FROM PHONEMES TO SENTENCES IN THE BRAIN: COMPOSING NEURAL PRIMITIVES FOR SPEECH DATA SYNTHESIS

Anonymous authors
Paper under double-blind review

ABSTRACT

Compositionality has been proposed as an organizing principle for how neural dynamical primitives are combined to produce movements in simple motor control tasks, such as reaching. Here, we extend this idea to speech by proposing a simple and interpretable mechanistic model of how premotor cortex constructs sentence level dynamics from reusable neural trajectories across multiple hierarchical linguistic levels: phonemes, biphones, syllables, and words. We compare it with a temporal convolutional network (TCN) that can capture non-linear interactions over a longer temporal context. We verify whether the neural activity generated by both models reproduces temporal and spectral properties of the recorded data. Both models preserve speech relevant information: a decoder trained on real neural recordings achieves comparable phoneme decoding performance on synthetic and held out real activity, across all four datasets tested. Finally, we demonstrate that augmenting real datasets with model generated activity improves speech decoding in low data regimes, reducing phoneme error rate by up to 27 percentage points with the TCN and 22 percentage points with the linear model relative to training on real data alone. This is particularly relevant for brain computer interfaces (BCIs), where neural recordings are scarce, while modern decoding models are increasingly data hungry.

1 INTRODUCTION

Recent BCI speech decoding approaches, from recurrent neural networks to foundation models, have pushed phoneme error rate (PER) below 10% (Card et al., 2024; Willett et al., 2023; 2024; Zhang et al., 2026; Boccato et al., 2026b). However, these models typically improve with increasing amounts of training data, while intracortical speech recordings remain costly, scarce, and difficult to collect at scale.

One line of research has tried to address this lack of available data at scale by pursuing sample-efficient decoders that are trained on collections of recordings spanning multiple subjects, tasks, species, and acquisition modalities (Zhang et al., 2026; Azabou et al., 2023; 2025; Ye & Pandarinath, 2021; Le et al., 2026; Willeke et al., 2026; Boccato et al., 2026a; Fogg et al., 2026; Boccato et al., 2026b).

An alternative strategy, is to augment a single participant's own limited recordings with synthetically generated samples. Data augmentation is standard practice for images (Krizhevsky et al., 2012), audio (Park et al., 2019), and text (Wei & Zou, 2019), because it is well understood which transformations preserve the physics of how data was generated: a rotated or cropped image is still a valid image, a pitch-shifted waveform is still a valid sound. However, for invasive neural recordings of speech, we do not have a principled model of how phonemes, words, and sentences are encoded in spiking or field-potential activity. Ad hoc approaches instead add Gaussian noise and offsets to neural activity, channel dropout and time-masking modelling (Feghhi et al., 2026). Recently, a diffusion-based generative model with a low-dimensional latent space (LDNS) has been proposed to generate realistic spiking data (Kapoor et al., 2024), but it is a general-purpose model of population dynamics, not built or evaluated for speech decoding augmentation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Despite the strong decoding performance of recent BCI speech decoders, the neural mechanisms by which the brain constructs speech motor plans remain poorly understood (Silva et al., 2024). Compositionality has been proposed as a fundamental organizing principle of motor control (Thoroughman & Shadmehr, 2000; Zimnik & Churchland, 2021; Shah et al., 2025). We suggest that speech motor plans may also be constructed from reusable neural primitives. This compositional view could both provide a mechanistic account of speech production and enable the principled generation of synthetic neural data for all possible words and sentences. More recently, using a controlled task in which participants attempted to produce pseudowords differing by a single phoneme, Krasa et al. (2026) showed that premotor cortical activity exhibits a compositional neural geometry during word preparation. Importantly, the proposed model predicts a single population activity vector over the preparatory period of individual words by combining position-specific phoneme regressors.

Here, we test this hypothesis by using neural trajectories recorded from the human premotor cortex in an open-ended naturalistic task, where participants imagine or attempt to produce full sentences (Fig. 1 (a)). We introduce a simple linear model that composes reusable dynamical neural primitives across multiple linguistic levels: phonemes, biphonemes, syllables, and words, to produce sentence level neural activity during speech production (Fig. 1 (b)). We then compare it with the real neural data, in terms of dynamics, statistical properties, and speech decoding accuracy. We further test whether a temporal convolutional network (TCN) that models nonlinear interactions among these primitives over a longer temporal context more accurately captures neural activity and improves speech decoding. Finally, we investigate whether augmenting the training set with synthetically generated neural trajectories, from both the linear and TCN models, improves decoding accuracy (Fig. 1 (c)).

2 METHODS

2.1 TESTING FOR SIGNATURES OF COMPOSITIONALITY IN HUMAN PREMOTOR CORTEX

In order to test for compositionality in the premotor cortex we used publicly available datasets in which human participants imagined or attempted to produce sentences presented on a screen while neural activity was recorded from premotor cortex (Card et al., 2024; Willett et al., 2023; Kunz et al., 2025) (check Appendix A.2 for a full description of the datasets). We aligned the neural trajectories to individual words using forced-alignment (Viterbi, 1967; Graves et al., 2006; Hira) and constructed a template for each phoneme by averaging its trajectory across all words (Fig. 2(a)).

We tested whether word level neural trajectories could be explained as compositions of phoneme level trajectories. Specifically, we asked whether the *shape* of a phoneme trajectory was preserved across words in the subspace spanned by the first five PCs, up to a Procrustes transformation (Gower, 1975), which allows translation, rotation, reflection, and global scaling. Trajectories were first resampled to a common length of ten time points via linear interpolation, to account for variable phoneme duration. For each phoneme occurrence within a word, we applied the Procrustes transformation to align every candidate phoneme template (computed while excluding the word being tested), to the observed trajectory, and then computed the residual error (Fig. **??**(b), right). We ranked the candidate templates by their residuals, with lower ranks indicating a better match. To construct the null distribution, we shuffled phoneme labels within each recording day before recomputing templates 10,000 times (Fig. 2(c)). If phoneme trajectories are reused across words, the true phoneme template should yield a lower residual, and therefore a lower rank, than expected after label shuffling.

2.2 COMPOSITIONAL MODEL OF SENTENCE PRODUCTION IN PREMOTOR CORTEX

We designed a model built from reusable kernels, or *stamps*, one per phoneme, that combine to form words and sentences. Each kernel describes how activity across all channels changes in a window of time following a given phoneme's onset. In particular, to generate activity for a word or sentence, we place a copy of each occurring phoneme's kernel at its onset time and sum them, so the same kernel is reused every time that phoneme appears. The predicted activity is given by

$$\hat{x}(c,t) = \sum_{p} \sum_{\tau=0}^{T} K_p(c,\tau)\, \delta(t - t_p - \tau) + b_{\text{day}}(c)\,, \tag{1}$$

where c denotes a channel, t a time step, p indexes each phoneme occurrence in the sentence with onset time t_p, T the kernel duration, and $K_p(c,\tau)$ the kernel fit for phoneme p, channel c, and time lag τ. The unit impulse $\delta(t - t_p - \tau)$ places this kernel value at time $t = t_p + \tau$, and $b_{\text{day}}(c)$ is a channel-specific bias for each recording day. We then extended the phoneme only model with kernels representing progressively higher hierarchical levels of linguistic structure: biphones, syllables, and words

$$\hat{x}(c,t) = \sum_{L} \sum_{p_L} \sum_{\tau=0}^{T} K_{p_L}(c,\tau)\, \delta(t - t_{p_L} - \tau) + b_{\text{day}}(c)\,, \tag{2}$$

where L denotes the hierarchical level and p_L the primitives for that hierarchical level (e.g. phonemes in the first level, biphonemes in the second level, and so on). Kernels and day biases were fit by ridge regression, with the regularization parameter cross-validated to maximize the variance in real activity explained by the model.

We model residuals $r(c,t) = x(c,t) - \hat{x}(c,t)$ as a first-order autoregressive (AR(1))

$$\mathbf{r}_t \;=\; \boldsymbol{\phi} \odot \mathbf{r}_{t-1} \;+\; \boldsymbol{\eta}_t, \qquad \boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0},\, \Sigma), \tag{3}$$

where $\mathbf{r}_t \in \mathbb{R}^C$ is the residual vector across all C channels at time t, $\boldsymbol{\phi} \in (-1,1)^C$ are channel-specific AR(1) coefficients controlling temporal persistence, and $\Sigma \in \mathbb{R}^{C\times C}$ is the innovation covariance matrix, capturing how the unpredictable component of the residuals co-vary across channels at a given instant.

The fitted parameters $\hat{\phi}$ and $\hat{\Sigma}$ determine the temporal and cross-channel covariance structure of the noise. A separate scalar gain γ scales a residual trace $r_{\text{sampled}}(c,t)$, simulated from this fitted AR(1) process, before adding it to the deterministic kernel-sum prediction

$$x_{\text{synth}}(c,t) = \hat{x}(c,t) + \gamma\, r_{\text{sampled}}(c,t)\,. \tag{4}$$

Unlike ϕ and Σ, which are fit directly from residual statistics, γ is tuned as a hyperparameter, chosen to minimize PER of the downstream BiT decoder on a held-out validation split, kept separate from the test split used for the decoder's own checkpoint selection in later experiments. To generate a complete synthetic trial, each phoneme's duration is additionally sampled from the empirical distribution of durations observed for that phoneme in the real data. An example of the synthetic neural activity our model generates for "How are you?" is shown in Fig. 1.

