Behavior Research Methods
2010, 42 (2), 571-585
doi:10.3758/BRM.42.2.571

A working memory test battery for MATLAB

Stephan Lewandowsky
University of Western Australia, Perth, Australia

Klaus Oberauer
University of Zurich, Zurich, Switzerland

Lee-Xieng Yang
National Cheng-Chi University, Taipei City, Taiwan

and

Ullrich K. H. Ecker
University of Western Australia, Perth, Australia

We present a battery of four working memory tasks that are implemented using MATLAB and the free Psychophysics Toolbox. The package includes preprocessing scripts in R and SPSS to facilitate data analysis. The four tasks consist of a sentence-span task, an operation-span task, a spatial short-term memory test, and a memory updating task. These tasks were chosen in order to provide a heterogeneous set of measures of working memory capacity, thus reducing method variance and tapping into two content domains of working memory (verbal, including numerical, vs. spatial) and two of its functional aspects (storage in the context of processing and relational integration). The task battery was validated in three experiments conducted in two languages (English and Chinese), involving more than 350 participants. In all cases, the tasks were found to load on a single latent variable. In a further experiment, the latent working memory variable was found to correlate highly but not perfectly with performance on Raven's matrices test of fluid intelligence. We suggest that the battery constitutes a versatile tool to assess working memory capacity with either English- or Chinese-speaking participants. The battery can be downloaded from www.cogsciwa.com ("Software" button).

There has been rapidly growing interest in the relationship between working memory capacity (WMC) and other aspects of higher level cognition. WMC is closely related to reasoning ability (Kyllonen & Christal, 1990; Süß, Oberauer, Wittmann, Wilhelm, & Schulze, 2002) and to fluid intelligence (Engle, Tuholski, Laughlin, & Conway, 1999), and it is among the best predictors of individual differences in a variety of complex cognitive activities such as text comprehension (Daneman & Merikle, 1996), learning of complex skills (Shute, 1991), and arithmetic (Bayliss, Jarrold, Gunn, & Baddeley, 2003). Individual differences in WMC are also related to the ability to control cognitive processes both in laboratory tasks (Kane, Conway, Hambrick, & Engle, 2007) and in everyday life (Kane, Brown, et al., 2007). Changes in WMC have been identified as one of the major driving forces of cognitive growth over childhood (Kail, 2007) and cognitive decline in old age (Hertzog, Dixon, Hultsch, & MacDonald, 2003). In recent years, WMC has been increasingly used as a predictor of phenomena in social psychology and clinical psychology, such as stereotype threat (Schmader, Johns, & Forbes, 2008), choking under pressure (Beilock & Carr, 2005), self-regulatory behavior (Hofmann, Gschwendner, Friese, Wiers, & Schmitt, 2008), and clinical conditions such as schizophrenia (Lee & Park, 2005).

An indispensable prerequisite of progress in this line of research is the availability of a well-validated tool to measure WMC. Psychometric research has established that WMC can be characterized by a general factor that accounts for a large proportion of variance, together with lower order factors capturing specific content domains (verbal vs. spatial; Kyllonen & Christal, 1990; Oberauer, Süß, Schulze, Wilhelm, & Wittmann, 2000) and specific cognitive functions (storage in the context of processing vs. relational integration; Oberauer, Süß, Wilhelm, & Wittmann, 2003). A variety of tasks have been shown to have high loadings on the general WMC factor (Oberauer et al., 2000), so that, in principle, there is a wealth of options for measuring WMC. Nevertheless, in practice, WMC is mostly assessed by a single paradigm: the complex-span paradigm (e.g., reading span, operation span [OS]; cf. Conway et al., 2005). Complex-span tasks (e.g., OS) interleave a memory component (e.g., remember a set of items in order of presentation) with a secondary processing task (e.g., judge the correctness of equations). Unsworth, Heitz, Schrock, and Engle (2005) made

L.-X. Yang, lxyang@nccu.edu.tw

available a particularly useful and automated version of the OS task, which has facilitated much research; at the same time, it might have contributed to seducing researchers into using a single task to measure WMC, which is unfortunate, because it results in a contamination of measured WMC with variance that is not due to the intended construct (i.e., general WMC), but rather to the specifics of the task paradigm chosen. To reduce the contribution of task-specific variance, researchers should use a battery of heterogeneous indicators to measure a general construct such as WMC (Oberauer, 2005b; Wittmann, 1988).

We believe that one reason for the often narrow operationalization of WMC is the lack of an easily available, easy-to-use instrument for assessing WMC through a heterogeneous battery of psychometrically validated tasks. The purpose of our present work is to provide such a tool: a balanced and tested battery of tasks that permits measurement of WMC at the latent-variable level, relatively free from task-specific variance.

We present a battery for assessing WMC through four tasks: an updating task (memory updating, MU), two span tasks (OS; sentence span, SS), and a spatial task (spatial short-term memory, SSTM). These tasks were chosen for three reasons: First, these tasks have been shown to have high loadings on WMC factors (Oberauer, 2005a; Oberauer et al., 2000) and were strongly correlated with new tasks designed to measure WMC (Oberauer et al., 2003). Second, these four tasks were substantially correlated with a test of reasoning ability and a test of general intelligence. To illustrate, Table 1 shows the correlations obtained in two large studies (Oberauer et al., 2000; Oberauer et al., 2003). Third, the four tasks represent both content domains of WMC (verbal, including numerical, and spatial) and both functional aspects (storage in the context of processing and relational integration).

We included two versions of the complex-span paradigm (reading span and OS) in the present battery, not so much because we believe these tasks to be particularly good measures of WMC, but because we found it desirable that results obtained with our battery be comparable with other results that are frequently based on just one or two versions of the complex-span paradigm. The complex-span tasks in the present battery were improved in several regards over those used in previous studies.

Table 1
Correlations Between Working Memory Tasks and Measures of Reasoning Obtained in Studies by Oberauer et al. (2000) and Oberauer et al. (2003)

	Reasoning		*gF*	
Task	2000	2003	2000	2003
MU	.67	.62	.62	.54
SS	.57	.47	.64	.51
OS	.54	.24	.52	.29
SSTM	.59	.58	.55	.44

Note—MU, memory updating (called "memory updating numerical" in the original publications); SS, sentence span (called "reading span" in the original publications); OS, operation span (called "computation span" in the original publications); SSTM, spatial short-term memory; *gF*, general fluid intelligence score.

These improvements were in part based on research on the psychometric properties of complex-span tasks (Conway et al., 2005; Friedman & Miyake, 2004; Lépine, Barrouillet, & Camos, 2005) and in part on experience in our labs and other labs.[1] First, trials of different list lengths are presented in random order so that the participants cannot anticipate the length of each list. Second, we separated the memory contents from the material to be processed: *Memory items* were letters presented separately from the sentences or arithmetic equations to be processed in the complex-span tasks. The third and most important feature of our complex-span tasks is that the presentation times for memory items, as well as the maximum presentation time for the processing stimuli, were held constant and controlled by the computer program, not by the participant or the experimenter. This feature has been shown to reduce the contribution of individual differences in strategy to the scores (Friedman & Miyake, 2004; Lépine et al., 2005).

We next report three validation experiments, the Method sections of which describe the tasks in full detail. The installation and usage of the task battery is described in the subsequent section.

VALIDATION EXPERIMENTS

We report three validation experiments, each involving more than 100 participants. For each study, we present a structural equation model that confirms that all of the tasks have substantial loadings on a common WMC factor. The only difference between the three experiments was the sentence stimuli for the SS task: In Experiment 1, the stimuli included garden-path sentences whose grammaticality was rather difficult to determine (and performance was therefore expected to be lower). In Experiment 2, we used simpler sentences (which hence yielded higher performance on the processing task), and Experiment 3 was conducted with Chinese-speaking participants and Chinese sentences for the SS task.

All of the experiments also involved a second session, which variously involved a memory-updating task (Experiments 1 and 2) or a category-learning task (Experiment 3). Those data are not reported here; a joint analysis of the present Experiment 1 and the memory-updating task can be found in Ecker, Lewandowsky, Oberauer, and Chee (2010).

Experiment 1

The purpose of the first experiment was to provide a validation of the WMC battery with a particularly difficult variant of the SS task.

Method

Participants. The participants were 110 third-year psychology students (16 male; age range = 19–43 years, mean age = 21.4) from the University of Western Australia. They participated in a single experimental session of 40 min for partial course credit.

Stimuli, Design, and Procedure. The four WMC tasks were MU, OS, SS, and SSTM. The tasks were always administered in this order. Each task took approximately 10 min.

