1 Methods

1.1 Probabilistic Model of Visual-Speller BCIs

We derive a generative model of the BCI speller system and then use this model to guide the generation and selection of codebooks which are optimal in terms of information-theoretic and psychophysiological criteria. We assume an N-letter alphabet Γ and an N-letter by L-bit codebook C. The basic demodulation and decoding procedure consists of finding the letter $\hat{T}$ among the possible letters $t \in \Gamma$ showing the largest probability $\Pr(t|X)$ of being the target letter T, given C and the measured brain signals $X = [x_1, \ldots, x_L]$, i.e.,

$$\hat{T} = \underset{t\in\Gamma}{\operatorname{argmax}} \Pr(t|X) = \underset{t\in\Gamma}{\operatorname{argmax}} \frac{\Pr(X|t)\Pr(t)}{\Pr(X)} , \tag{1}$$

where the second equality follows from Bayes' rule. A simple approach to decoding is to treat the individual binary epochs, with binary labels $\underline{c} = (C_{t1} \ldots C_{tL})$, as independent. This allows us to factor $\Pr(X|t)$ into per-epoch probabilities $\Pr(x_j|\underline{c})$ to give

$$\Pr(t|X) = \frac{\Pr(t)}{\Pr(X)} \prod_{j=1}^{L} \Pr(x_j|\underline{c}) = \frac{\Pr(t)}{\Pr(X)} \prod_{j=1}^{L} \frac{\Pr(C_{tj}|x_j)\Pr(x_j)}{\Pr(C_{tj})} = f_t(X) , \tag{2}$$

where the second equality again follows from Bayes' rule.

This form of Bayesian decoding [3] forms the basis for our decoding scheme. We train a probabilistic discriminative classifier, in particular a linear logistic regression (LR) classifier [1, pp82-85], to estimate $\Pr(C_{tj}|x_j) = p_j$ in (2) for $j = 1 \ldots L$. As a result, we can obtain estimates of the probability $\Pr(t|X)$ that a particular letter t corresponds to the user-selected codeword. Note that for decoding purposes the terms $\Pr(X)$ and $\Pr(x_j)$ can be ignored as they are independent of t. Furthermore, the product $\prod_j \Pr(C_{tj})$ depends only on the positive-class prior of the binary classifier, $\Pr(+)$. In fact, it is easy to show that during decoding this term cancels out the effect of the binary prior, which may therefore be set arbitrarily without affecting the decisions made by our decoder. The simplest thing to do is to train classifiers with $\Pr(+) = 0.5$, in which case the denominator term is constant for all t.

1.1.1 Modelling demodulation and decoding errors

We used a simple model of subjects' responses in each epoch in order to estimate the probability of making a prediction error with the above decoding method. The predicted letter $\hat{T}$ is the letter with maximal *estimated* $\Pr(t|X) = f_t(X)$. The probability that we mistakenly predict some other target W when the true target is T is $\Pr(f_T(X) < f_W(X)|T) = \Pr(\log f_T(X) < \log f_W(X)|T) = \Pr\left(\log \frac{f_T(X)}{f_W(X)} < 0|T\right)$. Thus the probability of error is simply the probability that the *estimated* log likelihood ratio of the two letters, given that T was actually transmitted, is less than 0. Using (2) to substitute for $f_{(.)}(X)$ we obtain

$$\log \frac{f_T(X)}{f_W(X)} = \log \frac{\Pr(T)}{\Pr(W)} + \sum_{j=1}^{L} \log \frac{\Pr(x_j|C_{Tj})}{\Pr(x_j|C_{Wj})} = \log \frac{\Pr(T)}{\Pr(W)} + \sum_{j=1}^{L} (C_{Tj} - C_{Wj}) \log \frac{\Pr(x_j|+)}{\Pr(x_j|-)}, \tag{3}$$

where the last equality follows from expanding the 4 possible combinations of values for C_{Tj} and C_{Wj}. Note, all the values in (3) are implicitly conditioned on the true transmitted letter being T. We will use the superscript notation $x^{|T}$ to flag quantities for which this dependence must be modelled.

Let ${z_j}^{|T}$ denote the terms over which we sum in (3). ${z_j}^{|T}$ is simply the logged likelihood ratio as estimated by the classifier. In general this ratio is difficult to estimate. However, if (as is used in the derivation of the linear LR classifier) we assume that the positive and negative classes are drawn from equal covariance Gaussian distributions, then the ${z_j}^{|T}$ will also have a Gaussian distribution.

