Numerical orthographic coding: merging Open Bigrams and Spatial Coding theories

Pierre Courrieu, Sylvain Madec, Arnaud Rey

Numerical orthographic coding: merging Open Bigrams and Spatial Coding theories

Pierre Courrieu, Sylvain Madec, and Arnaud Rey

Laboratoire de Psychologie Cognitive, CNRS & Aix-Marseille University, Marseille, France

Running Head: Orthographic coding

Total number of words: 7896

Tables: 4; Figures: 5

Corresponding author:

Pierre Courrieu
Laboratoire de Psychologie Cognitive
UMR 7290, CNRS & Aix-Marseille University
3, place Victor Hugo - Bat. 9, Case D
13331 Marseille Cedex 03 – France

Authors E-mail:

P. Courrieu: courrieu@free.fr, pierre.courrieu@univ-amu.fr

S. Madec: symadec@gmail.com

A. Rey: arnaud.rey@univ-amu.fr

Abstract. Simple numerical versions of the Spatial Coding and of the Open Bigrams coding of character strings are presented, together with a natural merging of these two approaches. Comparing the predictive performance of these three orthographic coding schemes on orthographic masked priming data, we observe that the merged coding scheme always provides the best fits. Testing the ability of the orthographic codes, used as regressors, to capture relevant regularities in lexical decision data, we also observe that the merged code provides the best fits and that both the spatial coding component and the open bigrams component provide specific and significant contributions. This gives us a new lighting on probable mechanisms involved in orthographic coding, together with new tools for modelling behavioural and electrophysiological data collected in word recognition tasks.

Key words. Orthographic Code; Spatial Coding; Open Bigrams; Orthographic Similarity; Orthographic Regressors

1. Introduction

Encoding symbol strings in relevant and convenient numerical formats is a recurrent problem in various scientific domains such as bioinformatics, computational linguistic, artificial intelligence, psycholinguistic, and neurosciences. In at least the last two cited domains, it is of importance to use string codes whose properties are compatible with those of the human perception of character strings. For instance, psycholinguists use string codes to model the data of visual word perception experiments with orthographic priming (Davis & Bowers 2006; Grainger, Granier, Farioli, Van Assche, & van Heuven, 2006). In neurosciences, one uses various word properties to analyse cerebral event related potentials (ERPs) in word perception tasks (Hauk, Davis, Ford, Pulvermüller, & Marslen-Wilson, 2006; Rey, Madec, Grainger, & Courrieu, 2013). Numerical string codes are also useful in neurocomputational modelling as input or output layers of multi-layer neural networks. When they are used as an output (for instance the output of a handwritten character string recognition system), numerical string codes must also be decodable into the corresponding character string (Courrieu, 2012).

In the present work, we propose new numerical string codes whose properties are as much as possible compatible with those of the human perception. Incidentally, we also propose a simple and natural way of solving the conflict between the two main current and concurrent theories of orthographic coding, namely the Open Bigrams theory (Dehaene, Cohen, Sigman, & Vinckier, 2005; Grainger et al., 2006; Hannagan & Grainger, 2012; Whitney, 2001), and the Spatial Coding theory (Davis, 1999, 2010).

In the next section, we present the main concepts of these two theories. In sections 3, 4, and 5, we propose simple numerical models belonging to each of these theoretical families, together with a natural merging of these two approaches. In section 6, we test the three models on available orthographic priming data. In section 7, we use the numerical orthographic codes as regressors to test the models on available lexical decision data. Then we conclude in section 8, and we provide useful Matlab/Octave programs in Appendix 1. As usual in Matlab/Octave codes, instructions for use are included as comments (at right of %).

2. Two usual orthographic coding theories

One of the main families of orthographic coding models available to date is based on the concept of "open bigrams" (Dehaene et al., 2005; Grainger et al., 2006; Hannagan

& Grainger, 2012; Whitney, 2001). An open bigram is an ordered pair of not-necessarily adjacent characters, and the open bigrams code of a character string is basically the list of all the open bigrams of the string (e.g. {SO, SN, ON} for the word SON). Open bigrams are commonly associated with numerical values depending on the gap between the two symbols in the string and the number of occurrences of the open bigram. For instance, following Hannagan and Grainger (2012), the open bigram ME appearing in the string MEMES is associated with the numerical value $2\lambda^2+\lambda^4$ (for some real λ such that $0>\lambda>1$) because the open bigram ME appears two times in sub-sequences of two characters, and one time in a subsequence of four characters. Unfortunately, several empirical arguments against the open bigrams coding theory have recently been stated (Davis & Bowers, 2006; Kinoshita & Norris, 2013; Lupker, Zhang, Perry, & Davis, 2015), so that there is now a serious doubt about the capability of this family of models to make relevant predictions.

Another important family of orthographic coding models is based on the concept of "spatial coding" (Davis, 1999, 2010). The spatial coding principle originates from Grossberg's theory of the encoding of event sequences (Grossberg, 1978; Grossberg & Pearson, 2008). The spatial coding model developed by Davis is a complete simulation model of visual word identification, including a number of possibly realistic but complex features. Empirical arguments supporting the spatial coding principle in word recognition can be found in the paper of Davis and Bowers (2006).

In short, in spatial coding, one associates a dedicated detector to each possible symbol of an alphabet, this detector being activated when an input string includes the corresponding symbol. In the simplest approaches, the activation value of each detector depends on the serial position of the corresponding symbol in the current input symbol string. For instance, Davis (1999) suggested that the activation at time T for a character appearing at the ith position in a string is of the form $\mu\omega^{(T-i)}$, for some real μ and ω>1. There was a difficulty whenever several occurrences of the same symbol appeared in the same string. Davis (1999, 2010) solved this problem assuming that there are several detectors for each alphabetical character, that is, one detector for each possible occurrence of this character in a string. This requires that one a priori fixes the maximum number of occurrences of a given character in a string (e.g. four in common English words), and that the total number of nodes (code length) is equal to the alphabet length times the maximum number of occurrences. A simplified approach of the spatial coding of character strings was proposed by Courrieu (2012) to encode the output of handwritten words recognition systems. It allows one to compactly encode every symbol string in the form of a fixed length numerical vector. An important property of spatial coding models is that every code vector can be exactly decoded back into the corresponding symbol string, which guaranties that the code completely and unequivocally represents the string, and allows one to use it as a decodable numerical output of various systems. The code format also allows one to use such orthographic codes as multidimensional predictors in regression analyses.

3. New Spatial Coding (SC) model

3.1 Code definition

Consider an alphabet of n symbols $\{s_1, s_2, ..., s_n\}$, for instance the 26 lower-case letters of the Roman alphabet. The spatial coding associates to each symbol of the alphabet one component of a real vector $(c_1, c_2, ..., c_n)$. Let X be a symbol string of m characters, one first determines the "symbol position bits" as $b_{k,i} = 1$ if the symbol s_i appears at rank k in X, else one has $b_{k,i} = 0$. Then the components of the orthographic code are given by:

$$c_i(X) = (\Sigma_{k=1..m}\, b_{k,i}\, 2^{-k})^p \ , \ i=1..n,\ 0 < p \leq 1, \qquad (1)$$

where p is a free parameter. For instance, the code for the word "parabola", in the 26 letters Roman alphabet, is the following 26 components vector:

C(parabola) =

$[(2^{-2}+2^{-4}+2^{-8})^p, (2^{-5})^p, 0,0,0,0,0,0,0,0,0,0, (2^{-7})^p, 0,0, (2^{-6})^p, (2^{-1})^p, 0, (2^{-3})^p, 0,0,0,0,0,0,0,0]$.

The Matlab/Octave function "str2scob" listed in Appendix 1 allows the computation of the Spatial Codes of character strings. Examples of string Spatial Codes and of the effect of p are visualized in Fig. 1.