The MU task. This task was originally designed by Salthouse, Babcock, and Shaw (1991) and was adapted for psychometric purposes by Oberauer et al. (2000). The task was to encode an initial set of digits, each presented in a separate frame on the screen, and to subsequently update these digits through arithmetic operations.

On each trial, the participants were presented with a set of frames that contained the to-be-remembered digits; this set size varied from three to five across trials (three frames were presented side by side in one row; four- and five-frame displays were presented in two rows). The participants initiated each trial with a keypress. The starting digits were then presented in their frames, one by one, for 1 sec each. Following encoding, cues for the arithmetic operations, such as "+2" or "−4," were displayed in individual frames for 1.3 sec each, followed by a 250-msec blank interval. The participants had to apply the operation to the digit that they currently remembered in that frame and to replace the memorized content by the result. After a varying number of updating operations (between two and six), final recall was signaled by question marks appearing one by one in each frame. The participants responded by typing the remembered digit for that particular frame; all nondigit keys were blocked. The typed digits appeared in their frames for 1 sec. There was no time constraint for recall, and no performance feedback was provided in this or any of the remaining tasks.

The operations ranged from −7 to +7, excluding 0. The interim and final results ranged from 1 to 9. Operations exceeding ±7 were not used, because one could then infer both the initial value (e.g., 1) and the result (9) from the operation (+8). Not every frame was necessarily updated within a trial, and the same frame could be updated multiple times. However, each updating step involved a different randomly chosen frame to keep the contribution of switch costs constant across operations, trials, and participants.

There were 15 trials in total, generated by crossing the set sizes of three, four, and five with the number of updating operations (two, three, four, five, and six). The 15 test trials were preceded by 2 practice trials (1 with two and 1 with three frames). All of the sets of initial digits; operations; updating, recall prompt, and trial orders were generated randomly. Yet, in order to control and minimize method variance, all randomly generated parameters were kept constant for all participants.

The OS task. This task was originally designed by Turner and Engle (1989). On each trial, the participants saw an alternating sequence of arithmetic equations (e.g., 3 + 2 = 5) and to-be-remembered consonants (which excluded Y and Q). The participants had to judge the correctness of each equation and to encode the following consonant for later serial recall.

Commencement of a trial was indicated by a fixation cross presented for 1.5 sec. Then, the first equation appeared centrally on the screen. It disappeared when the participants made a response or after the maximum response time of 3 sec had elapsed. The participants used the "/" and "Z" keys to make *Yes, this is correct* and *No, this is not correct* responses, respectively. The keys were labeled with "Y" and "N" accordingly.[2]

After the equation disappeared, a consonant was presented centrally for 1 sec. After a 100-msec blank interval, the next equation appeared. This sequence repeated four to eight times, depending on list length. Following list presentation, recall of the letters was prompted with a question mark and a blinking underscore. The participants then typed the remembered series of letters in their order of presentation; all letter keys were accepted, but nonletter keys were blocked. Every typed letter appeared next to the question mark for 200 msec. The participants had to type as many letters as were actually presented in the trial. They were informed that the order of letters mattered and were hence instructed to guess if necessary, rather than skip letters that they could not remember. There was no timing constraint for recall. The intertrial interval was 500 msec; there was a self-paced break after every 3 trials.

List length (i.e., the number of equations and letters) ranged from four to eight. There were 15 trials altogether, 3 trials per list length. The first equation operand was randomly drawn from the 1–10 range, the second operand was drawn from between −9 and +10 (excluding 0), and the results ranged from +1 to +20.[3] Half of the equations were correct. Three practice trials (with list lengths of three, four, and five) preceded the experimental trials.

Letter sequences, equations, and trial order were placed in one random order that remained the same for all of the participants. The consonant lists for each trial contained no repetitions.

The SS task. This task was a variant of the task originally designed by Daneman and Carpenter (1980) and was very similar to the OS task just described, except that the processing task was to judge the meaningfulness of sentences. There were equal numbers of meaningful and meaningless sentences, and regardless of meaningfulness, there were three types of sentence structures. About a third of the meaningful sentences were so-called *garden-path* sentences, which require updating of an initial parsing solution within the sentence (e.g., *As Toby sang a song played on the radio*). They were paired with structurally analogous sentences that were meaningless regardless of how they were parsed (e.g., *As Danny drank the milk rolled over the hill*). Another third of the sentences had the same structure but were disambiguated early by an additional pronoun with reference to the object, thus avoiding the garden path (e.g., *While Susan wrote the letter it fell off the table* [meaningful] vs. *As the chef stirred the soup it veered into the ditch* [meaningless]). In the remaining third of the sentences, the pronoun referred to the participant (*While Lisa drank the water she drove down the street* [meaningful] vs. *As the man walked the poodle he slept calmly* [meaningless]).

All of the sentences contained between 8 and 11 words, and sentence length, the use of *while* versus *as*, and the use of concrete first names were broadly counterbalanced across sentence categories. Because this processing task was more difficult than the processing of equations, the maximum response time to sentences was set to 5 sec, and list lengths only ranged from three to seven. The practice phase comprised three trials of set sizes of two, three, and four. In all other respects, the SS and OS tasks were identical.

The SSTM task. This task closely followed the original version by Oberauer (1993). The participants had to remember the location of a number of dots in a 10 × 10 grid. Following central presentation of a fixation cross for 1 sec, the grid was shown, and a variable number of solid dots appeared, one by one, in individual cells of the grid for 900 msec each (interstimulus interval = 100 msec). The participants were instructed to remember the spatial relations between the dots. That is, absolute dot positions were irrelevant; only the overall pattern of dots was to be remembered. After all of the dots were presented, the participants were cued to reproduce the pattern of dots. The cue was presented using a pattern mask that was the same size as the grid.

The participants reproduced the remembered pattern of dots by clicking the cells using a standard computer mouse. The order in which the dots were reproduced was irrelevant. Clicking on a dot again made it disappear, and the participants were allowed to correct the generated dots until they were satisfied with their response. They proceeded to the next trial by clicking a "Next" button at the bottom of the screen, followed by pressing the space bar. No feedback was given.

The number of to-be-remembered dots varied from two to six, and for each set size, all of the dots on a trial occurred either within a 5 × 5 area (i.e., a quarter of the grid) or anywhere in the grid, with equal probability. Altogether, there were 30 trials, 6 at each set size. Dot positions were generated at random, with the constraint that no dots appeared in corner positions to avoid verbal coding. Again, the order of trials and dot sequences was fixed for all of the participants.

Results

Scoring procedure. In all three experiments, MU, OS, and SS were scored as the proportion of items recalled correctly (in the context of complex-span paradigms, this has also been referred to as *partial credit scoring*; see Conway et al., 2005). For instance, a participant correctly

Table 2
Performance on the Working Memory Task in All Three Validation Experiments

Task	*M*	*SE*	Minimum	Maximum	Skewness	Kurtosis	Cronbach's α
			Experiment 1 ($N = 110$)				
MU	.54	.016	.18	.99	0.39	−0.49	.86
OS	.68	.012	.23	.94	−0.96	1.84	.78
OS_{pt}	.91	.005	.73	1.00	−0.85	0.45	
SS	.63	.017	.18	.96	−0.37	−0.54	.85
SS_{pt}	.80	.009	.48	.93	−1.19	1.75	
SSTM	.85	.004	.73	.94	−0.37	−0.26	.84
			Experiment 2 ($N = 114$)				
MU	.61	.015	.23	.93	−0.17	−0.49	.85
OS	.71	.012	.31	.93	−0.33	−0.38	.80
OS_{pt}	.90	.007	.59	1.00	−1.32	2.18	
SS	.69	.013	.17	.96	−0.85	1.40	.84
SS_{pt}	.91	.005	.70	.99	−1.09	1.42	
SSTM	.84	.005	.69	.99	−0.13	0.08	.86
			Experiment 3 ($N = 160$)				
MU	.72	.014	.20	1.00	−0.48	−0.57	.91
OS	.79	.008	.46	1.00	−0.61	0.31	.77
OS_{pt}	.95	.003	.74	1.00	−1.84	6.14	
SS	.84	.009	.41	1.00	−1.31	1.78	.76
SS_{pt}	.80	.008	.33	.95	−1.32	3.39	
SSTM	.89	.004	.68	.99	−0.98	1.76	.87

Note—MU, memory updating; OS, operation span; OS_{pt}, OS processing task; SS, sentence span; SS_{pt}, SS processing task; SSTM, spatial short-term memory.

remembering five out of six letters in a span-task trial would score 5/6 on that trial, and the total score would be calculated as the mean of these partial scores across trials. In the case of OS and SS, an item was scored as correct if it was reproduced in the correct list position.

