Distance-Constrained Orthogonal Latin Squares for Brain-Computer Interface

Gang Luo • Wanli Min
IBM T.J. Watson Research Center, 19 Skyline Drive, Hawthorne, NY 10532, USA
{luog, wanlimin}@us.ibm.com

Abstract The P300 brain-computer interface (BCI) using electroencephalogram (EEG) signals can allow amyotrophic lateral sclerosis (ALS) patients to instruct computers to perform tasks. To strengthen the P300 response and increase classification accuracy, we proposed an experimental design where characters are intensified according to orthogonal Latin square pairs. These orthogonal Latin square pairs satisfy certain distance constraint so that neighboring characters are not intensified simultaneously. However, it is unknown whether such distance-constrained, orthogonal Latin square pairs actually exist. In this paper, we show that for every matrix size commonly used in P300 BCI, thousands to millions of such distance-constrained, orthogonal Latin square pairs can be systematically and efficiently constructed and are sufficient for the purpose of being used in P300 BCI.

Keywords Brain-computer interface · Orthogonal Latin squares · Experimental design · Distance constraint

1. Introduction

A few diseases, e.g., end-stage amyotrophic lateral sclerosis (ALS) and severe cerebral palsy, can make patients fully paralyzed. These paralyzed patients can neither speak nor move their body parts. Brain computer interface (BCI) [1, 2] can allow these patients to instruct computers to perform tasks and is often the only method for them to communicate with the outside world. As patients think about what they want, their thinking is classified based on their electroencephalogram (EEG) waves reflecting their brains' electrical activities and the computer automatically executes the corresponding instructions. Accurate EEG wave classification is critical for the computer to issue the correct instructions.

A	B	C	D	E	F	G
H	I	J	K	L	M	N
O	P	Q	R	S	T	U
V	W	X	Y	Z	0	1
2	3	4	5	6	7	8
9	⊔	(	)	!	@	#
$	%	^	&	*	,	.

Fig. 1 Example of a matrix used in the P300 brain-computer interface.

A	B	C	D	E	F	G
H	I	J	K	L	M	N
O	P	Q	R	S	T	U
V	W	X	Y	Z	0	1
2	***3***	***4***	***5***	***6***	***7***	***8***
9	⊔	(	)	!	@	#
$	%	^	&	*	,	.

(a)

A	B	C	D	***E***	F	G
H	I	J	K	***L***	M	N
O	P	Q	R	***S***	T	U
V	W	X	Y	***Z***	0	1
2	3	4	5	***6***	7	8
9	⊔	(	)	***!***	@	#
$	%	^	&	*******	,	.

(b)

Fig. 2 Seven characters are intensified simultaneously. One row of characters is intensified in (a), and one column of characters is intensified in (b).

In order to be widely adopted, a BCI system needs to be noninvasive, easy to use, easy to set up, and portable. The P300 BCI system using EEG signals satisfies these requirements and is one of the most promising among all types of BCIs. P300 refers to a neurally evoked potential component of EEG. The current P300 BCI communicates one symbol at a time and works as shown in Fig. 1 and Fig. 2 [3]. A matrix of characters or pictures is displayed on the computer screen. A predetermined number of intensification rounds are performed for communicating one character. During these rounds, the user focuses on the matrix cell containing the desired character that she intends to communicate. The user is also instructed to count the number of times this desired character is intensified. In each round, all the rows and columns of the matrix are intensified once in a random order - one row or column at a time. The row and column containing the desired character form the rare set (the target), and the other rows and columns form the frequent set (the nontargets). The physiological rationale behind P300 BCI is that intensification of the target row or column should elicit a P300 response because it is an unexpected rare event in the sequence of row or column intensifications. Using properly calibrated signal processing algorithms, we can detect P300 responses from the recorded EEG signals of the user. Then we can use these responses to classify the target row and column whose intersection cell contains the character the user intends to communicate.