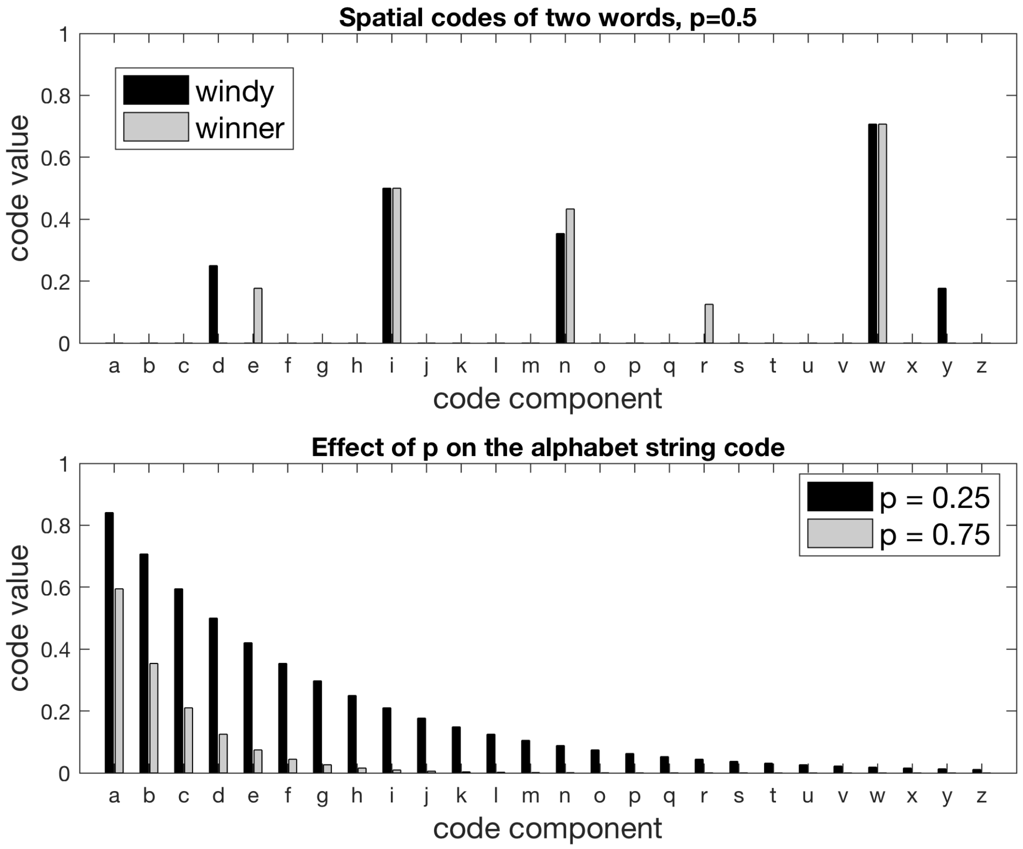

Figure 1. Visualization of the numerical spatial codes of two words (windy, winner), and of the effect of the parameter p (p = 0.25, or p = 0.75) on the alphabet string code (abcdefghijklmnopqrstuvwxyz).

3.2 Decoding

Such a code can be completely and unequivocally decoded back into the corresponding character string in all cases. If a component is zero, then the corresponding character does not appear in the string. For each non-zero component, the corresponding character appears one or several times in the string. To know where it appears, it suffices to raise the component to the power *1/p*, and to compute the binary form of the result. The non-zero bits of this form correspond to the symbol position bits. For instance, in the above example of the word "parabola", consider the code vector component corresponding to the letter "a", its value is $(2^{-2}+2^{-4}+2^{-8})^{p}$. Raising it to the power *1/p*, we obtain $(2^{-2}+2^{-4}+2^{-8})$ =0.31640625, whose binary form is (.01010001), which indicates that the letter "a" appears at ranks 2, 4, and 8 in the character string.

The use of base 2 exponential functions for the coding is motivated by the fact that 2 is the minimum base that allows complete and unequivocal decoding. On the other hand, the exponential function of base 2 decreases very fast as the rank of letters increases, which tends to crush the code values for most letters in the string, except the initial ones. The use of the parameter *p* allows us to correct for this drawback, and to obtain a function that is possibly more suitable to cognitive modelling. In particular, the use of an appropriate *p* value allows minimizing the influence of noise and of approximation errors occurring in natural or artificial systems, because *p* determines the minimum difference (spacing) between two distinct exact values in the code (including 0). The effect of *p* on spatial codes is visualized in the lower part of Fig. 1.

The actual decoding procedures must manage the possible presence of noise and approximation errors in realistic models using spatial coding. This is what does de Matlab/Octave function “scob2str” listed in Appendix 1, and used hereafter.

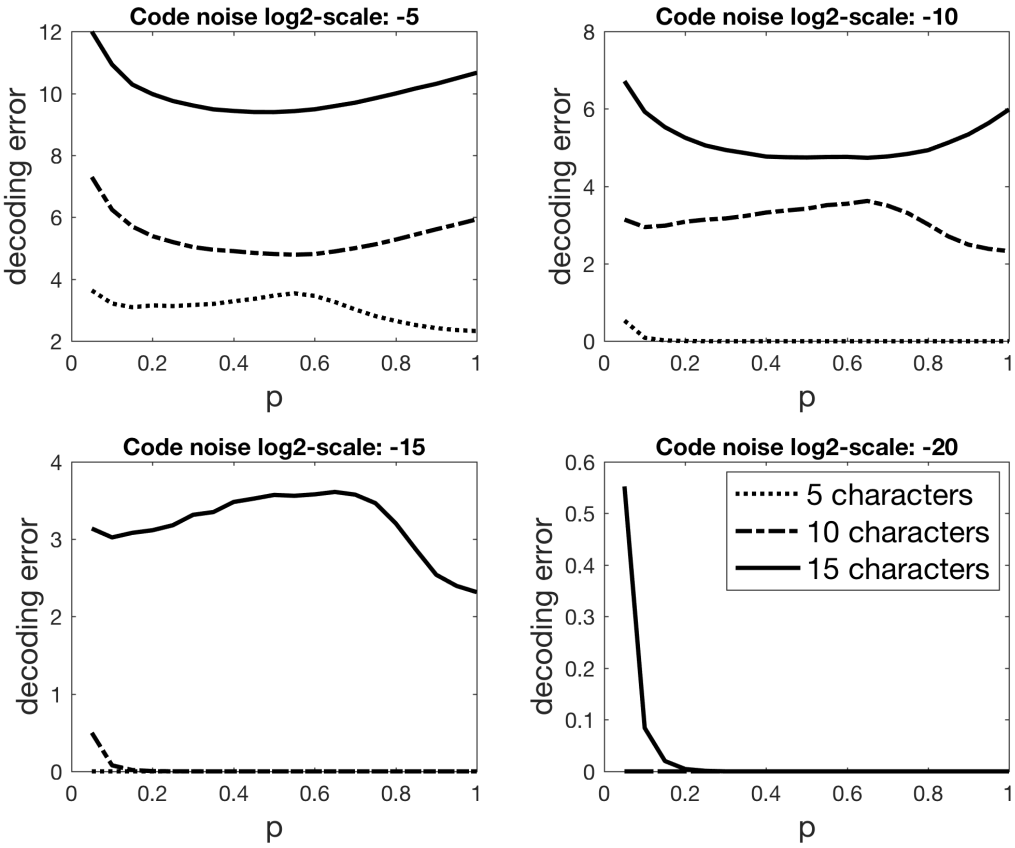

Figure 2. Summary of a computational experiment measuring the decoding error of noisy spatial codes as a function of the scale of the Gaussian noise in the codes (2^{-5}, 2^{-10}, 2^{-15}, 2^{-20}), the length of the original character string (5, 10, or 15 letters), and the p parameter value (from 0.05 to 1 by steps of 0.05).

