Supplemental Materials

Sections

Tables

Figures

References

Section 1: Normalizing the decoding matrix

Normalizing the decoding matrix *D* allows the gain of the cursor to be reported precisely (otherwise, D contains an unknown gain multiplier embedded within it). We normalized *D* so that $Df_t \sim N(\begin{bmatrix}\cos\theta \\ \sin\theta\end{bmatrix}, P)$ when the user is trying to move at full speed in direction θ (where *P* is a 2 x 2 covariance matrix that describes the decoding noise). To normalize it, we projected u_t at each time step onto a unit vector pointing from the cursor to the target, averaged the magnitude of these projections when the cursor was far from the target, and divided *D* by this average value [this procedure was employed previously in (Willett et al. 2017)].

This procedure can be justified as follows. First, we start with the assumption that u_t is the sum of an encoded control vector c_t and a decoding noise vector e_t:

$$u_t = c_t + e_t.$$

Our intent is to normalize *D* such that $\|c_t\| = 1$ when the user is intending to move the cursor at maximum speed. To do this, we first estimate $\|c_t\|$ using unnormalized u_t from time steps when the user intends to move at maximum speed. We then divide *D* by this estimate (which we call c_{max}) to normalize it. Note that simply averaging $\|u_t\|$ over time steps when the user intends to move at maximum speed does not accurately estimate c_{max} (since $\|u_t\| = \|c_t + e_t\|$ follows a "Rice" distribution whose mean is not equal to $\|c_t\|$ in general).

To estimate c_{max} we made the assumption that, when far from the target and when the cursor gain is slow, the encoded control vector c_t points straight from the cursor to the target at its maximum magnitude. Then, we can estimate c_{max} by taking the average of the projection of u_t onto a unit vector that points from the cursor to the target:

$$\begin{aligned} E\left[\left(\frac{g_t - p_t}{\|g_t - p_t\|}\right)\cdot(u_t)\right] &= E\left[\left(\frac{g_t - p_t}{\|g_t - p_t\|}\right)\cdot(c_t + e_t)\right] \\ &= \left(\frac{g_t - p_t}{\|g_t - p_t\|}\right)\cdot(c_t) + E\left[\left(\frac{g_t - p_t}{\|g_t - p_t\|}\right)\cdot(e_t)\right] \\ &= \|c_t\| + 0 \end{aligned}$$

where g_t is the position of the target and p_t is the position of the cursor at time step *t*. The final step follows based on the assumption that c_t is parallel to $\frac{g_t - p_t}{\|g_t - p_t\|}$ and that the error vector is independent of the g_t and p_t and has zero mean.

D can be normalized by dividing by an empirical estimate of c_{max}:

$$\frac{1}{N}\sum_{t=1}^{N}\left[\left(\frac{g_t - p_t}{\|g_t - p_t\|}\right)\cdot(u_t)\right],$$

where *t* indexes over time steps where the user is far from the target (at a distance from the target that is greater than 80% of the starting target distance). We used closed-loop decoder calibration blocks to compute the normalization coefficient, since cursor gain was kept intentionally slow during these blocks.

Section 2: Movement time model fitting

We fit the parameters for the LinPow movement time equation (and other candidate equations that were non-linear in the parameters) iteratively using Matlab2014b's lsqcurvefit function (using the "trust-region reflective" algorithm). To determine an exponent for the *R* term of the proposed LinPow equation, we used lsqcurvefit to find the exponent that led to the largest fraction of variance accounted for (FVAF) across all "datasets" (unique combinations of session date and dwell time setting). To estimate a 95% confidence interval, we used a bootstrapping procedure. We randomly resampled 10,000 times from the 14 datasets to yield 10,000 new sets of 14 datasets; for each set, we found the exponent that led to the greatest mean FVAF across all conditions. We found a lower bound of -3.28 and an upper bound of -1.78. We chose to round the exponent to -2 since it performed similarly to -2.21 when cross-validated and the extra precision was unwarranted given the wide confidence interval.