Experimental design is the term describing how characters are arranged and how intensification is performed. To maximize both the classification accuracy and communication speed of the P300 BCI system, a good experimental design is needed to obtain strong P300 responses. However, the above-mentioned, row-column experimental design is nonoptimal due to an undesirable effect caused by neighboring characters. This effect particularly affects ALS patients, who have eye movement problems and form one of the most important user groups of BCI. When neighboring characters in a row or a column are intensified simultaneously, an ALS patient's attention can be distracted from the desired character [4]. This weakens the P300 response and hence reduces the classification accuracy. To reduce this interference, we need to maintain minimal pair-wise distance among simultaneously intensified characters. The larger the distances between simultaneously intensified characters, the less interference the ALS patient will receive.

In Min and Luo [5], we proposed using the mathematical structure of the Latin square to intensify non-neighboring characters simultaneously. A Latin square of order n is an $n \times n$ matrix based on a set of n symbols, so that each row and column contains each symbol exactly once [6, 7, 8]. Without loss of generality, the symbols are assumed to be 1, 2, …, and n. Fig. 3 shows an example of a Latin square of order seven.

0	2	1	4	3	6	5
3	1	6	0	5	4	2
4	5	2	6	0	1	3
5	6	4	3	2	0	1
6	3	5	1	4	2	0
1	0	3	2	6	5	4
2	4	0	5	1	3	6

Fig. 3 A self-orthogonal Latin square of order seven.

A	B	***C***	D	E	F	G
H	***I***	J	K	L	M	N
O	P	Q	R	S	***T***	U
V	W	X	Y	Z	0	***1***
2	3	4	***5***	6	7	8
9	⊔	(	)	!	@	#
$	%	^	&	*	,	.

Fig. 4 Seven characters are intensified simultaneously according to the corresponding positions of the symbol "1" in the Latin square in Fig. 3.

If we intensify characters according to a Latin square (Fig. 4), the simultaneously intensified characters will not be direct neighbors either horizontally or vertically. To ensure the desired character can be uniquely determined within each round of intensification, we proposed resorting to the concept of orthogonal Latin squares [5]. Intuitively, Latin squares L_1 and L_2 of the same order n are orthogonal if the cells in L_1 containing the same symbol can be regarded as a conceptual row, the cells in L_2 containing the same symbol can be regarded as a conceptual column, and each conceptual row and column has a unique intersection cell. The formal definition of orthogonal Latin squares is as follows [6, 7, 8]:

Definition of orthogonal Latin squares. Given two Latin squares, L_1 and L_2, of order n, we can superimpose them on one another and construct an $n \times n$ superposition matrix N of ordered pairs. Here, (h, k) occupies position (i, j) of N if and only if h occupies position (i, j) of L_1 and k occupies position (i, j) of L_2. L_1 and L_2 are said to be orthogonal if each of the n^2 possible ordered pairs occurs exactly once in N. L_1 is said to be self-orthogonal if L_1 and its transpose are orthogonal.

The Latin square in Fig. 3 is self-orthogonal. Many more self-orthogonal Latin squares are provided in Burger *et al.* [6].

For an $n \times n$ character matrix M, the following new experimental design proposed in Min and Luo [5] can ensure unique character determination by mapping M to the superposition of L_1 on L_2. Whenever the experimental design intensifies the hth ($1 \le h \le n$) row of M, the new experimental design intensifies the characters in M corresponding to the hth conceptual row in L_1. Whenever the experimental design intensifies the kth ($1 \le k \le n$) column of M, the new experimental design intensifies the characters in M corresponding to the kth conceptual column in L_2. By detecting the P300 responses from the recorded EEG signals of the user, we can classify the target conceptual row and column whose unique intersection cell contains the character the user intends to communicate. If we expand nonsquare matrices into square matrices by adding dummy rows or columns, this method also works for nonsquare character matrices.

When choosing Latin squares, we can impose various distance constraints [5]. One is that, in the Latin square, the distance between any pair of cells containing the same symbol is larger than a predetermined threshold t. Here, the distance between two cells is defined as the Euclidean distance between the centers of these two cells. Using this constraint, we can ensure that, at any time, the distance between any two simultaneously intensified characters is larger than t, which can reduce interference for ALS patients, lead to stronger P300 responses, and improve classification accuracy.

Among all reasonable distance thresholds, the minimal one is that $t = \sqrt{2}$ units so that the simultaneously intensified characters will not be direct neighbors horizontally, vertically, or diagonally. However, it is unknown whether orthogonal Latin square pairs satisfying this minimal distance constraint actually exist. In this paper, we show that for every matrix size commonly used in P300 BCI, thousands to millions of orthogonal Latin square pairs satisfying this minimal distance constraint can be systematically and efficiently constructed and are sufficient for the purpose of being used in P300 BCI.