Fig. 2 summarizes a computational experiment illustrating the behavior of the decoding process as a function of *p*, the amount of noise in the spatial codes, and the length of the character strings. A total of 480000 computational tests has been performed, each of them using a randomly generated character string of a given length (5, 10, or 15 characters), a *p* parameter value (varying from 0.05 to 1 by steps of 0.05), and a given amount of Gaussian noise (with mean 0 and standard deviation 2^{-5}, 2^{-10}, 2^{-15}, or 2^{-20}) added to the spatial code components of the string, resulting in a noisy spatial code which was decoded back using the "scob2str" routine. The resulting string was then compared to the original one using the Damerau-Levenshtein string distance (Damerau, 1964) as a "decoding error" measure. This was repeated 2000 times for each combination of the experimental variables modalities, and the average decoding error was used as the dependent variable in the plots of Fig. 2. Note that for zero noise, the decoding error is always zero if the length of strings does not exceed the precision of the used real numbers (the maximum is 52 characters with the usual standard IEEE 754 double-precision binary floating-point format). With non-zero noise, one can observe in Fig. 2 that the decoding error increases with the amount of noise and with the length of strings, which is not surprising. However, the decoding error is not a monotonic function of *p*, and its shape depends on the relation between the noise scale and the length of the string. In short, let L be the number of characters of the string, then there is almost no effect of *p* on the decoding error (zero) if the noise scale is lower than 2^{-L}, except a small increase for very low *p* values. However, the decoding error function has a maximum on the middle zone of the *p* values if the noise scale is close to 2^{-L}, while it has a minimum on the middle zone of the *p* values if the noise scale is greater than 2^{-L}. Thus, in weakly noisy systems, one can use any value of p, even 1, which is equivalent to remove the *p* parameter. In highly noisy systems, the critical string length is low, and most words are longer than this critical length, thus it is preferable for the decoding accuracy to choose an intermediate value for *p* (in a neighborhood of 0.5). Now, in a system where (unfortunately) the noise scale corresponds to the modal string length, the plots in Fig. 2 show that the best choice is *p*=1.

4. New Open Bigrams Coding (OB) model

The coding model described hereafter is a variant of the one described in Hannagan and Grainger (2012), and in Lodhi, Saunders, Shawe-Taylor, Cristianini, and Watkins (2002). Contrarily to the original model, this variant does not detect one-character strings since it encodes only open bigrams, thus at least two character strings. For instance, in the word 'hat', the open bigrams are 'ha', 'ht', and 'at', while the word 'at' is itself a bigram, but in the one-letter word 'a', there is no bigram in the usual sense. In fact, the following model was designed to be compatible with the above spatial coding scheme (1), in the perspective of merging the two approaches, as described in the next section.

In an alphabet of *n* characters, the open bigrams code of a string *X* of *m* characters is defined as a real matrix of $n \times n$ components c_{ij}, each one corresponding to a possible open bigram whose first character has the index *i* in the alphabet, and whose second character has the index *j*. The symbol position bits $b_{k,i}$ are defined as previously, and one has:

$$c_{ij}(X) = \left(\Sigma_{k=1..m-1} \Sigma_{l=k+1..m} b_{k,i}\, b_{l,j}\, 2^{-(l-k)}\right)^p, \quad i, j = 1..n, \quad 0 < p \leq 1, \qquad (2)$$

For instance, using the 26 letters Roman alphabet, the open bigram 'aa' in the word 'parabola', has the code value $c_{1,1}(\text{parabola})=(2^{-(4-2)}+2^{-(8-2)}+2^{-(8-4)})^p$, while the open bigram 'oa' has the code value $c_{15,1}(\text{parabola})=(2^{-(8-6)})^p$. Since it is more convenient to store the codes in the form of row vectors than in the form of matrices, one vectorizes the code matrix by concatenating its rows one after the other, which results in a row vector of n^2 components. The Matlab/Octave function “str2scob” listed in Appendix 1 allows the computation of the Open Bigrams codes of character strings.

The size of an Open Bigrams code is the square of the size of a Spatial Code, which is somewhat cumbersome, but also much more redundant, and thus potentially more robust in case of approximation errors and noisy code. An Open Bigrams code is easy to decode if the target character string does not include more than one repeated character (as the A in PARABOLA). However, in the general case, decoding an Open Bigrams code is a hard-to-solve problem and there is no known practical solution for large-scale applications that require fast decoding.

5. Merging Spatial and Open Bigrams Codes (SCOB)

There is in fact a very simple and quite natural solution to the main drawbacks of the open bigrams coding. Assume that the beginning of every character string is not its first symbol (letter), but a "start character". For instance, one can consider the left whitespace as a tag of the start character for printed words. One can append a start character at the beginning of the alphabet, and append a start character at the beginning of each character string. If the size of the original alphabet was n characters, then it is now $n+1$, but the size of the corresponding open bigrams code is $(n+1)n$, since the start character is not a stop character and it never appears at the second position in a bigram. The code is computed in the same way as the basic OB code, and one obtains an Open Bigrams code where one character strings are represented as (start character + symbol). Assigning to the start character the index 0 in the alphabet, and the serial position 0 in the character strings, one sets $b_{0,0}=1$, and the code components are defined as:

$$c_{ij}(X) = (\Sigma_{k=0..m-1} \Sigma_{l=k+1..m} b_{k,i} b_{l,j} 2^{-(l-k)})^p, \quad i = 0..n, j = 1..n, \quad 0 < p \leq 1. \qquad (3)$$

This code is easily decodable in all cases since its first n components (c_{0j} components) are exactly equal to those of the spatial code (1) of the same string, which is decodable. For this reason, we will abbreviate this code as SCOB, for "Spatial Code + Open Bigrams", but also remembering "Start Character + Open Bigrams". In this context, one can consider that *the Spatial Coding is just the initial part (n components) of an Open Bigrams coding using an alphabet that includes a start character*.

The Matlab/Octave function “str2scob” listed in Appendix 1 allows the computation of the SCOB codes of character strings, and the function “scob2str” allows their decoding.

6. Tests on masked orthographic priming data

6.1. Test on Adelman et al.’s (2014) data

Orthographic coding models are commonly tested using masked orthographic priming techniques (Davis & Bowers, 2006; Grainger et al., 2006; Van Assche & Grainger, 2006; Welvaert, Farioli, & Grainger, 2008), where one assumes that the more the prime and the target are orthographically similar, the more the priming effect is large (in first approximation). The orthographic similarity of two character strings depends on the considered coding model, together with an associated similarity function. In the case where the orthographic codes are fixed length numerical vectors, say x and y, one can for instance use a similarity function of the form $S(x, y)= <x, y>/(||x||.||y||)$, where $<., .>$ denotes the inner (dot) product, and $||.||$ is the Euclidean norm (Hannagan & Grainger, 2012). If the considered orthographic coding model is correct, then one can expect a strong positive correlation between S(x, y) and the perceptual priming effect of the string whose code is x on the string whose code is y.

We used 27 masked priming effects obtained in a lexical decision task with different prime structures, reported by Adelman et al. (2014), in order to test the predictive capability of the orthographic coding schemes SC, OB, and SCOB described above. The parameter *p* was optimized for each model in order to obtain the best possible correlation between S(x, y) and the corresponding empirical priming effects.

For SC, one obtained p=0.54, r=0.59; for OB, one obtained p=0.55, r=0.90; finally, for SCOB, one obtained p=0.81, and r=0.92. Using Williams T2 test (Steiger, 1980; Williams, 1959), we observed that the OB fit was significantly better than the SC fit (T2(24)=3.73, p<.001), and the SCOB fit was significantly better than the SC fit (T2(24)=4.45, p<.001). However, the correlations were not significantly different for OB and SCOB, although the fit was a bit better for SCOB (T2=1.01, n.s.). Thus, contrarily to what was expected, it seems that the open bigrams coding, even alone, is more suitable than the spatial coding for predicting orthographic priming effects in a lexical decision task.

