Working Memory Updating Latency Reflects the Cost of Switching Between Maintenance and Updating Modes of Operation

Article *in* Journal of Experimental Psychology Learning Memory and Cognition · November 2013

DOI: 10.1037/a0035545

Yoav Kessler
Ben-Gurion University of the Negev

Klaus Oberauer
University of Zurich

Journal of Experimental Psychology:
Learning, Memory, and Cognition
2014, Vol. 40, No. 3, 738–754

0278-7393/14/$12.00 DOI: 10.1037/a0035545

Working Memory Updating Latency Reflects the Cost of Switching Between Maintenance and Updating Modes of Operation

Yoav Kessler
Ben-Gurion University of the Negev

Klaus Oberauer
University of Zurich

Updating and maintenance of information are 2 conflicting demands on working memory (WM). We examined the time required to update WM (updating latency) as a function of the sequence of updated and not-updated items within a list. Participants held a list of items in WM and updated a variable subset of them in each trial. Four experiments that vary the number of to-be-updated and to-be-maintained items, as well as their positions in the list, are reported. The pattern of latencies was best explained by a model assuming forward scanning of the list, updating modified items, and maintaining nonmodified items. Switching between updating and maintenance incurred a response time cost, which increased with overall set-size. The formation of new item-position associations accounted for an additional response time component. The finding of an update-switch cost provides novel behavioral support for a class of physiologically inspired computational models, in which updating and maintenance require 2 different states of WM.

Keywords: working memory, updating, maintenance, switching, gating

Working memory (WM) is a core cognitive mechanism that enables the short-term maintenance and mental manipulation of information. A well-functioning WM must balance two demands that are in perpetual conflict with each other. On the one side, WM must maintain relevant information, shielding it from interference from external (e.g., perceptual) or internal (e.g., long-term memory) input. On the other side, the content of WM must be rapidly updated when required, in order to keep up with changes in the outer world or as part of mental computations (Braver & Cohen, 2000; Cohen, Braver, & O'Reilly, 1996; D'Ardenne et al., 2012; Frank, Loughry, & O'Reilly, 2001; see also Goschke, 2000, for an analogous dilemma in goal-directed action control). These conflicting demands of stability and flexibility often arise at the same time. For example, mentally multiplying two 2-digit numbers requires solving substages of the problem and updating WM with their interim results, while keeping previous results intact for use in later stages. Control processes are therefore needed to resolve the stability–flexibility dilemma by coordinating the flow of information into WM and hence determining what will be updated and when.

Whereas much research has focused on the maintenance function of WM, relatively little is known about updating and the way in which it is controlled. Several researchers have argued that WM updating is not a unitary process but is rather composed of several subprocesses (Ecker, Lewandowsky, Oberauer, & Chee, 2010; Kessler & Meiran, 2008; Zhang, Verhaeghen, & Cerella, 2012) that may supplement each other or be recruited in different combinations by different task demands. For example, Ecker et al. (2010) decomposed updating into three components: retrieval, transformation, and substitution. Retrieval refers to the need to retrieve information as part of the updating operation when the new information must be computed from old information in WM (e.g., in mental arithmetic). Transformation refers to the mental manipulation of the representation as part of the updating process, and substitution is the actual change in the content of WM. Furthermore, substitution can in turn be decomposed into the removal of old, no-longer-relevant information from WM (Oberauer, 2001) and the encoding of new information to supplant it.

Kessler and Meiran (2008) have recently argued that substitution of information in WM is carried out by two complementary processes, namely local and global updating. Participants were required to keep track of a series of one to three items. Each trial involved either retaining the entire set unchanged (no update) or updating some or all of the items by replacing them with new stimuli. The participants were required to update their WM according to the information presented visually and then press a key to proceed, thus enabling measurement of response time (RT) for updating. Kessler and Meiran introduced two versions of the paradigm. In the *full-display paradigm*, all the items were pre-

This article was published Online First January 20, 2014.

Yoav Kessler, Department of Psychology and Zlotowski Center for Neuroscience, Ben-Gurion University of the Negev, Beer-Sheva, Israel; Klaus Oberauer, Department of Psychology–Cognitive Psychology, University of Zurich, Zurich, Switzerland.

This research has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under Grant Agreement PCIG09-GA-2011-293832, awarded to the first author, and from the Swiss National Science Foundation, under Grants 100014_126766 and 100014_143333, awarded to the second author.

Correspondence concerning this article should be addressed to either Yoav Kessler, Department of Psychology and Zlotowski Center for Neuroscience, Ben-Gurion University of the Negev, Beer-Sheva, Israel 84105, or Klaus Oberauer, Department of Psychology–Cognitive Psychology, University of Zurich, Binzmühlestrasse 14/22, 8050 Zürich, Switzerland. E-mail: kesslery@bgu.ac.il or k.oberauer@psychologie.uzh.ch

sented in each trial, namely both the updated and nonupdated items. For example, if the WM set was composed of the letters A–B–C and the middle letter was updated to E, then the entire new set A–E–C was presented on the screen. In contrast, the *partial-display paradigm* involved presenting only the updated items, while asterisks served for place holders. For example, in the above case, the stimulus *–E–* was presented.

The two paradigms resulted in different patterns of RTs for updating steps. With a full display, RT increased monotonically with the number of updated items. This suggests that the content of WM is not updated as a whole but on an item-to-item basis ("local updating"). In addition, the difference between the nonupdate and update conditions increased with WM set-size. This was taken as evidence for an additional, global updating process, which takes place after the item-level updating process is completed and presumably serves for binding the new WM set together (see also Kessler & Meiran, 2006, for a similar argument). The results were similar in the partial-display paradigm, except for one notable difference: updating times increased within each set-size N as more items were updated, but only until N – 1 items. Updating the entire set was faster than updating any subset of the items, creating a nonmonotonic relation between the number of updated items and RT. This latter finding was interpreted as evidence for global updating of the bindings integrating the memory set. Specifically, when a subset of items is to be updated in the partial-display paradigm, participants need to dismantle the bindings among the old items before they can selectively replace some of them while keeping others, and then they need to establish new bindings to integrate the new set. In contrast, when the whole set is updated, participants can simply replace the old set by a new set, without the need for unbinding the old set.

The Present Study

Previous analyses of WM updating have tacitly treated updating of items at all positions of the list as equivalent. However, serial order is an essential property of WM and as such is likely to also affect updating. The present study analyzes the updating process as a function of the serial positions of updated and of not-updated items in a list. This enables us to pin down more precisely what information is modified and in what order. As we show, the distribution of updated items across serial positions is a central factor in understanding the dynamics of updating.

The structure of this article is as follows. We begin by describing the general experimental method, followed by specific hypotheses regarding the subcomponents of updating. We use a regression approach to model the pattern of updating latencies across conditions. Each hypothesis is represented by a separate predictor that reflects the pattern of updating latencies predicted from that hypothesis. This approach enables evaluation of the explanatory power of each hypothesis for our results. Experiment 1 uses a full-display paradigm, in which all items, including the nonupdated ones, are presented in each trial. Experiment 2 employs a partial-display paradigm, in which only the updated items are presented and asterisks serve as placeholders for the nonupdated items. Identifying similar predictors that explain RT in both paradigms would provide converging evidence for their general contribution to updating latency. Experiments 3 and 4 test the emerging model by applying it to a wider range of experimental conditions, thereby further establishing its generality.

We begin by reporting two experiments. Experiment 1 used the full-display paradigm, and Experiment 2 used the partial-display paradigm of Kessler and Meiran (2008). Participants were required to keep track of four letters appearing within frames on the screen. In each trial, either the entire set or a subset was replaced by new letters, and the participants were required to update their WM accordingly and to press a key to proceed, enabling RT measurement. After a few trials, the participants needed to recall the last letter that appeared in each frame. Table 1 presents the updating conditions used in the present study. The conditions differ in the number of updated items and their serial positions. We also examined situations where subsets of items were shifted to new positions, such that item-to-item associations were maintained while all item-position associations had to be updated. These conditions help to distinguish the potential contributions of item-position associations and item-item associations, as explained below. We chose those conditions from the many possible patterns of updated and maintained items in four-item lists because we identified the chosen subset as most diagnostic for distinguishing between the hypotheses introduced below, while keeping the complexity of the design within bounds.

Theories of memory for serial order propose different kinds of associations between the items and between items and their context (see Lewandowsky & Farrell, 2008, for review). Our working hypothesis was that all the associations of an item, either to other items or to its context (spatial or temporal; see below), are updated whenever that item is modified. In other words, updating an item also entails updating the entire network of associations in which that item is involved. Using this logic we were able to model updating times in various situations as a function of the number and type of associations that need to be

Table 1
Conditions for Experiments 1 and 2

Condition	Description
RUUU	First item repeated, last three items updated
RRUU	First two items repeated, last two items updated
RRRU	First three items repeated, last item updated
UUUR	First three items updated, last item repeated
UURR	First two items updated, last two items repeated
URRR	First item updated, last three items repeated
RURU	First and third items repeated, second and fourth items updated
URUR	First and third items updated, second and fourth items repeated
URRU	First and fourth items updated, second and third items repeated
UR_1R_2U	First and fourth items updated, items 1 and 2 from the previous trial reappear in positions 2 and 3, respectively
UUR_2R_3	First and second items updated, items 2 and 3 from the previous trial reappear in positions 3 and 4, respectively
UUUU	All four items updated

Note. R = repeated; U = updated.

updated (see Hitch, Fastame, & Flude, 2005, for a similar approach to investigating which associations are retained in long-term learning).

We investigated four hypotheses. These hypotheses are not mutually exclusive but rather can be combined. Therefore, we represented each of them by a predictor for a regression model, such that the predictor reflects the pattern of updating RT that would be expected from the hypothesis (see Table 2 for predictor values). We evaluated the explanatory power of each predictor through its contribution to explaining variance in updating RT across conditions in the regression model. The linear combination of predictors in our regression approach implies the assumption that updating latencies can be modeled as additive combinations of the durations of individual processing steps, with each processing step captured by one predictor. This assumption is tested by the ability of the models to fit the data. If we had made the stronger assumption that individual processes are strictly sequential discrete stages, rather than partly overlapping or organized in cascade, we could have interpreted the regression weights of each model as estimates of the durations of processing stages specified in that model.