The rest of the paper is organized as follows. Section 2 presents our method for constructing distance-constrained, orthogonal Latin square pairs. Section 3 provides some experimental results. Section 4 concludes.

2. Distance-constrained orthogonal latin squares

For any positive integer n that is neither 2 nor 6, many pairs of orthogonal Latin squares of order n exist [7, 8]. For the order of 2 or 6, it has been proven that orthogonal Latin squares do not exist. The pair of orthogonal Latin squares used to communicate a character can vary from one character to another through random selection. This provides much flexibility and makes the character intensification pattern more unexpected by the user. As mentioned by Sellers *et al.* [3], such unexpectedness can lead to stronger P300 responses and improve classification accuracy.

In general, for a given order, a small number of orthogonal Latin square pairs can be quickly constructed using the mathematical structure of the finite field [7, 8]. To obtain more orthogonal Latin square pairs, the state-of-the-art method is to perform an exhaustive search by building a search tree and using various pruning rules to limit the size of the tree [6]. This search procedure often runs for multiple days without being finished and produces a subset of all possible orthogonal Latin square pairs that satisfy certain property. At present, many enumeration results of orthogonal Latin square pairs have been compiled and are publicly available [6]. In this section, we show how to use these orthogonal Latin square pairs as seeds to construct distance-constrained, orthogonal Latin square pairs.

In the rest of the paper, we focus on the minimal distance constraint, where the predetermined distance threshold is $t=\sqrt{2}$ units. Other distance constraints can be handled in a similar way, whereas fewer orthogonal Latin square pairs will satisfy them compared to the case of the minimal distance constraint. For the purpose of P300 BCI, there is no need to find all possible orthogonal Latin square pairs satisfying the minimal distance constraint. Rather, it is sufficient to find many orthogonal Latin square pairs satisfying the minimal distance constraint. This observation can be used to improve the efficiency of constructing distance-constrained, orthogonal Latin square pairs, as early stopping becomes feasible in the construction process.

To systematically construct orthogonal Latin square pairs satisfying the minimal distance constraint, the following property of orthogonal Latin squares plays a key role [7, 8]: **Property of orthogonal Latin squares**. Given a pair of orthogonal Latin squares, we can simultaneously permute their rows or columns. The resulting two matrices still form a pair of orthogonal Latin squares.

Starting from a single "seed" orthogonal Latin square pair, this property allows us to systematically construct a large number of orthogonal Latin square pairs through simultaneous row or column permutation. As mentioned above, for any order n that is neither 2 nor 6, multiple seed orthogonal Latin square pairs can be quickly constructed or obtained from a publicly-available, pre-compiled pool of orthogonal Latin square pairs [6, 7, 8]. There are $n!$ distinct permutations for the n symbols 1, 2, …, and n. Consequently, there are $n!$ distinct row permutations and another $n!$ distinct column permutations. Starting from a single seed orthogonal Latin square pair, we can obtain $(n!)^2$ orthogonal Latin square pairs through simultaneous row or column permutation. For each obtained orthogonal Latin square pair, we can check whether it satisfies the minimal distance constraint. For example, using the self-orthogonal Latin square shown in Fig. 3 and its transpose as the seed orthogonal Latin square pair, we can obtain 588 orthogonal Latin square pairs satisfying the minimal distance constraint. One such distance-constrained, orthogonal Latin square pair is shown in Fig. 5. If one seed orthogonal Latin square pair cannot produce enough distance-constrained, orthogonal Latin square pairs, we can resort to more seed orthogonal Latin square pairs.

0	1	3	6	5	4	2
3	6	5	4	2	0	1
5	4	2	0	1	3	6
2	0	1	3	6	5	4
1	3	6	5	4	2	0
6	5	4	2	0	1	3
4	2	0	1	3	6	5

0	4	6	1	2	5	3
2	5	3	0	4	6	1
4	6	1	2	5	3	0
5	3	0	4	6	1	2
6	1	2	5	3	0	4
3	0	4	6	1	2	5
1	2	5	3	0	4	6

Fig. 5 A pair of orthogonal Latin squares of order seven that satisfy the minimal distance constraint.