6.2 Test on Kinoshita and Norris's (2013) data

This unexpected result led us to reconsider arguments recently reported against the Open Bigrams theory, in particular those of Kinoshita and Norris (2013). In three experiments using the same-different match task (Norris & Kinoshita, 2008), with masked primes including open bigrams and reversed open bigrams, the authors observed that reversed bigram primes (e.g. ob-ABOLISH), as well as widely non-contiguous open bigram primes (e.g. bs-ABOLISH) produced robust orthographic priming effects, while this is not possible with current open bigrams models. The authors then concluded: "letter order is not coded by open bigrams". However, such priming effects are possible in spatial coding models, and if one simply appends a start character at the beginning of each word, performing an open bigrams coding on such a string will generate a SCOB code whose initial part is in fact equal to a Spatial Code. Then one can expect that this particular open bigrams model is able to account for the critical priming effects. In order to verify this on Kinoshita and Norris (2013) data, we computed the prime-target orthographic similarities predicted by our three models (SC, OB, and SCOB), using for each model the optimal *p* parameter value previously estimated on Adelman et al.'s (2014) data, and we compared these orthographic similarities with the empirical priming effects.

Table 1. Orthographic priming effects obtained by Kinoshita and Norris (2013) in three experiments using the same-different match task with masked primes including open bigrams and reversed open bigrams. The corresponding orthographic similarities computed by the SC, OB, and SCOB models between the primes and the targets are also reported, with the model fits at the bottom of the table.

Prime - Target	Priming effect (ms)	SC model (p=0.54)	OB model (p=0.55)	SCOB model (p=0.81)
Experiment 1				
of – OF	97	1.0	1.0	1.0
fo – OF	57	0.9338	0	0.4906
the – THE	82	1.0	1.0	1.0
hte – THE	48	0.9425	0.5538	0.6355
Experiment 2				
bo, is – ABOLISH	24	0.4027	0.3223	0.3056
bl, ls – ABOLISH	24	0.4067	0.2202	0.2149
bs – ABOLISH	28	0.4759	0.1027	0.1668
Experiment 3				
bo, is – ABOLISH	29	0.4027	0.3223	0.3056
bs – ABOLISH	25	0.4759	0.1027	0.1668
ob, si – ABOLISH	18	0.3760	0	0.0814
sb – ABOLISH	13	0.3760	0	0.0814
Model fit		r=0.91, p<.0001	r=0.85, p<.001	r=0.97, p<.0001

The results are reported in Table 1, where one can see that the OB model behaves as Kinoshita and Norris expected, predicting zero priming for reversed bigrams, and it provides the worst fit to the data (r=0.85). As we expected, the SC model provides a good account of these data (r=0.91), however, its fit is not significantly better than that of the OB model (T2(8)=0.77, n.s.). Finally, the SCOB model provides the best fit to the data (r=0.97), which is significantly better than the OB model fit (T2(8)=3.91, p<.005), but not significantly better than the SC model fit (T2(8)=1.49, n.s.). Thus the SCOB model suitably account for Kinoshita and Norris's data and it remains the best predictor, however, the hierarchy of performance is reversed for the two other models with respect to what was observed with lexical decision data. This suggests the possibility that the same-different match task and the standard masked priming lexical decision task do not involve the exact same mechanisms.

7. Numerical string codes as multidimensional regressors

Another way of examining the relevance of numerical orthographic codes is to test their ability to capture significant regularities in word processing behavioral data as those collected in large-scale item-level behavioural databases (Balota, Yap, Cortese, Hutchison, Kessler, Loftis, Neely, Nelson, Simpson, & Treiman, 2007; Ferrand, New, Brysbaert, Keuleers, Bonin, Méot, Augustinova, & Pallier, 2010). In this context, we use string codes as multidimensional regressors.

7.1. Methodological considerations

One can directly use the numerical string codes as multidimensional regressors on large-scale item level behavioural databases, where the number of items is much larger than the number of regressor dimensions (it is usually recommended to have at least 10-20 times more items than regressor dimensions). In large OB or SCOB codes, there are frequently components corresponding to open bigrams that never occur in a given database. The regression coefficients of these components are *a priori* zero, and the corresponding components must be temporarily removed to compute the other coefficients. As a result, the number of degrees of freedom of the multidimensional regressors in regression analyses can be lower than the number of code components.

However, high dimension independent variables tend to mechanically account for a large part of the data variance in multiple regression analyses, even if they are purely random, and it is known that the usual R^2 statistic is positively biased. This problem can be partially solved using the so-called "adjusted R^2", denoted $\underline{R}^2$ hereafter, which is a well-known unbiased estimator of the corresponding population parameter, and is designed to be independent of the regressor dimension (Cohen, Cohen, West, & Aiken, 2003, pp. 83-84; Theil, 1961, p. 212).

Another possibility is to use well-known validation methods such as the Monte-Carlo cross-validation procedure to estimate the generalization power of a built regressor. Cross-validation, as a generalization process, avoids the overfitting problems that systematically occur in least squares multiple regression models (Picard & Cook, 1984). In short, using a large-scale database, one repeatedly randomly sample a subset of items of a given size as the cross-validation (generalization) set, and one uses the remaining subset of items as the learning set. One computes the regression coefficients of the orthographic code components on the learning set data, then one applies these coefficients to the orthographic codes of the items of the cross-validation set, and one computes the correlation coefficient of the resulting one-dimensional orthographic regressor with the target data of the cross-validation set. One obtains a sample of cross-validation correlation coefficients, which can be summarized by its mean and a confidence interval (for instance the 99% one). To compute the useful statistics, it is convenient to use the r-to-z Fisher transformation, then to compute the mean and 99% confidence interval on the z values, and finally to transform the results in r values using the inverse Fisher transformation.

7.2. Test on lexical decision data of the English Lexicon Project

The English Lexicon Project (Balota et al., 2007) includes two behavioural databases. The first one provides lexical decision data, and the second one provides speeded word naming data for 40481 English words. For the present study, we used only the lexical decision data, and the items selection was conditional to the availability of valid data concerning: the word spelling, its pronunciation, its frequency, its OLD20, OLD20 frequency, the mean lexical decision z-time and response accuracy. The number of selected words was finally 39302. Both uppercase and lowercase letters were used, resulting in an alphabet of 52 characters:

ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz.

The average number of valid observations per item was 27.8. Using the ICC method (Courrieu, Brand-D'Abrescia, Peereman, Spieler, & Rey, 2011; Courrieu & Rey, 2011, 2015), we computed the proportion of systematic item variance available in the RT z-scores, which gave the ICC= 0.8954, with a 99% CI= [0.8934, 0.8973].

7.2.1. Determination of the optimal code and p parameter value

We tried to determine a suitable orthographic code for the ELP lexical decision data by fitting (least squares method) the RT z-scores and the accuracy, on the basis of spatial codes (SC), open bigrams codes (OB), and merged spatial and open bigrams codes (SCOB), as a function of the p parameter value, which was varied from 0.05 to 1.00 by steps of 0.05. One can observe in Fig. 3 that open bigrams codes always provided better fits (Pearson's r) than the spatial code alone, while the SCOB codes seem to have a small but regular advantage on OB codes. The smallest p parameter values are optimal for the spatial code alone, while p=1 is optimal for the OB and SCOB codes. Thus, globally, p=1 is the optimal choice, which is equivalent to remove the p parameter.