Item-Position Associations

According to the first hypothesis, lists or sets of items are represented in WM by binding each item to its position, namely its context. This idea is incorporated in many models of serial-order memory (Lewandowsky & Farrell, 2008). The position can be either the item's spatial position (i.e., the frame in which it appears on the screen), its temporal position within a sequence, or both. Updating some items in a set simply means to update the item-position associations in the positions that are changed. This leads to the prediction that updating RT is a function of the number of to-be-changed item-position bindings, which is equal to the number of updated items. Item-position associations were modeled using a predictor that indicated the number of such associations that had to be updated in each condition. For example, condition UURR involves updating two item-position associations, for the first and second items. The predictor value for this condition is therefore 2. In contrast, condition URRR requires only one item-position association to be updated, so the predictor has the value 1 for this condition. Therefore, the item-position predictor predicts RTs to be larger in condition UURR than in condition URRR, and the size of the predicted RT difference is reflected in the regression weight of the predictor. The regression weight can therefore be interpreted as the time cost of updating one item-position association.

Item-Item Associations

According to this hypothesis, lists or sets involve bindings between each item and each other item. The idea that each item in a memory set is bound to every other item underlies the notion of global updating in the theory proposed by Kessler and Meiran (2008). It entails the assumption that each updated item would require establishing new associations with all the other items in the set. This hypothesis leads to the prediction that updating RTs increase with the number of item-item associations that need to be changed. The item-item predictor values reflect the number of new item-item associations that had to be created in each condition. These associations are both between the new items and between each new item and the maintained old items. For example, condition UURR involves updating five item-item associations—those between the items in Positions 1–2, 1–3, 1–4, 2–3, and 2–4.

Chaining

A special case of item-item associations is the chaining hypothesis, according to which each item is bound to only its immediate successor in a list, rather than to all the other items. Chaining has been part of several earlier models of memory for serial order (e.g., Lewandowsky & Murdock, 1989). Chaining theories were criticized on many grounds (see Henson, 1998; Lashley, 1951), but we add the chaining hypothesis because it represents an alternative interpretation of the idea that list items are integrated into a whole that needs to be dismantled and re-created by a global-updating

Table 2
Observed RT and Predictor Values by Condition for Experiments 1 and 2

Condition	Mean RT (ms) Experiment 1 (full display)	Mean RT (ms) Experiment 2 (partial display)	Predictor value Item-position	Item-item	Chaining	Update-switch	Modified item-position[a]	Switch-back[a]
RUUU	2,650	3,084	3	6	3	1	3	0
RRUU	2,247	2,882	2	5	2	1	2	0
RRRU	2,138	2,778	1	3	1	1	1	0
UUUR	2,686	3,590	3	6	3	2	4	0
UURR	2,601	3,503	2	5	2	2	4	1
URRR	2,366	3,931	1	3	1	2	4	2
RURU	2,515	4,737	2	5	3	3	3	0
URUR	2,786	4,988	2	5	3	4	4	0
URRU	2,736	4,619	2	5	2	3	4	1
UR_1R_2U	2,792	2,907	4	5	2	1	4	0
UUR_2R_3	2,832	2,997	4	5	2	1	4	0
UUUU	2,730	2,972	4	6	3	1	4	0

Note. RT = response time; R = repeated; U = updated.
[a] The rationale behind these predictors is presented in the Results section of Experiment 1.

process (Kessler & Meiran, 2008).[1] Like the preceding hypothesis, the chaining hypothesis engenders the prediction that updating latency increases with the number of item-item associations to be updated, but here we count only associations between neighboring items in a list. Accordingly, for example, condition UURR involves updating two chaining associations: between Items 1–2 and 2–3.

We included two conditions in the experiment specifically designed to test the item-item association and chaining hypotheses: Conditions UR_1R_2U and UUR_2R_3 each shifted a pair of items in the list, thereby changing their item-position associations but maintaining their item-item association. If item-item associations (between neighboring items) need to be updated whenever a subset of a list is updated, these conditions should be updated faster than the condition UUUU. In contrast, if only item-position associations need to be updated, the two pair-shift conditions should not differ from UUUU, because all three conditions require updating four item-position associations.

Update-Switch

Whereas the first three hypotheses reflect assumptions about the representational structure to be updated, our fourth hypothesis expresses assumptions about the processing schedule of updating. We assumed that updating involves scanning the list from beginning to end. Forward scanning has been shown to be the preferred mode of operation in retrieval from lists (Farrell & Lelièvre, 2009; Lange, Cerella, & Verhaeghen, 2011; Nairne, Ceo, & Reysen, 2007) and therefore is the most plausible schedule for sequential updating of list items. Each trial begins in a maintenance mode. Moving along the list, some of the items should be updated while others need to be retained. Accordingly, WM needs to switch between its maintenance and updating functions, and each of these switch operations requires time, thereby incurring a switching cost. For simplicity, we assumed that switching from maintenance to update and vice versa require the same amount of time. The update-switch hypothesis therefore predicts that updating RTs increase with the number of switches between updating and retention when scanning across the list from beginning to end. The update-switch predictor values reflect the number of switches between updating and maintenance, or vice versa, when scanning through the list in forward order. For example, condition UURR involves two switches—from maintenance to updating the first item, and from updating the second item to maintaining the third. The predictor value for this condition is therefore 2. The switch cost is reflected in the regression weight of the update-switch predictor.

Experiment 1

Method

Participants. Twenty undergraduate students (19 female; age: $M = 22.90$, $SD = 0.89$) from Ben-Gurion University of the Negev participated in the experiment in exchange for course credit. All the participants reported normal or corrected-to-normal vision and no learning disabilities or neuropsychological dysfunction.

Stimuli and apparatus. The experiments were programmed in E-Prime 2.0 (Schneider, Eschman, & Zuccolotto, 2002) and run on personal computers with 17-in. monitors. The stimuli were uppercase consonants, excluding M and W. The letters were presented in 18-point Arial font within four frames that were positioned horizontally in the center of the screen. Each frame subtended a visual angle of approximately $1.43° \times 1.43°$, assuming a 60-cm viewing distance. The gap between the frames was approximately 0.67°. All the stimuli were presented in white on a black background. Items were not allowed to repeat in different positions within the same list.

Procedure. The experiment consisted of 65 runs of trials, preceded by five practice runs. Each run started with the presentation of four frames with letters inside (see Figure 1). The participants were instructed to memorize the letters and then press the space bar to continue. Then a sequence of – one to five trials began. The number of trials was determined randomly in each run to minimize strategic effects of anticipating the end of a run: Participants had to update in each trial because they could be tested for the final list after any trial. In each trial, letters appeared inside all the frames, some or all of which differed from the letters presented in the preceding trial. The participants were instructed to update their memory in each trial and to remember only the last letter that appeared in each of the frames. The participants had to press the space bar to continue to the next trial, and RTs were measured between display onset and response. The participants were instructed to be as fast and accurate as possible. Between each response and the next display there was an intertrial interval of 1,000 ms during which empty frames were presented. This was done to ensure that updating steps would not attract attention automatically merely due to an apparent visual change.

In each trial, one of the 12 update conditions was randomly selected and applied. Table 1 describes the conditions. Each condition was administered in 10% of the trials except for RURU, URUR, UR_1R_2U, and UUR_2R_3, which were administered in 5% of the trials.[2] A run ended with four recall screens that required the participants to enter the last letter that appeared in each of the frames, in forward order.

Results and Discussion

Recall at the end of a run was correct in 91% of the runs. RT was analyzed only for trials that came from runs in which the final recall was correct. In addition, RTs longer than 10 s were removed from the analysis (0.7% of the data). Mean RTs for the remaining trials were submitted to a one-way analysis of variance (ANOVA) with condition as a within-subject independent variable, confirming that there were significant differences among the conditions, $F(11, 209) = 11.25$, $MSE = 91{,}985.42$, $\eta_p^2 = .37$ (see Figure 2a)

Table 3 presents the correlation between the observed RT and each of the predictors, across the 12 conditions. The correlation pattern confirms that RTs covary with the predictor values of item-position and item-item. Multilevel linear regression (also known as linear mixed-effects modeling or hierarchical linear

[1] Item–item association and chaining form two extreme poles of a continuum of models in which interitem associations vary in strength as a function of interitem distance.

[2] When planning the experiment, we regarded RURU and URUR as two instances of the same condition (alternating U and R) and UR_1R_2U and UUR_2R_3 as two instances of another condition (shifting an adjacent item pair); therefore each of those conditions was presented in only 5% of trials.

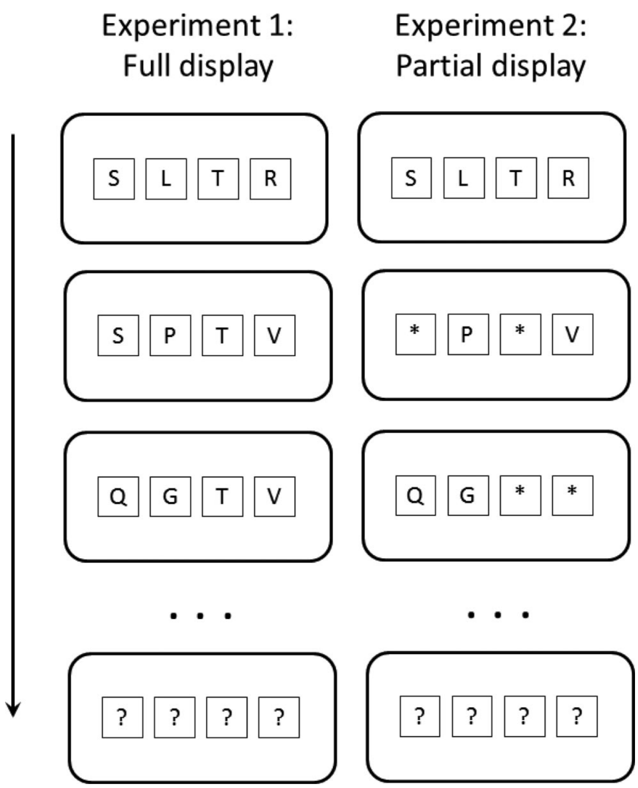

Figure 1. Schematic description of the paradigm. Each step was followed by a 1,000-ms intertrial interval, in which only the empty frames were presented on the screen.

modeling; Pinheiro & Bates, 2000) was used as the main analysis framework. The advantage of this approach is to allow the regression coefficients to vary between subjects, thus controlling for between-subjects variability in the size of within-subject effects (cf. Lorch & Myers, 1990). The models were implemented using the *lme* function of the *nlme* package in the R programming language (Pinheiro, Bates, DebRoy, Sarkar, and the R Development Core Team, 2012). Model comparison and selection were based on the Bayesian information criterion (BIC) statistic. The BIC corrects the $-2 \cdot \log(\text{likelihood})$ lack-of-fit statistic by adding a penalty for the number of free parameters, thereby rewarding parsimony. Fifteen models were tested, corresponding to all possible subsets of the four predictors and the full set. Table 4 presents model fits for these models.