We considered autoregressive noise models with higher order, and additionally vector autoregressive (VAR) models that model correlations between channels across time. These approximated better the real data but led to higher PER on the validation data split (Appendix A.3), and we therefore considered the AR(1) model for the future augmentation experiments.

2.3 Extending to deep non-linear architecture: temporal convolution network

The compositional model above assumes neural activity is a linear superposition of each primitive's kernel, triggered by it's onset. We tested whether relaxing these assumptions would improve synthetic data fidelity and, in turn, data augmentation, by fitting a temporal convolutional network (TCN) (Bai et al., 2018). The TCN can model nonlinear interactions among primitives, integrate information over a longer temporal context, and account for the duration of each primitive.

At each time bin, the TCN receives a 64-dimensional learnable embedding of the phoneme identity active at that bin, held constant across the full duration of the segment, together with a binary onset indicator (1 at the exact bin where a new phoneme begins, 0 otherwise) and a day embedding. The TCN is trained to minimize the mean squared error between the predicted and observed neural activity

$$\mathcal{L} = \frac{1}{C \sum_n D_n} \sum_{n} \sum_{c=1}^{C} \sum_{t=1}^{D_n} \Big(x_n(c,t) - \hat{x}_n(c,t) \Big)^2\,,$$

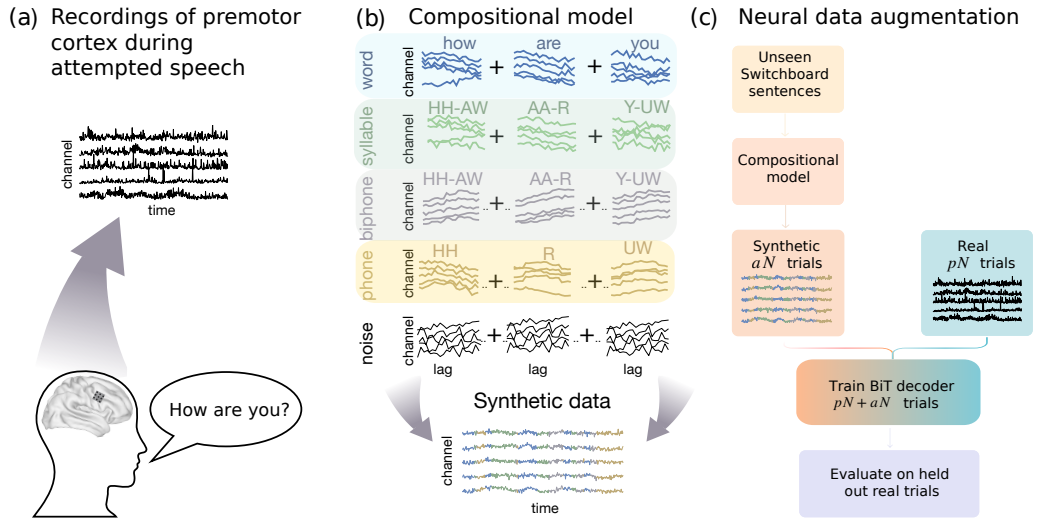


Figure 1: Overview of the experimental paradigm. (a) We use public available data from participants implanted with intracortical microelectrode arrays in premotor cortex who imagined or attempted to produce speech. (b) Decomposing the sentence "How are you?" into primitives at different levels of the linguistic hierarchy. The kernel learned for each primitive for five example channels is shown below; these kernels are shifted to their primitive's onset and summed to obtain the predicted activity for the full sentence. Finally, noise is added to produce the synthetic activity shown at the bottom. (c) We use the fitted model to generate synthetic activity for unseen sentences from the Switchboard dataset. Denoting by N the total number of real trials, we run experiments retaining pN real trials and augmenting them with aN synthetic trials, training BiT decoder and evaluating the phoneme error rate (PER) on held-out real data. The parameters p and a denote the fraction of real and augmented data respectively.

where n indexes trials and D_n is the number of time bins in trial n. The network consists of a first convolution layer projecting the input to 768 channels, followed by 12 residual blocks and a final head projecting back down to the number of recorded channels. Each residual block contains two sublayers, each comprising group normalization, a GELU nonlinearity, and a dilated non-causal convolution. Dilation doubles across successive blocks, progressively expanding the temporal receptive field.

The residuals were modeled as AR(1) processes, as in the linear model. To compare with the increasing hierarchical levels used for the linear model (Eq. 2), we also consider a hierarchical TCN variant, in which each linguistic level (phoneme, biphone, syllable, and word) was processed by its own independent sub-network, with the final signal given by the sum of all active levels' outputs, mirroring the additive combination of level-specific kernels in the linear model.

To evaluate the linear and TCN models, we compare phoneme decoding accuracy on real and synthetic data across the Card, Willett, Kunz T12, and Kunz T15 datasets, using BiT, a transformer-based neural decoder trained separately for phoneme decoding on each subject (Feghhi et al., 2026; Zhang et al., 2026). For each dataset, we generate the same number of synthetic training trials as real ones, so decoding performance is compared at matched sample sizes. We also verify if the synthetic data reproduces key temporal and spectral statistics of the real neural signals.

2.4 Generating synthetic data and evaluating on held-out real data

We investigated whether augmenting the training set with synthetically generated neural trajectories improves decoding accuracy. Specifically, we generated trajectories from sentences in the Switchboard dataset (Godfrey et al., 1992) that participants had not previously spoken. We then trained the BiT for 800 epochs using mixtures of real and synthetic data, varying the fraction of real (p) and synthetic data (a) (Benchetrit et al., 2026). Performance was evaluated as the best phoneme error rate (PER) on a fixed held-out real test set for trajectories generated by the linear and TCN models. We considered the sets $p = \{0, 0.1, 0.25, 0.5, 0.75, 1\}$ and $a = \{0, 0.1, 0.2, 0.5, 1, 2, 4, 8\}$.

The linear and TCN generative models were each trained either on all available training trials (N) or only on the subset of real trials used to train the decoder (pN). Finally, we repeated a reduced set of p and a conditions without smoothing or noise-injection augmentation in the BiT training pipeline.

3 RESULTS

3.1 SIGNATURES OF COMPOSITIONALITY IN HUMAN PREMOTOR CORTEX

We constructed a template for each phoneme by averaging its trajectory across all words (Fig. 2(a)). As an initial qualitative test, we compared frequent words that shared one or more phonemes (Fig. 2(b)). Trajectory segments corresponding to the same phoneme appeared more similar across words than segments corresponding to different phonemes. To quantify this, for each phoneme occurrence, we compared its trajectory with templates for every candidate phoneme, each constructed while excluding the word containing that occurrence. After aligning with Procrustes transformation the templates to the observed trajectory, we ranked them by residual error: a low rank indicates that the template for the phoneme actually produced is a close match. The observed mean rank of the true phonemes was significantly lower than the mean rank after shuffling phoneme labels within recording days and rebuilding the phoneme templates (one-sided permutation test, n=222 phoneme occurrences, $p < 0.0001$), providing evidence that phoneme-specific trajectory shape is preserved across words.

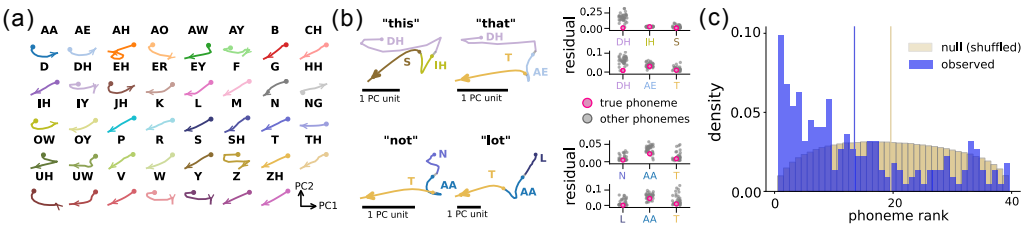


Figure 2: (a) Mean trajectory across words of each phoneme projected into the space spanned by the first two PCs. The start and end of the trajectories are represented with a circle and an arrow respectively. (b) Left: example word trajectories. Right: procrustes residuals for the true phoneme (pink) and alternative candidate phonemes (gray). (c) Distribution of true and shuffled phoneme residual rank. The vertical lines represent the mean.

3.2 EVALUATING THE COMPOSITIONAL MODELS

We compare phoneme decoding accuracy on real and synthetic data across the Card, Willett, Kunz T12, and Kunz T15 datasets, using BiT, a transformer-based neural decoder trained separately for phoneme decoding on each subject (Table 1). For synthetic data generated from the linear model, decoding performance approaches, and in some cases surpasses, that of real neural activity across datasets, demonstrating that decoding accuracy is preserved when phoneme neural trajectories are linearly combined into word and sentence trajectories. Additionally, phoneme decoding accuracy generally improved as higher-order levels were added, though not strictly monotonically for every dataset. This suggests that higher-order linguistic structure captures neural variability not explained by phoneme kernels alone.