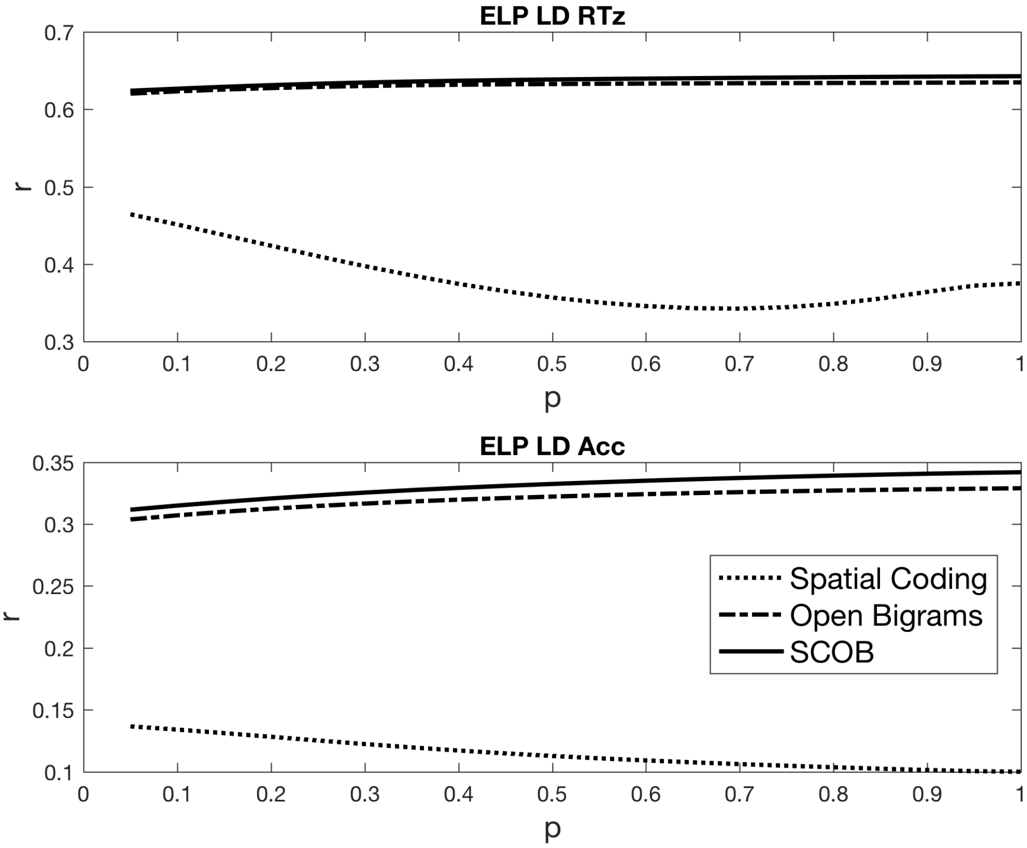

Figure 3. Correlation of ELP lexical decision z-times (upper panel), and response accuracy (lower panel) with their least squares approximations based on spatial codes (SC), on open bigrams codes (OB), and on merged SC and OB codes (SCOB), as functions of the code parameter p.

In order to see whether or not both the SC and the OB components provided specific significant contributions to the performance of the SCOB code regressor, we performed a series of hierarchical multiple regressions, with the RT z-scores as dependent variable, while the orthographic codes SC and OB were entered as multidimensional regressors in this order, and in the reverse order, with various values of the p parameter. These analyses are presented in Table 2, where one can see that in all cases, both the SC and the OB components provide specific significant contributions to the data fit, while p=1 for the two components is globally the best parameter choice.

We conclude that the SCOB code without p parameter is the most appropriate orthographic code for these data, while the contribution of the SC component is small but always relevant.

Table 2. Hierarchical multiple regression analyses of the lexical decision z-times of the English Lexicon Project (Balota et al., 2007). The multidimensional orthographic regressors are the spatial code (SC) and the open bigrams code (OB) of the words, with various values of the code parameter p.

Regressor 1	R^2	Regressor 2	R^2	ΔR^2	ΔR^2 significance	Adj. R^2
SC (p=1)	0.1410	OB (p=1)	0.4132	0.2723	F(1243, 38006)=14.19	0.3932
OB (p=1)	0.4031	SC (p=1)	0.4132	0.0102	F(52, 38006)=12.65	0.3932
SC (p=0.70)	0.1175	OB (p=0.70)	0.4106	0.2931	F(1243, 38006)=15.20	0.3905
OB (p=0.70)	0.4017	SC (p=0.70)	0.4106	0.0089	F(52, 38006)=11.00	0.3905
SC (p=0.05)	0.2158	OB (p=0.05)	0.3895	0.1737	F(1243, 38006)=8.70	0.3687
OB (p=0.05)	0.3849	SC (p=0.05)	0.3895	0.0047	F(52, 38006)=5.57	0.3687
SC (p=0.05)	0.2158	OB (p=1)	0.4089	0.1931	F(1243, 38006)=9.99	0.3887
OB (p=1)	0.4031	SC (p=0.05)	0.4089	0.0058	F(52, 38006)=7.17	0.3887

Underscored F-values are highly significant (p<0.0001)

7.2.2. Relation with usual regressors

In Table 3, one can see the inter-correlations of the SCOB based one-dimensional orthographic regressors (A.RTz and A.Acc), the corresponding targeted data variables (RTz and Acc), and four usual regressors, namely the word length, the word log-frequency, the OLD20, and the OLD20 frequency (Yarkoni, Balota, & Yap, 2008). For the word frequency, we used the logarithm of the HAL frequency plus one (to avoid log(0) for very rare words). We note the high correlations of A.RTz with the word length and the OLD20.

Table 3. Inter-correlations of the ELP one-dimensional SCOB based regressors, A.RTz and A.Acc, targeting the lexical decision z-times (RTz) and the accuracy (Acc), respectively. Correlations with 4 usual regressors: word length, word log-frequency, old20 and old20 frequency are also provided.

r_{39302}	A.RTz	A.Acc	RTz	Acc	length	log-Fr	old20	old20F
A.RTz	-	-0.3627	0.6428	-0.1240	0.8637	-0.3559	0.8568	-0.6412
A.Acc	-0.3627	-	-0.2332	0.3418	0.0474	0.1054	-0.1438	0.0305
RTz	0.6428	-0.2332	-	-0.5974	0.5552	-0.6594	0.6114	-0.4313
Acc	-0.1240	0.3418	-0.5974	-	0.0162	0.4915	-0.1154	0.0191
length	0.8637	0.0474	0.5552	0.0162	-	-0.3514	0.8683	-0.7217
log-Fr	-0.3559	0.1054	-0.6594	0.4915	-0.3514	-	-0.4016	0.4583
old20	0.8568	-0.1438	0.6114	-0.1154	0.8683	-0.4016	-	-0.6622
old20F	-0.6412	0.0305	-0.4313	0.0191	-0.7217	0.4583	-0.6622	-

In Table 4, one analysed the relations of the multidimensional SCOB regressor with the four usual regressors in fitting the RT z-scores. This was done using a series of hierarchical multiple regression analyses from which we can observe that all tested regressors provided specific significant contributions to the fit, except the word length whose contribution was completely explained by the SCOB (but not the reciprocal).

Table 4. Hierarchical multiple regression analyses of the ELP lexical decision z-times. The regressors are the multidimensional SCOB code of the words and the usual regressors: word length, word log-frequency, and old20, and old20 frequency. All regressors provide significant specific contributions, except the word length whose effect is completely explained by the SCOB.

Regressor 1	R^2	Regressor 2	R^2	ΔR^2	ΔR^2 significance	Adj. R^2
old20 + old20F + log-Fr + Length	0.5959	SCOB	0.6744	0.0785	F(1295, 38002)=7.08	0.6632
SCOB	0.4132	old20	0.4505	0.0372	F(1, 38005)=2575.4	0.4317
SCOB	0.4132	old20F	0.4148	0.0015	F(1, 38005)=98.45	0.3948
SCOB	0.4132	log-Fr	0.6604	0.2472	F(1, 38005)=27660	0.6488
SCOB	0.4132	length	0.4132	0	F(1, 38005)= 0	0.3932

Underscored F-values are highly significant ($p<0.0001$)

7.2.3. Orthographic regressors cross-validation

The learning and cross-validation of SCOB orthographic regressors were tested on ELP lexical decision z-times and on the response accuracy. We used cross-validation sets of 2000 items, learning sets of 39302-2000=37302 items, and 120 random test repetitions.

The distributions of 120 cross-validation r values for ELP lexical decision z-times, and ELP lexical decision accuracy are shown in Fig. 4. For the RT z-scores, one obtained the learning average r = 0.6437, 99% CI = [0.6435, 0.6438], and the cross-validation average r = 0.4313, 99% CI = [0.3888, 0.4720]. For the response accuracy, one obtained the learning average r = 0.3440, 99% CI = [0.3438, 0.3443], and the cross-validation average r = 0.1430, 99% CI = [0.1267, 0.1592]. The cross-validation mean r value obtained for the RT z-scores corresponds to 18.6% item variance accounted for, and to 20.78% systematic item variance accounted for, given the ICC=0.8954. The difference between the cross-validation r and the learning r reveals a substantial overfitting resulting from the least squares fit of the regressors to the data.