For each of the $(n!)^2$ orthogonal Latin square pairs obtained through simultaneous row or column permutation, the minimal distance constraint is checked by processing the n^2 matrix cells one by one for either of the two Latin squares in the orthogonal Latin square pair. As shown in Fig. 6, when processing the cell (i, j) of a Latin square, where $1 \leq i \leq n$ and $1 \leq j \leq n$, we need to consider its eight direct neighboring cells to see whether any of them contains the same symbol as that of itself. If so, this Latin square violates the minimal distance constraint. The cell (i, j) has four horizontal or vertical direct neighbors: the cells $(i, j\text{-}1)$, $(i, j+1)$, $(i\text{-}1, j)$, and $(i+1, j)$. According to the definition of Latin square, these four cells, which are on the same row or column as the cell (i, j), do not contain the same symbol as that of the cell (i, j) and hence need not to be checked. Among the remaining four cells ($i+1$, $j-1$), ($i+1$, $j+1$), ($i-1$, $j-1$), and ($i-1$, $j+1$) that are the diagonal direct neighbors of the cell (i, j), only the two cells ($i+1$, $j-1$) and ($i+1$, $j+1$) need to be checked. The checking of whether the cell ($i-1$, $j-1$) and the cell (i, j) contain the same symbol is performed when the cell ($i-1$, $j-1$) is processed. The checking of whether the cell ($i-1$, $j+1$) and the cell (i, j) contain the same symbol is performed when the cell ($i-1$, $j+1$) is processed. In summary, two checks need to be performed for each matrix cell.

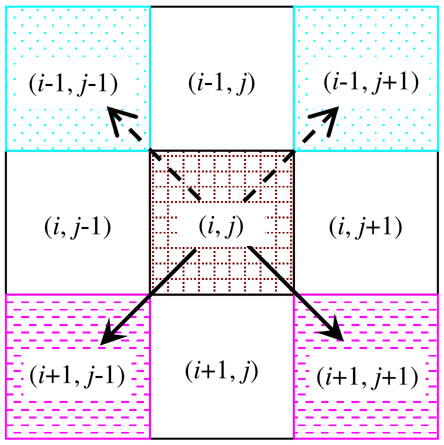

Fig. 6 Neighboring cells of the cell (i, j) in the matrix.

$(n!)^2$ is a large number for a moderately large n. Consequently, even for a single seed orthogonal Latin square pair, it can be time-consuming to enumerate all $(n!)^2$ orthogonal Latin square pairs through simultaneous row or column permutation and check whether they satisfy the minimal distance constraint. Nevertheless, the time overhead of constructing distance-constrained, orthogonal Latin square pairs is not a major concern in practice for the following reasons.

First, the distance-constrained, orthogonal Latin square pairs need to be constructed only once. Then they can be saved and repeatedly used in P300 BCI.

Second, the order n used in P300 BCI is usually not large. If n is large, the computer screen will be cluttered with too many objects (characters or pictures). Due to insufficient space between neighboring objects, the performance of the P300 BCI system will degrade. Actually a multi-step selection procedure will be used when the user needs to select from a large number of objects [9]. For example, suppose 256 objects are available for selection. We divide these 256 objects into 16 disjoint groups, each containing 16 objects. To select a particular object, the user proceeds in two steps. In the first step, he is presented with 16 groups and selects the group that contains this object. In the second step, he is presented with 16 objects in that group and selects this object.

Third, similar to the method described in [6], our procedure of searching distance-constrained, orthogonal Latin square pairs can be easily parallelized to achieve near-linear speedup on multiple computers.

Fourth, as will be shown in Section 3, exhaustive enumeration is often unnecessary. Usually through partial enumeration of simultaneous row or column permutations, we can quickly obtain a large number of orthogonal Latin square pairs satisfying the minimal distance constraint. These distance-constrained, orthogonal Latin square pairs are sufficient for the purpose of being used in P300 BCI. On the other hand, suppose we use a distance constraint that is much stricter than the minimal distance constraint and hence can be satisfied by very few orthogonal Latin square pairs. Then a lot of time may have to be spent on enumerating many orthogonal Latin square pairs through simultaneous row or column permutation, possibly using multiple seed orthogonal Latin square pairs.