In contrast, for the TCN model, adding towers for higher hierarchical primitives improves decoding only from phoneme to phoneme+biphone; further levels (syllable, word) do not improve, and in some datasets slightly worsen, decoding performance. This may reflect the TCN's convolutional architecture already implicitly capturing longer temporal horizon structure, making explicit higher-level towers redundant. Nonetheless, the TCN reached a lower absolute PER than the linear model in three of the four datasets (all but Kunz T15), suggesting that the non-linear interactions between phoneme occurrences and the longer temporal context available to the TCN allow its synthetic data to carry more information relevant to decoding speech than a purely linear combination can capture.

We note that for all datasets, the best synthetic decoding accuracy exceeds that of real neural activity. This likely reflects the noise gain selection procedure: for both models, the noise gain was chosen to minimize decoded PER in a held-out validation set, rather than to match the real data's noise

statistics, so some of the apparent improvement over real data may reflect a less noisy, and therefore more easily decoded, synthetic signal, rather than a genuine gain in how well the compositional structure captures neural variability.

		Synth. Linear				Synth. TCN			
Dataset	**Real**	L1	L1→L2	L1→L3	L1→L4	L1	L1→L2	L1→L3	L1→L4
Card	7.89	28.41	11.90	10.41	9.67	7.00	**6.19**	6.19	6.26
Willett	16.00	28.48	15.77	14.42	13.65	10.80	10.25	10.28	**10.17**
Kunz T12	28.41	21.43	15.80	14.72	14.94	12.73	**10.70**	11.25	12.18
Kunz T15	38.32	29.48	**18.86**	20.72	20.19	28.63	28.31	28.53	28.63

Table 1: Phoneme error rate (PER, %) on synthetic data generated by the linear and TCN models, decoded using a BiT model trained on real data. L1: phoneme; L2: biphone; L3: syllable; L4: word. L1→L4 denotes the hierarchical linguistic levels considered, from phonemes (L1) to words (L4).

To test whether the compositional model generalizes to novel words absent from training, we compared PER on out-of-vocabulary (OOV) test words to matched in-vocabulary control words present in both the training and test data. Each OOV word was matched one-to-one to the closest in-vocabulary word occurring in test, using z-scored Euclidean distance over four features: phoneme count, syllable count, duration, and mean phoneme frequency, with matching performed jointly across all OOV words so that no in-vocabulary word was reused. This controls for confounds such as OOV words simply being longer or containing rarer phonemes, rather than comparing against an unconditional average of all in-vocabulary words.

Since the OOV words have no word-specific kernels, their synthetic trajectories are built from sub-word phoneme, biphoneme, and syllable kernels, so how well the model reconstructs OOV words is a direct test of whether that sub-word structure generalizes to unseen wholes. We considered two controls: a day specific bias only L0 model and a model that takes in account the onset of each phoneme, but where the phoneme identity is shuffled (L0S).

For both the linear and TCN models, OOV PER is far below what would be expected with no compositional structure: the day specific bias only L0 model gives PER of approximately 98% for both, and the model with correct timing but shuffled phoneme identity gives 84.1% (linear) and 87.2% (TCN) (Table 2). This demonstrates that both models generalize to unseen words using shared sub-word structure. The gap relative to matched in-vocabulary words for the linear model starts smaller than the gap observed in real data and grows toward it as biphone and syllable structure are added, reaching a comparable gap to real data by the syllable level. The linear model's OOV PER improves as higher-order primitives are added, while the TCN's PER is already low at the phoneme level and stays approximately flat as further levels are added. This is likely because the TCN's convolutional architecture already captures, from phoneme identity and timing alone, context that the linear model only gets through explicit biphone and syllable kernels.

		Synth. Linear					Synth. TCN				
Metric	**Real**	L0	L0S	L1	L1→L2	L1→L3	L0	L0S	L1	L1→L2	L1→L3
OOV	33.2	98.0	84.1	52.4	35.8	35.6	98.2	87.2	32.9	29.8	30.0
Matched	12.5	97.9	84.6	38.1	16.9	14.0	97.9	86.7	11.2	8.5	8.5
Gap	20.8			**14.3**	18.9	21.5			21.8	21.3	21.6

Table 2: Phoneme error rate (PER, %) for out-of-vocabulary (OOV) sentences of Card and a matched in-vocabulary control group, on real data and on synthetic data from the linear and TCN models. L0: day specific bias; L0S: correct phoneme onset timing but shuffled phoneme identity; L1: phoneme; L2: biphone; L3: syllable; L4: word. L1→L4 denotes the hierarchical linguistic levels considered, from phonemes (L1) to words (L4). Gap is in percentage points. All columns score the same 296 OOV words one-to-one to 296 in-vocabulary control words.

We also verified whether the synthetic data reproduced key temporal and spectral statistics of the real neural signals (Fig. 3). Across most channels, the real time series shows a rise in autocorrelation at lags roughly matching phoneme duration (40–150 ms, Fig. 3 (b)), consistent with sustained activity locked to phoneme production; the TCN model partially captures this rise, while the linear model does not. The power spectrum shows a similar pattern: the TCN model approximately follows the dip in real data at low frequencies, whereas the linear model's spectrum is comparatively flat (Fig. 3 (c)). Both models generate realistic trial-to-trial variability (Fig. 3 (d)). Finally, the phoneme

representational geometry of both synthetic populations closely resembles that of the real data (Fig. 3 (f)).

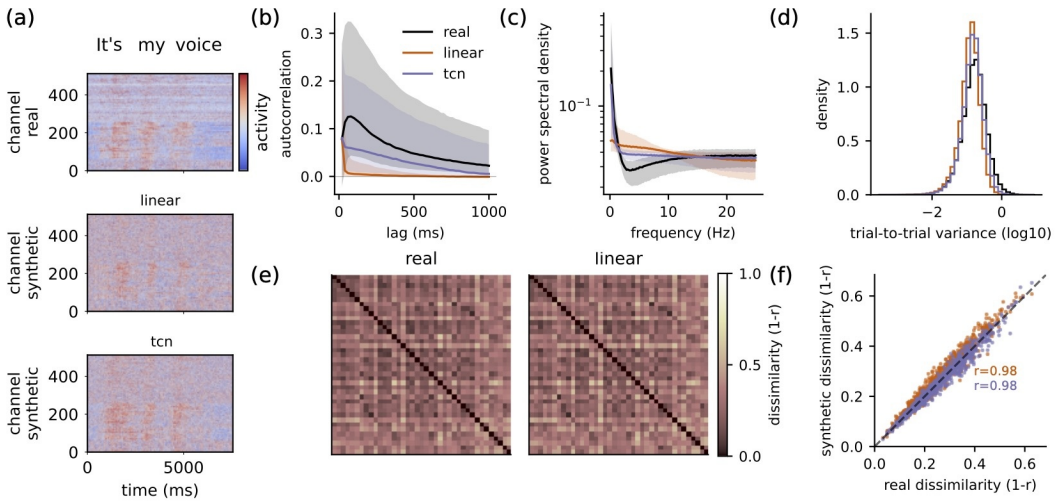


Figure 3: (a) Example single-trial neural activity for real data and for synthetic data generated by the linear and TCN models for the sentence "It's my voice". (b) Autocorrelation for real and synthetic data (lag 0 omitted). For each channel, the autocorrelation is computed per trial and averaged across trials. The line is the median and the shaded area the 10th–90th percentile across channels. (c) Power spectral density for real and synthetic data (excluding zero frequency). For each channel, the power spectrum is computed per trial and averaged across trials. The line is the median and the shaded area the 10th–90th percentile across channels. (d) Per-channel variance, across repeated occurrences of the same word (matched for phoneme identity and within-word position), of the trial-averaged response. (e) Representational dissimilarity matrices (RDMs) for real data and for synthetic data generated by the linear model, over the 39 phonemes AA, AE, AH, AO, AW, AY, B, CH, D, DH, EH, ER, EY, F, G, HH, IH, IY, JH, K, L, M, N, NG, OW, OY, P, R, S, SH, T, TH, UH, UW, V, W, Y, Z, and ZH. (f) Off-diagonal RDM dissimilarity values for the linear and TCN synthetic data plotted against the corresponding real dissimilarity values; each point is one phoneme pair, and the dashed line indicates equality.

3.3 Synthetic data improves speech decoding when data is scarce

We investigated whether augmenting the training set with synthetically generated neural trajectories improves decoding accuracy. Specifically, we generated trajectories for sentences from the Switchboard dataset (Godfrey et al., 1992) that participants had never spoken. We then trained BiT for 800 epochs on varying proportions of real (p) and synthetic (a) data, with synthetic data generated by either the linear or the TCN model, and report the best PER on a fixed test set (Fig. 4). We considered all hierarchical levels for the linear model and only gave as input, phoneme information to TCN. Initially we fit the linear and TCN models on all N training trials.