The best fitting model explained updating RTs in the full-display paradigm by update-switch and item-position (see Table 5 for parameter estimates). To ensure that the BIC difference between the best and second-best fitting model was substantial, we transformed the BIC difference (ΔBIC) between these two models into a Bayes factor (BF), using the formula BF $=$ exp (ΔBIC/2). BF greater than 10 is considered as strong evidence supporting the better fitting model (Jeffreys, 1961). In the present experiment, the data provided strong evidence for the best fitting model over the second-best model (i.e., the one including update-switch, item-position, and item-item as predictors; ΔBIC $= 7.58$, BF $= 44.26$).

The best fitting model suggests the following picture of the updating process: People scan through the list from beginning to end and decide at each position whether the current item needs to be replaced or not.[3] Only those memory items that do not match the new array of stimuli are replaced by the corresponding stimulus. Thus, control processes need to switch between updating and not updating, and this switch costs time, reflected in the update-switch predictor. In addition, each time an item needs to be replaced by a new stimulus, removing the old item-position binding and establishing a new item-position binding takes time, and this time is reflected by the item-position predictor.

This scenario raises the question whether the described processing schedule is optimal. Specifically, repeatedly switching between updating and not updating within a scan might take more time than simply updating the entire list. This is because the cost of update-switching is longer than that of establishing a new item-position association. A compromise strategy would be to start the scanning procedure in a maintenance mode until the first new item is encountered and then to switch to update mode only once without switching back, such that all items scanned after the first switch are updated, regardless of whether they are old or new. In this way the WM system could save the time of unnecessarily substituting unchanged item-position bindings at the beginning of the list (whenever the first list item or items remain unchanged) and at the same time minimize the cost of update-switching. However, the advantage of strictly remaining in updating mode is mitigated as more nonupdate items appear after the first new item; when several nonupdate items follow, it might be advantageous to switch-back into maintenance mode. We therefore considered a flexible variant of the new processing schedule where, after the first switch into updating mode, the system switches back into maintenance mode with a certain probability every time it encounters an old (unchanged) item in the stimulus display.

To examine this idea, we created two new predictors. *Modified item-position* is the number of items that are updated if the system switches into updating mode at the first updated position and strictly remains in updating mode for the remaining list. The regression weight of this predictor therefore reflects the cost of switching into updating mode once, plus the updating cost per item, assuming that every list item following the first updated item is updated, regardless of whether it is old or new. The second new predictor, *switch-back*, represents the possibility that people switch-back to maintenance model after the first updated item. In combination with modified item-position, the switch-back predictor reflects the potential reduction of the updating cost by switching back into maintenance mode after the first updated item. A reduction in RTs is expected from saving the time for updating all following old items. The value of the switch-back predictor is the number of further old items after the first old item that follows the first new item. Because it predicts an RT reduction relative to the prediction by modified item-position, switch-back should re-

[3] The forward-scan model assumes that scanning is done from left to right, compatible with the reading direction of Latin letters. Although our participants were native Hebrew speakers, they were highly experienced in left-to-right reading when English words or numbers are presented. Moreover, the fact that the recall phase at the end of each trial was carried out in a sequential left-to-right order encouraged using this reading direction. Finally, assuming a right-to-left scanning direction does not fit the data, since it affects the update–switch predictor values. For example, this would predict longer RTs for RURU than URUR, which is the opposite of the observed finding.

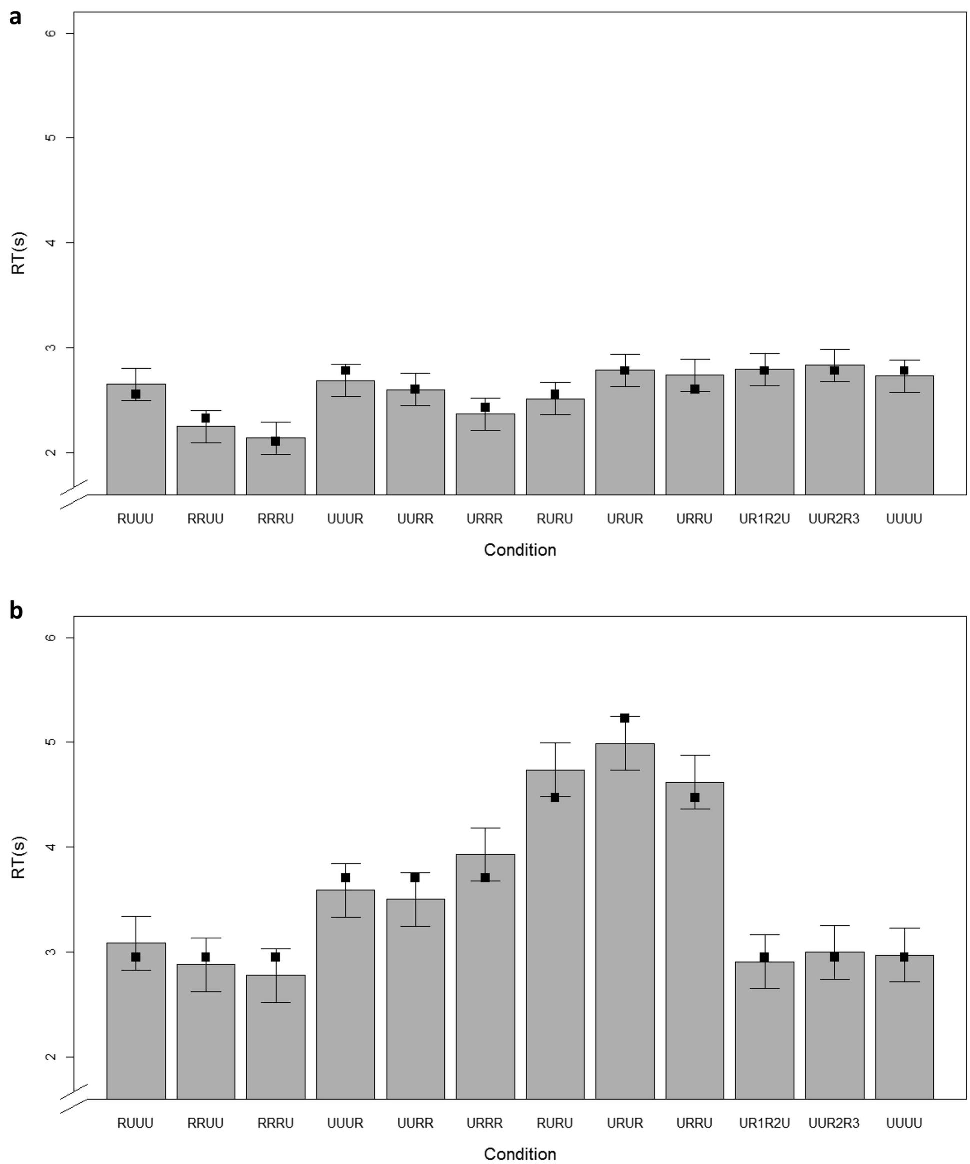

Figure 2. Observed and predicted mean response time (RT) by condition. Bars represent observed RT. R = repeated; U = updated. Panel a: Experiment 1. Solid squares indicate flexible model variant prediction. Panel b: Experiment 2. Solid squares indicate best fitting model prediction. Error bars represent 95% within-subject confidence intervals (Masson & Loftus, 2003).

ceive a negative regression weight. A higher probability of switching back into maintenance mode implies larger time savings and therefore a larger (absolute) regression weight. A model with the above two predictors fit the data better than the original update-switch and item-position model (ΔBIC = 5.09, BF = 12.77). Also, it fit better than a model with modified item-position only (ΔBIC = 17.12, BF = 5,224.95; see Table 4 and Figure 2a). Table 5 provides the predictor estimates for this model.

The switch-back predictor accounts for the general benefit in switching back to maintenance mode, defined as the time difference between the duration of switching back (and not updating) and of updating (but not switching back), multiplied with the probability of switching back. Unfortunately, we cannot identify separate parameters for the three hypothetical components, the time costs of switching and of updating and the probability of switching back. Therefore, whereas other model parameters can be interpreted as the estimated duration of specific processes, the switch-back parameter estimate cannot be interpreted in that way, because it reflects the gain in time from a process that occurs only probabilistically, with an unknown probability. Separately estimating the components of the switch-back parameter would require separate estimates of the switch cost and the updating cost, which are wrapped into one in the modified item-position predictor. In principle, those two components could be separated as an intercept (for the time for switching into updating mode once) and a slope (for the number of updating steps), but estimating these two

Table 3
Correlations Among Observed RT and Predictor Values for Experiments 1 and 2

Predictor	Observed RT	
	Experiment 1	Experiment 2
Item-position	.74**	−.39
Item-item	.66*	−.02
Chaining	.58*	.29
Update-switch	.24	.98***
Modified item-position	.82**	.36
Switch-back	−.13	.25

Note. RT = response time.
* $p < .05$. ** $p < .01$. *** $p < .001$.

parameters separately would require a condition without any switch (such as RRRR) to estimate the intercept and a large range of numbers of updated items (to estimate the slope), both of which were not available in the present experiment.

We conclude that when people need to update all or part of a memory list according to a fully presented new list, they scan through the list in forward order, starting in maintenance mode. Upon encountering a new item in the display that needs to replace an item in memory, they switch into updating mode and substitute the old item's binding to its position by a binding of the new item to that position. From there on they tend to substitute all further items by the stimuli in the display but switch-back into maintenance mode probabilistically after encountering a stimulus that matches the item in memory.

Experiment 2

The full-display paradigm used in Experiment 1 potentially confounds two processes: detecting which items were updated, and the actual modification of those items. Because the participant does not know in advance which items within the set are to be updated, forward scanning could be necessary for detecting these items rather than for the updating process per se. To disentangle detection and updating, Experiment 2 used a partial-display paradigm, in which only the updated (new) items were presented in each trial (see Kessler & Meiran, 2008, for a similar logic). Here, in contrast to Experiment 1, participants do not need to scan the list in order to identify which items were updated. Any evidence for forward scanning in this paradigm would be solely attributed to the updating requirement.

Method

Participants. Twenty undergraduate students (16 female; age: $M = 23.00$, $SD = 1.05$) from Ben-Gurion University of the Negev who did not participate in Experiment 1 participated in the experiment in exchange for course credit. All the participants reported normal or corrected-to-normal vision and no learning disabilities or neuropsychological dysfunction.

Procedure. The procedure was identical to that for Experiment 1 with one exception. In each trial only the updated letters were presented inside the frames. An asterisk appeared inside the nonupdated frames.