During our experiments, we notice that each orthogonal Latin square pair seems to have some inherent, distance-related property. For example, if we cannot use a seed orthogonal Latin square pair to quickly (e.g., within two minutes) obtain any orthogonal Latin square pair satisfying the minimal distance constraint, then we are unlikely to find any orthogonal Latin square pair satisfying the minimal distance constraint even after all possible, simultaneous row or column permutations are exhausted. On the other hand, if we can use this seed orthogonal Latin square pair to quickly obtain a few orthogonal Latin square pairs satisfying the minimal distance constraint, then we are likely to find many orthogonal Latin square pairs satisfying the minimal distance constraint by exhausting all possible, simultaneous row or column permutations.

One possible explanation of this phenomenon is as follows. For a seed orthogonal Latin square pair of order n, the corresponding search space includes the $(n!)^2$ orthogonal Latin square pairs obtained through simultaneous row or column permutation. A small difference in two permutations (e.g., the positions of two symbols are switched) can affect the neighboring relationships of many symbols in the two matrices of the orthogonal Latin square pair (e.g., two rows or columns are simultaneously switched). For the n symbols 1, 2, …, and n, we can enumerate all $n!$ distinct permutations in a regular way. For example, first enumerate all n possibilities of the first element in the permutation, then enumerate all $n-1$ possibilities of the second element in the permutation, etc. Nevertheless, the resulting search in the search space is essentially performed in a "random" way by quickly jumping through dissimilar matrices whose contained symbols have dramatically different neighboring relationships. In a short amount of time, we can probe many scattered places of the search space. If no qualified orthogonal Latin square pair is found during this period, then the likelihood that the search space contains any qualified orthogonal Latin square pair would be low. Further investigation of the inherent, distance-related property of orthogonal Latin square pairs seems to be non-trivial and is an interesting area for future work.

Our goal is to quickly construct enough distance-constrained, orthogonal Latin square pairs for the purpose of P300 BCI, rather than find all possible distance-constrained, orthogonal Latin square pairs. The above phenomenon suggests using a time-based heuristics to improve the efficiency of constructing distance-constrained, orthogonal Latin square pairs. The idea is to quickly discard those seed orthogonal Latin square pairs that are unlikely to produce any distance-constrained, orthogonal Latin square pair. More specifically, we have a predetermined time threshold of *m* (e.g., *m*=2) minutes. If within this amount of time, we cannot use a seed orthogonal Latin square pair to obtain any orthogonal Latin square pair satisfying the minimal distance constraint, then we switch to a new seed orthogonal Latin square pair rather than keep trying with the old seed orthogonal Latin square pair.

Fig. 7 shows the flow chart of constructing distance-constrained, orthogonal Latin square pairs. The counter c_S keeps track of the number of distance-constrained, orthogonal Latin square pairs that have been constructed from the seed orthogonal Latin square pair *S*. The search procedure stops when we have found enough (e.g., a few hundred) distance-constrained, orthogonal Latin square pairs for the purpose of P300 BCI.