In each dataset table there are two visual gradients: PER decreases as the fraction of real data increases, and generally decreases with the augmentation factor as well. Synthetic augmentation produced the largest improvement for both models on the Kunz dataset, reducing PER by up to 20 percentage points (pp) for linear and 38pp for TCN relative to no augmentation, at 50% real data. It also helped when only a small fraction of the Card and Willett datasets was available: at 10% real data, PER decreased by 9pp (Card) and 8pp (Willett) for the linear model, and by 23pp (Card and Willett) for the TCN model, suggesting synthetic augmentation is particularly valuable in data-scarce settings. Overall, the TCN leads to lower PER comparing with the linear model.

To verify this improvement is not an artifact of BiT's own spike smoothing and noise-injection augmentation, we repeated the experiment with that augmentation disabled, over a reduced grid of real fraction and augmentation factor conditions on the Card dataset. The same qualitative pattern held: synthetic augmentation continued to help most when less real data was available (Fig. 7). For 10% real data the benefit of augmentation was larger with BIT's own augmentation disabled, sugesting

that synthetic data may useful variability that partially overlaps with the variability introduced by BiT's built-in augmentation.

On the other hand, to assess whether the improvement depends on the generative model having access to more real data than would be available in practice, we repeat the experiment in an online setting, where the generative model is fit using only the p fraction of real data available in that condition, rather than the full dataset (Fig. 6). The same qualitative pattern held: synthetic augmentation helped most when real data was scarce, whether because the real fraction p was small, or because the dataset itself was small. Comparing the online and offline experiments at the smallest real fraction tested (p=0.1), offline augmentation led to a larger improvement in decoding accuracy than online for the TCN model, but was similar for the linear model. On Card (a=1), the TCN improved by 18.8pp offline versus 7.4pp online, while the linear model improved by 6.6pp offline versus 5.5pp online. On Willett (a=2), the TCN improved by 22.8pp offline versus 10.8pp online, while the linear model improved by 7.4pp offline versus 6pp online. On Kunz T15 (a=2), the TCN again showed a larger offline improvement (15.6pp vs. 9.4pp online), but the linear model instead showed a larger online improvement (13.2pp vs. 11pp offline). In summary, synthetic augmentation improves phoneme decoding across datasets, with the largest gains at low real-data fractions, in both the offline and online settings.

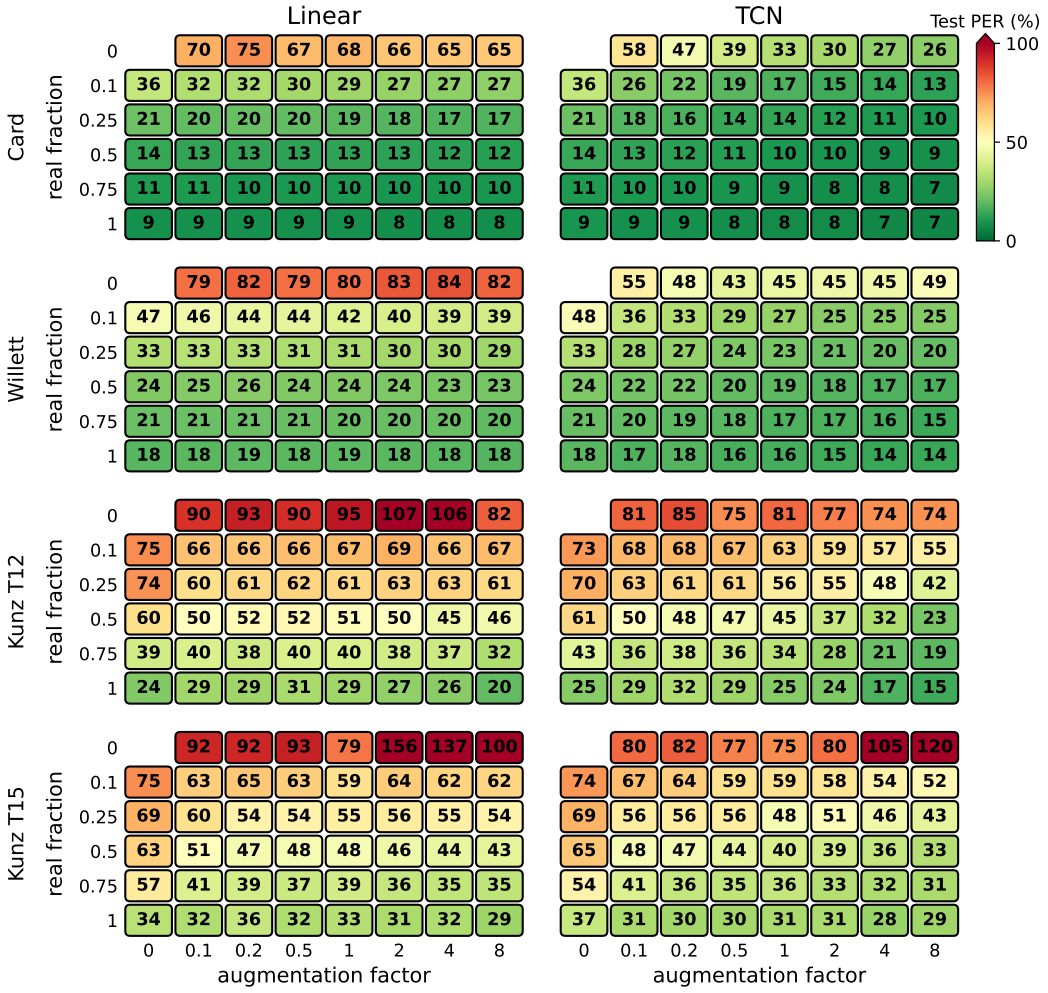


Figure 4: We retain pN real training trials and add aN synthetic trials, generated from unseen Switchboard sentences by the linear or TCN model. p is the fraction of real data retained, a is the augmentation factor, and N is the total number of training trials. We train BiT on the combined dataset and evaluate it on held-out real data using phoneme error rate (PER). The tables report test-set PER across values of p and a.

4 DISCUSSION

In this work, using publicly available datasets in which participants imagined or attempted to produce sentences while premotor cortex activity was recorded, we found evidence that phoneme-level neural trajectories are reused to construct word-level activity. Inspired by these signatures of compositionality we fit a simple linear model that composes kernels at phoneme onsets. Considering additional hierarchical levels: biphones, syllables, and words, synthetic data generated from the model achieves a PER, decoded by a real data trained model, close to that of real recordings across all datasets. This provides further evidence that neural trajectories at different hierarchical levels are linearly combined into word and sentence-level trajectories. We then tested whether a non-linear composition of these primitives, over a longer temporal context, better explains neural activity by training a temporal convolutional network (TCN), which receives, in addition to onset information, a duration-held representation of each primitive's identity. Using the BIT decoder trained on real data, the TCN model's synthetic data further reduced PER. Additionally, this model better captures the temporal and spectral properties of the real data.

Both the linear and the non-linear TCN model significantly improve speech decoding in data-scarce regimes, with maximum improvements of 21.8pp (linear) and 38.5pp (TCN) when the generative model is fit on the full dataset (offline), and 21.8pp (linear) and 27.2pp (TCN) when it is restricted to the same scarce real-data fraction as the decoder (online). This is very useful for BCIs, where neural recordings remain costly and difficult to collect at scale and foundation models are starting to show promising results, but require large amounts of training data (Zhang et al., 2026; Feghhi et al., 2026).

However, the benefit of both models decreases as more real data becomes available. Real data itself does not show this pattern: additional real trials keep reducing PER even at the largest training sizes we tested. This suggests the shrinking benefit is specific to synthetic data, not a general saturation effect. Our models likely add the most value when they can fill in phoneme and word combinations not yet covered by a small real training set; as real data grows, it increasingly covers these combinations, so synthetic data becomes redundant. A similar effect was also observed on a recent augmentation study for fMRI using a large encoding model (TRIBE v2) (Benchetrit et al., 2026).

The linear model does not perfectly match the temporal and spectral statistics of real data. In fact, for several datasets, the PER of synthetic data generated from this model is lower than the PER of real data, which means the synthetic activity is easier to decode than actual recordings. This is consistent with how the noise scale was chosen: it was tuned to minimize decoded PER, not to match real signal statistics. We tested whether a more complex noise model would close this gap, but it instead led to much worse PER. This suggests the problem might not be the noise model's complexity, but its structure: our models add noise as an independent process, unrelated to which primitives are active on a given trial, while real trial-to-trial variability is likely coupled to the underlying compositional signal.

A natural future direction is replacing the independent noise term with structured sources of variability, coupled to the same compositional primitives (Yu et al., 2008; Macke et al., 2011; Pandarinath et al., 2018; Duncker & Sahani, 2018; Kapoor et al., 2024). On the other hand, a more biophysically grounded account would require simulating the underlying spiking network directly (e.g., with tools such as Brian Stimberg et al. (2019) or NEST Gewaltig & Diesmann (2007)) which in turn requires connectivity and cell-type data for premotor speech circuits that does not currently exist at the necessary resolution. Another possible future direction is to combine our data augmentation approach with cross-subject, cross-species decoder training (Fogg et al., 2026; Boccato et al., 2026b;a), towards high-performance, data-efficient neural decoders.

