SUPPLEMENTARY INFORMATION

Singular Value Decomposition (SVD)

The SVD procedure decomposes an arbitrary *m*-by-*n* matrix into three matrices. As illustrated in Figure S1, the matrix *C*, which is the input data for SVD analysis, can be represented as a product of three matrices, *U, Σ*, and *V´*, whose dimensions are indicated in Figure S1. The matrices *U* and *V´* are orthogonal. *Σ* is a diagonal matrix with non-negative singular values on its diagonal. The size of dimension *r* of three matrices can be less than or equal to the smaller of *m* and *n* of matrix *C*, which is *n* in our example. If *r* is equal to *n*, then the production of three matrices, *U*, *Σ*, and *V´*, reproduce the original matrix, *C*, exactly. If *r* is less then *n*, then the production of three matrices is said to approximate the original matrix. This qualifies SVD as a dimensionality reduction method as it has been used for this purpose in many studies (Alter, et al., 2000; Landauer, 1999; Landauer, McNamara, Dennis, & Kintsch, 2007). The eigenvalue decomposition, a mathematical basis for factor analysis, becomes a special case of SVD when the matrix *C* is symmetric and positive definite. In this case, the components of *Σ* become the eigenvalues of *C*, and *U* and *V* are the same set of eigenvectors of *C*.

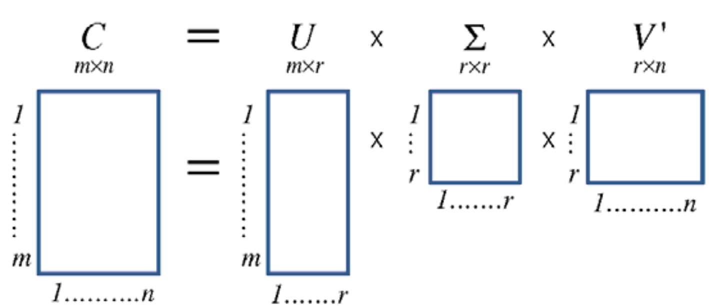


Figure S1. Matrix representation of SVD.

Let us translate this introduction into a category fluency analysis. (a simplified example of SVD analysis is presented at the end of this section). The matrix *C* now represents a word-by-protocol (or subject) matrix, whose entry, c_{ij}, becomes 1 if subject *j* says the word *i,* and 0 otherwise, which makes the *C* a binary matrix. Sometimes researchers use weighting function(s) to improve the result of SVD but it is not a required process (Quesada, 2007). In the current study, raw binary data were used without using any weighting functions, mainly because the group differences we seek in the current study seem to emerge clearly without weightings.

As reported by many studies (Giovannetti, et al., 2003; Troyer, et al., 1997), the exemplars that people give on fluency tasks often form semantic clusters or subcategories of words that share one or more properties. This implies that the patterns of binary values for semantically related words (i.e., row vectors) would be similar in the matrix *C*. Thus, we would expect the matrix *U*, a result of SVD procedure, to include *m* different word vector points that form semantically meaningful clusters in *r* vector space (or *r* number of different properties) if there are any systematic patterns in *C*. The matrix V', which represents the vector space of protocols, is not relevant to the current study since we analyze two homogenous groups (SZ and NC) separately. Note that the value *r* is what a researcher needs to determine. It is not automatically determined by SVD. Usually, choosing a specific *r*-dimension depends on the interpretability of the cluster outputs and the type of data (Quesada, 2007). Choosing *r* in advance does not affect the actual solutions one gets. That is, the r of 20 and 40 will give exactly the same solutions up to 20^{th} dimensions, although they are normalized. But the additional 20 (dimension 21 to 40) dimensional information will be available only from 40 dimensional solutions. Another noteworthy point is that the first dimension of any SVD solution usually is determined by how frequently words occur in whole dataset (Hu et al., 2003). This is a mathematical consequence of the analysis applied to frequency matrices.