Results

Recall was correct in 84% of the runs. RT was analyzed for only trials that came from runs in which the final recall was correct. In addition, RTs longer than 10 s were removed from the analysis (1.7% of the data). An ANOVA on mean RTs yielded a significant effect of condition, $F(11, 209) = 49.91$, $MSE = 257{,}577.37$, $\eta_p^2 = .72$ (see Figure 2b).

Table 3 presents the correlation between the observed RT and each of the predictors across the 12 conditions. Only update-switch

Table 4
Model Fit for Experiments 1 and 2

Predictor	Experiment 1 (full display)			Experiment 2 (partial display)		
	AIC[a]	BIC[a]	Log likelihood	AIC[a]	BIC[a]	Log likelihood
ItPos	260.34	277.74	−125.17	693.74	711.14	−341.87
ItIt	268.86	286.27	−129.43	719.90	737.30	−354.95
Chain	280.92	298.33	−135.46	706.32	723.73	−348.16
UpSw	305.54	322.94	−147.77	**427.55**	**444.95**	**−208.77**
ItPos + ItIt	258.07	282.44	−122.04	671.61	695.98	−328.81
ItPos + Chain	257.98	282.35	−121.99	632.82	657.19	−309.41
ItPos + UpSw	**224.37**	**248.74**	**−105.19**	428.82	453.19	−207.41
ItIt + Chain	272.86	297.23	−129.43	646.18	670.55	−316.09
ItIt + UpSw	266.56	290.92	−126.28	427.34	451.70	−206.67
Chain + UpSw	284.44	308.80	−135.22	428.68	453.04	−207.34
ItPos + ItIt + Chain	259.56	290.88	−120.78	625.33	656.66	−303.67
ItPos + ItIt + UpSw	224.99	256.32	−103.50	430.97	462.30	−206.49
ItPos + Chain + UpSw	225.15	256.48	−103.58	432.11	463.43	−207.05
ItIt + Chain + UpSw	266.02	297.35	−124.01	430.97	462.30	−206.49
ItPos + ItIt + Chain + UpSw	224.20	262.48	−101.10	434.59	472.88	−206.30
ModItPos	243.36	260.77	−116.68	698.36	715.76	−344.18
SwBack	307.61	325.02	−148.81	706.34	723.75	−348.17
ModItPos + SwBack	**219.28**	**243.65**	**−102.64**	696.56	720.93	−341.28

Note. Data for best fitting models are indicated in bold. AIC = Akaike information criterion; BIC = Bayesian information criterion; ItPos = item-position; ItIt = item-item; Chain = chaining; UpSw = update-switch; ModItPos = modified item-position; SwBack = switch-back.
[a] Smaller AIC and BIC values indicate better fit.

Table 5
Parameter Estimates for Best Fitting Models in Experiments 1 and 2

Experiment/model	Predictor	Estimate (s)	*SE*	*df*	*t*
		Experiment 1			
Full model	Intercept	1.64	0.25	216	6.48***
	item-position	0.22	0.03	216	7.12***
	Item-item	0.08	0.06	216	1.36
	Chaining	−0.16	0.07	216	−2.18*
	Update-switch	0.18	0.03	216	6.20***
Best fitting model in initial set of models	Intercept	1.81	0.24	218	7.52***
	Update-switch	0.15	0.02	218	6.01***
	Item-position	0.21	0.02	218	8.50***
Flexible model variant	Intercept	1.88	0.24	218	7.86***
	modified item-position	0.23	0.03	218	8.12***
	switch-back	−0.18	0.04	218	−4.92***
		Experiment 2			
Full model	Intercept	2.32	0.30	216	7.78***
	Item-position	0.02	0.05	216	0.37
	Item-item	−0.06	0.09	216	−0.71
	chaining	0.07	0.11	216	0.61
	Update-switch	0.75	0.07	216	10.55***
Best fitting model	Intercept	2.19	0.22	219	9.78***
	Update-switch	0.76	0.06	219	11.95***

* $p < .05$. *** $p < .001$.

correlated significantly with RT ($r = .98$). Fifteen models, corresponding to all possible subsets of four predictors, as well as the full set, were tested. The data were best fit by a model including update-switch as the only predictor (see Tables 4 and 5 for model fit and predictor estimate). The data provide strong evidence for the best fitting model over the second-best model, the one including update-switch and item-item as predictors (ΔBIC = 6.75, BF = 29.24).

According to the best fitting model, processing the list starts in maintenance mode, because the maintenance mode is needed for maintaining the items from the previous trial. Participants switch from maintenance to update and vice versa while serially progressing in the list. In contrast to the full-display paradigm, with partial display, switching back to maintenance is required to keep non-updated items in memory, because those items are not visually available. This reasoning is supported by testing models with modified item-position and switch-back as predictors, which clearly fit the data worse than the update-switch model did (see Table 4). Thus, our modeling provided one explanation for the difference in RT patterns with full display and with partial display: Whereas with full display participants can afford to update items that do not change, thereby partially saving the costs of switching, with partial display this strategy is not feasible.

In contrast to Experiment 1, the number of item-position bindings that needed to be substituted did not seem to play a role in the partial-display paradigm of Experiment 2. Adding item-position as a predictor resulted in a poorer model fit compared to a model with update-switch alone. This seems to be at odds with the conclusion from Experiment 1 that updating item-position bindings costs time. One potential reason for this discrepancy is that the contribution of item-position was not detected in Experiment 2 due to lower statistical power than in Experiment 1, because RTs were slower and more variable in Experiment 2. Another possible reason is that in the partial-display paradigm the entire list needs to be reconstructed in each trial. This is done by substituting item-position bindings for the updated items *and* retrieving the nonupdated items missing in the display, perhaps to refresh them in memory (Raye, Johnson, Mitchell, Greene, & Johnson, 2007). Because the number of updated items and the number of nonupdated items add to a constant, the effect of item-position binding will not be detected if the duration of updating an item-position binding is similar to the duration of retrieving an item.

Discussion

As in the full-display paradigm, the pattern of updating latencies in the partial-display paradigm is explained by forward scanning of the list and the cost of switching between updating and maintenance. This implies that scanning is required not only for detecting which items need to be updated. People also scan the list even in a situation where the updated items could potentially be accessed directly, without going through the entire set. A model assuming direct access to the updated items would predict that updating latencies reflect primarily the number of to-be-updated items, regardless of their serial positions, as incorporated in the item-position predictor. A model with item-position as a single predictor failed to fit the present experiment, and this fact speaks strongly against a direct-access model. As suggested above, the reason for forward scanning in the partial-display situation might be the need to refresh the repeated items. If that is the case, participants need to access every list item regardless of whether it is to be updated or not, and access to list items is easiest in forward order (Lange et al., 2011).

Notably, the estimate of update-switch differs greatly in the full- and partial-display paradigms, being .15 s and .76 s, respectively. The comparison of these estimates adds further evidence for our conclusion from Experiment 1. The estimate from the full-display paradigm used in Experiment 1 (.15 s) is from the best fitting model among the standard models, the item-position plus update-switch model. The flexible model variant, which provided a better fit to the full-display data, implies that the participants did not switch from maintenance to updating and vice versa in all trials but rather tended to continue in updating mode in some of the trials. This implies that people switch less often, on average, than assumed in the item-position plus update-switch model. Hence, if the flexible model is correct, the estimate from the standard model is an underestimation of the real switch cost. Because it is plausible that the switch cost is approximately the same for the partial-display and the full-display paradigms, it is plausible to assume that the .15 estimate from Experiment 1 is an underestimation, as implied by the flexible model.

Accounting for Kessler and Meiran's (2008) finding. Kessler and Meiran's (2008) finding of slower RTs for updating part of the set, compared to updating the entire set, was interpreted as evidence for binding among WM items that require global updating. According to that account, updating part of the set requires unbinding, modification, and rebinding, which together are slower than replacing the entire set. The results of Experiment 2 show that, although partial-set updating is slower than updating the entire set in *some* of the conditions (UUUR, UURR, URRR, RURU, URUR, URRU), this is not the case in others (RUUU, RRUU, RRRU) and therefore is not a general rule.

The update-switch model suggests a different explanation for the finding of Kessler and Meiran (2008): Updating the entire set is done by switching from maintenance to update mode only once, at the beginning of a list, whereas updating a subset of the items requires more switches between updating and not updating, on average, across all conditions of updating a subset (see Table 1). To test this account of Kessler and Meiran's results, we reanalyzed the data of their partial-display experiment (Experiment 3). Instead of analyzing mean RT as a function of the number of updated items, as in the original article, we broke the data into different sequences of updated and maintained items. Because the update-switch model does not currently account for set-size effects (but see Experiments 3 and 4 below for extension), we limited the reanalysis to a set-size three only (see Figure 3).

The main effect of condition (RRR through UUU) was significant, $F(7, 77) = 28.82$, $MSE = 93{,}373.68$, $\eta_p^2 = .72$. RTs for conditions URR, RUR, and RRU, which share the same number of updated items, differed significantly, $F(2, 22) = 8.00$, $MSE = 36{,}870.28$, $\eta_p^2 = .42$. So did RTs for conditions UUR, URU, and RUU, $F(2, 22) = 12.14$, $MSE = 89{,}699.57$, $\eta_p^2 = .52$. Therefore, disaggregating those conditions reveals a systematic source of variance that was occluded in the more aggregated analysis of Kessler and Meiran (2008). This source of variance is the number of update switches: A model with a single update-switch predictor (log likelihood = −39.95, AIC = 89.89, BIC = 102.71) fit the data better than did a model with both update-switch and item-position (log likelihood = −38.87, AIC = 91.74, BIC = 109.69, ΔBIC = 6.98, BF = 32.79). A model with only item-position fit the data worse than did both models (log likelihood = −80.25, AIC = 170.50, BIC = 183.32), as can be expected given the differences among conditions that share the same number of updated items. The novel explanation emerging from our model is more powerful than the global updating hypothesis, as it explains not only the difference between partial set and whole set updating but also the difference among the various partial-set-updating conditions. Moreover, the global updating account is based on dismantling and re-creating item-item associations, but no evidence for the latter was found in the present experiments:

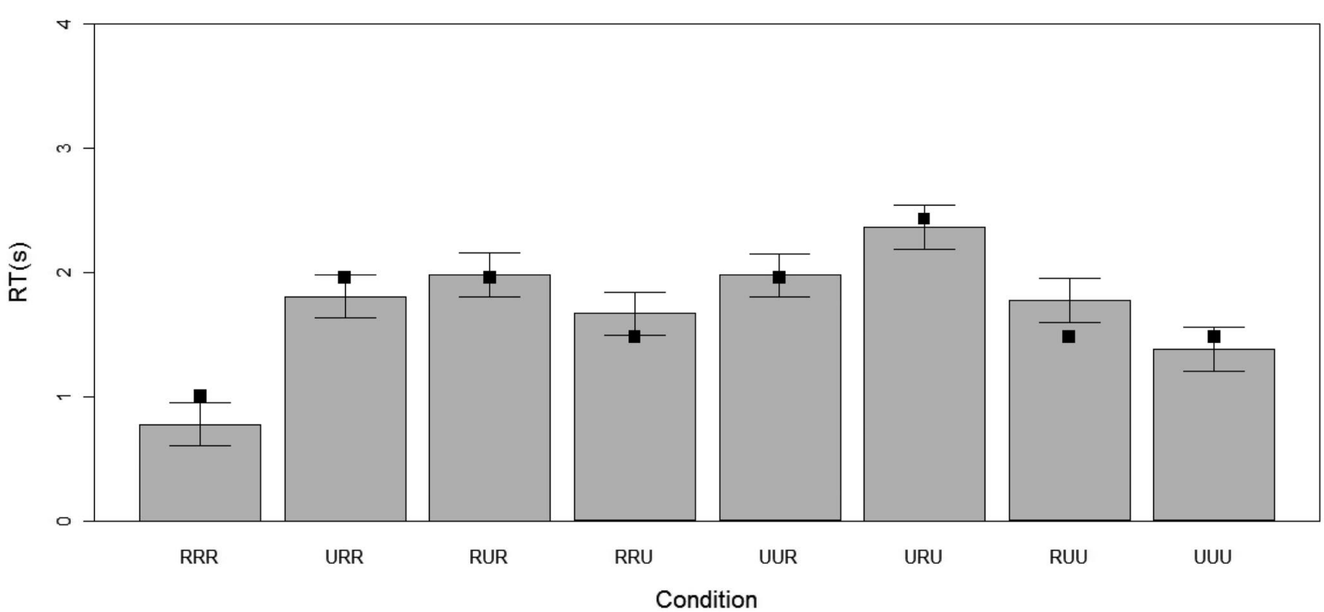

Figure 3. Reanalysis of Kessler and Meiran's (2008) Experiment 3 data, with set-size equal to three items. Bars represent observed mean response times (RTs); error bars represent 95% within-subject confidence intervals (Masson & Loftus, 2003). Model prediction is indicated by solid squares. R = repeated; U = updated.

Models including the item-item association or the chaining predictors fared worse than did the update-switch model.

Experiment 3

Experiments 3 and 4, using the full-display and partial-display paradigm, respectively, were carried out in order to validate and extend our findings. We manipulated the number of updated and maintained items separately. As a result, the total number of items that needed to be held in WM, that is, the memory set-size, was varied between two and four items. Set-size manipulations played an important role in previous analyses of WM updating latencies. In particular, Kessler and Meiran (2006, 2008) interpreted set-size effects as evidence for global updating (see also Badre, 2012, for a discussion of global vs. selective updating). According to this hypothesis, the entire content of WM is updated as a whole, including both maintained and updated items. The duration of this process increases with the set-size and occurs only whenever any of the items is updated (but not when all the items need to be maintained). Therefore, the interaction of updating (vs. pure maintenance) with set-size was interpreted as evidence for global updating.

However, Experiments 1 and 2 did not support the global updating hypothesis, for two reasons. First, both item-item and chaining predictors failed to account for the data. These predictors are two plausible instantiations of the idea that global updating entails creating new associations among the items in WM. Second, the advantage of updating the entire set compared to a subset of the items in the partial-display paradigm was fully explained by the update-switch process. Accordingly, no evidence was found for a bound set representation and therefore for a global updating process.

In order to account for set-size effects, the model that emerged from Experiments 1 and 2 also suggests an alternative explanation of the set-size effect on updating latencies. In contrast to the global updating hypothesis (Kessler & Meiran, 2008), our explanation of the set-size effect on the updating cost does not involve the assumption that updating of larger sets takes more time because larger sets require disassembling and reassembling a larger memory set. Rather, our model explains the set-size effect through two generic effects, the linear increase of the number of processing steps and the proportional increase by slowing of individual processing steps, as follows. First, according to this model, participants scan the entire set in all conditions and decide whether to update each of the items. Larger sets require a larger number of scanning steps. Therefore, scanning time (and the associated decision process for each item) increases linearly with set-size, regardless of the number of updated items. The mere effect of scanning time is manifested in RT differences between no-update conditions that differ in set-size (i.e., R, RR, RRR; see Kessler & Meiran, 2008). Second, in addition, we expected that with increasing set-size, each individual processing step would be slowed down. It is well established that processes involving access to items in WM are slowed as set-size increases (Oberauer, Demmrich, Mayr, & Kliegl, 2001; Oberauer & Göthe, 2006). There are several possible explanations for this slowing, among them the competition for retrieval between items in the memory set and the degradation of item representations through interference; in the present context we did not aim to adjudicate between those explanations. Regardless of why larger set-sizes slow down processes on WM items, any such slowing affects RTs not in a linear but in a multiplicative fashion. For instance, if increasing set-size from two to four slows down all processes by Factor 2, then RTs in all conditions with set-size four will be twice as long as the corresponding conditions with set-size two (Myerson, Adams, Hale, & Jenkins, 2003). This multiplicative effect of proportional slowing is captured by an interaction of set-size with other predictors.

Method

Participants. Twenty undergraduate students (16 female; age: $M = 23.05$, $SD = 1.53$) from Ben-Gurion University of the Negev who did not participate in the previous experiments participated in the experiment in exchange for course credit. All the participants reported normal or corrected-to-normal vision and no learning disabilities or neuropsychological dysfunction.

Procedure. We used a full-display procedure similar to that of Experiment 1. The numbers of updated and of maintained items within a set were manipulated independently, each between zero and four items. This was achieved by allowing the set-size to vary from one run to another. Within each run, the numbers of updated and of maintained items were manipulated, but the set-size was kept constant. In each trial, the number of frames presented on the screen was 2, 3, or 4, depending on the set-size. The total number of runs was 30, 45, and 60 for set-sizes 2, 3, and 4, respectively, plus three runs of each set-size that were given as practice in the beginning of the experimental session. Each run included an unpredictable number of updating trials ranging from one to five. The number of update-switches was manipulated (0, 1, or 2) between trials. In total, 18 conditions were examined and were administered with equal probabilities throughout the experiment (see Table 6 for the full list of conditions).

Results

Recall was correct in 99%, 92%, and 93% of the runs for set-sizes 2, 3, and 4, respectively. The main effect of set-size on recall rate was significant, $F(2, 38) = 17.90$, $MSE = 0.0017$, $\eta_p^2 = .49$, reflecting an advantage for two items over three and four, $F(1, 19) = 54.64$, $MSE = 0.0011$, $\eta_p^2 = .74$. The latter conditions did not differ significantly ($F < 1$).

RT was analyzed only for trials that came from runs in which the final recall was correct. In addition, a single trial with RT longer than 10 s was removed from the analysis. An ANOVA on mean RTs yielded a significant effect of condition, $F(17, 323) = 101.40$, $MSE = 70{,}229.23$, $\eta_p^2 = .84$ (see Figure 4a).

Extending the update-switching model. In order to extend the best fit model of Experiment 1 to account for set-size effects, we added a set-size predictor (2, 3, or 4 items) and examined its main effect and its interaction with each of the other predictors. A set-size effect on overall updating RTs can arise through an effect on the scanning time, which was reflected in the intercept of the model for Experiment 1, and through an effect on additional processes captured by the model predictors, updating and update-switching. In both cases, the effect can be a linear increase of RT arising from an increased number of steps, as well as a proportional effect from slowing of individual processing steps. Any effect of set-size on scanning time—whether linear or proportion-

Table 6
Observed RT and Predictor Values by Condition for Experiments 3 and 4

Condition	Mean RT (ms): Experiment 3 (full display)	Mean RT (ms): Experiment 4 (partial display)	Predictor value: Item-position	Predictor value: Update-switch	Predictor value: Set-size	Predictor value: Modified item-position	Predictor value: Switch-back
RR	632	798	0	0	2	0	0
RU	763	1,235	1	1	2	1	0
UR	826	1,277	1	2	2	2	0
UU	880	1,040	2	1	2	2	0
RRR	881	969	0	0	3	0	0
RRU	1,089	1,640	1	1	3	1	0
URR	1,355	2,122	1	2	3	3	1
RUU	1,372	1,845	2	1	3	2	0
UUR	1,565	2,091	2	2	3	3	0
UUU	1,557	1,665	3	1	3	3	0
RRRR	1,289	1,230	0	0	4	0	0
RRRU	1,621	2,588	1	1	4	1	0
URRR	2,032	3,290	1	2	4	4	2
RRUU	1,874	2,543	2	1	4	2	0
UURR	2,199	2,881	2	2	4	4	1
RUUU	2,355	2,845	3	1	4	3	0
UUUR	2,420	3,146	3	2	4	4	0
UUUU	2,378	2,794	4	1	4	4	0

Note. RT = response time; R = repeated; U = updated.

al—translates into a main effect of set-size as predictor in the model. The linear effect of set-size arising from an increased number of updating and update-switch processes is captured by those predictors directly, because they reflect the number of each of those processing steps. The proportional slowing of updating and update-switching by an increase of set-size translates into two-way interactions of set-size with the update and with update-switch, respectively. Our aim in this model was not to *explain* why a larger set-size slows down cognitive operations on WM contents. Rather, our goal was to demonstrate that this slow-down is sufficient to account for set-size effects on updating latencies, without assuming additional processing steps such as global updating.

To examine whether set-size effects are linear as suggested by our model, we conducted an ANOVA on conditions RR, RRR, and RRRR with set-size as an independent variable. The main effect was significant, $F(2, 38) = 97.95$, $MSE = 22{,}477.68$, $\eta_p^2 = .84$. The linear trend contrast among these conditions was significant, $F(1, 19) = 112.01$, $MSE = 38{,}570.03$, $\eta_p^2 = .85$, and so was the quadratic trend, $F(1, 19) = 13.00$, $MSE = 6{,}385.33$, $\eta_p^2 = .41$. Whereas the linear contrast explained 98% of the variance among the conditions, the quadratic trend explained only the remaining 2%. Accordingly, we conclude that the nonlinear RT component of set-size effects is small and negligible and a linear model is adequate.