AI USE STATEMENT

In this work, we have not used generative AI tools to help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis and interpret results.

Additionally, we used generative AI tools to create or edit parts of the software code, edit specific sections or sentences of the research paper to improve readability and improve the quality of some scientific figures. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

In this work we used publicly available datasets which contain human intracortical recordings. A full list of the datasets is in the Appendix A.1, including references to the official sources. We do not acquire any additional data, from humans or animals. The data were used in accordance with the access conditions and licenses specified by the original data providers. Ethical oversight and informed-consent procedures for the original collections are described in the corresponding dataset publications. Although improved neural-speech decoding may support communication for people with severe speech impairment, such technologies also raise privacy, autonomy, and misuse concerns. We therefore emphasize that the present work is intended for research purposes and does not establish clinical safety, efficacy, or suitability for deployment.

Reproducibility statement

The code to replicate the experiments performed is submitted as supplementary materials. In the appendix A we provide details of how we processed the publicly available data used in the paper.

References

Mehdi Azabou, Vinam Arora, Venkataramana Ganesh, Ximeng Mao, Santosh Nachimuthu, Michael Mendelson, Blake Richards, Matthew Perich, Guillaume Lajoie, and Eva Dyer. A unified, scalable framework for neural population decoding. *Advances in Neural Information Processing Systems*, 36:44937–44956, 2023.

Mehdi Azabou, Krystal Xuejing Pan, Vinam Arora, Ian Jarratt Knight, Eva L Dyer, and Blake Aaron Richards. Multi-session, multi-task neural decoding from distinct cell-types and brain regions. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=IuU0wcO0mo.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

Yohann Benchetrit, Marlène Careil, Simon Dahan, Hubert Banville, Stéphane d'Ascoli, and Jean-Rémi King. Boosting brain-to-image decoding with tribe v2 data augmentation. *arXiv preprint arXiv:2606.06345*, 2026.

Tommaso Boccato, Michal Olak, and Matteo Ferrante. Cross-subject decoding of human neural data for speech brain computer interfaces. *Journal of Neural Engineering*, 23(4): 046018, jul 2026a. doi: 10.1088/1741-2552/ae8576. URL https://doi.org/10.1088/1741-2552/ae8576.

Tommaso Boccato, Michal Olak, and Matteo Ferrante. Brainwhisperer: Leveraging large-scale asr models for neural speech decoding. *arXiv preprint arXiv:2603.13321*, 2026b.

Nicholas S Card, Maitreyee Wairagkar, Carrina Iacobacci, Xianda Hou, Tyler Singer-Clark, Francis R Willett, Erin M Kunz, Chaofei Fan, Maryam Vahdati Nia, Darrel R Deo, et al. An accurate and rapidly calibrating speech neuroprosthesis. *New England Journal of Medicine*, 391(7):609–618, 2024.

Lea Duncker and Maneesh Sahani. Temporal alignment and latent gaussian process factor inference in population spike trains. *Advances in neural information processing systems*, 31, 2018.

Ebrahim Feghhi, Shreyas Kaasyap, Nima Hadidi, and Jonathan Kao. Time-masked transformers with lightweight test-time adaptation for neural speech decoding. *Advances in Neural Information Processing Systems*, 38:93047–93072, 2026.

Zachery M. Fogg, Nicholas S. Card, Maitreyee Wairagkar, Aparna Srinivasan, Tyler Singer-Clark, Xianda Hou, Elizaveta Okorokova, Hamza Peracha, Carrina Iacobacci, Tiffany Brailow, Justin Jude, Hadar Levi-Aharoni, Trung Le, Domenick Mifsud, Pranav I. Deevi, Samuel R. Nason-Tomaszewski, Anna L. Pritchard, Yunnuo Zhang, Brice Richards, Payton Bechefsky, Leigh R. Hochberg, Ziv Williams, Kiarash Shahlaie, Nicholas Au Yong, Daniel B. Rubin, Chethan Pandarinath, David M. Brandman, and Sergey D. Stavisky. A generalizable speech neuroprosthesis. *bioRxiv*, pp. 2026.07.23.739430, July 2026. doi: 10.64898/2026.07.23.739430. URL https://doi.org/10.64898/2026.07.23.739430. Preprint.

Marc-Oliver Gewaltig and Markus Diesmann. Nest (neural simulation tool). *Scholarpedia*, 2(4): 1430, 2007.

John J Godfrey, Edward C Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pp. 517–520. IEEE, 1992.

John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006.

Moto Hira. Forced alignment with wav2vec2. URL https://docs.pytorch.org/audio/stable/tutorials/forced_alignment_tutorial.html.

Jaivardhan Kapoor, Auguste Schulz, Julius Vetter, Felix Pei, Richard Gao, and Jakob H Macke. Latent diffusion for neural spiking data. *Advances in Neural Information Processing Systems*, 37: 118119–118154, 2024.

Benyamin Abramovich Krasa, Erin M Kunz, Foram Kamdar, Donald Avansino, Nick Hahn, Akansha Singh, Nicholas S Card, Maitreyee Wairagkar, Carrina Iacobacci, Leigh R Hochberg, et al. Premotor cortex uses a compositional neural geometry to plan words. *bioRxiv*, 2026.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Erin M Kunz, Benyamin Abramovich Krasa, Foram Kamdar, Donald T Avansino, Nick Hahn, Seonghyun Yoon, Akansha Singh, Samuel R Nason-Tomaszewski, Nicholas S Card, Justin J Jude, et al. Inner speech in motor cortex and implications for speech neuroprostheses. *Cell*, 188(17): 4658–4673, 2025.

Trung Le, Hao Fang, Jingyuan Li, Tung Nguyen, Lu Mi, Amy L Orsborn, Uygar Sümbül, and Eli Shlizerman. Spint: Spatial permutation-invariant neural transformer for consistent intracortical motor decoding. *Advances in Neural Information Processing Systems*, 38:25815–25835, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Jakob H Macke, Lars Buesing, John Cunningham, Byron M Yu, Krishna V Shenoy, and Maneesh Sahani. Empirical models of spiking in neural populations. *Advances in neural information processing systems*, 24, 2011.

Chethan Pandarinath, Daniel J O'Shea, Jasmine Collins, Rafal Jozefowicz, Sergey D Stavisky, Jonathan C Kao, Eric M Trautmann, Matthew T Kaufman, Stephen I Ryu, Leigh R Hochberg, et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature methods*, 15(10):805–815, 2018.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*, 2019.

Nishal P Shah, Donald Avansino, Foram Kamdar, Claire Nicolas, Anastasia Kapitonava, Carlos Vargas-Irwin, Leigh R Hochberg, Chethan Pandarinath, Krishna V Shenoy, Francis R Willett, et al. Pseudo-linear summation explains neural geometry of multi-finger movements in human premotor cortex. *Nature Communications*, 16(1):5008, 2025.

A. B. Silva, K. T. Littlejohn, J. R. Liu, et al. The speech neuroprosthesis. *Nature Reviews Neuroscience*, 25(7):473–492, 2024. doi: 10.1038/s41583-024-00819-9. URL https://doi.org/10.1038/s41583-024-00819-9.

Marcel Stimberg, Romain Brette, and Dan FM Goodman. Brian 2, an intuitive and efficient neural simulator. *elife*, 8:e47314, 2019.

Kurt A Thoroughman and Reza Shadmehr. Learning of action through adaptive combination of motor primitives. *Nature*, 407(6805):742–747, 2000.

Andrew Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269, 1967.

Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 6382–6388, 2019.

Konstantin F Willeke, Polina Turishcheva, Alex Gilbert, Goirik Chakrabarty, Hasan Bedel, Paul Fahey, Yongrong Qiu, Marissa Weis, Michaela Vystrčilová, Taliah Muhammad, et al. Omnimouse: Scaling properties of multi-modal, multi-task brain models on 150b neural tokens. In *International Conference on Learning Representations*, volume 2026, pp. 134382–134411, 2026.

Francis R Willett, Erin M Kunz, Chaofei Fan, Donald T Avansino, Guy H Wilson, Eun Young Choi, Foram Kamdar, Matthew F Glasser, Leigh R Hochberg, Shaul Druckmann, et al. A high-performance speech neuroprosthesis. *Nature*, 620(7976):1031–1036, 2023.

Francis R. Willett, Jingyuan Li, Trung Le, Chaofei Fan, Mingfei Chen, Eli Shlizerman, Yue Chen, Xin Zheng, Tatsuo S. Okubo, Tyler Benster, Hyun Dong Lee, Maxwell Kounga, E. Kelly Buchanan, David Zoltowski, Scott W. Linderman, and Jaimie M. Henderson. Brain-to-text benchmark '24: Lessons learned, 2024. URL https://arxiv.org/abs/2412.17227.

Joel Ye and Chethan Pandarinath. Representation learning for neural population activity with neural data transformers. 2021. doi: 10.48550/ARXIV.2108.01210. URL https://arxiv.org/abs/2108.01210.

Byron M Yu, John P Cunningham, Gopal Santhanam, Stephen Ryu, Krishna V Shenoy, and Maneesh Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Advances in neural information processing systems*, 21, 2008.

