevin Pariseau (Narrator), Penguin Group USA and Audible (Publisher)
https://www.amazon.com/Who-Were-the-Beatles/dp/B002EDTVA6

Pictures in the picture naming paradigms are from the following paper:
Rossion B, Pourtois G. Revisiting Snodgrass and Vanderwart's object pictorial set: The role of surface detail in basic-level object recognition. Perception 2004; 33: 217–36.
http://journals.sagepub.com/doi/10.1068/p5117

11. License