The scoring of SSTM was slightly more complex, because the participants were instructed to focus on relative position, rather than on absolute location, during reconstruction. Hence, the similarity between the presented and recalled patterns formed the SSTM score. Pattern similarity was computed as the sum of the dot-to-dot similarities, which in turn were scored by awarding 2 points for no distance between a recalled dot and a presented dot, 1 point for a deviation of one cell in any direction, and 0 points if the deviation exceeded one cell. When computing the similarity, the presented and recalled dot patterns were first aligned on the location of one dot pair (i.e., presented and recalled), before computing the dot-to-dot similarity in the manner just described. Because there are $n!$ possible ways in which two patterns of size n can be aligned on a single dot pair, similarity was computed for all $n!$ alignments, and the largest similarity value was used as a participant's score on a given trial. For consistency with the other WMC tasks, each pattern similarity score was converted to a proportion by dividing it by the full-match similarity (i.e., if all recalled dots matched the presented dot locations), before the proportions were averaged across all trials in the same manner as for the other tasks.

Descriptives and correlations. Table 2 provides summary information for all of the tasks and for all of the experiments. The table shows that performance across participants ranged from safely above floor to near ceiling. An examination of the underlying distributions (not shown here) revealed that the scores in all of the tasks were approximately normally distributed. Two participants fell more than 3 *SD*s below the mean for the OS task, but because their performance was acceptable for the other tasks, all observations were retained for the structural equation modeling. The table also shows values of Cronbach's α for each task, computed by splitting the trials for each task into three sets and computing α across these sets. The pairwise correlations between the WMC tasks are shown in Table 3.

Structural equation modeling. The data from the four tasks were submitted to a structural equation model analysis, and a single latent variable was found to be sufficient to accommodate performance on all tasks. The final model is shown in Figure 1 and includes a correlation be-

Table 3
Correlations Between Working Memory Capacity Tasks for Experiments 1–3

Task	MU	OS	SS	SSTM
		Experiment 1		
OS	.36	–		
SS	.30	.50	–	
SSTM	.49	.29	.20	–
		Experiment 2		
OS	.68	–		
SS	.53	.64	–	
SSTM	.32	.26	.28	–
		Experiment 3		
OS	.45	–		
SS	.43	.56	–	
SSTM	.42	.32	.25	–

Note—MU, memory updating; OS, operation span; SS, sentence span; SSTM, spatial short-term memory.

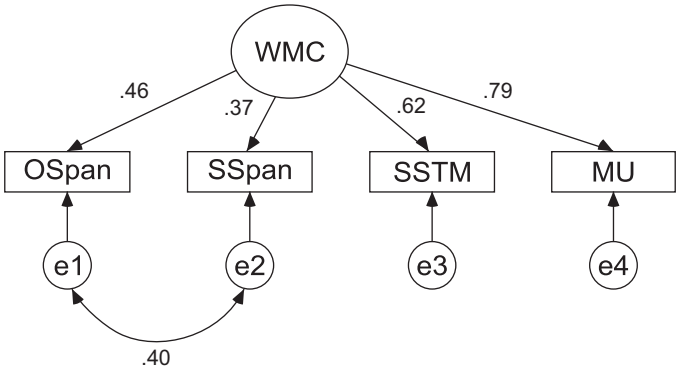

Figure 1. Structural equation model for Experiment 1. All correlations and loadings are standardized estimates. WMC, working memory capacity; OSpan, operation span; SSpan, sentence span; SSTM, spatial short-term memory; MU, memory updating; e1, e2, . . . , measurement error or unique variance in the indicator variable that is not explained by the latent variable.

tween the errors for the two span tasks, which was included on the basis of modification indices. The fit of this model was excellent [$\chi^2(1) = 0.3$, $p > .1$; comparative fit index (CFI) = 1.0; root-mean square error of approximation (RMSEA) = 0.0; and standardized root-mean square residual (SRMR) = 0.009]. Removal of the pairwise correlation between the two span tasks would have significantly worsened fit [$\Delta\chi^2(1) = 14.2$]. The composite reliability coefficient for this model, computed using the phantom factor technique recommended by Raykov (1997; see also Fan, 2003) was .56.

Experiment 2

The purpose of the second experiment was to repeat the first but with a considerably easier version of the SS task.

Method

Participants and Procedure. In Experiment 2, we recruited a new set of 114 undergraduates at the University of Western Australia for voluntary participation in a single experimental session. The participants were in their second or third year of the psychology degree and were enrolled in a course on cognitive psychology (27 male, age range = 18–62 years, mean age = 20.8).[4]

Experiment 2 was identical to Experiment 1, with the exception of the SS task: The garden-path sentences from Experiment 1 were eliminated and replaced by shorter and easier sentences, which had to be judged as true or false. Many sentences had an "X is a Y" or "X has a Y" structure, such as *A hammer is a tool* (true) or *An eagle has a fin* (false); others were comparative or simple common sense statements, such as *Children are younger than adults* (true) or *Cows live underground* (false). All of the sentences contained between three and six words; the number of words was counterbalanced across true and false categories. Furthermore, sentence presentation time was shortened to 4 sec (down from 5 sec), we used set sizes of between four and eight (instead of between three and seven), and the three practice trials used set sizes of three, four, and five (up from two, three, and four).

Results

Descriptives and correlations. Descriptive statistics and correlations between tasks are shown in Tables 2 and 3. An inspection of the distributions (not shown here) revealed approximate normality. One participant fell more than 3 *SD*s below the mean for OS, and a different 2 participants were identified as outliers by the same criterion for SS. Because no participant fell below that threshold on more than one task, all observations were retained for the structural equation modeling.

A comparison of the means in Table 2 for the first two experiments shows that the simpler sentences in the second experiment raised performance on the processing task considerably relative to that in the first study. This increase in processing performance was accompanied by a smaller increase in SS memory performance. Memory performance on the other three tasks was nearly identical across the two experiments; therefore, we believe that the increase in SS memory performance was due to the easier processing task. The pattern of correlations among the four tasks also changed somewhat relative to that in Experiment 1. Whereas in Experiment 1 MU and SSTM tended to cluster together, probably because both tasks have a spatial component, in Experiment 2 MU clustered together with SS and OS, and all three tasks had relatively low correlations with SSTM. It seems that in Experiment 1, variance in MU was determined jointly by numerical and spatial abilities, whereas in Experiment 2 numerical abilities dominated.

Structural equation model. The structural equation model for this experiment was the same as that for Experiment 1 and is shown in Figure 2. The model's fit was again excellent [$\chi^2(1) = 1.1$, $p > .1$; CFI = 0.999; RMSEA = 0.032; SRMR = 0.019]. Removal of the pairwise correlation between the two span tasks would have affected fit very little [$\Delta\chi^2(1) = 1.8$]. The composite reliability coefficient for this model was .70.

Experiment 3

The purpose of the third experiment was to generalize the WMC battery to a different linguistic environment. We therefore generated stimuli for the SS task in traditional Chinese and tested participants in Taiwan.

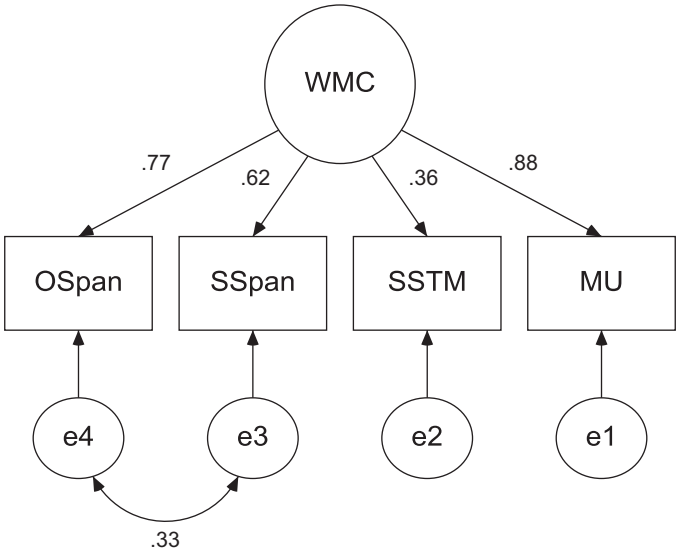

Figure 2. Structural equation model for Experiment 2. All correlations and loadings are standardized estimates. WMC, working memory capacity; OSpan, operation span; SSpan, sentence span; SSTM, spatial short-term memory; MU, memory updating.