As explained briefly in the text, the major difference between MDS and SVD procedures is that the input matrix for SVD does not include any type of similarity measure. In principle, all possible words generated in a category fluency task can be analyzed using SVD, but they are not all equally informative. Also, the resulting Euclidian distance between two word positions in *r*-dimensional space obtained via SVD cannot be interpreted the same way it is in MDS. The cosine of angle between two word vectors is a better measure of similarity than is Euclidian distance (Landauer & Dumais, 1997). A cosine value can be interpreted as a clustering measure between any pair of words. A cosine close to 1.0 indicates that people frequently generate the two words together. A value close to 0.0 implies that two words are generated more independently of each other (Landauer, 2007), assuming that SVD solution is valid.

A Simplified Example of SVD Analysis

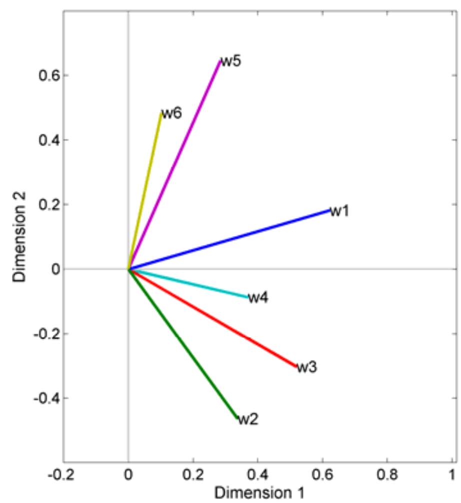
Here we present a SVD analysis of a make-up dataset. Although simplified, it gives an intuitively clear result of SVD analysis which can be readily apprehensible from the input data for SVD. The input data is a 6 by 4 binary matrix, which represents 6 different words named by 4 different subjects. From the matrix below (Figure S2), we see that the subject 1 (s1) named first three words but not the other words. Similarly, the fourth column tells us that the subject 4 (s4) named the first, fifth and sixth words during a fluency test. Note that the order of columns or rows do not matter in SVD.

$$\begin{matrix} w1 \\ w2 \\ w3 \\ w4 \\ w5 \\ w6 \end{matrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

s1 s2 s3 s4

Figure S2. Simplified input matrix for demonstration.

From this matrix, we can reasonably guess that a vector representing the first word (w1) would be located in a neutral position as a result of SVD analysis since it co-occurs with all other words. Also, the vectors w2 and w5 would be separated very wide since they are mutually exclusive. In general, there seem to be two major clusters of vectors, one with w2, w3, and w4 and the other with w5 and w6. The result of SVD is presented in Figure S3, which shows word



vectors in U matrix (see Figure S1) positioned in the first 2 dimensional space (i.e., r = 2).

Figure S3. Word vectors represented in 2-dimensional vector space as a result of SVD analysis.

As expected, we see two major clusters in Figure S2. That is, w2, w3, and w4 form on cluster and w5 and w6 form another one along the dimension 2 based on the angles between these vectors. One interesting thing is that the vector angle between w4 and w1 is smaller than that between w3 and w1, which is counter-intuitive since w3 co-occurs with w1 more frequently than w4 does (see Figure S2). When 3 dimensional space is considered (dimensions 1-3), however, the angle between w1 and w4 is much greater [87.1°; cos(87.1)=0.05] than the angle between w1 and w3 [58.4°; cos(58.4)=0.52]. Considering the simplicity of the example, this result critically demonstrates the importance of examining high-dimensionality of clustering analysis in verbal fluency.

Word Rank Table

Frequency ranks of animal names and supermarket items. Words are sorted by the frequencies calculated from a large verbal fluency database (All; n=780) including various patients groups and normal controls, some of which are used for current study (SZ; n=102 and NC; n=109).