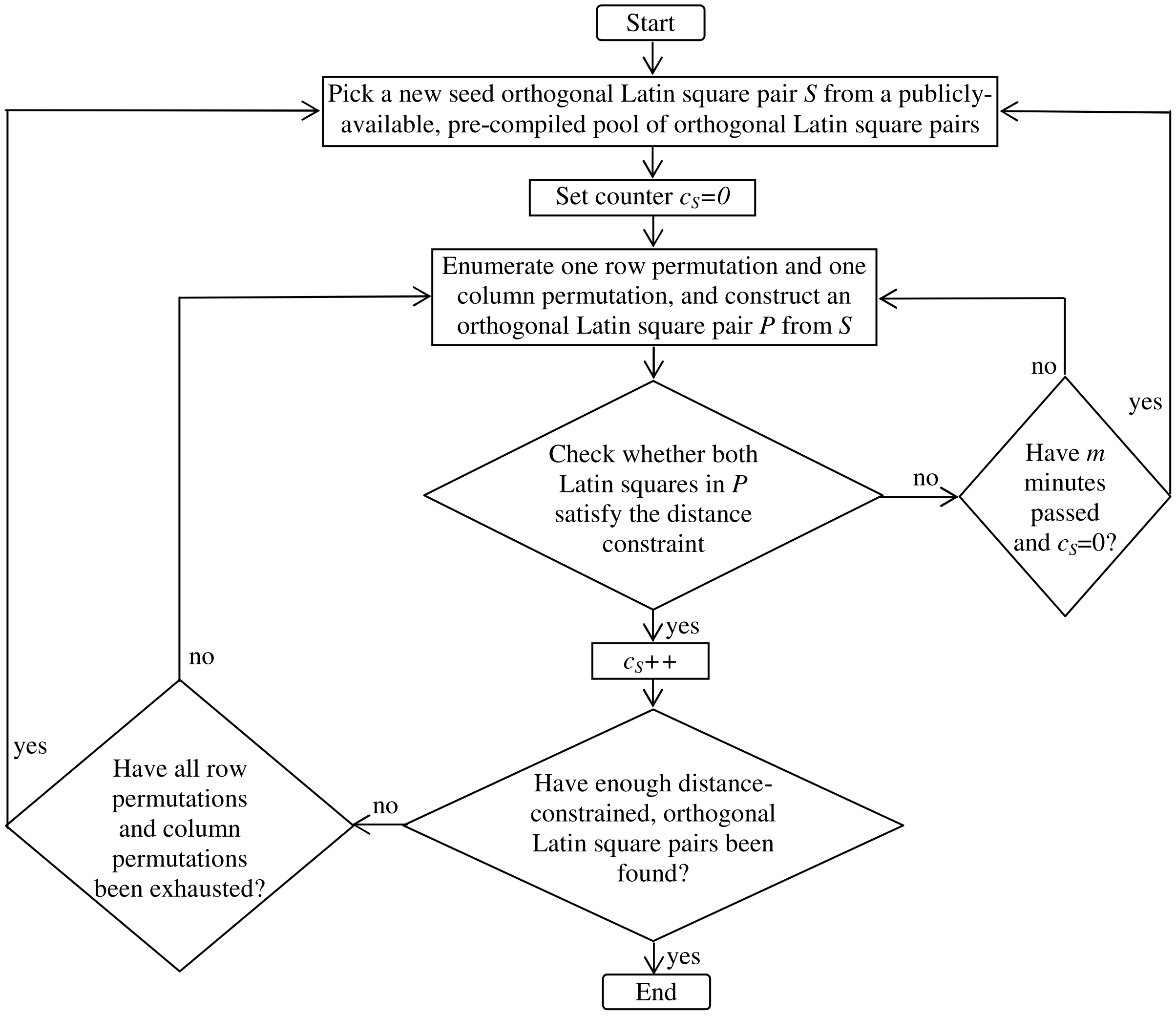
Start
Pick a new seed orthogonal Latin square pair S from a publicly-available, pre-compiled pool of orthogonal Latin square pairs
Set counter $c_S=0$
Enumerate one row permutation and one column permutation, and construct an orthogonal Latin square pair P from S
Check whether both Latin squares in P satisfy the distance constraint
no
Have m minutes passed and c_S=0?
no
yes
yes
c_S++
Have enough distance-constrained, orthogonal Latin square pairs been found?
no
Have all row permutations and column permutations been exhausted?
no
yes
yes
End

Fig. 7 Flow chart of constructing distance-constrained, orthogonal Latin square pairs.

3. Experimental results

We performed some experiments to demonstrate that many orthogonal Latin square pairs satisfying the minimal distance constraint can be systematically and efficiently constructed using the method described in Section 2. 26 characters, 10 digits, and a few symbols require more than 36 matrix cells but no more than 81 matrix cells. In this section, we focus on Latin squares of order seven, eight, and nine, which represent the matrix sizes commonly used in P300 BCI. All used seed self-orthogonal Latin squares come from Burger *et al*. [6]. Our experiments were performed on a computer with two 3GHz processors, 2GB memory, and one 111GB disk.