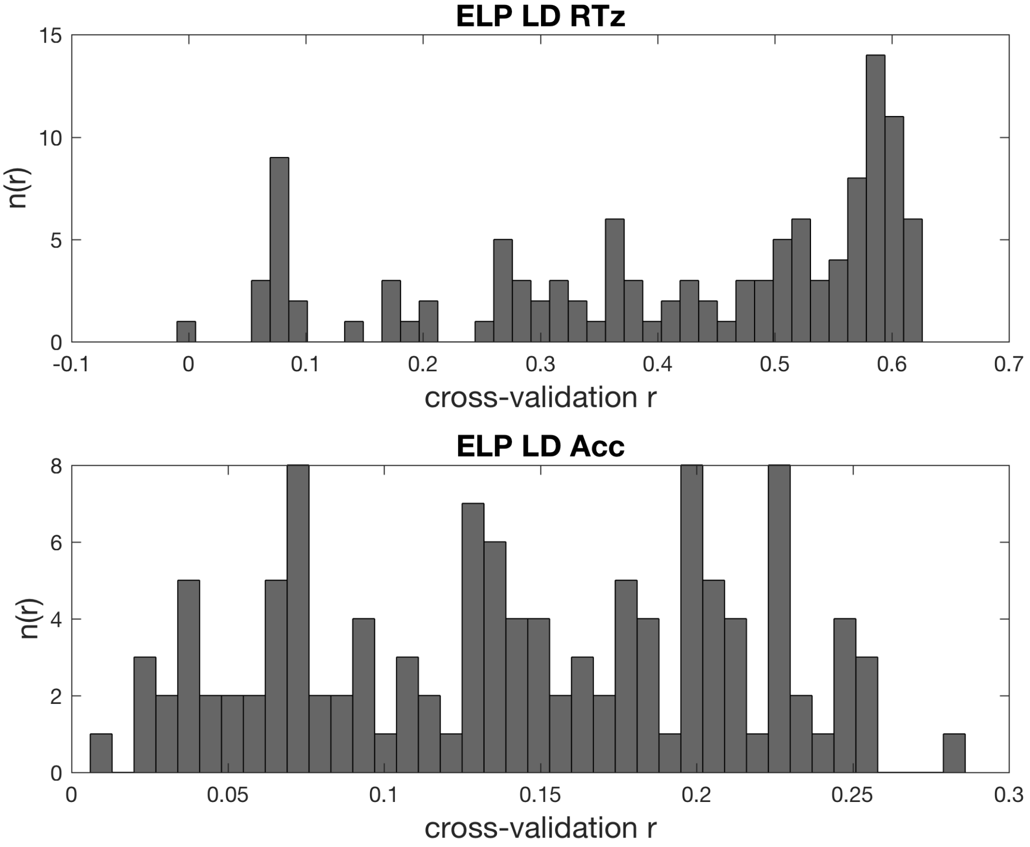

Figure 4. Distribution of 120 cross-validation r values for ELP lexical decision RTz, and ELP lexical decision accuracy. Each r value was computed using 2000 randomly selected generalization test words, while 37302 other words were used to compute regression coefficients of the SCOB code components in order to approximate the data (learning).

Finally, we must note that very similar results were obtained with other large-scale behavioural databases such as the ELP word-naming database (Balota et al., 2007), and the French Lexicon Project (Ferrand et al., 2010). These results clearly confirm the conclusions obtained with masked orthographic priming data.

8. Conclusion

We presented simple numerical versions of the Spatial Coding and of the Open Bigrams coding of character strings, together with a merging of these two approaches. This merging was obtained making the simple and natural hypothesis that all character strings begin with a "start character" (tagged by the left whitespace of printed words, for instance). In these conditions, the initial part of an open bigrams code is equal to the spatial code of the same string, which makes the open bigrams code decodable. Comparing the predictive performance of the three orthographic coding schemes on orthographic masked priming data, as well as on large-scale lexical decision data, we observe that the merged coding scheme always provides the best performance, and that both the spatial coding component and the open bigrams component provide specific and significant contributions. This new lighting on probable mechanisms involved in orthographic coding also provides new tools for modelling behavioural and electrophysiological data collected in word recognition tasks. In order to illustrate this last point, an example of application of the Spatial Coding model in the analysis of cerebral event related potentials (ERPs), initially reported by Rey et al. (2013), is summarized in Appendix 2.

Funding: This work, carried out within the Labex BLRI (ANR-11-LABX-0036) and the Institut Convergence ILCB (ANR-16-CONV-0002), has benefited from support from the French government, managed by the French National Agency for Research (ANR) and the Excellence Initiative of Aix-Marseille University (A*MIDEX)

Declarations of interest: none.

References

Adelman, J. S., Johnson, R. L., McCormick, S. F., McKague, M., Kinoshita, S., Bowers, J. S., Perry, J. R., Lupker, S. J., Forster, K. I., Cortese, M. J., Scaltritti, M., Aschenbrenner, A. J., Coane, J. H., White, L., Yap, M. J., Davis, C., Kim, J., Davis, C. J. (2014). A behavioral database for masked form priming. *Behavior research methods*, *46*(4), 1052-1067.

Balota, D. A., Yap, M. J., Cortese, M. J., Hutchison, K. A., Kessler, B., Loftis, B., Neely, J. H., Nelson, D. L., Simpson, G. B., & Treiman, R. (2007). The English Lexicon Project. *Behavior Research Methods,* 39, 445–459.

Cohen, J., Cohen, P., West, S.G., & Aiken, L.S. (2003). *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences (3rd Ed.)*. London, Lawrence Erlbaum Associates, Publishers.

Courrieu, P. (2012). Density Codes, Shape Spaces, and Reading. *ERMITES 2012 : Representations and Decisions in Cognitive Vision*. La Seyne-sur-Mer, August 30-31 and September 1. Proceedings : http://glotin.univ-tln.fr/ERMITES12/

Courrieu, P., Brand-D'Abrescia, M., Peereman, R., Spieler, D., & Rey, A. (2011). Validated intraclass correlation statistics to test item performance models. *Behavior Research Methods, 43*, 37-55. doi: 10.3758/s13428-010-0020-5

Courrieu, P., & Rey, A. (2011). Missing data imputation and corrected statistics for large-scale behavioral databases. *Behavior Research Methods, 43*, 310-330. doi: 10.3758/s13428-011-0071-2

Courrieu, P., & Rey, A. (2015). General or idiosyncratic item effects : what is the good target for models ? *Journal of Experimental Psychology : Learning, Memory, and Cognition, 41(5),* 1597-1601. DOI : 10.1037/xlm0000062

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM, 7(3)*, 171-176.

Davis, C. J. (1999). *The Self-Organising Lexical Acquisition and Recognition (SOLAR) model of visual word recognition*. Unpublished doctoral dissertation, University of New SouthWales, Australia.

Davis, C.J. (2010). The spatial coding model of visual word identification. *Psychological Review, 117(3)*, 713-758.

Davis, C. J., & Bowers, J. S. (2006). Contrasting five different theories of letter position coding: Evidence from orthographic similarity effects. *Journal of Experimental Psychology: Human Perception and Performance*, *32*(3), 535-557.

Dehaene, S., Cohen, L., Sigman, M., & Vinckier, F. (2005). The neural code for written words: A proposal. *Trends in Cognitive Sciences, 9*, 335–341.

Ferrand, L., New, B., Brysbaert, M., Keuleers, E., Bonin, P., Méot, A., Augustinova, M., & Pallier, C. (2010). The French Lexicon Project: Lexical decision data for 38,840 French words and 38,840 pseudowords. *Behavior Research Methods, 42*, 488–496.