Rank	Supermarket items	All	SZ	NC	Rank	Animals	All	SZ	NC
1	Milk	582	60	84	1	cat	719	83	98
2	Bread	503	44	70	2	dog	714	82	99
3	Cheese	427	36	65	3	lion	610	69	95
4	Eggs	369	36	64	4	tiger	550	59	80
5	Apples	324	36	35	5	elephant	519	54	76
6	Meat	310	31	42	6	giraffe	406	56	58
7	Chicken	297	35	47	7	bear	404	44	66
8	Lettuce	292	31	38	8	horse	396	52	54
9	Cereal	288	30	48	9	zebra	379	51	57
10	ice cream	282	36	47	10	monkey	347	54	52
11	Oranges	275	35	32	11	snake	343	55	48
12	Soda	254	39	41	12	cow	339	42	46
13	Tomatoes	230	28	29	13	bird	306	43	46
14	Vegetables	217	22	30	14	pig	235	28	26
15	Potatoes	215	15	31	15	deer	207	19	35
16	Butter	213	18	27	16	fish	206	29	30
17	Fish	211	31	31	17	mouse	194	23	23
18	candy	208	26	35	18	rabbit	192	17	33
19	bananas	202	16	22	19	hippopotamus	190	32	28
20	fruit	191	23	29	20	rhinoceros	176	22	20
21	carrots	177	15	19	21	rat	169	21	20
22	cookies	175	24	21	22	alligator	162	24	28
23	cake	169	24	24	23	squirrel	160	13	21
24	onions	166	13	21	24	sheep	153	14	30
25	steak	162	26	20	25	chicken	153	16	20
26	sugar	157	18	21	26	gorilla	153	25	26
27	yogurt	145	9	17	27	whale	147	22	22
28	soup	138	16	12	28	goat	140	19	20
29	juice	133	12	18	29	leopard	138	13	17
30	pears	128	12	11	30	eagle	115	13	16
31	beef	126	11	18	31	crocodile	111	19	20
32	toilet paper	126	9	21	32	fox	109	7	17
33	ham	124	13	18	33	kangaroo	107	11	12
34	bacon	124	14	26	34	shark	104	24	9
35	potato chips	124	24	21	35	lizard	102	18	16
36	coffee	122	12	20	36	raccoon	102	8	10
37	celery	122	7	15	37	ape	93	14	10
38	turkey	122	15	11	38	dolphin	88	13	12
39	paper towels	119	9	17	39	duck	87	11	7
40	lunch meat	110	8	29	40	donkey	85	16	11

SVD analysis on even and odd numbered NCs

The goal of this analysis is to demonstrate the stability of clusters by NCs we reported in the paper. One hundred nine healthy controls were divided into two even- and odd-numbered sub-groups. The results of SVD analysis is presented in Figure S3 and S4, each shows 20 animal names (rank 1-20 for Figure S3 and 21-40 for Figure S4).

Figure S4. Top 20 animals clusters of even and odd numbered NC.