Based on the results of Experiment 1 and the above analyses, we compared two general models:

$$(1)\ \text{update-switch} + \text{item-position} + \text{set-size} + (\text{update-switch}*\text{set-size}) + (\text{item-position}*\text{set-size})$$

and

$$(2)\ \text{modified item-position} + \text{switch-back} + \text{set-size} + (\text{modified item-position}*\text{set-size}) + (\text{switch-back}*\text{set-size}).$$

Table 7 presents the correlation between the observed RT and the predictors of these models across the 18 conditions. Model 2 fit the data better, in accordance with the modeling result of Experiment 1 (Model 1: log likelihood = 15.12, AIC = −4.24, BIC = 46.28; Model 2: log likelihood = 22.07, AIC = −18.14, BIC = 32.38; ΔBIC = 13.90, BF = 1,043.15). We continued by improving this model by examining all its nested models that include set-size, or an interaction with set-size, as predictors. The best fitting model included four parameters modified item-position, switch-back, set-size, and modified item-position*set-size (ΔBIC = 11.73, BF = 352.24, compared to the best fit model). See Figure 4b and Table 8 for model fit, and see Table 9 for parameter estimates. We withhold the discussion of these results until after Experiment 4.

Experiment 4

Method

Participants. Twenty undergraduate students (14 female; age: $M = 23.15$, $SD = 1.06$) from Ben-Gurion University of the Negev who did not participate in the previous experiments participated in the experiment in exchange for course credit. All the participants reported normal or corrected-to-normal vision and no learning disabilities or neuropsychological dysfunction.

Procedure. The procedure was identical to that for Experiment 3 with one exception. In each trial only the updated letters were presented inside the frames. An asterisk appeared inside the nonupdated frames.

Results

Recall was correct in 97%, 84%, and 84% of the runs for set-sizes 2, 3, and 4, respectively. The main effect of set-size on recall rate was significant, $F(2, 38) = 50.81$, $MSE = 0.0024$, $\eta_p^2 =$

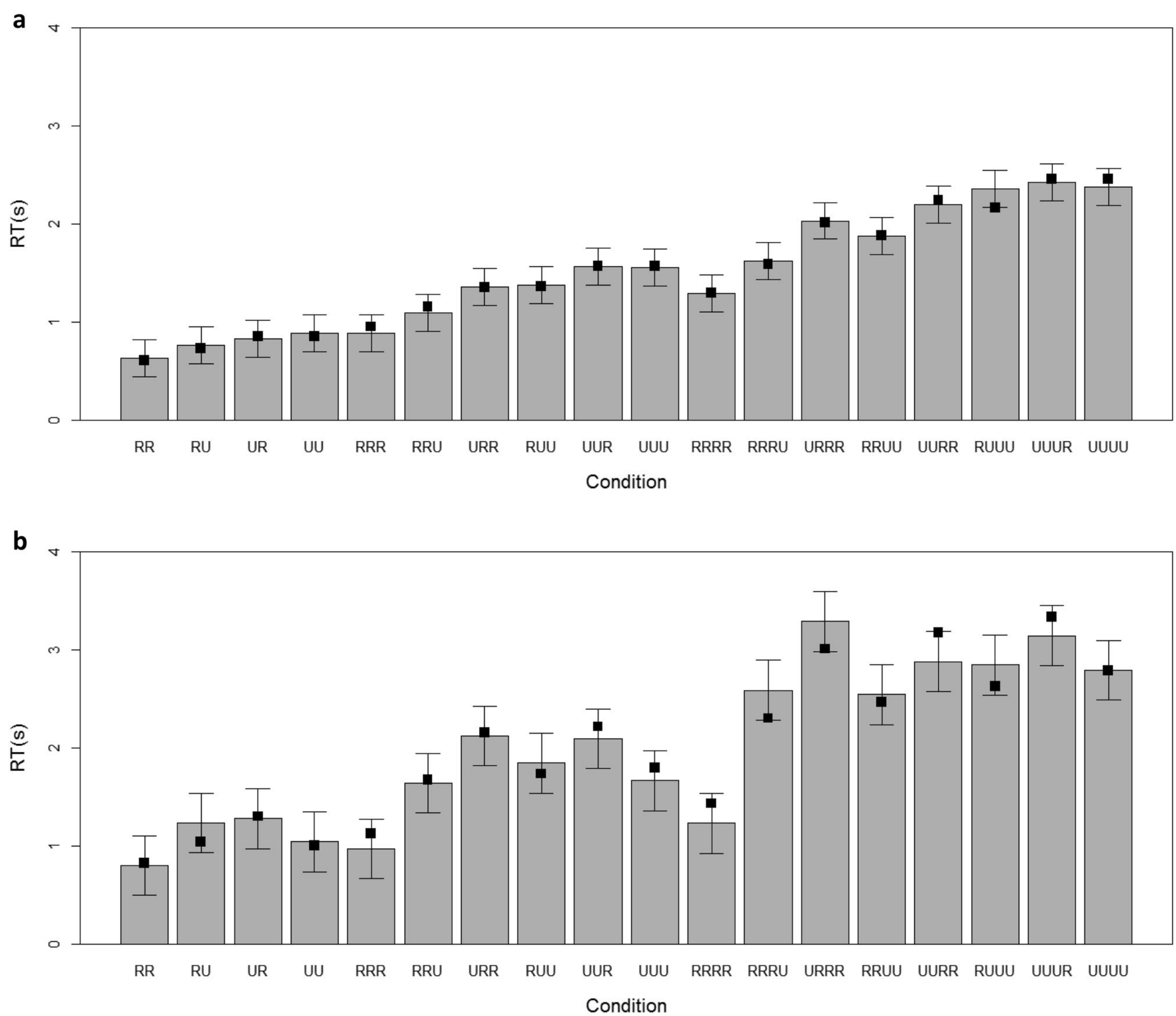

Figure 4. Observed and predicted mean response times (RTs) by condition. R = repeated; U = updated. Panel a: Experiment 3. Panel b: Experiment 4. Bars represent observed RTs; solid squares indicate best fitting model prediction. Error bars represent 95% within-subject confidence intervals (Masson & Loftus, 2003).

.73, reflecting an advantage for two items over three and four, $F(1, 19) = 92.53$, $MSE = 0.0026$, $\eta_p^2 = .83$. The latter conditions did not differ significantly ($F < 1$).

RT was analyzed only for trials that came from runs in which the final recall was correct. In addition, RTs longer than 10 s were removed from the analysis (0.2% of the data). An ANOVA on mean RTs yielded a significant effect of condition, $F(17, 323) = 71.67$, $MSE = 183{,}114.46$, $\eta_p^2 = .79$ (see Figure 4b).

We continued by extending the best fitting model of Experiment 2 to include set-size effects. As in the analysis of Experiment 3, we added a set-size predictor and its two-way interactions with the other predictors. To test for the linearity of set-size effect, an ANOVA was conducted on conditions RR, RRR, and RRRR, with set-size as an independent variable. The main effect was significant, $F(2, 38) = 25.44$, $MSE = 37{,}320.20$, $\eta_p^2 = .57$. Only the linear trend contrast among these conditions was significant, $F(1, 19) = 32.98$, $MSE = 56{,}760.63$, $\eta_p^2 = .63$, and not the quadratic trend, $F(1, 19) = 1.51$, $MSE = 17{,}879.78$, $\eta_p^2 = .07$. Accordingly, set-size effects can be modeled by a linear RT component.

Two models were compared as follows:

$$(1)\ \text{update-switch} + \text{set-size} + (\text{update-switch}*\text{set-size})$$

and

$$(2)\ \text{update-switch} + \text{item-position} + \text{set-size} + (\text{update-switch}*\text{set-size} + \text{item-position}*\text{set-size}).$$

Model 2 fitted the data better (Model 1: log likelihood = −238.29, AIC = 494.58, BIC = 529.55; Model 2: log likelihood = −204.85, AIC = 435.70, BIC = 486.22; ΔBIC = 43.33, BF = $2.56^{*}10^{9}$). Table 7 presents the correlation between

Table 7
Correlations Among Observed RT and Predictor Values for Experiments 3–4

Predictor	Observed RT: Experiment 3	Observed RT: Experiment 4
Update-switch	.43	.60**
Item-position	.72**	.58*
Set-size	.86***	.81***
Modified item-position	.78***	.79***
Switch-back	.31	.50*

Note. RT = response time.
* $p < .05$. ** $p < .01$. *** $p < .001$.

the observed RT and each of the predictors across the 18 conditions. BIC for this general model was smaller than for any of its nested models. However, compared to the second-best model, with update-switch, item-position, set-size, and update-switch*set-size as predictors, the general model was not strongly favored (ΔBIC = 1.49, BF = 2.11). See Table 10 and Figure 4b for model fit and Table 9 for parameter estimates.

Discussion of Experiments 3–4

The results of Experiments 3 and 4 replicated the earlier findings. The forward-scan and update-switching model fit well across different set-sizes. Two points should be emphasized.

First, the flexible strategy model was again superior to the "standard" item-position plus update-switch model in the full-display paradigm. Apart from replicating the findings of Experiment 1, this finding suggests that the decision whether to update an item or not is not only based on the bottom-up properties of the item, namely whether it is the same or different from the existing representation in WM. The flexible strategy model also implies that even repeated items are sometimes "updated," or re-created, in order to avoid the cost of switching between WM modes.

Second, the findings of Experiment 2, in which item-position (i.e., the number of to-be-updated item-position bindings) was not part of the best fitting model, were not replicated in Experiment 4. This finding confirms our assumption that the failure to detect the effect of number of updated item-position bindings in Experiment 2 was mere coincidence: In Experiment 2, the time for retrieving (and arguably refreshing) not-updated items happened to be close to the time for updating item-position bindings of updated items, so that the total time for both processes was constant with the constant set-size of Experiment 2. In Experiment 4, we saw evidence for a main effect of item-position, implying that in this experiment the time for updating an item-position binding was longer than the time spent on retrieving and refreshing a not-updated item.