Golub, G. H., & Reinsch, C. (1970). Singular value decomposition and least squares solutions. *Numerische Mathematik, 14(5)*, 403-420.

Grainger, J., Granier, J.P., Farioli, F., Van Assche, E., & van Heuven, W. (2006). Letter position information and printed word perception: The relative-position priming constraint. *Journal of Experimental Psychology: Human Perception and Performance, 32*, 865–884.

Grossberg, S. (1978). A theory of human memory: Self-organization and performance of sensory-motor codes, maps, and plans. *In* R. Rosen & F. Snell (Eds.), *Progress in theoretical biology* (pp. 233–374). New York, NY: Academic Press.

Grossberg, S., & Pearson, L. R. (2008). Laminar cortical dynamics of cognitive and motor working memory, sequence learning and performance: toward a unified theory of how the cerebral cortex works. *Psychological Review*, *115*(3), 677.

Hannagan, T., & Grainger, J. (2012). Protein analysis meets visual word recognition: A case for string kernels in the brain. *Cognitive Science, 36*, 575–606.

Hauk, O., Davis, M.H., Ford, M., Pulvermüller, F., and Marslen-Wilson, W.D. (2006). The time course of visual word recognition as revealed by linear regression analysis of ERP data. *NeuroImage, 30*, 1383–1400.

Kinoshita, S., & Norris, D. (2013). Letter order is not coded by open bigrams. *Journal of memory and language*, *69*(2), 135-150.

Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., & Watkins, C. (2002). Text classification using string kernels. *Journal of Machine Learning Research, 2*, 419–444.

Lupker, S. J., Zhang, Y. J., Perry, J. R., & Davis, C. J. (2015). Superset versus substitution-letter priming: An evaluation of open-bigram models. *Journal of Experimental Psychology: Human Perception and Performance*, *41*(1), 138-151.

Madec, S., Le Goff, K., Anton, J-L., Longcamp, M., Velay, J-L., Nazarian, B., Roth, M., Courrieu, P., Grainger, J., & Rey, A. (2016). Brain correlates of phonological recoding of visual symbols. *NeuroImage, 132*, 359-372. doi: 10.1016/j.neuroimage.2016.02.010

Norris, D., & Kinoshita, S. (2008). Perception as evidence accumulation and Bayesian inference: Insights from masked priming. *Journal of Experimental Psychology: General, 137*, 434–455. http://dx.doi.org/10.1037/a0012799

Picard, R., & Cook, D. (1984). Cross-validation of regression models. *Journal of the American Statistical Association, 79 (387),* 575–583*.*

Rey, A., Madec, S., Grainger, J., Courrieu, P. (2013). Accounting for variance in single-word ERPs. Oral communication presented at the *54th Annual Meeting of the Psychonomic Society*, Toronto, Canada, November 14-17*.*

Steiger, J. H. (1980). Tests for comparing elements of a correlation matrix. *Psychological Bulletin, 87(2)*, 245–251.

Theil, H. (1961). *Economic Forecasts and Policy* (2nd ed., 3rd printing, 1970). Amsterdam, North-Holland Publishing Company.

Van Assche, E., & Grainger, J. (2006). Astudy of relative-position priming with superset primes. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 32*, 399–415.

Welvaert, M., Farioli, F., & Grainger, J. (2008). Graded effects of number of inserted letters in superset priming. *Experimental Psychology, 55(1)*, 54–63.

Whitney, C. (2001). How the brain codes the order of letters in a printed word: The SERIOL model and selective literature review. *Psychonomic Bulletin & Review, 8*, 221–243.

Williams, E. J. (1959). The comparison of regression variables. *Journal of the Royal Statistical Society, Series B, 21*, 396–399.

Yarkoni, T., Balota, D. A., & Yap, M. J. (2008). Moving beyond Coltheart's N: A new measure of orthographic similarity. *Psychonomic Bulletin & Review, 15*, 971–979.

Appendix 1

Matlab/Octave code of useful functions (for academic use only).

```
function [v,alphabet,lsaprx,lscoef] = str2scob(s,p,alphabet,lscoef,data,RL)
% Spatial Coding and/or Open Bigrams coding of character strings.
% Optionally compute regression coefficients and data approximation
% ----------------------------------------------------------------------
%                 Input arguments:
% s: cell/char array of m strings (m >= 1).
% p: 1x2 vector; SC included if p(1)>0, OB included if p(2)>0.
%  default ([]): p=[1,1], i.e. both SC and OB with power equal to 1.
% data: optional data vector or matrix to be approximated (m-by-dw).
% RL: if provided then the strings are encoded from right to left.
%
%                 Input or output arguments:
% alphabet: optional string of lenght N (set to '' if unknown).
% lscoef: optional least square approximation coefficients such that
%    lsaprx=[ones(m,1),v]*lscoef; (set lscoef to [] if unknown)
%
%                 Output arguments:
% v: table of numerical codes of all strings. The size of v is:
%    m-by-N for SC, m-by-N*N for OB, or m-by-N(N+1) for SC + OB.
% lsaprx: optional least square approximation of data on the v basis.
%
%                 Usage:
% Exemple 1. Simple SCOB encoding
% v=str2scob('word',[1/3 1],'a':'z');
%     result:
% size(v) = [1 702]
%
% Exemple 2. SC encoding, LS coefficients & LS approximation of data
% s{1}='caba'; s{2}='bab'; s{3}='bacaba'; s{4}='ababa'; data=[4;3;6;5];
% p=[1/(6*log(2)),0];      % Note: this is a simple SC since p(2)=0
% [v,alphabet,lsaprx,lscoef]=str2scob(s,p,'',[],data);
%     result:
% v = 0.7560    0.6065    0.8465
%     0.7165    0.8931         0
%     0.7649    0.8589    0.6065
%     0.9037    0.7560         0
% alphabet = 'abc'
% lsaprx = [4.0000; 3.0000; 6.0000; 5.0000]
% lscoef = [-20.4385; 18.8399; 11.1284; 4.0703]
%
% Exemple 3. Reuse of coefficients for generalization on new strings
%     new input:
% t{1}='baa'; t{2}='cabb';
% [v2,alphabet,aprx2]=str2scob(t,p,alphabet,lscoef);
%     result:
% v2 = 0.7899    0.8465         0
%      0.7165    0.6686    0.8465
% aprx2 = [3.8632; 3.9471]
% -----------------------------------------------------------------------
if ischar(s), s=cellstr(s); end
m=length(s);
if (nargin>5) && ~isempty(RL), RL=true; else RL=false; end
if nargin<2 || length(p)<2, p=[1,1]; end
scflag=false; obflag=false;
if p(1)>0, scflag=true; end
if p(2)>0, obflag=true; end
if (nargin<3) || isempty(alphabet)             % Compute the alphabet
    alphabet='';
    for i=1:m
        alphabet=unique(strcat(alphabet,s{i}));
    end
end
N=length(alphabet);
if scflag && obflag
    v=zeros(m,(N+1)*N);
else if scflag
        v=zeros(m,N);
    else if obflag
            v=zeros(m,N*N);
        else
            error('No coding method selected')
        end
    end
end
sc=[]; ob=[];
for i=1:m                        % Compute codes of the m strings
    si=s{i}; L=length(si);
    if RL, si=fliplr(si); end
    if scflag                    % Spatial Code or start-OB
        sc=zeros(1,N);
     for j=1:L
        c=strfind(alphabet,si(j));
        sc(1,c)=sc(1,c)+2^(-j);
     end
     sc=sc.^p(1);
    end
    if obflag                    % Open Bigrams coding
        ob=zeros(N,N);
     for j1=1:(L-1)
        for j2=(j1+1):L
            c1=strfind(alphabet,si(j1));
            c2=strfind(alphabet,si(j2));
            gap=j2-j1;
            ob(c1,c2)=ob(c1,c2)+2^(-gap);
        end
     end
     ob=ob.^p(2); ob=ob'; ob=ob(:)';
    end
    v(i,:)=[sc,ob];
end
if nargin<4, lscoef=[]; lsaprx=[]; end
% Reuse given lscoef on the codes of new input strings
if (nargin>=4) && ~isempty(lscoef)
    lsaprx=[ones(m,1),v]*lscoef;
end
% Compute lscoef and lsaprx from given data to be approximated
if (nargin>=5) && ~isempty(data) && isempty(lscoef)
    [vh,vw]=size(v); [dh,dw]=size(data);
    if vh~=dh, error('data size error'); end
    lscoef=zeros(vw+1,dw); nzv=find(sum(v)>0);
    x=pinv([ones(m,1),v(:,nzv)])*data; lsaprx=[ones(m,1),v(:,nzv)]*x;
    lscoef([1;nzv(:)+1],:)=x;
end
end


function st = scob2str(v,p,alphabet,RL)
% Decoding of a SC or a SCOB numerical code of a character string
% ---------------------------------------------------------------
%              Input arguments:
% v: SC or SCOB numerical code of a character string
% p: power parameter of the code, or only p(1).
% alphabet: character string including all reference characters
% RL: if provided then the output string is reversed.
%
%              Output argument:
% st: character string resulting from the decoding of v
%
%              Usage:
%    Preliminary encoding:
% v=str2scob('word',[1/3 1],'a':'z');
%    result:
% size(v) = [1 702]
%    Decoding:
% st=scob2str(v,1/3,'a':'z')
%    result:
% st = word
% -----------------------------------------------------------------
if (nargin>3) && ~isempty(RL), RL=true; else RL=false; end
N=length(alphabet); v=v(1:N); maxlen=-log2(eps);
v(v>1)=1-eps; v(v<0)=0; v= v.^(1/p(1));
  st=''; nextk=1;
  vmax=max(v,[],2);
    while (vmax>=eps) && (nextk<=maxlen)
      j=find(v==vmax,1,'first');
      ch=alphabet(j);
      k=ceil(-log2(v(j)));
      if abs(k-nextk)>1, break, end
      st=strcat(st,ch);
      k=min(k,nextk);
      v(j)=v(j)-2^(-k);
      vmax=max(v,[],2);
      nextk=nextk+1;
    end
if RL, st=fliplr(st); end
end
```