We first consider orthogonal Latin square pairs of order seven. As mentioned in Section 2, using the self-orthogonal Latin square in Fig. 3 and its transpose as the seed orthogonal Latin square pair, we can obtain 588 orthogonal Latin square pairs satisfying the minimal distance constraint. Some of these distance-constrained, orthogonal Latin square pairs are each composed of one self-orthogonal Latin square and its transpose. One such distance-constrained, self-orthogonal Latin square is shown in Fig. 8. The enumeration of all possible, simultaneous row or column permutations is finished within one minute.

0	2	4	5	6	3	1
3	1	0	2	4	5	6
5	6	3	1	0	2	4
2	4	5	6	3	1	0
1	0	2	4	5	6	3
6	3	1	0	2	4	5
4	5	6	3	1	0	2

Fig. 8 A self-orthogonal Latin square of order seven that satisfies the minimal distance constraint.

Although we start from a self-orthogonal Latin square, the seed orthogonal Latin square pair does not have to be composed of a self-orthogonal Latin square and its transpose. For example, using simultaneous row or column permutation, any orthogonal Latin square pair obtained through simultaneous row or column permutation of a self-orthogonal Latin square *L* and *L*'s transpose will produce the same set of orthogonal Latin square pairs as *L* and *L*'s transpose, as the composition of two permutations is still a permutation.

0	2	1	4	5	6	7	3
3	1	7	0	2	4	5	6
4	6	2	5	7	3	0	1
5	7	6	3	0	1	2	4
7	5	3	6	4	2	1	0
1	3	0	7	6	5	4	2
2	0	4	1	3	7	6	5
6	4	5	2	1	0	3	7

Fig. 9 A self-orthogonal Latin square of order eight.

Next, we consider orthogonal Latin square pairs of order eight. Fig. 9 shows a self-orthogonal Latin square of order eight. Using this self-orthogonal Latin square and its transpose as the seed orthogonal Latin square pair, we can obtain 55,296 orthogonal Latin square pairs satisfying the minimal distance constraint. One such distance-constrained, orthogonal Latin square pair is shown in Fig. 10. The enumeration of all possible, simultaneous row or column permutations is finished within one hour.

0	2	3	5	6	7	4	1
3	1	6	2	4	5	0	7
5	7	4	0	1	2	3	6
4	6	1	7	3	0	5	2
7	5	0	4	2	1	6	3
1	3	2	6	5	4	7	0
2	0	5	3	7	6	1	4
6	4	7	1	0	3	2	5

0	3	6	7	1	2	5	4
2	1	4	5	3	0	7	6
4	0	2	6	7	1	3	5
1	7	5	3	0	4	6	2
5	2	1	4	6	3	0	7
6	4	0	2	5	7	1	3
7	5	3	1	4	6	2	0
3	6	7	0	2	5	4	1

Fig. 10 A pair of orthogonal Latin squares of order eight that satisfy the minimal distance constraint.

0	2	1	4	5	6	7	3
3	1	6	7	0	2	4	5
4	3	2	5	7	0	1	6
2	0	7	3	6	1	5	4
6	5	3	1	4	7	0	2
7	6	4	0	2	5	3	1
1	7	5	2	3	4	6	0
5	4	0	6	1	3	2	7

Fig. 11 Another self-orthogonal Latin square of order eight.

Not every orthogonal Latin square pair can be used as a seed to guarantee the construction of at least one orthogonal Latin square pair satisfying the minimal distance constraint. For example, the self-orthogonal Latin square shown in Fig. 11 and its transpose form an orthogonal Latin square pair, which cannot be used a seed to construct any orthogonal Latin square pair satisfying the minimal distance constraint.

0	2	3	4	5	6	7	8	1
4	1	5	8	6	3	2	0	7
5	8	2	6	1	7	4	3	0
6	0	1	3	7	2	8	5	4
7	5	0	2	4	8	3	1	6
8	7	6	0	3	5	1	4	2
1	3	8	7	0	4	6	2	5
2	6	4	1	8	0	5	7	3
3	4	7	5	2	1	0	6	8

Fig. 12 A self-orthogonal Latin square of order nine.