The mean ${\mu_j}^{|T}$ and standard deviation ${\sigma_j}^{|T}$ of ${z_j}^{|T}$ depend on the true letter T, making it difficult to estimate them for new codewords in general. In constructing a model of ${\mu_j}^{|T}$ and ${\sigma_j}^{|T}$ based

on the two competition data-sets, we find that this Gaussianity assumption is well satisfied if we only condition on the current bit C_{tj} and the time elapsed since the last target stimulus. Thus we hypothesize that it is possible to learn $\mu(\underline{c}, \text{TPT})$ and $\sigma(\underline{c}, \text{TPT})$ on one set of codebooks and use them to predict the error rate on new ones.

Being a sum of Gaussians, (3) is also Gaussian distributed, with mean ${b_W}^{|T} = \log \Pr(T) / \Pr(W) + \sum_j (C_{Tj} - C_{Wj}){\mu_j}^{|T}$ and variance $({s_W}^{|T})^2 = \sum_j (C_{tj} - C_{ij})^2 ({\sigma_j}^{|T})^2$. Thus, the probability of confusing the true transmitted letter T with letter W is $\phi({b_W}^{|T}/{s_W}^{|T})$, where ϕ is the cumulative Gaussian function. The probability of confusing T with *any* other letter is $\Pr(\bigvee_W f_T(X) < f_W(X)|T)$ which, due to the common conditioning variable in all the ${z_j}^{|T}$, can be evaluated as an N-dimensional cumulative Gaussian. For this, we use the numerical algorithm `MVNDST` of [2].

We use this function in two ways. Firstly, we use it to compute the *codebook loss* $\mathcal{L}(\mathrm{C})$ given a complete codebook C. This is the sum of error probabilities, weighted by the probability of transmission of each letter, $\mathcal{L}(C) = \sum_T \Pr(T) \Pr(\bigwedge_i f_T(X) < f_W(X)|T)$. Secondly, we use the pairwise probability of confusing 2 codewords $\mathcal{L}_2(u, v) = [\Pr(f_u(X) < f_v(X)|u) + \Pr(f_v(X) < f_u(X)|v)]/2$ as a heuristic to guide the selection candidate codewords to add to a partially filled codebook.

1.1.2 Codebook Optimization

In order to explore the relative influence of the information theoretic and psychophysiological effects, two optimized codebooks were constructed; one with maximal minimal Hamming distance (denoted **D10**), and one which minimized the codebook loss (denoted $\mathbf{D8}_{\text{opt}}$).

In general, finding an such optimal codebook is a hard combinatorial optimization problem, where we must choose N codewords each from a potential space of 2^L possibilities. We approximately solve this problem using a branch-and-bound search strategy with a heuristic to decide greedily which codeword to add next to the current solution. Unfortunately, the heuristics alone were not powerful enough to generate good solutions in reasonable time. Therefore, additional pruning criteria, based on Hamming distance and the mean TTI were used to reduce the set of candidate codewords considered at each branch point. Although the use of heuristics and pruning rules means that each candidate codebook was not constructed based purely on maximization of the stated objective, in both cases the final selection among candidates was performed purely according to the maximum value of the respective objective functions. For simplicity we assumed a flat prior over letters (see last paragraph of discussion).

The two "optimized" codes were generated as follows:

- **D10** was a 24-bit code with the largest minimum Hamming distance we could achieve ($d_{\text{min}} = 10$) using the search method. The heuristic for codeword selection was the minimum distance to previously selected codewords, and the criteria for selection were (first) d_{min} and (second, to select among a large number of $d_{\text{min}} = 10$ candidates) the lowest number of consecutive targets.
- $\mathbf{D8}_{\text{opt}}$ was a 24-bit code optimized according to our model. The heuristic for greedy selection of a new candidate codeword u was the pairwise codebook loss $\mathcal{L}_2(u, v)$ summed across each previously selected codebook entry v. The final selection criterion was our overall codebook loss function $\mathcal{L}()$.

We should note that, since this experimental study was performed, we have discovered that $d_{\text{min}} = 12$ is possible for this problem, using a Hadamard code. This would have been preferable to **D10**for use as an experimental examination of the extreme worry-about-Hamming-distance-and-not-at-all-about-TTI end of the tradeoff.

References

[1] Bishop CM (1995) Neural Networks for Pattern Recognition. *Clarendon Press, Oxford.*

[2] Genz A. (1992) Numerical Computation of Multivariate Normal Probabilities. *Journal of Computational and Graphical Statistics* **1**: 141–149

[3] Gestel T, *et al*. (2002) Multiclass LS-SVMs: Moderated Outputs and Coding-Decoding Schemes *Neural Processing Letters*, **15**: 45–48