Method

Participants and Procedure. The final experiment was identical to the first two experiments, with the following two exceptions. First, the participants were 160 undergraduate students from National Cheng-Kung University in Taiwan. Second, for the SS task, the sentences were composed of 17 Chinese characters. Half of the sentences were meaningful, and half were meaningless. For example, a meaningful sentence might be *I went out without taking any money, but fortunately I ran into an old friend who helped me out*; a parallel meaningless version of this sentence would replace *fortunately* with *unfortunately*. The participants were instructed to judge whether each sentence was meaningful.

The remaining three tasks were unchanged from the two experiments conducted in English.

Results

Descriptives and correlations. Summary results and correlations are again shown in Tables 2 and 3, respectively. An inspection of the distributions (not shown here) again revealed approximate normality. One participant who scored 0 on the OS processing task was eliminated from further consideration. After the elimination of that participant, there were 0, 2, 2, and 3 participants who fell more than 3 *SD*s below the means for the MU, OS, SS, and SSTM tasks, respectively. Because no participant fell below that threshold on more than two tasks, those observations were retained for the structural equation modeling, which is therefore based on data from 159 participants.

Owing to equipment failure and clerical data management problems, the values of Cronbach's α could only be computed from a subset of participants ($n = 124$ for the SSTM task, and $n = 100$ for the remaining three tasks). All other summary statistics and the structural equation model were based on observations from the full sample of 159 participants for all of the tasks.

Structural equation model. The structural equation model for this experiment was the same as those for Experiments 1 and 2 and is shown in Figure 3. The model's fit was again excellent [$\chi^2(1) = 0.6, p > .1$; CFI = 1.0; RMSEA = 0.0; SRMR = 0.01]. Removal of the pairwise correlation between the two span tasks would have significantly worsened fit [$\Delta\chi^2(1) = 9.0$]. All four tasks had substantial loadings on the factor, demonstrating that they all shared a substantial amount of variance that is represented by the factor. The composite reliability coefficient was .79.

Summary of Validation Experiments and Discussion

The three experiments converge on the conclusion that our task battery has considerable internal validity: In all cases, the tasks were at least moderately correlated, reflecting a common source of variance that is captured by a latent factor with substantive loadings of at least three out of the four tasks. This result was obtained across different types of sentences in the SS task, and it was also found to be invariant across two very different languages (English and Chinese) and two very different cultures (Australian and Taiwanese). Nevertheless, the data also indicate that each task had some task-specific variance that was not accounted for by the common factor. The amount of specific variance plus measurement error can be computed by subtracting the squared standardized loading of each task on the latent factor from 1. The amount of measurement error alone can be estimated as 1 minus the task's reliability, as given by Cronbach's α. These considerations imply two conclusions. First, each individual task reflects a large amount of specific variance plus error variance. Those values ranged from .23 for the MU task in Experiment 2 to .87 for the SSTM task in the same experiment.

Second, because of the high values of Cronbach's α, only a modest fraction of this variance can be attributed to mea-

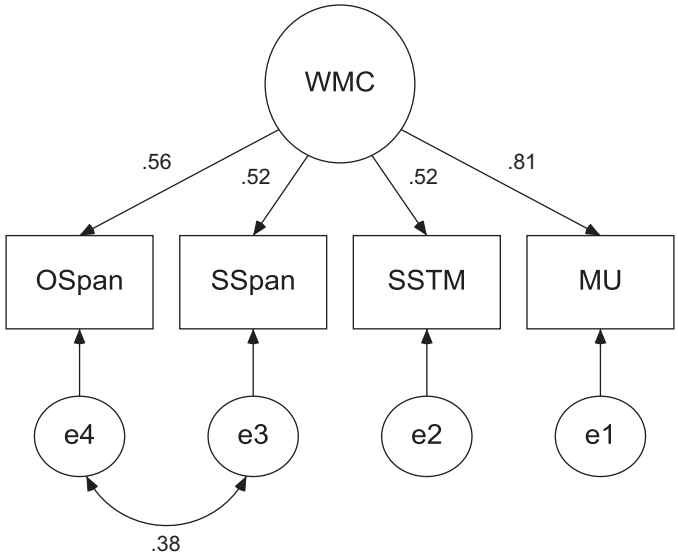

Figure 3. Structural equation model for Experiment 3. All correlations and loadings are standardized estimates. WMC, working memory capacity; OSpan, operation span; SSpan, sentence span; SSTM, spatial short-term memory; MU, memory updating.

surement error (e.g., .15 and .16, respectively, for the MU and SSTM tasks in Experiment 2). It follows that a large proportion of variance must be considered to be reliable task-specific variance. This, in turn, implies that any measure of WMC that is based on a single task is likely to reflect more variance due to *specific* features of that task than variance due to the construct that it is meant to measure. As a consequence, correlations between WMC thus measured with other variables are difficult to interpret, because it is not clear to what degree they are due to genuine variance in WMC, as opposed to variance arising from specific features of the task. Our present results and thinking are thus in line with a recent demonstration by Schmiedek, Hildebrandt, Lövdén, Lindenberger, and Wilhelm (2009) that using only complex-span tasks yields too narrow a measure of WMC.

This problem also holds to some (albeit lessened) degree when several versions of the complex-span paradigm are used to measure WMC. The need to include a correlation between the error terms of the SS and OS tasks shows that the two versions of complex-span tasks have variance in common that they do not share with the other two tasks. This shared variance is likely to be paradigm specific: Thus, even composite scores of two or more complex-span tasks are likely to be contaminated by paradigm-specific variance that does not reflect the intended construct. This pitfall can be avoided by use of a battery of heterogeneous tasks, such as those presented in this article. Our tasks combine multiple methods and reflect multiple content domains and diverse aspects of WMC functioning. When combined with the use of structural equation modeling to establish a measurement model for WMC, the battery of tasks yields a latent factor that reflects a pure estimate of the common variance of the indicator tasks, thus removing both task-specific variance and error variance and yielding a purer measure of the WMC construct than is obtainable with one task or with a number of complex-span tasks alone.

In addition, external validity of our task battery was established by another experiment—not reported in detail here—that was conducted by Stewart Craig in the first author's laboratory. That experiment involved 135 participants (sampled from the same pool as were those in Experiments 1 and 2) and was identical to Experiment 1 (i.e., the difficult sentences for the SS task were used), but in addition, Raven's advanced progressive matrices (Raven, 1989) test was administered to the same participants. This test is considered one of the most reliable tests of fluid intelligence.

We first generated a measurement model of WMC in the same manner as for the preceding three experiments [$\chi^2(1) = 2.7$, $p > .1$; CFI = 0.99; RMSEA = 0.11; SRMR = 0.028]. The resultant latent WMC variable was then correlated with performance on the Raven's test, and the correlation was found to be $r = .67$ ($p < .0001$). This replicated the widely established finding that WMC correlates highly with measures of fluid intelligence. For example, in an extensive review of 14 data sets, Kane, Hambrick, and Conway (2005) found the correlation between WMC and *gF* to be .72; that is, general intelligence shared 50% of the variance with people's abilities to perform fairly simple memory tasks. The correlation observed with our battery thus meshes precisely with the existing body of literature. We conclude that our WMC battery has high external in addition to high internal validity.

USING THE TASK BATTERY

All four tasks are implemented by a set of MATLAB functions. Our programs have been designed and tested for

use with MATLAB 7.5 (R2007b) and later for Windows XP or Vista. Note that certain features of the battery will not work with earlier versions of MATLAB, and we caution against use of any MATLAB version older than 7.5.

Our programs require the Psychophysics Toolbox (Brainard, 1997; Pelli, 1997), which is freely available on the Web (http://psychtoolbox.org/). There are more than 13,000 installations to date of Psychtoolbox Version 3.0.8, and a similar or greater number of installations of the earlier version, 2.54. Psychtoolbox Versions 2.54 and 3.0.8 are typically considered to be incompatible with each other, but our code has been written to detect the installed version and to adapt to its requirements. The "touch and feel" of the tasks differs slightly between versions of Psychtoolbox, and we find the appearance under Version 2.54 more pleasing. We also recommend Version 2.54 because unlike the later version, it is more forgiving of hardware peculiarities. During our extensive use of the WMC battery, we have not encountered a single problem under Psychtoolbox 2.54. If you use the battery under Psychtoolbox Version 3.0.8, please test your installation thoroughly, by running though all demos that come with the toolbox, before using our WMC battery. Particular caution is advised on computers with multiple monitors or low-powered graphics cards: If there are any problems, in our experience they will be revealed during execution of the Psychtoolbox demos. In our experience, Version 3.0.8 requires at least 4 GB of RAM and 512 MB of display memory. Please refer to the Psychtoolbox Web site for further information about graphics hardware requirements for Version 3.0.8.