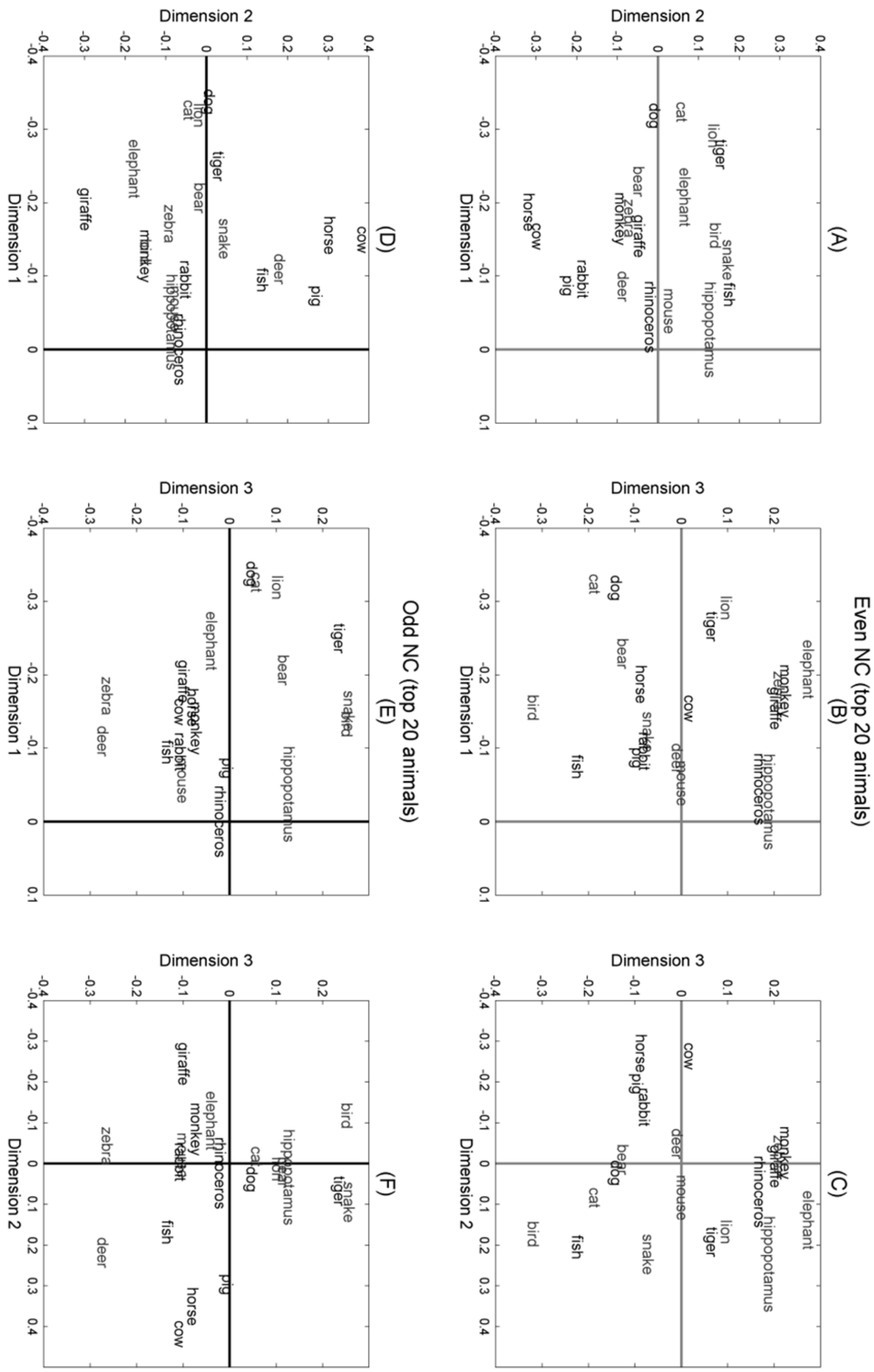
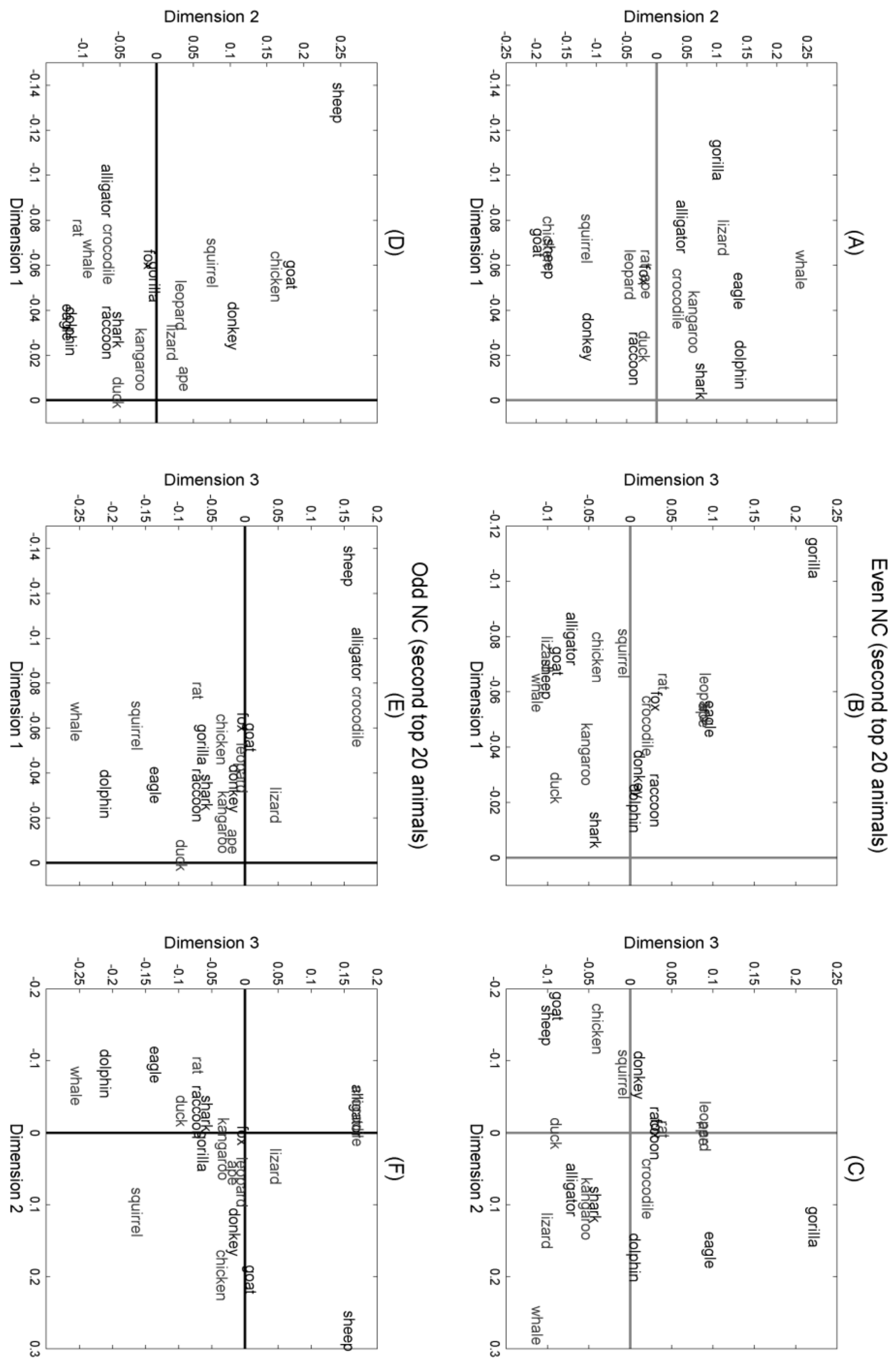