Finally, we consider orthogonal Latin square pairs of order nine. Fig. 12 shows a self-orthogonal Latin square of order nine. Using this self-orthogonal Latin square and its transpose as the seed orthogonal Latin square pair, within three hours we can obtain one million orthogonal Latin square pairs satisfying the minimal distance constraint. One such distance-constrained, orthogonal Latin square pair is shown in Fig. 13. Some of these distance-constrained, orthogonal Latin square pairs are each composed of one self-orthogonal Latin square and its transpose. One such distance-constrained, self-orthogonal Latin square is shown in Fig. 14. It takes two days to enumerate all possible, simultaneous row or column permutations to obtain the complete set of 13,716,864 orthogonal Latin square pairs satisfying the minimal distance constraint. This shows that early stopping via partial enumeration can be valuable, as hundreds of distance-constrained, orthogonal Latin square pairs should be sufficient for the purpose of being used in P300 BCI.

0	2	3	4	1	5	8	7	6
4	1	5	8	7	6	0	2	3
5	8	2	6	0	1	3	4	7
6	0	1	3	4	7	5	8	2
8	7	6	0	2	3	4	1	5
1	3	8	7	5	0	2	6	4
2	6	4	1	3	8	7	5	0
7	5	0	2	6	4	1	3	8
3	4	7	5	8	2	6	0	1

0	4	5	6	3	7	2	1	8
2	1	8	0	4	5	6	3	7
3	5	2	1	7	0	4	8	6
4	8	6	3	5	2	1	7	0
6	3	7	2	1	8	0	4	5
7	2	4	8	0	3	5	6	1
8	0	3	5	6	1	7	2	4
5	6	1	7	2	4	8	0	3
1	7	0	4	8	6	3	5	2

Fig. 13 A pair of orthogonal Latin squares of order nine that satisfy the minimal distance constraint.

0	2	3	4	6	7	8	5	1
4	1	5	8	3	2	0	6	7
5	8	2	6	7	4	3	1	0
6	0	1	3	2	8	5	7	4
8	7	6	0	5	1	4	3	2
1	3	8	7	4	6	2	0	5
2	6	4	1	0	5	7	8	3
7	5	0	2	8	3	1	4	6
3	4	7	5	1	0	6	2	8

Fig. 14 A self-orthogonal Latin square of order nine that satisfies the minimal distance constraint.

4. Conclusions

Orthogonal Latin square pairs satisfying certain distance constraint can be used to increase both the classification accuracy and communication speed of the P300 BCI system. This paper shows that for every matrix size commonly used in P300 BCI, thousands to millions of such distance-constrained, orthogonal Latin square pairs can be systematically and efficiently constructed. A direction for future work is to investigate quantitatively the benefits that these distance-constrained, orthogonal Latin square pairs can bring to P300 BCI, particularly on ALS patients.

Acknowledgment We thank Dennis J. McFarland, Selena B. Thomas, Theresa M. Vaughan, and Catherine Wolf for helpful discussions.

References

1. Dornhege, G., Millan, J.R., and Hinterberger, T. *et al.*, *Toward Brain-Computer Interfacing (Neural Information Processing)*. MIT Press, 2007.
2. Nijholt, A., Tan, D.S., and Pfurcheller, G. *et al.*, Brain-computer interfacing for intelligent systems. *IEEE Intelligent Systems* 23(3): 72-79, 2008.
3. Sellers, E.W., Krusienski, D.J., and McFarland, D.J. *et al.*, A P300 event-related potential brain-computer interface (BCI): the effects of matrix size and inter stimulus interval on performance. *Biological Psychology* 73(3): 242-252, 2006.
4. Okamoto, K., Hirai, S., and Amari, M. *et al.*, Oculomotor nuclear pathology in amyotrophic lateral sclerosis. *Acta Neuropathologica* 85(5): 458-462, 1993.
5. Min, W., and Luo, G., Medical applications of EEG wave classification. *Chance* 22(4): 14-20, 2009.
6. Burger, A.P., Kidd, M.P., and van Vuuren, J.H., Enumeration of self-orthogonal Latin squares. Available at http://www.vuuren.co.za/papers/SOLS.pdf, 2009.
7. Laywine, C.F., and Mullen, G.L., *Discrete Mathematics Using Latin Squares*. Wiley-Interscience, 1998.
8. Street, A.P., and Street, D.J., *Combinatorics of Experimental Design*. Oxford University Press, 1987.
9. Muller, K., and Blankertz, B., Toward noninvasive brain-computer interfaces. *IEEE Signal Processing Magazine* 23(5): 125-128.