Yizi Zhang, Linyang He, Chaofei Fan, Tingkai Liu, Han Yu, Trung Le, Jingyuan Li, Scott Linderman, Lea Duncker, Francis R Willett, et al. A cross-species neural foundation model for end-to-end speech decoding. In *The Fourteenth International Conference on Learning Representations*, 2026.

Andrew J Zimnik and Mark M Churchland. Independent generation of sequence elements by motor cortex. *Nature neuroscience*, 24(3):412–424, 2021.

A Appendix 1

A.1 Data

All experiments rely on publicly available intracortical recordings collected via Utah arrays from BrainGate[1] participants, spanning attempted and imagined speech tasks across two participants. Three datasets are considered: Card et al. (2024), Willett et al. (2023) and Kunz et al. (2025). A

[1] https://www.braingate.org

full description of each dataset is given below, and Table A.1 describes the number of sentences for training and validation split for each dataset. For datasets which have no predefined validation or test split, or in cases where the test split does not include labels, we randomly split the validation set in half to create a validation and a test split both with labels. Data was binned into 20 ms time bins where not already provided at that resolution. For each dataset, spike counts were z-scored per channel; normalization statistics were computed across all available sessions of the same subject where possible, and within individual sessions otherwise.

Card et al. (2024) Neural activity was recorded during attempted speech from the left precentral gyrus of participant T15, a person with ALS and severe dysarthria. Recordings contain threshold spike counts and spike-band power (SBP) from 256 electrodes spanning four arrays implanted in areas i6v, 4, s6v, and 55b, covering a 125,000-word vocabulary. A held-out, unlabeled test split of this dataset was released as part of the Brain-to-Text '25 competition.[2] We refer to this dataset as Card throughout.

Willett et al. (2023) This dataset includes recordings from participant T12, also diagnosed with ALS. Sentences are drawn from both a 50-word and a 125,000-word vocabulary, collected under an attempted speech paradigm. Threshold spikes and SBP are available from 128 electrodes in arrays implanted in areas i6v and s6v. An unlabeled competition split was released in connection with the Brain-to-Text '24 challenge.[3] We refer to this dataset as Willett.

Kunz et al. (2025) This dataset accompanies a study on the decodability of imagined speech from motor cortex activity. It includes recordings from four participants (T12, T15, T16, T17) across attempted and imagined speech conditions. For our purposes, we use only data from participants T12 and T15, and discard all single-word trials. Neural features consist of threshold spikes and SBP from the same electrode arrays described above. We abbreviate the name of this dataset as Kunz.

Dataset	Train	Val	Test
Card	8069	713	713
Willett	8800	440	440
Kunz T12	752	42	42
Kunz T15	934	53	53

Table 3: Number of trials in the training, validation and test splits for each dataset.

A.2 Details of neural data analysis

We used publicly available data for these analyses (Card et al., 2024). To align the neural time series data with the occurrence of words and phonemes we used forced alignment algorithm (Viterbi, 1967; Graves et al., 2006; Hira) on the trained BrainWhisperer model (Boccato et al., 2026b). To avoid circularity, we used the BiT (Feghhi et al., 2026) model for all subsequent speech decoding analyses.

We started by fitting the principal components (PCs) for the SBP data, that includes data from 256 channels in 20 ms bins ($X \in \mathbb{R}^{n_{\text{time bins}} \times n_{\text{channels}}}$). To visually compare trajectories of words and sentences, we projected into the space spanned by the first two PCs.

To test whether the *shape* of a phoneme trajectory was preserved across words, we compared each observed phoneme trajectory with reference trajectories for all candidate phonemes. For each occurrence of phoneme p in word w, we constructed a leave-one-word-out reference trajectory by averaging all occurrences of p in words other than w. Because phoneme durations varied, trajectories were first resampled to a common number of time points.

We assessed statistical significance using a one-sided permutation test. For each permutation, phoneme labels were shuffled among occurrences within each recording day. We then rebuilt the phoneme templates and repeated the complete Procrustes alignment and ranking procedure. This

[2] https://www.kaggle.com/competitions/brain-to-text-25

[3] https://eval.ai/web/challenges/challenge-page/2099/overview

process was repeated $B = 10{,}000$ times to obtain a null distribution of mean ranks. The p-value was computed as

$$p = \frac{1 + \sum_{b=1}^{B} \mathbb{I}\left(\overline{r}_{\text{null}}^{(b)} \leq \overline{r}_{\text{obs}}\right)}{B + 1} ,$$

where $\overline{r}_{\text{obs}}$ is the observed mean rank and $\overline{r}_{\text{null}}^{(b)}$ is the mean rank obtained in the b-th permutation. Only words with more than 100 occurrences were included to ensure reliable estimates of the mean phoneme trajectories.

A.3 Fitting other noise models

The noise model parameters are estimated from the residuals $r(c,t) = x(c,t) - \hat{x}(c,t)$ of the fitted deterministic model, pooled across all training trials. For each channel c, the AR(1) coefficient is estimated as

$$\hat{\phi}_c = \frac{\sum_n \sum_{t=2}^{T_n} r_n(c,t)\, r_n(c,t-1)}{\sum_n \sum_{t=2}^{T_n} r_n(c,t-1)^2} , \tag{5}$$

where n indexes trials. Lag-1 products are computed within each trial and summed across trials, so no product spans a trial boundary.

Let $\hat{P}$ denote the pooled covariance matrix of the residual vector across channels. If $\hat{\sigma}_c$ is the residual standard deviation in channel c and $\hat{R}$ is the corresponding residual correlation matrix, then

$$\hat{P} = \operatorname{diag}(\hat{\sigma}_c)\, \hat{R}\, \operatorname{diag}(\hat{\sigma}_c) .$$

Under the multivariate noise model

$$\mathbf{r}_t = \hat{\Phi}\mathbf{r}_{t-1} + \boldsymbol{\eta}_t , \qquad \hat{\Phi} = \operatorname{diag}(\hat{\boldsymbol{\phi}}) , \qquad \boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, \hat{\Sigma}) ,$$

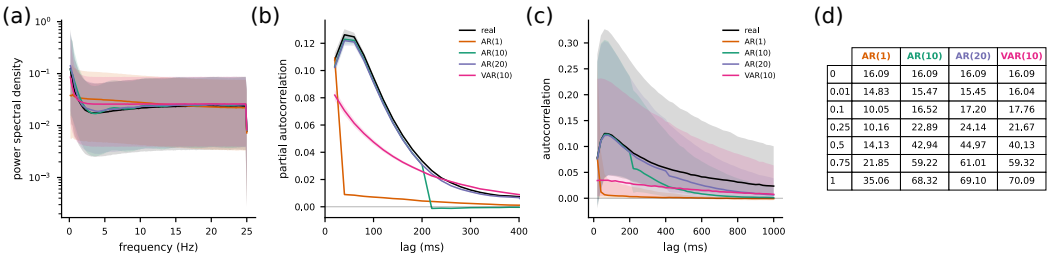
the innovation covariance under the stationarity assumption is given by

$$\hat{\Sigma} = \hat{P} - \hat{\Phi}\hat{P}\hat{\Phi}^{\top} , \tag{6}$$

with entries

$$\hat{\Sigma}_{cd} = \hat{\sigma}_c \hat{\sigma}_d \hat{R}_{cd} \left(1 - \hat{\phi}_c \hat{\phi}_d\right) . \tag{7}$$

We considered autoregressive noise models with higher order, and additionally vector autoregressive (VAR) models that model correlations between channels across time. The AR models with order 10 and 20 approximated better the power spectrum and partial autocorrelation of the real data, but led to significant higher PER on the validation data split, and we therefore considered the AR(1) model for the future augmentation experiments.



	AR(1)	AR(10)	AR(20)	VAR(10)
0	16.09	16.09	16.09	16.09
0.01	14.83	15.47	15.45	16.04
0.1	10.05	16.52	17.20	17.76
0.25	10.16	22.89	24.14	21.67
0.5	14.13	42.94	44.97	40.13
0.75	21.85	59.22	61.01	59.32
1	35.06	68.32	69.10	70.09

Figure 5: (a) Autocorrelation for real and synthetic data (lag 0 omitted). For each channel, the autocorrelation is computed per trial and averaged across trials. The line is the median and the shaded area the 10th–90th percentile across channels. (b) Partial autocorrelation for real and synthetic data (lag 0 omitted). For each channel, the autocorrelation is computed per trial and averaged across trials. The line is the median and the shaded area the 10th–90th percentile across channels. (c) Power spectral density for real and synthetic data (excluding zero frequency). For each channel, the power spectrum is computed per trial and averaged across trials. The line is the median and the shaded area the 10th–90th percentile across channels.(d) PER for validation data split for different noise gain parameters and noise models.

A.4 Architectures and Training Hyperparameters

A.4.1 Linear model

The kernel duration T was set to 20, since each bin is 20ms, these corresponds to 400ms. In practice, we use forced alignment, as described above, to align the neural data with the occurrence of each primitive, so the window used to fit the kernels contains some anticipatory neural activity. The ridge regularization parameter λ was swept across the values [10, 30, 100, 300, 1000, 3000, 1e4, 3e4, 1e5] and was selected to maximize the explained variance on held-out validation set. The noise scale was swept over [0.0, 0.01, 0.1, 0.25, 0.5, 0.75, 1.0] and was selected to minimize the decoded PER on the real held-out validation split.