Figure S5. Second top 20 animal clusters of even and odd numbered NC.



List of software programs and files for 2- and 3-D dimensional plots and cosine plots

The following programs and files are written by the authors for readers to freely examine various aspects of SVD results not reported in the paper due to space limitation (available by request). These programs are designed to run on PC (will not work on Apple computers). Windows XP and Vista OSs have been tested and confirmed to work with these programs. Windows 7 may or may not work, depending on the computer system configurations.

- 'MCRInstaller.exe': A runtime library needed to run Matlab programs for Windows. This needs to be installed on user's PC in advance to run the programs (user may skip installation of this library if Matlab is already installed on user's computer). This program is proprietary (Mathworks Inc.) and subject to limitation in its usage, although there is no charge for using this program. 'MCRInstaller.exe' can be used by readers without any charge <u>only</u> to run the programs that we provide here. It <u>cannot</u> be used for other purposes.
- 'instruction.docx': a short instruction for the programs
- 'New_LSA_data_all_variables': data file. Needed for all programs to run
- 'run3d_bw.exe': 3-D plot of 2, 3, and 4 dimensions for animal names (NC and SZ)
- 'run3d_bw_sup.exe': 3-D plot of 2, 3, and 4 dimensions for supermarket items (NC and SZ)
- 'cos_valueplot.exe': cosine measure plot
- 'dimensionProfile.exe': 2-D plot of any combinations of two dimensions out of 25

References

Hu, X., Cai, Z., Franceschetti, D., Penumatsa, P., Graesser, A. C., Louwerse, M. M., . . . TRG. (2003). *LSA: The first dimension and dimensional weighting.* Paper presented at the Proceedings of the 25th Annual Conference of the Cognitive Science Society, Boston.

Landauer, T. K. (1999). Latent semantic analysis: A theory of the psychology of language and mind. *Discourse Processes, 27*(3), 303-310.

Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review, 104*(2), 211-240.

Landauer, T. K., McNamara, D. S., Dennis, S., & Kintsch, W. (2007). *Handbook of Latent Semantic Analysis*. Mahwah, NJ: LEA.