Appendix 2

Example of application in electrophysiological data analysis (Rey et al., 2013)

As mentioned in the introduction, an application field of special interest of orthographic or phonological regressors is the analysis of cerebral event related potential data (ERPs) in various psycholinguistic tasks (lexical decision, word naming, ...). The problem of the dimension of regressors is particularly critical in this area because the number of distinct stimuli used in ERP experiments is usually limited. As an example, we rapidly summarize the work of Rey, Madec, Grainger, and Courrieu (2013). These authors collected ERPs associated to 200 printed French test words, 4-8 letters long, in a speeded word naming task, using averaged ERPs on 4 repetitions per word for

48 French participants. The EEG activity was recorded continuously using 64 electrodes, positioned on the scalp according to the 10-10 International System, in a time window of -100 ms to +500 ms with respect to the stimulus onset. The between-participant consistency of ERPs, as measured by the ICC, allowed detecting latencies and scalp locations where systematic electrophysiological responses occurred. At these spatiotemporal points, various regressors were applied (with test inflation control) to attempt to identify the nature of the involved processes. In particular, one used an orthographic Spatial Code (1) of 26 lowercase letters to detect orthographic processing, a phonological Spatial Code (1) of 35 French phonemes to detect phonological encoding, and the usual word log-frequency to detect a lexical level processing. Spatial codes were computed using a *p* parameter value of about 1/3 in order to obtain non-negligible code values at all serial positions. However, the size of the orthographic code matrix was 200-by-26, while the size of the phonological code matrix was 200-by-35, which in both cases resulted in substantial regressor overfitting for 200 words. So, it was necessary to lower the dimension of the regressors, which was done by replacing each of the two string code matrices by its first three (left) singular vectors (Golub & Reinsch, 1970). This provided acceptable three-dimensional regressors (i.e. 200-by-3 matrices), while preserving a maximum part of each original regressor variation for the considered 200 test words.

Fig. 5 shows the obtained time course of the detected processes. The first systematic (significant ICCs) but unidentified processes appeared before 100 ms, while the beginning of an orthographic processing was detected at a latency of about 148 ms on a right occipital area, migrating to an occipital area at 188 ms, where and when also appeared the beginning of a phonological encoding. The phonological processing then migrated to a left occipital area at about 238 ms, and was followed by a word-frequency effect beginning at about 246 ms, also on a left occipital area. The sequence of detected processes seems logical in a word naming task, and the scalp locations of corresponding ERPs are consistent with those observed with other methods in other tasks involving visual character processing and phonological transcoding (Madec et al., 2016).

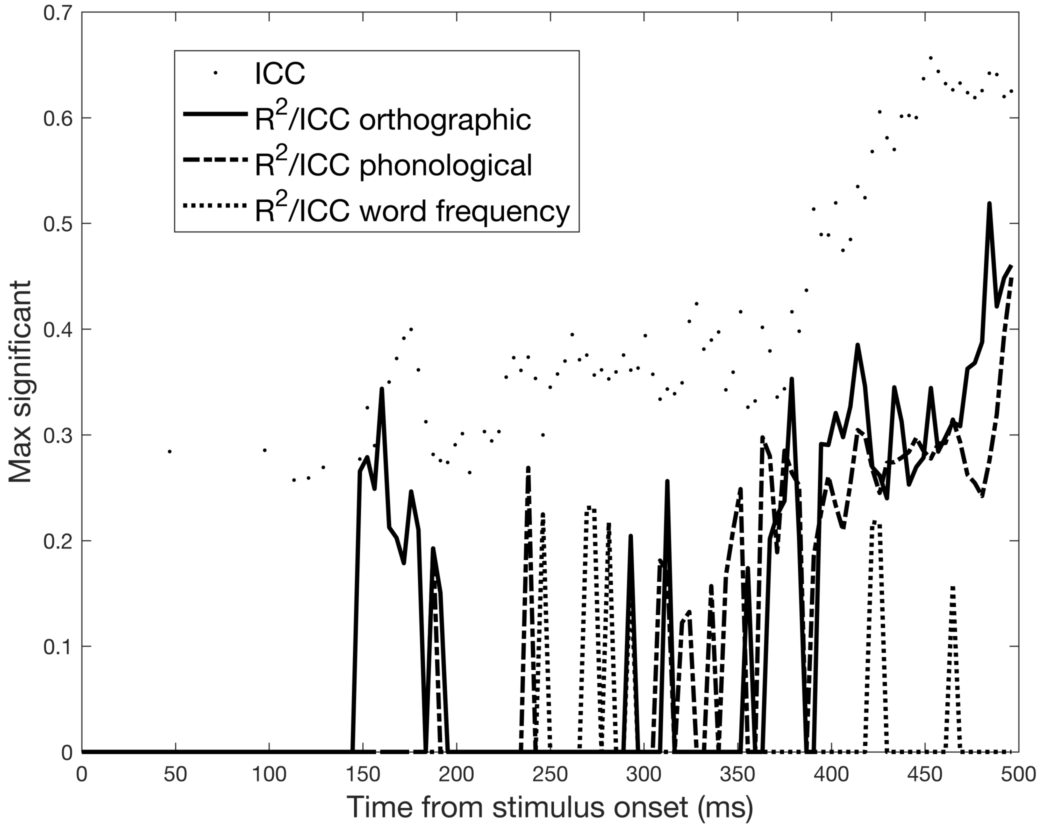

Figure 5. Time course of the between-participant consistency (ICC) and regressor fits (R^2/ICC), for orthographic, phonological, and word frequency regressors applied to cerebral Event Related Potentials in a word-naming task. Non-significant statistics are set to zero for readability, and only the maximal significant values among 64 electrodes are displayed for each latency (after Rey, Madec, Grainger, & Courrieu, 2013).