A.4.2 TCN model

Our TCN model uses non-causal (symmetric) convolutions. The temporal convolution network was trained for each dataset using the AdamW optimizer (Loshchilov & Hutter, 2017). Dropout was applied to avoid overfitting. The full set of hyperparameters is listed in Table A.4.2

Hyperparameter	Value
Architecture	
TCN blocks (layers)	12 (two repeated cycles of dilations 1,2,4,8,16,32)
Channel width	768
Conv. kernel size	3
GroupNorm groups	32
Activation	GELU
Dropout	0.1 (channel/spatial dropout)
Input embeddings	phoneme (64-dim) + day (64-dim)
Training	
Optimizer	AdamW
Learning rate	2×10^{-4}
Weight decay	1×10^{-3}
Gradient clipping (max norm)	1.0
Loss	masked MSE (per-channel, over valid timesteps)
Training crop length	256 bins ($\approx$5.12 s), randomly cropped per trial
Batch size	16
Max. epochs	900
Validation R^2 evaluated every	15 epochs
Decoded PER evaluated every	25 epochs, on the val split
Early stopping	no improvement in val PER for 4 consecutive evaluations (100 epochs)
Checkpoint selection	best val PER

Table 4: TCN architecture and training hyperparameters.

A.4.3 BiT

The BiT decoder used for decoding both the sythetic and real trials in Table 1 was trained on scratch for each dataset (Feghhi et al., 2026; Zhang et al., 2026). All models were trained with AdamW (Loshchilov & Hutter, 2017) for a fixed maximum of 800 epochs, with the best checkpoint selected based on the validation PER. The BiT encoder consisted of the following architecture: a 7-layer Transformer with hidden size 384, 6 attention heads, an intermediate size of 512, GELU activations, rotary positional embeddings (RoPE), and a dropout of 0.4. Neural activity was patched into segments of 5 time bins and linearly embedded before being passed to the encoder, with a CTC head used for phoneme decoding. Spike smoothing and noise injection were applied to the input during training for all four datasets. The full set of hyperparameters is listed in Table A.4.3.

Hyperparameter	Value
Architecture	
Transformer layers	7
Hidden size	384
Attention heads	6
Intermediate size	512
Activation (transformer)	GELU
Positional encoding	RoPE ($\theta = 10000$)
Dropout (transformer)	0.4
Patch size	5
Embedder activation	Softsign
Dropout (embedder)	0.2
Decoding head	CTC (vocabulary size 41)
Temporal masking	inactive
Optimization	
Optimizer	AdamW
Peak learning rate	5×10^{-4}
LR schedule	one-cycle (cosine)
`div_factor`	25 (initial LR $= 2 \times 10^{-5}$)
`final_div_factor`	10^4 (final LR $\approx 2 \times 10^{-9}$)
Weight decay	1×10^{-5}
Training	
Batch size	32
Max. epochs	800
Precision	bfloat16
Checkpoint selection	best validation PER
Input augmentation	Gaussian smoothing ($\sigma = 2$) + noise ($\sigma = 0.2$) + constant offset ($\sigma = 0.05$)

Table 5: BiT architecture and training hyperparameters.

For the augmentation experiments, with results reported in Fig. 4, we used similar hyperparameters except from batch size, that was set to 64.

B CONTROLS FOR AUGMENTATION EXPERIMENTS

In the main experiments, the generative model (linear or TCN) is always fit on the full real dataset, regardless of the real fraction p used to train the downstream decoder, a design choice that could inflate the apparent benefit. To test this, we repeated the experiment in an online setting, where the generative model is fit using only that condition's own p-fraction of real data (Fig. 6).

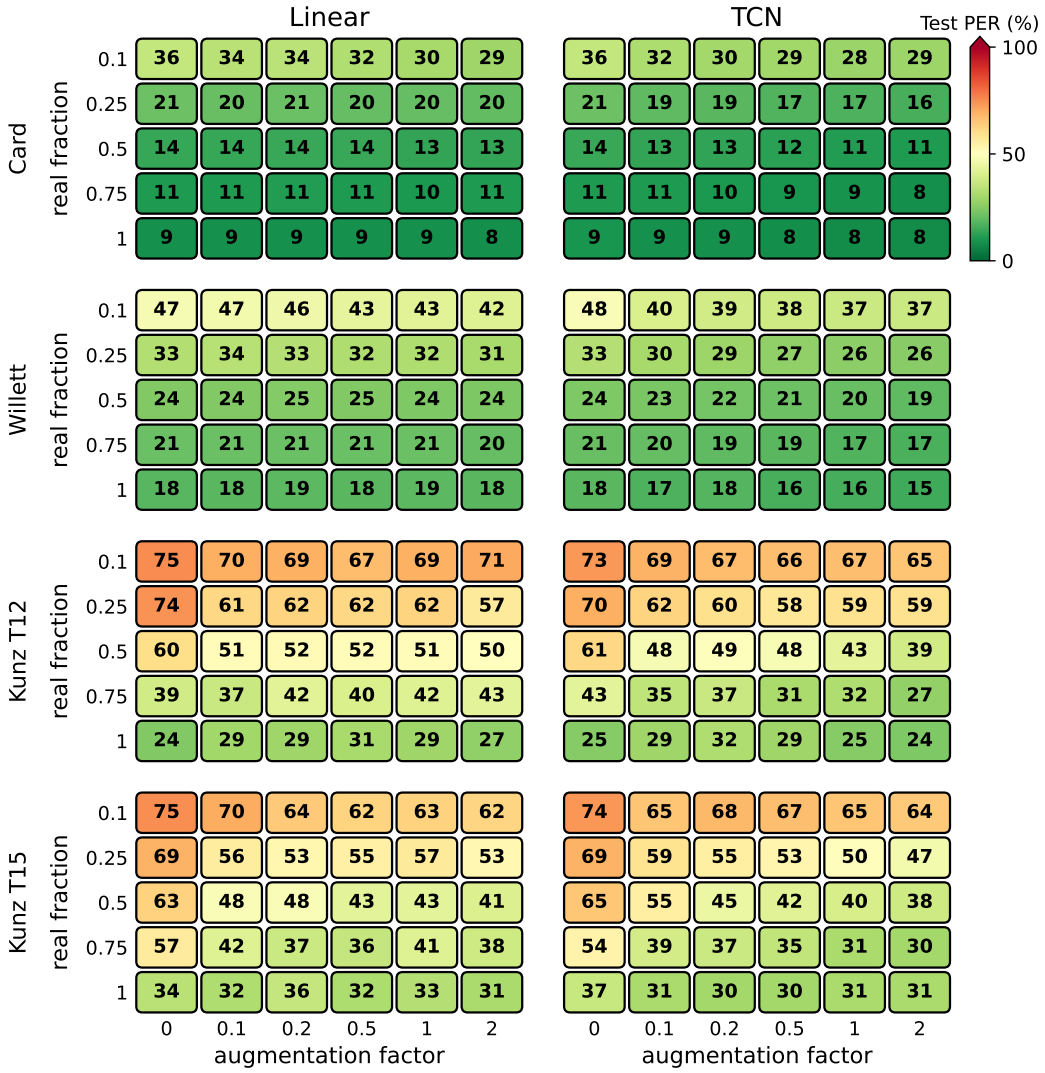


Figure 6: Similar to the augmentation experiments described in Section 3.3 but using the fraction pN of trials (instead of N) to fit the generative linear or TCN model. This corresponds to an *online* version of the results presented in Section 3.3. Augmentation fraction $a = 2, 4$ were not considered in these experiments due to computational constraints.

On the other hand, BiT applies its own spike-smoothing and noise-injection augmentation (Gaussian temporal smoothing plus independent white and constant-offset noise) during decoder training, which could in principle account for part of the augmentation benefit reported in the main text rather than the synthetic data itself. To rule this out, we repeated the Card experiment with BiT's augmentation disabled, over a reduced grid (real fraction $p \in 10\%, 50\%, 100\%$, augmentation factor $a \in 0, 0.2, 1, 2$) for both the linear and TCN generative models.

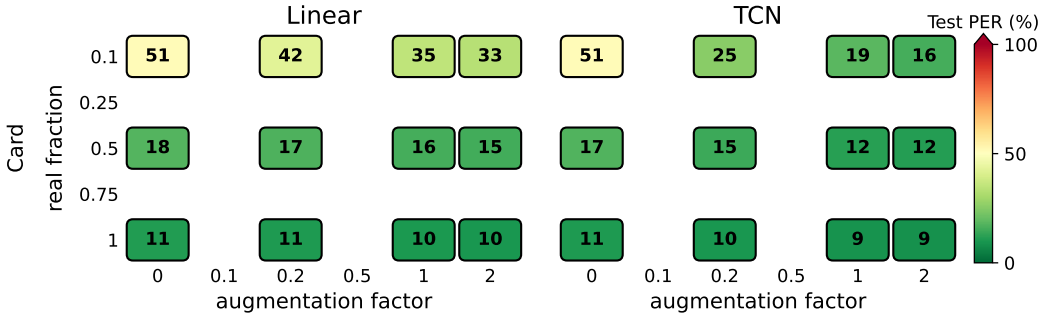


Figure 7: Similar to the augmentation experiments described in Section 3.3 but considering a smaller grid and considering no smoothing and noise injection augmentation when training the BiT.