General Discussion

The results of the present study can be summarized as follows. Updating RT in both the full-display and partial-display paradigms is well explained by a model in which the strongest source of variance between conditions is the cost of switching between updating and maintenance. In order to update the list, participants scan through it in forward order, and the overall updating time depends on the number of required switches between the two states. In the full-display paradigm, participants can reduce the number of update-switches by applying a more flexible strategy of updating all items, starting from the first to-be-updated location. However, this cannot be done in the partial-display paradigm. This

Table 8
Model Fit for Experiment 3

Predictor	AIC[a]	BIC[a]	Log likelihood
ModItPos + SwBack + SS + ModItPos*SS + SwBack*SS	−18.14	32.38	22.07
ModItPos + SwBack + SS + ModItPos*SS	**−22.10**	**20.65**	**22.05**
ModItPos + SwBack + SS + SwBack*SS	32.57	75.32	−5.29
ModItPos + SwBack + SS + ModItPos*SS + SwBack*SS	34.62	69.59	−8.31
ModItPos + SwBack + ModItPos*SS + SwBack*SS	468.34	511.08	−223.17
ModItPos + SwBack + ModItPos*SS	501.49	536.46	−241.74
ModItPos + SwBack + SwBack*SS	464.74	499.71	−223.37
ModItPos + SS + ModItPos*SS + SwBack*SS	16.08	58.83	2.96
ModItPos + SS + ModItPos*SS	74.26	109.24	−28.13
ModItPos + SS + SwBack*SS	75.72	110.70	−28.86
ModItPos + ModItPos*SS + SwBack*SS	475.72	510.69	−228.86
ModItPos + ModItPos*SS	502.05	529.25	−244.02
ModItPos + SwBack*SS	471.86	499.06	−228.93
SwBack + ModItPos*SS + SwBack*SS	671.48	706.46	−326.74
SwBack + ModItPos*SS	737.76	764.96	−361.88
SwBack + SwBack*SS	684.97	712.17	−335.48
SS + ModItPos*SS + SwBack*SS	334.73	369.71	−158.37
SS + ModItPos*SS	335.89	363.09	−160.94
SS + SwBack*SS	403.50	430.70	−194.75
ModItPos*SS	755.86	775.29	−372.93
SwBack*SS	758.01	777.44	−374.01

Note. Bold font indicates the best fitting model. AIC = Akaike information criterion; BIC = Bayesian information criterion; ModItPos = modified item-position; SwBack = switch-back; SS = set-size.
[a] Smaller AIC and BIC values indicate better fit.

Table 9
Parameter Estimates for Best Fitting Models in Experiments 3–4

Experiment/model	Predictor	Estimate (s)	*SE*	*df*	*t*
		Experiment 3			
Full model	Intercept	1.46	.09	335	17.02***
	Modified item-position	.22	.02	335	12.76***
	Switch-back	−.22	.03	335	−6.19***
	Set-size	.53	.04	335	11.92***
	Modified item-position*set-size	.08	.02	335	5.36***
	Switch-back*set-size	−.01	.05	335	−.14
Best fitting model	Intercept	1.47	.09	336	17.07***
	Modified item-position	.22	.02	336	12.70***
	Switch-back	−.22	.02	336	−9.84***
	Set-size	.53	.04	336	12.02***
	Modified item-position*set-size	.08	.01	336	5.55***
		Experiment 4			
Full and best fitting model	Intercept	1.33	.09	335	14.12***
	Update-switch	.53	.05	335	9.86***
	Item-position	.08	.04	335	2.28*
	Set-size	.47	.07	335	6.43***
	Update-switch*set-size	.22	.04	335	5.56***
	Item-position*set-size	.10	.03	335	3.59***

Note. Due to the inclusion of interaction terms in the models, we centered the predictors item-position, modified item-position, switch-back, and set-size at zero. This was done by subtracting the mean predictor value from the specific value in each condition. This procedure ensures that the main effects of the predictors are not generated through the interaction terms.
* $p < .05$. *** $p < .001$.

difference in strategy affordance appears to be the only qualitative difference between the two paradigms.

The update-switch cost provides evidence for two operation modes of WM for controlling its contents: maintenance and updating. A maintenance mode enables WM to keep information intact and resilient to the influence of other input, whereas the updating mode allows input to modify representations in memory. These two operation modes serve the two competing demands on WM: maintaining the stability of representations as long as they remain relevant, and flexibility of modifying representations whenever they need to be updated. Changing from one operation mode to the other takes time, generating the update-switch cost.

Table 10
Model Fit for Experiment 4

Predictor	AIC[a]	BIC[a]	Log likelihood
ItPos + UpSw + SS + ItPos*SS + UpSw*SS	**435.70**	**486.22**	**−204.85**
ItPos + UpSw + SS + ItPos*SS	467.47	510.22	−222.74
ItPos + UpSw + SS + UpSw*SS	444.96	487.71	−211.48
ItPos + UpSw + SS + ItPos*SS + UpSw*SS	501.98	536.96	−241.99
ItPos + UpSw + ItPos*SS + UpSw*SS	511.79	554.54	−244.90
ItPos + UpSw + ItPos*SS	829.50	864.48	−405.75
ItPos + UpSw + UpSw*SS	508.48	543.46	−245.24
ItPos + SS + ItPos*SS + UpSw*SS	663.67	706.41	−320.83
ItPos + SS + ItPos*SS	676.55	711.52	−329.27
ItPos + SS + UpSw*SS	659.98	694.95	−320.99
ItPos + ItPos*SS + UpSw*SS	694.24	729.21	−338.12
ItPos + ItPos*SS	894.45	921.65	−440.22
ItPos + UpSw*SS	696.10	723.30	−341.05
UpSw + ItPos*SS + UpSw*SS	559.61	594.58	−270.80
UpSw + ItPos*SS	883.64	910.84	−434.82
UpSw + UpSw*SS	557.30	584.51	−271.65
SS + ItPos*SS + UpSw*SS	718.76	753.73	−350.38
SS + ItPos*SS	721.17	748.38	−353.59
SS + UpSw*SS	728.70	755.90	−357.35
ItPos*SS	996.86	1,016.29	−493.43
UpSw*SS	774.80	794.23	−382.40

Note. Data in bold font indicate the best fitting model. AIC = Akaike information criterion; BIC = Bayesian information criterion; ItPos = item-position; UpSw = update-switch; SS = set-size.
[a] Smaller AIC and BIC values indicate better fit.

A bistate WM system as describe above is based on a gating mechanism that controls its input (Badre, 2012; O'Reilly, 2006), allowing for intact maintenance when the gate is closed and rapid updating when it opens. O'Reilly and colleagues (e.g., Frank et al., 2001; O'Reilly & Frank, 2006) suggested that retention of information over the short term is carried out in two parallel sets of representation. Information represented in the posterior/sensory cortex matches the perceptual input, and is constantly changing as new input flows in. The posterior cortex is connected through a gating system, implemented in the basal ganglia, with the prefrontal cortex. Opening the gate allows for updating the prefrontal cortex with the information held in the posterior cortex. Closing the gate enables keeping the prefrontal cortex representations intact, despite new perceptual input. While updating information in the posterior system can be regarded as automatic, subject to bottom-up input from the environment, updating the prefrontal system depends on opening the gate between the two.

The finding of an update-switch cost fits nicely with the gating model. Performance in the present tasks requires keeping and coordinating two sets of representations: the current memory list, which resulted from updating operations in the previous trial, and the new input specifying the required updates. In order to update correctly, participants need to scan both lists in synchrony and in forward order. Finding a mismatch between the two lists requires opening the gate between the perceptual representation of the display (in the posterior cortex) and WM (in the frontal cortex, according to the gating model) to allow updating. The gate should then be closed for nonupdated items in order to allow maintenance. Closing the gate is also required at the end of the trial in order to maintain the information until the next display, or the final-recall prompt, is presented. Within the above framework, our finding of an update-switching cost reflects the time needed for switching the gate state.

The update-switch hypothesis assumes that each trial begins in a maintenance mode. This assumption is supported by the idea that robust maintenance is the default state of the system. That is, the prefrontal cortex tends to maintain information intact over time, and this tendency is occasionally overridden by phasic bursts of activation that enable updating (Frank et al., 2001; O'Reilly, 2006; see also Olivers & Meeter, 2008, for an application of this idea in a model of attentional blink). Default maintenance is also implemented in a model of event perception and segmentation (Zacks, Speer, Swallow, Braver, & Reynolds, 2007). In sum, there are precedents for our assumption of default maintenance in several independent strands of research.

The finding of a switch cost between the two modes of WM operation corresponds with the existing literature on task-switching phenomena. Task switching (for reviews see Kiesel et al., 2010; Monsell, 2003; Vandierendonck, Liefooghe, & Verbruggen, 2010) refers to a situation in which the set of task-related rules change, either by external demands (Meiran, 1996; Rogers & Monsell, 1995) or by an internal decision (Arrington & Logan, 2004; Kessler, Shencar, & Meiran, 2009). Switching the present task is typically associated with a behavioral cost, compared to no-switch situations. Accordingly, one could think of the two gating modes of WM as two task sets controlling how the WM system treats old and new information. From this perspective, switching between updating and maintenance mode could be regarded as an instance of task switching. We argue that this interpretation is plausible for the partial-display paradigm but not the full display. Specifically, while performing the partial-display paradigm, participants switch between updating WM with the new information and refreshing old information given the cue "*." In order to perform this task, one needs to interpret the stimulus correctly, update WM with the given letters, and refresh existing representations given a "*." However, this is not the case in the full-display paradigm, where all letters are presented in each trial. In this case, the only instruction given to the participants is to update their memory with the given string of letters. Accordingly, no switching between two different intended tasks is involved. Because the general framework of the forward scan and update-switch model fit both paradigms, it is reasonable to assume that update-switch costs do not reflect switching between task sets but rather between two states of the WM system within a single task.

There is also a logical reason that makes us hesitate to describe the switch between updating and maintenance modes as a task-set switch: Switching from one task set to another is an updating process itself. The representation of the task to be carried out must be updated in working memory before response selection is carried out (Mayr & Kliegl, 2000; Monsell, 2003; Oberauer, Souza, Druey, & Gade, 2013). In a typical task-switch experiment, people switch between two tasks A and B on some trial-to-trial transition (i.e., the task-switch trials) but not on others (i.e., the task-repetition trials). Thus, the mode of WM has to switch between updating mode and maintenance mode. If switching between updating and maintenance mode is described as task-set switching, that would imply that a task-set switch (between maintenance and updating mode) is required to enable a task-set switch (between task A and task B). This theoretical framing gives rise to an infinite regress.

One important difference between task switching and mode switching is that task switching enables people to switch between arbitrarily defined tasks given to them through instruction. Therefore, representations of these task sets need to be learned, and in task-switching situations the appropriate representation needs to be retrieved from long-term memory and established in procedural WM (Mayr & Kliegl, 2000; Oberauer et al., 2013). In contrast, the maintenance and the updating mode can be regarded as two modes of operation built into the WM system. Selecting one of them does not require the (re)configuration of a representation of the intended task in procedural WM, but simply setting a parameter of the WM system to one of two possible values. Control of this parameter can be achieved by learning associations between features of the current situation that require updating (e.g., perceiving a new letter) and eliciting a "go" signal that switches the WM system from its default maintenance mode into updating mode (O'Reilly, 2006). Once acquired through reinforcement learning, this association controls the gate without further top-down input.