The WMC battery may also work with other operating systems (e.g., Apple Macintosh), although this has not been tested. Because the battery relies on the use of the Psychophysics Toolbox, we request that authors who use our battery cite not only this article in all relevant publications, but also the articles by Brainard (1997) and Pelli (1997).

Installation

MATLAB and the Psychophysics Toolbox must already be installed to run the task battery. A .zip file containing a "WMCBattery" folder with the programs for all tasks in various subfolders is available online (at www.cogsciwa.com, under the "Software" button on the main menu at the left). Follow the instructions at that Web site to download the .zip file.

The unzipped "WMCBattery" folder can be placed anywhere, but it must be made the current directory within MATLAB before the task battery can be run. (But the directory need not be added to MATLAB's search path.) The following instructions assume that "WMCBattery" is the current directory within MATLAB, that the Psychophysics Toolbox has been installed and is known to work, and that the MATLAB command window is accessible and ready to execute commands.

Running the Tasks

By default, the four tasks are presented in a set order (i.e., MU, OS, SS, and SSTM, as in the experiments above) by running the batch function WMCBattery.m. By default, the batch function presents the SS task in English, with the easy sentences used in Experiment 2 above. The Appendix contains information about how to change those defaults.

When the WMCBattery.m function is run, either by dragging the function name from MATLAB's "Current Directory" window into the Command Window or by calling the function from the Command Window directly, the experimenter must first enter the participant number (`SubjID`; an arbitrary integer in the range of 1–1,000 that will be used to identify data from all four tasks) before general instructions for the participant appear on the screen in the appropriate language. From then on, the participant presses the space bar to advance to the next page of instructions and/or to the next task.

Alternatively, each task can be run separately, and on its own, by calling the main task function from within MATLAB after the appropriate folder has been made the current directory in MATLAB. For example, the MU task is run by executing the MU.m program within the "MU" folder. When tasks are run individually, the participant number must be entered before the task begins. No instructions are shown to the participants if the tasks are run individually. Note that an escape to the MATLAB command window is possible from all tasks by pressing the "F12" key (although this is only effective when the program is awaiting input from the keyboard, and this does not work with older versions of MATLAB). We also recommend that no other programs are run during data collection; for instance, an incoming-mail sound would still be heard if an e-mail program were running in addition to MATLAB. It is also imperative that the data files (see below for details) are not open in any other application, since this would cause the programs to fail.

The instructions for the participants are contained within appropriately named folders within the "WMCBattery" folder. One folder contains English instructions and another one the Chinese instructions. Each page of instructions is contained within a separate file in .jpeg format. The file names must not be changed; however, the contents of the files are arbitrary and can be adjusted as needed. We have provided the Word files for the English instructions to facilitate editing; note that changes are not reflected unless they are exported to the appropriate .jpeg file.

When running the WMC battery on a Windows machine, we recommend that certain keys, such as the "Windows" key, be disabled (especially since the "Windows" key is close to one of the default response keys, "Z," on the keyboard). The "Windows" key calls up the Start menu, which will stall MATLAB. Disabling of keys can be achieved with a utility called Sharpkeys, which is available on the Web at www.codeplex.com/sharpkeys. We have used this program in conjunction with the present task battery, as well as with other programs, and found it to work satisfactorily.

In addition, when running the WMC battery with Psychtoolbox Version 3.0.8, the Windows taskbar should be set so that it does not automatically hide, because it may otherwise appear when the mouse is moved during the SSTM task. Right click on the taskbar, choose "Properties" from the menu, and turn off the "Auto-hide the taskbar" option.

Note that this is not necessary when running the WMC battery under Psychtoolbox Version 2.54.

Although all of the tasks operate quite quickly (e.g., entering the participant number is nearly immediately followed by the prompt to commence the experiment), there are two exceptions to this. First, when running the SS task for the first time, reading the sentences from the disk causes a notable delay at the beginning of the task. Second, at the end of the SSTM task, there is a notable time during which the data are scored and written to disk. The programs are not responsive to input during those times, and appropriate on-screen messages are shown.

Note that to minimize method-related variance, by default all of our programs use a single common random sequence for all participants. This can be altered, and a different random sequence can be used for each participant, by following the instructions in the Appendix.

The tasks can be modified in several other ways—for example, by changing the number of trials, list lengths, and so on. The Appendix provides detailed instructions for a number of likely modifications to the code.

DATA ANALYSIS

The data from the four tasks are contained within the "Data" folder within the "WMCBattery" folder. Data files are updated after each participant is tested and contain the data of all tested participants. Thus, duplicate use of a participant number does not cause overwriting of data but creates two copies of the same participant. It follows that existing data files should be moved or deleted before each new experiment.

The data can be analyzed using the statistics package of one's choice. We provide two scripts for preprocessing, one in the freely available R programming language (R Development Core Team, 2005) and one as a syntax file for SPSS. These two scripts achieve the same result, which is to preprocess the data and convert the results from all tasks into a format that is easily imported into standard statistics packages for further analysis (or which forms the basis for further analysis in R or SPSS, respectively). Running the R script additionally gives a summary of descriptive group statistics. Both scripts assume that the data files from all of the tasks are located in the same folder (although its location is arbitrary), and both assume that the data files are called SSTM.dat, OS.dat, SS.dat, and MU.dat. Those file names are not arbitrary but *mandatory*. The details of those scripts are as follows.

Data Preprocessing in R

The preprocessing script is in the same folder as the WMCBattery.m function. The script is called WMCBatteryAnalysis.r. In order to use the script, the location of the folder containing the data must be provided by changing the argument to the `setwd` command at the top of the script. To clarify, consider the following segment of the code:

```
# all data files (SSTM.dat, OS.dat,
# SS.dat, MU.dat) reside in the 'Data'
# folder within the 'WMCBattery' folder
# by default.
# The location of this folder (or any
# other location) must be provided to R
# in the following statement (note the
# double "\\" in the path):
# setwd("\\xxxxxxxxx")
```

The last line of the above excerpt tells R where to find the data; this must be replaced by the appropriate path specification, which will be unique to the particular installation. This is likely to be a path that ends in `WMCBattery\\Data`; for example, `C:\\Program Files\\MATLAB\\WMCBattery\\Data`. Note that R requires a double backslash, rather than a single backslash, in the path specifier. Note also that the path must be surrounded by double quotation marks. The R script will run (e.g., by opening it within the R GUI using the "File," "Open script . . ." menu sequence and then using the "Edit," "Run All" menu sequence in the R script editor) without further changes once the path to the data is specified.

The R script generates an ASCII text file, called WMCBattery.dat (located within the "Data" folder), which contains mean performance for each task and participant. Means for each participant are computed across trials using the partial-credit scoring described earlier in connection with our experiments (cf. Conway et al., 2005).

The data file thus conforms to the standard format expected by most statistical packages. To illustrate the format, the first four lines of a hypothetical but representative file are shown in Listing 1, located at the bottom of the page (note that the data values have been truncated for typographical reasons; in reality, all numbers are represented in full precision).

The first row contains the variable names, which are largely self-explanatory. Note that "pt" within a variable name refers to the processing component of a span task (either OS or SS). The subsequent rows contain the observations for all participants, with the participant number provided during data collection recorded in the first column. The results of the three experiments reported earlier were based on data files of this format, which were then converted into SPSS data sets for the structural equation model analysis (using AMOS).

A detailed description of the data file format for each individual task is provided in the Appendix. Note that knowledge of those details is not required to extract the standard summary data from the R script just described.

Listing 1

"SubjID"	"MUMean"	"OSMean"	"OSptMean"	"SSMean"	"SSptMean"	"SSTMMean"
1	0.61071	0.62539	0.91111	0.54031	0.82666	0.88753
2	0.33666	0.53785	0.9	0.50269	0.81333	0.85412
4	0.69777	0.69476	0.81111	0.51476	0.88	0.79164

Data Preprocessing in SPSS

The SPSS syntax file is called WMCbattery.read.sps and also resides in the same folder as the WMCBattery.m function. In order to use the syntax file, the location of the folder containing the data must be provided by changing the argument to the `CD` command at the top of the syntax file. Specifically, the first few lines of the syntax file are

```
%this should be WMCBattery

*** Syntax for reading the output of the
WMCbattery.

CD '/xxxxxx'.

**Replace '/xxxxxx' with the full path
**name of the directory where your data
**are located. Do NOT remove the quotes
**(' and ') and period at the end. After
**replacing the '/xxxxxx', click on
**"Run" followed by "All."
```

The line beginning with `CD` in the above excerpt must be replaced by the appropriate path specifier. This is likely to be a path that ends in `WMCBattery/Data`—for example, `C:/Program Files/MATLAB/WMCBattery/Data`. It is imperative that the single quotation marks around the path name are retained and that there is a period (".") at the end of the line. The SPSS syntax file will run without further changes once the path to the data is specified.