C FROM THE CONVOLUTIONAL KERNEL MODEL TO RIDGE REGRESSION

C.1 THE GENERATIVE MODEL

In this work we define the predicted neural activity on channel c at time t as a sum of phoneme-triggered kernels:

$$\hat{x}(c,t) \;=\; \sum_{p} \sum_{\tau=0}^{T} K_p(c,\tau)\, \delta(t - t_p - \tau) \;+\; b_{\text{day}}(c), \tag{8}$$

where p indexes phoneme occurrences with onset time t_p, $K_p(c,\tau)$ is the kernel for phoneme identity p (channel c, lag τ), and $b_{\text{day}}(c)$ is a day-specific channel bias. Note K_p depends only on the *identity* of the phoneme at occurrence p, not on the occurrence index itself, every instance of phoneme "AE" reuses the same kernel. This is the compositionality assumption.

The goal of fitting is: given the real neural time series $x(c,t)$ and known phoneme onset times t_p (from forced-alignment), estimate $K_p(c,\tau)$ for every phoneme identity, plus $b_{\text{day}}(c)$.

C.2 REWRITING THE CONVOLUTION AS A LINEAR MAP

The key observation is that Eq. 8 is *linear* in the unknown kernel values $K_p(c,\tau)$. For a fixed channel c, define a vector of unknowns by stacking every (phoneme identity, τ) pair:

$$\boldsymbol{\beta}_c = \big[\, K_{\text{AA}}(c,0), \ldots, K_{\text{AA}}(c,T),\; K_{\text{AE}}(c,0), \ldots, K_{\text{AE}}(c,T),\; \ldots \,\big]^{\top} \in \mathbb{R}^M, \tag{9}$$

with $M = (\#\text{ phoneme identities}) \times (T+1)$.

We build a design matrix $X \in \mathbb{R}^{N \times M}$, one row per discrete time bin $t = 1, \ldots, N$, whose columns are indicator (event) regressors:

$$X\big[t, , (p,\tau)\big] = \sum_{t_p \in \mathcal{T}_p} \mathbb{I}\left[t = t_p + \tau\right] \;, \tag{10}$$

where $\mathcal{T}_p$ denotes the set of onset times of all occurrences of phoneme p. In words: column (p,τ) is an impulse train with a 1 placed τ bins after every onset of phoneme identity p (the sum allows overlapping impulses when two onsets of the same phoneme fall within τ bins of each other). This is exactly the lagged-indicator, or finite-impulse-response (FIR), design matrix used in temporal receptive field (TRF) and event-related regression estimation. Each phoneme onset contributes K overlapping shifted-delta columns (one per lag $\tau = 0, \ldots, K-1$), so the design matrix is sparse.

Appending a column (or block of one-hot columns, one per recording day) to absorb $b_{\text{day}}(c)$, the whole model in Eq. 8 becomes, per channel c,

$$\mathbf{x}_c \;=\; X\,\boldsymbol{\beta}_c \;+\; \boldsymbol{\varepsilon}_c, \tag{11}$$

where $\mathbf{x}_c \in \mathbb{R}^N$ is the observed neural time series on channel c and $\boldsymbol{\varepsilon}_c$ is residual noise. The convolution has become an ordinary linear regression: the temporal structure (which phoneme happened when, and for how many lags after) lives entirely in X; the kernel shapes to be learned live entirely in $\boldsymbol{\beta}_c$.

C.3 EXTENDING TO HIERARCHICAL LEVELS

The full model (Eq. 2) adds analogous terms for biphones, syllables, and words:

$$\hat{x}(c,t) \;=\; \sum_{L} \sum_{p_L} \sum_{\tau=0}^{T} K_{p_L}(c,\tau)\, \delta(t - t_{p_L} - \tau) \;+\; b_{\text{day}}(c). \tag{12}$$

This is handled by simply *concatenating* design matrices column-wise:

$$X_{\text{full}} = \big[\, X_{\text{phon}} \;\big|\; X_{\text{biphon}} \;\big|\; X_{\text{syll}} \;\big|\; X_{\text{word}} \;\big|\; X_{\text{day}} \,\big], \qquad \boldsymbol{\beta}_c = \begin{bmatrix} \boldsymbol{\beta}_c^{\text{phon}} \\ \boldsymbol{\beta}_c^{\text{biphon}} \\ \boldsymbol{\beta}_c^{\text{syll}} \\ \boldsymbol{\beta}_c^{\text{word}} \\ \mathbf{b}_{\text{day}}(c) \end{bmatrix}. \tag{13}$$

Each block of X_{full} is built the same way, just keyed on the onset times of that level's primitives. Fitting L1, L1+L2, L1–L3, L1–L4 progressively (as in Table 1) is just fitting with progressively more column-blocks included.

C.4 Why ridge, not OLS

Ordinary least squares would solve

$$\hat{\boldsymbol{\beta}}_c^{\text{OLS}} = \arg\min_{\boldsymbol{\beta}_c} \|\mathbf{x}_c - X\boldsymbol{\beta}_c\|_2^2 = (X^\top X)^{-1} X^\top \mathbf{x}_c. \tag{14}$$

But $X^\top X$ is typically near-singular here: adjacent lag columns within a kernel are highly collinear (overlapping shifted deltas), and rare phonemes/biphones/words give only a handful of onsets to estimate a whole $T{+}1$-length kernel from. This is the classic setting where ridge regression is used to regularize FIR estimation. Ridge adds an ℓ_2 penalty on the kernel weights:

$$\hat{\boldsymbol{\beta}}_c^{\text{ridge}}(\lambda) = \arg\min_{\boldsymbol{\beta}_c} \|\mathbf{x}_c - X\boldsymbol{\beta}_c\|_2^2 + \lambda\|\boldsymbol{\beta}_c\|_2^2, \tag{15}$$

with closed-form solution

$$\hat{\boldsymbol{\beta}}_c^{\text{ridge}}(\lambda) = \left(X^\top X + \lambda I\right)^{-1} X^\top \mathbf{x}_c. \tag{16}$$

The λI term makes $X^\top X + \lambda I$ well-conditioned and shrinks kernel estimates toward zero, which is exactly the right prior for a kernel that should be mostly a short, smooth transient rather than an arbitrary $T{+}1$-dimensional vector fit from a handful of trials.

C.5 Fitting procedure

1. **Design matrix construction.** Using forced-alignment onset times t_{p_L} for every hierarchical level L (phoneme, biphone, syllable, word), build the lagged indicator blocks X_L and concatenate into X_{full}, plus one-hot day regressors for b_{day}.
2. **Joint regression across channels.** Since channels are treated independently in Eq. 8 ($K_p(c,\tau)$ has a free c index), the ridge problem separates over channels. However, X is shared, so in practice this is solved as one multivariate ridge regression:

$$\hat{B}^{\text{ridge}}(\lambda) = \left(X^\top X + \lambda I\right)^{-1} X^\top Y\,, \tag{17}$$

   where $Y \in \mathbb{R}^{N\times C}$ stacks all C channels' time series as columns and $\hat{B} \in \mathbb{R}^{M\times C}$ stacks all channels' kernel weights. This requires only one matrix inversion of $X^\top X{+}\lambda I \in \mathbb{R}^{M\times M}$, reused across all channels. In practive we compute this via Cholesky factorization rather than explicit matrix inversion, which is algebraically equivalent but numerically more stable. Even for the largest (word-level) kernel set, M remains orders of magnitude smaller than the number of time bins used to accumulate $X^\top X$, so this normal-equations formulation is cheaper than the alternative dual formulation, which would instead require inverting an $N \times N$ matrix. The step of summing each trial's contribution into $X^\top X$ and $X^\top Y$ exploits the sparsity of X (each time bin has only a handful of active kernels) and only the resulting $M \times M$ and $M \times C$ matrices are ever formed densely.
3. **Selecting λ.** λ is cross-validated: we sweep a grid of λ values, refit $\hat{\beta}_c(\lambda)$ on a training split, and choose the λ that maximizes R^2 on held-out validation trials.
4. **Recovering the kernels.** Once $\hat{\boldsymbol{\beta}}_c$ is fit, the entries corresponding to each (p,τ) block are read back off as $\hat{K}_p(c,\tau)$, i.e. ridge regression estimates the same object as the convolutional model, just via a reparameterization that turns "sum of shifted per-identity kernels" into "one big linear regression with a structured design matrix."
5. **Generation.** At synthesis time, Eq. 12 is evaluated in the forward direction: given a target sentence's phoneme, biphone, syllable, and word onset times, the fitted kernels $\hat{K}p_L(c,\tau)$ are placed at their corresponding onsets, summed, and combined with a sampled autoregressive residual noise term to produce $x_{synth}(c,t)$.