To conclude, the present work has shown the importance of the distribution of updated and not-updated items in a list for analyzing updating behavior, and it has identified the role of update-switching in explaining updating latency. Although gating of WM representations is a relatively established mechanism, the current work is the first to suggest that switching among the gate states results in a marked behavioral cost, which influences updating latency. Future work is required to understand the role of update-switching process in other updating paradigms and to implement this mechanism in detailed models of WM updating.

References

Arrington, C. M., & Logan, G. D. (2004). The cost of a voluntary task switch. *Psychological Science, 15,* 610–615. doi:10.1111/j.0956-7976.2004.00728.x

Badre, D. (2012). Opening the gate to working memory. *PNAS Proceedings of the National Academy of Science, USA, 109,* 19878–19879. doi:10.1073/pnas.1216902109

Braver, T. S., & Cohen, J. D. (2000). On the control of control: The role of dopamine in regulating prefrontal function and working memory. In S. Monsell & J. Driver (Eds.), *Attention and Performance XVIII: Control of cognitive processes* (pp. 713–737). Cambridge, MA: MIT Press.

Cohen, J. D., Braver, T. S., & O'Reilly, R. C. (1996). A computational approach to prefrontal cortex, cognitive control, and schizophrenia: Recent developments and current challenges. *Philosophical Transactions of the Royal Society of London B: Biological Sciences, 351,* 1515–1527. doi:10.1098/rstb.1996.0138

D'Ardenne, K., Eshel, N., Luka, J., Lenartowicz, A., Nystrom, L. E., & Cohen, J. D. (2012). Role of prefrontal cortex and the midbrain dopamine system in working memory updating. *PNAS Proceedings of the National Academy of Sciences, USA, 109,* 19900–19909. doi:10.1073/pnas.1116727109

Ecker, U. K. H., Lewandowsky, S., Oberauer, K., & Chee, A. E. H. (2010). The components of working memory updating: An experimental decomposition and individual differences. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 36,* 170–189. doi:10.1037/a0017891

Farrell, S., & Lelièvre, A. (2009). Is scanning in probed order recall articulatory? *Quarterly Journal of Experimental Psychology, 62,* 1843–1858. doi:10.1080/17470210802588400

Frank, M. J., Loughry, B., & O'Reilly, R. C. (2001). Interactions between frontal cortex and basal ganglia in working memory: A computational model. *Cognitive, Affective, & Behavioral Neuroscience, 1,* 137–160. doi:10.3758/CABN.1.2.137

Goschke, T. (2000). Intentional reconfiguration and involuntary persistence in task-set switching. In S. Monsell & J. Driver (Eds.), *Attention and Performance XVIII: Control of cognitive processes* (pp. 331–355). Cambridge, MA: MIT Press.

Henson, R. N. A. (1998). Short-term memory for serial order: The start-end model. *Cognitive Psychology, 36,* 73–137. doi:10.1006/cogp.1998.0685

Hitch, G. J., Fastame, M. C., & Flude, B. (2005). How is the serial order of a verbal sequence coded? Some comparisons between models. *Memory, 13,* 247–258.

Jeffreys, H. (1961). *Theory of probability*. Oxford, United Kingdom: Oxford University Press.

Kessler, Y., & Meiran, N. (2006). All updateable objects in working memory are updated whenever any of them is modified: Evidence from the memory updating paradigm. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 32,* 570–585. doi:10.1037/0278-7393.32.3.570

Kessler, Y., & Meiran, N. (2008). Two dissociable updating processes in working memory. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 34,* 1339–1348. doi:10.1037/a0013078

Kessler, Y., Shencar, Y., & Meiran, N. (2009). Choosing to switch: Spontaneous task switching despite associated behavioral costs. *Acta Psychologica, 131,* 120–128. doi:10.1016/j.actpsy.2009.03.005

Kiesel, A., Stenhauser, M., Wendt, M., Falkenstein, M., Jost, K., Philipp, A. M., & Koch, I. (2010). Control and interference in task switching—A review. *Psychological Bulletin, 136,* 849–874. doi:10.1037/a0019842

Lange, E. B., Cerella, J., & Verhaeghen, P. (2011). Ease of access to list items in short-term memory depends on the order of the recognition probes. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 37,* 608–620. doi:10.1037/a0022220

Lashley, K. S. (1951). The problem of serial order in behavior. In L. A. Jeffress (Ed.), *Cerebral mechanisms in behavior: The Hixon Symposium* (pp. 112–136). New York, NY: Wiley.

Lewandowsky, S., & Farrell, S. (2008). Short-term memory: New data and a model. *Psychology of Learning and Motivation, 49,* 1–48. doi:10.1016/S0079-7421(08)00001-7

Lewandowsky, S., & Murdock, B. B., Jr. (1989). Memory for serial order. *Psychological Review, 96,* 25–57. doi:10.1037/0033-295X.96.1.25

Lorch, R. F. Jr., & Myers, J. L. (1990). Regression analysis of repeated measures data in cognitive research. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 16,* 149–157. doi:10.1037/0278-7393.16.1.149

Masson, M. E. J., & Loftus, G. R. (2003). Using confidence intervals for graphically based data interpretation. *Canadian Journal of Experimental Psychology, 57,* 203–220. doi:10.1037/h0087426

Mayr, U., & Kliegl, R. (2000). Task-set switching and long-term memory retrieval. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 26,* 1124–1140. doi:10.1037/0278-7393.26.5.1124

Meiran, N. (1996). Reconfiguration of processing mode prior to task performance. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 22,* 1423–1442. doi:10.1037/0278-7393.22.6.1423

Monsell, S. (2003). Task switching. *Trends in Cognitive Sciences, 7,* 134–140. doi:10.1016/S1364-6613(03)00028-7

Myerson, J., Adams, D. R., Hale, S., & Jenkins, L. (2003). Analysis of group differences in processing speed: Brinley plots, Q-Q plots, and other conspiracies. *Psychonomic Bulletin & Review, 10,* 224–237. doi:10.3758/BF03196489

Nairne, J. S., Ceo, D. A., & Reysen, M. B. (2007). The mnemonic effects of recall on immediate retention. *Memory & Cognition, 35,* 191–199. doi:10.3758/BF03195954

Oberauer, K. (2001). Removing irrelevant information from working memory: A cognitive aging study with the modified Sternberg task. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 27,* 948–957. doi:10.1037/0278-7393.27.4.948

Oberauer, K., Demmrich, A., Mayr, U., & Kliegl, R. (2001). Dissociating retention and access in working memory: An age-comparative study of mental arithmetic. *Memory & Cognition, 29,* 18–33. doi:10.3758/BF03195737

Oberauer, K., & Göthe, K. (2006). Dual-task effects in working memory: Interference between two processing tasks, between two memory demands, and between storage and processing. *European Journal of Cognitive Psychology, 18,* 493–519. doi:10.1080/09541440500423038

Oberauer, K., Souza, A. S., Druey, M., & Gade, M. (2013). Analogous mechanisms of selection and updating in declarative and procedural working memory: Experiments and a computational model. *Cognitive Psychology, 66,* 157–211. doi:10.1016/j.cogpsych.2012.11.001

Olivers, C. N. L., & Meeter, M. (2008). A boost and bounce theory of temporal attention. *Psychological Review, 115,* 836–863. doi:10.1037/a0013395

O'Reilly, R. C. (2006, October 6). Biologically based computational models of high-level cognition. *Science, 314,* 91–94. doi:10.1126/science.1127242

O'Reilly, R. C., & Frank, M. J. (2006). Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. *Neural Computation, 18,* 283–328. doi:10.1162/089976606775093909

Pinheiro, J. C., & Bates, D. M. (2000). *Mixed-effects models in S and S-Plus*. New York, NY: Springer.

Pinheiro, J., Bates, D. M., DebRoy, S., Sarkar, D., & the R Development Core Team. (2012). *nlme: Linear and nonlinear mixed effects models* (R package Version 3.1–101) [Computer software]. Vienna, Austria: R Foundation for Statistical Computing.

Raye, C. L., Johnson, M. K., Mitchell, K. J., Greene, E. J., & Johnson, M. R. (2007). Refreshing: A minimal executive function. *Cortex, 43,* 135–145. doi:10.1016/S0010-9452(08)70451-9

Rogers, R. D., & Monsell, S. (1995). The costs of a predictable switch between simple cognitive tasks. *Journal of Experimental Psychology: General, 124,* 207–231. doi:10.1037/0096-3445.124.2.207

Schneider, W., Eschman, A., & Zuccolotto, A. (2002). *E-Prime user's guide*. Pittsburgh, PA: Psychology Software Tools.

Vandierendonck, A., Liefooghe, B., & Verbruggen, F. (2010). Task switching: Interplay of reconfiguration and interference control. *Psychological Bulletin, 136,* 601–626. doi:10.1037/a0019791

Zacks, J. M., Speer, N. K., Swallow, K. M., Braver, T. S., & Reynolds, J. R. (2007). Event perception: A mind-brain perspective. *Psychological Bulletin, 133,* 273–293. doi:10.1037/0033-2909.133.2.273

Zhang, Y., Verhaeghen, P., & Cerella, J. (2012). Working memory at work: How the updating process alters the nature of working memory transfer. *Acta Psychologica, 139,* 77–83. doi:10.1016/j.actpsy.2011.10.012

Received February 3, 2013
Revision received November 4, 2013
Accepted November 8, 2013 ■

Correction to Noreen and MacLeod (2012)

In the article "It's All in the Detail: Intentional Forgetting of Autobiographical Memories Using the Autobiographical Think/No-Think Task" by Saima Noreen and Malcolm D. MacLeod (*Journal of Experimental Psychology: Learning, Memory, and Cognition*, Vol. 39, No. 2, pp. 375-393. doi: 10.1037/a0028888), Figure 7 was erroneously identical to Figure 4. The correct version of Figure 7 appears below.

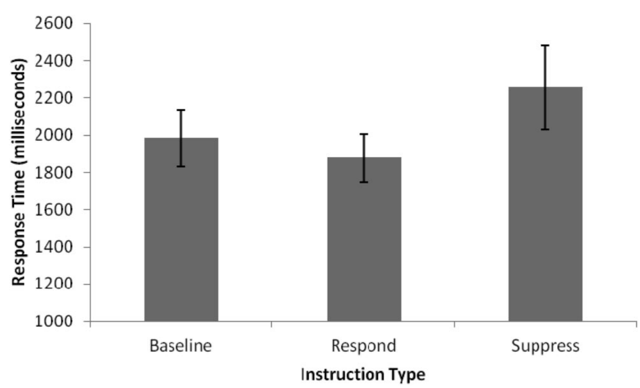

Figure 7. Study 2: Mean response times for generating memories recalled in the baseline, think, and no-think conditions at final test (error bars represent within-subjects standard errors in each condition).

DOI: 10.1037/xlm0000014