The SPSS syntax file creates an SPSS spreadsheet in the same format as the data file produced by the preceding R script: Each line corresponds to 1 participant, and the seven columns correspond to the participant number, followed by the mean percent correct values of the four tasks and the mean percent correct values of the processing components of the span tasks, in the same order as produced by the R script.

AUTHOR NOTE

Preparation of this article was facilitated by a Discovery Grant from the Australian Research Council and an Australian Professorial Fellowship to the first author. We thank Chung Yu Wang for programming assistance. We also thank Nicole Cruz, Claudia von Bastian, Stewart Craig, Charles Hanich, Abel Oh, Briony Swire, David Tang, Cameron Tieleman, Janice Wong, Chang-Hao Kao, Hsuan-Yu Lin, and May-Miao Liu for serving as testers who successfully installed and operated the software (including analysis scripts) based on an earlier draft of this article. Correspondence concerning this article should be addressed to L.-X. Yang, Research Center of Mind, Brain, and Learning and Department of Psychology, National Cheng-Chi University, No. 64, Sec. 2, ZhiNan Rd., Wenshan District, Taipei City 11605, Taiwan (R.O.C.) (e-mail: lxyang@nccu.edu.tw).

REFERENCES

Bayliss, D. M., Jarrold, C., Gunn, D. M., & Baddeley, A. D. (2003). The complexities of complex span: Explaining individual differences in working memory in children and adults. *Journal of Experimental Psychology: General*, **132**, 71-92.

Beilock, S. L., & Carr, T. H. (2005). When high-powered people fail: Working memory and "choking under pressure" in math. *Psychological Science*, **16**, 101-105.

Brainard, D. H. (1997). The Psychophysics Toolbox. *Spatial Vision*, **10**, 433-436.

Conway, A. R. A., Kane, M. J., Bunting, M. F., Hambrick, D. Z., Wilhelm, O., & Engle, R. W. (2005). Working memory span tasks: A methodological review and user's guide. *Psychonomic Bulletin & Review*, **12**, 769-786.

Daneman, M., & Carpenter, P. A. (1980). Individual differences in working memory and reading. *Journal of Verbal Learning & Verbal Behavior*, **19**, 450-466.

Daneman, M., & Merikle, P. M. (1996). Working memory and language comprehension: A meta-analysis. *Psychonomic Bulletin & Review*, **3**, 422-433.

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, & Cognition*, **36**, 170-189.

Engle, R. W., Tuholski, S. W., Laughlin, J. E., & Conway, A. R. A. (1999). Working memory, short-term memory and general fluid intelligence: A latent variable approach. *Journal of Experimental Psychology: General*, **128**, 309-331.

Fan, X. (2003). Using commonly available software for bootstrapping in both substantive and measurement analyses. *Educational & Psychological Measurement*, **63**, 24-50.

Friedman, N. P., & Miyake, A. (2004). The reading span test and its predictive power for reading comprehension ability. *Journal of Memory & Language*, **51**, 136-158.

Hertzog, C., Dixon, R. A., Hultsch, D. F., & MacDonald, W. S. (2003). Latent change models of adult cognition: Are changes in processing speed and working memory associated with changes in episodic memory? *Psychology & Aging*, **18**, 755-769.

Hofmann, W., Gschwendner, T., Friese, M., Wiers, R. W., & Schmitt, M. (2008). Working memory capacity and self-regulatory behavior: Toward an individual differences perspective on behavior determination by automatic versus controlled processes. *Journal of Personality & Social Psychology*, **95**, 962-977.

Kail, R. (2007). Longitudinal evidence that increases in processing speed and working memory enhance children's reasoning. *Psychological Science*, **18**, 312-313.

Kane, M. J., Brown, L. H., McVay, J. C., Silvia, P. J., Myin-Germeys, I., & Kwapil, T. R. (2007). For whom the mind wanders, and when: An experience-sampling study of working memory and executive control in daily life. *Psychological Science*, **18**, 614-621.

Kane, M. J., Conway, A. R. A., Hambrick, D. Z., & Engle, R. W. (2007). Variation in working memory capacity as variation in executive attention and control. In A. R. A. Conway, C. Jarrold, M. J. Kane, A. Miyake, & J. N. Towse (Eds.), *Variation in working memory* (pp. 21-46). Oxford: Oxford University Press.

Kane, M. J., Hambrick, D. Z., & Conway, A. R. A. (2005). Working memory capacity and fluid intelligence are strongly related constructs: Comment on Ackerman, Beier, and Boyle (2005). *Psychological Bulletin*, **131**, 66-71.

Kyllonen, P. C., & Christal, R. E. (1990). Reasoning ability is (little more than) working-memory capacity? *Intelligence*, **14**, 389-433.

Lee, J., & Park, S. (2005). Working memory impairments in schizophrenia: A meta-analysis. *Journal of Abnormal Psychology*, **114**, 599-611.

Lépine, R., Barrouillet, P., & Camos, V. (2005). What makes working memory spans so predictive of high-level cognition? *Psychonomic Bulletin & Review*, **12**, 165-170.

Oberauer, K. (1993). Die Koordination kognitiver Operationen: Eine Studie über die Beziehung zwischen Intelligenz und "working memory" (The coordination of cognitive operations: A study on the relation of intelligence and "working memory"). *Zeitschrift für Psychologie*, **201**, 57-84.

Oberauer, K. (2005a). Binding and inhibition in working memory: Individual and age differences in short-term recognition. *Journal of Experimental Psychology: General*, **134**, 368-387.

Oberauer, K. (2005b). The measurement of working memory capacity. In O. Wilhelm & R. W. Engle (Eds.), *Handbook of understanding and measuring intelligence* (pp. 393-408). Thousand Oaks, CA: Sage.

Oberauer, K., Süß, H.-M., Schulze, R., Wilhelm, O., & Wittmann, W. W. (2000). Working memory capacity—facets of a cognitive ability construct. *Personality & Individual Differences*, **29**, 1017-1045.

Oberauer, K., Süß, H.-M., Wilhelm, O., & Wittmann, W. W. (2003). The multiple faces of working memory: Storage, processing, supervision, and coordination. *Intelligence*, **31**, 167-193.

Pelli, D. G. (1997). The Video Toolbox software for visual psychophysics: Transforming numbers into movies. *Spatial Vision*, **10**, 437-442.

RAVEN, J. C. (1989). *Standard progressive matrices, 1956 revision*. Victoria: ACER.
RAYKOV, T. (1997). Estimation of composite reliability for congeneric measures. *Applied Psychological Measurement*, **21**, 173-184.
R DEVELOPMENT CORE TEAM (2005). R: A language and environment for statistical computing [Programming language]. Vienna: R Foundation.
SALTHOUSE, T. A., BABCOCK, R. L., & SHAW, R. J. (1991). Effects of adult age on structural and operational capacities in working memory. *Psychology & Aging*, **6**, 118-127.
SCHMADER, T., JOHNS, M., & FORBES, C. (2008). An integrated process model of stereotype threat effects on performance. *Psychological Review*, **115**, 336-356.
SCHMIEDEK, F., HILDEBRANDT, A., LÖVDÉN, M., LINDENBERGER, U., & WILHELM, O. (2009). Complex span versus updating tasks of working memory: The gap is not that deep. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, **35**, 1089-1096.
SHUTE, V. J. (1991). Who is likely to acquire programming skills? *Journal of Educational Computing Research*, **7**, 1-24.
SÜß, H.-M., OBERAUER, K., WITTMANN, W. W., WILHELM, O., & SCHULZE, R. (2002). Working-memory capacity explains reasoning ability—and a little bit more. *Intelligence*, **30**, 261-288.
TURNER, M., & ENGLE, R. (1989). Is working memory capacity task dependent? *Journal of Memory & Language*, **28**, 127-154.
UNSWORTH, N., HEITZ, R. P., SCHROCK, J. C., & ENGLE, R. W. (2005). An automated version of the operation span task. *Behavior Research Methods*, **37**, 498-505.
WITTMANN, W. W. (1988). Multivariate reliability theory: Principles of symmetry and successful validation strategies. In J. R. Nesselroade & R. B. Cattell (Eds.), *Handbook of multivariate experimental psychology* (pp. 505-560). New York: Plenum.

NOTES

1. We thank Randy Engle, Mike Kane, Andy Conway, and Akira Miyake for many discussions on how to optimize the protocol of complex-span tasks.

2. These keys represent the default setting but can be altered to suit different keyboard layouts as detailed below.

3. Given the constraint of a positive result, there were more additions than subtractions despite the random draw.

4. The analyses reported below contain data from all of the participants. We repeated the analysis with the oldest (62 years) and also with the second-oldest (45 years) participants excluded. In the latter case, the maximum age in the sample was 39. Both of these additional analyses yielded virtually unchanged results; in particular, the loadings of the manifest variable onto the common factor differed by no more than .01 from those obtained with all of the participants. We therefore only report the analysis with all participants included.

APPENDIX
Data Format and Programming Details

This Appendix contains further detailed information about the format of the data files for each task and ways in which the experimental parameters can be modified. None of the material in this Appendix is required to run the battery in its standard format.

Data Format for Each Task

Data from all tasks are recorded into the "Data" folder located within the "WMCBattery" folder and are identified by the name of the task. For example, the MU task writes to a file called MU.dat. The SSTM task creates additional data files as detailed below. All files are in ASCII (text) format.

The MU task. The following shows an excerpt from a hypothetical but representative MU.dat file.

```
1    3    821000    0    1    1   -1   -1   -1
1    4    574600    1    1    0    1   -1   -1
1    5    481000    0    0    0   -1   -1   -1
...
1   15    135700    0    0    0    0   -1   -1
1   16    397600    1    1    1    1   -1   -1
1   17    651380    1    0    1    0    0   -1
```

The first column contains the participant number and the second column the trial number (ranging from 3 to 17 in the present case because the first two practice trials are not recorded).

Columns 3–8 contain the responses (i.e., typed digits; padded with 0s for set size < 6), and Columns 9–14 indicate whether a response is correct (1) or incorrect (0). For set size < 6, the columns are padded with −1s. In this instance, during final recall of the first test trial (Trial 3), this participant typed 8, 2, and 1 (for Frames 1, 2, and 3, respectively), although 9, 2, 1 would have been the correct responses. Hence, the score on this trial was recorded as 0, 1, 1.

Note that Column 8 is always 0, and Column 14 is always −1 in this example, because set size never exceeded 5 in this experiment. We show below how those parameters can be altered.

The OS task. The data are written into the OS.dat file. A hypothetical excerpt is shown below. Note that records are too long to fit on a single line of text and hence wrap around. For additional clarity, records are separated by additional white space that is not present in the actual data file:

```
1   1   4   XMDN%%%%   XMDN%%%%   2.093 0.363 0.476 0.396 -1.000 -1.000
  -1.000 -1.000   1 0 0 1 -1 -1 -1 -1   1 0 1 1 -1 -1 -1 -1   2.807 1.408 1.825
   1.347 -1.000 -1.000 -1.000 -1.000

1   2   6   PKFGSJ%%   PKFGSJ%%   1.682 1.052 0.268 0.196 0.348 0.788
  -1.000 -1.000   0 0 0 1 1 1 -1 -1   1 1 1 1 0 1 -1 -1   1.989 2.830 1.336
   1.914 2.490 2.661 -1.000 -1.000

1   3   6   JKBVFD%%   JKBFDJ%%   1.446 0.082 0.213 0.339 0.188 7.611
  -1.000 -1.000   0 1 1 0 1 0 -1 -1   1 1 1 1 1 0 -1 -1   0.000 2.194 1.004
   2.197 0.000 2.511 -1.000 -1.000
```

APPENDIX (Continued)

Again, the first column codes for participant number, the second for trial number. Column 3 indicates the list length. Following the list length column, there are six additional sets of columns, each of which comprises eight entries. The first set records the letters presented on the screen (in the above instance, the first trial involved memoranda X, M, D, and N). Note that for list lengths < 8, the remaining columns are padded with "%" symbols. The second set records the participant's typed recall responses (the first trial above involved perfect recall, hence the letters X, M, D, and N are shown a second time, again padded with "%"). The third set of columns records the times (in seconds) taken to recall the letters, padded with -1. The fourth set of columns records whether or not the equations presented in between-list items were correct or not (1, *correct*; 0, *incorrect*), whereas the fifth set scores the responses to the equations (1, *correct*; 0, *incorrect*). The final set of columns reports the time taken for the equation judgments (in seconds). All numeric columns are padded with -1 for list lengths < 8.

The SS task. The data are written into the SS.dat file, which has the same structure as the data file for the OS task. Note that the name of this file, SS.dat, is identical, irrespective of which version (easy, hard, or Chinese) is being run. This facilitates analysis, but it requires some care if multiple versions of the SS task are conducted at the same time.

The SSTM task. The overall scores of all participants are recorded in the SSTM.dat file, which is located in the same location as the summary files from all other tasks (viz. in the "Data" folder within the "WMCBattery" folder). The format of this file is as follows:

```
SubID   Score    Date     Time   240
  5      195  2008/2/1   16:16
  2      205  2008/3/10  10:33
  4      190  2008/3/10  12:33
  8      225  2008/3/10  15:33
 10      214  2008/3/11  10:35
 11      222  2008/3/11  11:36
```

The first row is a header that specifies variable names and, in the last position, the maximum attainable score. All subsequent rows contain the data in the following format: The first column gives the participant number, the second provides the overall SSTM score (the sum of all trial scores; the maximum score is calculated as the number of dots presented throughout the experiment times 2; in the current version, with six trials per set size and set size ranging from two to six, the maximum score is 240). The last two columns give the date and time of testing.

The SSTM task provides further information about performance details that are available in separate files described in the next section.

Modifying Task Parameters

This section explains some key variables within the MATLAB code for the various tasks that can be modified to adjust the experiment to particular needs. All these changes are made by opening the appropriate MATLAB program file (with a "*.m" extension) by double-clicking on the file in MATLAB's current directory window. This will open the program in the MATLAB editor, and all changes must be saved to disk before they take effect the next time the program is run.

Changing global task parameters. By default, the four tasks are presented in a set order (i.e., MU, OS, SS, SSTM, as in the experiments above) by running the batch function WMCBattery.m. By default, the batch function presents the SS task in English, with the easy sentences used in Experiment 2 above. This can be altered by editing a line at the beginning of WMCBattery.m as shown below.

```
%determine which version of sentence span. Choices are:
% easy
% hard
% chinese
SSversion 5 'easy';
```

That is, the variable `SSversion` determines the choice of SS task, and it can be set to any one of the three choices (`easy`, `hard`, or `chinese`; note that those are case sensitive). Note that we did not provide an interactive interface for the choice of version of the SS task to reduce the potential for human error during data collection. Note also that for the Chinese version to operate correctly under a non-Chinese version of Windows, the Chinese character set must be installed (via the Windows Control Panel's "Regional and Language Options") and that Chinese must be selected as both the standard language ("Regional Options" tab) and the "Language for non-Unicode programs" ("Advanced" tab) for MATLAB to display the sentences correctly.

Several other modifications can be readily achieved by editing the batch function WMCBattery.m; for example, the order of tasks can be altered by reordering the corresponding segments of code within the `case` block within WMCBattery.m.

The WMCBattery.m file also determines the response keys for the *yes–no* decisions in the span tasks. By default, they are set to "/" and "Z" for *yes* and *no*, respectively, but they can be changed to any other key to accommodate different keyboard layouts or preferences. The following code excerpt, taken from the beginning of WMCBattery.m, clarifies how this can be achieved.

APPENDIX (Continued)

```
%determine response keys
expinfo.yeskey = '/';        %e.g., for right arrow: 'right';
expinfo.nokey = 'Z';         %e.g., for left arrow: 'left';
```

To change the response keys, the two assignment statements must be changed; for example, to use the right and left arrow keys, the first statement would read `expinfo.yeskey = 'right';`. The permissible values for the keys are the strings returned by the PsychToolbox function `KbName` (for keys that have two possible outputs, only the first one must be used; e.g., `'/'` for `'/?'`). Note that the experimental instructions refer to the default response keys, so any change in response keys here must be accompanied by a change to the instructions. (If the SS and OS tasks are run in stand-alone mode, the corresponding statements at the top of the SS.m and OS.m files can be changed in an analogous way—i.e., by assigning different strings to `expinfo.yeskey` and `expinfo.nokey`.)

Finally, if a different random sequence is desired for each participant (as opposed to a common sequence for all participants), this can be achieved by making the seed for the random sequence dependent on participant numbers. All four tasks contain a statement of the type `"seed = 54369;"` (where the actual number differs between tasks) which needs to be changed to `"seed = 54369 + subject;"`. This can be achieved by editing the lines in the relevant files, as is shown in Table A1.

Table A1
Location of Statements to Make Random Sequence Dependent on Participant Number

Task	File Location and Name	Line Number[a]
MU	`MU\mu.m`	44
SS	`SS\ss.m`	74
OS	`OS\os.m`	60
SSTM	`SSTM\RandLoc2.m`	26

[a]Each line shown here contains a statement of the form `"seed = 12345;"`, where the number differs between programs. This needs to be changed to `"seed = 12345 + subject;"` to ensure that each participant receives a different unique random sequence.

The MU task. The main function for this task is MU.m. The top of the program contains placeholders for relevant timing parameters:

```
expinfo.iii = 0.25;          %inter-item interval (all times in sec)
expinfo.iti = 1;             %inter-trial interval
expinfo.itesti = 1;          %inter-test interval
expinfo.startinterval = 1;   %interval between fixation cross and first start value
expinfo.prestime = [1, 1.3]; %presentation time for start values / for operations
expinfo.PauseAfterBreak = 2;
```

The beginning of the file also defines the nature and the number of trials. First, the number of trials for practice and for the experiment are specified in two variables. Then, two arrays are defined that spell out set size (i.e., number of frames on the screen) and the number of operations per trial, respectively:

```
Npracticetrials = 2;
Ntesttrials = 15;
setsize4mt = [2 3  3 4 3 4 5 4 5 3 5 3 3 5 4 4 5];
opnum4mt =   [4 3  3 4 6 5 5 3 2 2 3 5 4 4 6 2 6];
```

In the above example, the two practice trials involve two and three frames, respectively, and four and three updating steps, respectively. The remaining experimental trials involve three, four, three, and so on frames and three, four, six, and so on updating steps. Subject to the constraint that the number of frames and steps must not exceed six, any arbitrary sequence of trials can be defined within these arrays.

The OS task. Again, basic settings of the OS task can be changed by editing the top of main function OS.m:

```
Npracticetrials = 3; %must be ≤ length of pll
pll = [ 3 4 5 ];     %list lengths for practice trials
listlength = [4:8];  %array of list lengths, each replicated Ntpercond times
Ntpercond = 3;       %trials per condition (i.e., list length)
...
expinfo.iii = 0.1;                %inter-item interval (all times in sec)
expinfo.Presentationduration = 1; %presentation time for to-be-remembered stimuli
expinfo.StudyTestDelay = .5;
```

APPENDIX (Continued)

```
expinfo.InterTrialInterval = 0.5;
expinfo.responsevisible = 0.2;    %time each response letter is visible on screen
expinfo.PauseAfterBreak = 2;
expinfo.ptDuration = 3;
...
```

The maximum presentation time for the processing task (i.e., the equations) can be altered by changing the value of `expinfo.ptDuration`. Note that the maximum list length must not exceed eight.

The variable `expinfo.responsevisible` determines how long a response remains on screen after the letter has been entered; during that time, no further responses are recorded. If this presents a problem because participants type too rapidly, the time can be shortened. The interkeystroke delay that is due to `expinfo.responsevisible` can also lead to very brief response latencies for all but the first item if the participants type quickly.

Changing the difficulty of the equations requires editing the auxiliary functions maketrueop.m and makefalseop.m. For example, the range from which operands are drawn (default, 1–10) can be changed:

```
oprnd1 = num2str(ceil (rand*10)); %sets ceiling for the 1st operand (here 10)
oprnd2 = num2str(ceil (rand*10)); %sets ceiling for the 2nd operand (here 10).
```

The SS task. The basic settings of the SS task are identical to those of the OS task. The declaration part at the top of the program is nearly identical to that of the OS task.

There are three versions of the SS task, each with a different type of sentence. The default version that is called when running the batch function (WMCBattery.m) uses easy sentences (as in Experiment 2 in this article), but those can be replaced by more difficult sentences (as in Experiment 1) or Chinese sentences (as in Experiment 3). We noted above how this switch can be achieved in WMCBattery.m.

When the SS task is run outside the batch function, the version must be provided as an argument in order to override the *easy* default. Thus, for the Chinese version, the task must be invoked by typing `SS('chinese')` at the MATLAB command window (with "SS" being the current directory), and the difficult version is invoked by `SS('hard')`. The argument can be omitted when the *easy* default is desired.

In all versions, the sentences are drawn from a version-specific (and appropriately named) Excel file that contains four columns and that resides in the SS folder. Meaningful/true sentences are drawn from the first column, meaningless/false sentences are drawn from the second column. Columns 3 and 4 contain meaningful and meaningless sentences for the practice phase. (Note that in the actual test, the two kinds of sentences are drawn equally often. In the practice phase, there is a bias toward drawing meaningful sentences; hence, the third column must contain more sentences than would be needed for half the practice phase.) Sentences can be changed, and more sentences added, by editing the respective Excel spreadsheet.

The SSTM task. The main function for this task is SSTM.m. Again, basic parameters—though not the number of trials and set sizes—can be changed at the top of the program:

```
Time.Dot=.9;          % Individual dot presentation time
Time.PostDot=.1;      % Interval between dots
Time.Result=1;        % Presentation time for retention interval mask
Time.Alert=1;         % Fixation cross presentation time
Time.PostTrial=0;     % Interval between disappearance of dots and RI mask
Time.PostAlert=.5;    % Interval between fixation cross and first dot
...
NumT.Pract=2;         % Number of practice trials (Must be 2)
NumT.Test=30;         % MUST NOT BE CHANGED
```

Although calculations of spatial parameters assume a default screen resolution of 800 × 600 pixels, the task works under a range of resolutions without editing.

Data scoring in the SSTM task is more complex than in the other tasks. In consequence, there is a delay after the task has finished, during which the data are scored and recorded. When the "Experiment Over . . ." message appears on the screen, the data scoring process begins; this may take up to 20 sec. Note that the program will not respond to any keypresses during this period. An additional "Data Recorded . . ." message is then displayed on the screen, which requires the space bar to be pressed before the program terminates. For this task, a further subfolder is created within the "Data" folder, called "SSTMDetailed," which contains additional text files with detailed data for each participant (e.g., Sub-01.txt for Participant 1, and so on). None of the files within this additional folder are required for the standard summary analysis performed by our R script or by the SPSS syntax file; however, they provide detailed information that may be helpful in other situations.

The Sub-xx.txt file consists of two parts. The first part records the trial-based data; each line corresponds to a single trial:

```
ID    Trial    Score    RT    NumDot
 1      1        4    1.741     2
```

The first column records the participant number (`ID`), and the second shows the trial number. The third column contains the participant's score on the trial (the sum of the scores for each dot), the fourth shows the reaction

APPENDIX (Continued)

time (for reconstructing all dots; in seconds), and the last column shows the number of dots on this trial (`NumDot`; i.e., the set size, presently ranging from two to six).

The "SSTMDetailed" folder contains a MATLAB program, called ReadSSTM.m, which combines the initial parts of all individual participants' files into a single text file, called SSTMCombined.dat, that contains the same columns as the individual files just described. The program requires that the "SSTMDetailed" folder be made the current directory in MATLAB, and the final output file will be located within the "Data" folder.

The second part of the Sub-xx.txt file contains data concerning the actual response dots; each line represents one reconstructed dot:

```
%changed InTrial\# to InTrial#
Dot   Res   Transf-Res   Ans   Score   InTrial#
 1    3 4      3 4       3 4     2        1
```

The first column is an order number of generated dots in a given trial (`Dot`; e.g., 1 indexes the first generated dot), the second and third columns give the *x*–*y* matrix coordinates of that response dot (`Res`; e.g., [3 4] indicates that the dot was placed in a cell in the third row and fourth column of the matrix), and the sixth and seventh columns give the studied dot coordinates (`Ans` for a correct answer; e.g., [3 4] indicates that a dot was studied in the cell with [*x y*] coordinates 3 and 4).

To understand the fourth and fifth columns, one needs to know that scoring does not depend on absolute positions (i.e., whether a dot was reproduced in the exact same cell that it was studied in), but only on the spatial relationship between dots. Therefore, the generated dot pattern is virtually moved to the position where it best matches the study pattern. The fourth and fifth columns index the cell to which a given dot was moved in order to maximize the match of the whole pattern (`Transf-Res`; here, [3 4] indicates that this dot was not moved). The score for a given dot is given in Column 8. An exact match (i.e., a response dot reproduced in the exact study location, possibly after movement) scores 2 points, and a response dot that is off by one cell (possibly after the virtual movement) scores 1 point. Finally, the last column indexes the trial on which the dot was presented (`InTrial#`).

(Manuscript received September 23, 2009;
revision accepted for publication January 6, 2010.)