Section 3: Neural modulation and noise model

To study the properties of each participants' decoding noise, we needed to determine which part of what we decoded was "noise" and which part corresponded to volitional modulation. To do so, we followed the approach taken in (Willett et al. 2017) and modeled the decoded control vector (u_t) as the sum of an underlying "encoded" control vector (c_t) and decoding noise (e_t):

$$u_t = c_t + e_t.$$

We modeled the encoded control vector as a non-linear function of target position and cursor position:

$$c_t = \frac{g_t - p_t}{\|g_t - p_t\|} f_{targ}(\|g_t - p_t\|),$$

where g_t is the target position, p_t is the cursor position, and f_{targ} is a nonlinear, scalar weighting function that we fit empirically. This equation is a simplified version of the feedback control model reported in (Willett et al. 2017) and states that the user pushes the cursor straight towards the target with a force described by f_{targ} as a function of distance from the target. We parameterized f_{targ} as a continuous, piecewise linear function with 12 breakpoints set at the (0, 8.33, 16.66, ... 100) percentiles of target distance ($\|g_t - p_t\|$). Values of the function at the breakpoints were determined using least squares fitting (minimizing the error $\sum_{t=1}^{N} \|u_t - c_t\|^2$ over all time steps of a block, excluding a brief reaction time interval at the beginning of each movement).

To make the session-specific f_{targ} curves appearing in Figure 6D and 6E, we averaged the f_{targ} functions fit to each block in the session. We only included sessions that had enough blocks to accurately fit a radius-specific f_{targ} curve. To yield the block-specific noise signal-dependency curves shown in Figure 6G, the noise standard deviation was estimated separately in each of 20 bins corresponding to 20 levels of control vector magnitude $\|c_t\|$, then normalized by the total noise standard deviation to yield a signal-dependency curve.

Section 4: Comparing the noise properties of our iBCI to the able-bodied motor system

To estimate the SNR of the iBCI, we used data from closed-loop decoder calibration blocks (where the gain was always kept low), so that we could make the safe assumption that the user's encoded control vector was pointing straight from the cursor towards the target at full magnitude. We averaged the pre-smoothed decoder output (u_t) in a 300 ms time window at the beginning of each movement and modeled it as the sum of a vector pointing straight from the cursor to the target (signal) and neural variability (noise) by fitting the following linear model:

$$u_i = b_0 \frac{g_i - p_i}{\|g_i - p_i\|} + \varepsilon_i$$

where u_i is the averaged pre-smoothed decoder output for trial i, g_i is the target position, p_i Is the average cursor position, b_0 is a scalar model parameter and ε_i is spherical Gaussian noise. We fit b_0 (the signal) and the standard deviation of ε (the noise) using least squares regression. This can be done by stacking the X and Y components of the 2D vectors on top of each other to convert it into a univariate regression problem. After computing the SNR, we then used the signal-dependency curves in Figure 6G to estimate how the magnitude of the noise would attenuate with a smaller motor command.

Importantly, for our SNR computations we accounted for each participant's reaction time by starting the 300 ms window *after* a reaction time interval (first 260 ms after target appearance for T6 and 340 ms for T8). These reaction times were determined by visual inspection of the mean angular deviation between u_t and $\frac{g_t - p_t}{\|g_t - p_t\|}$ as a function of time after target appearance.

To estimate the SNR of able-bodied movements, we used published measurements in the literature (Schmidt et al. 1979; Meyer et al. 1988) or re-analyzed publically available data (Young, Pratt, and Chau 2009; Liao and Kirsch 2014) (Supplemental Table 3). When re-analyzing the stylus movements and reaching movements in (Young, Pratt, and Chau 2009; Liao and Kirsch 2014), we measured the accuracy of only the "ballistic" phase of the movement (which we defined to be equal to the first half of the movement, defined by the peak speed). This was necessary because the full movements lasted longer than 200 or 300 ms and included finer, near target corrections. To measure the signal-to-noise ratios of these "ballistic" half movements, we defined the signal as the movement amplitude (the mean distance traveled) and the noise as the standard deviation of the trial-to-trial movement amplitudes.

To make the signal to noise ratio curves plotted in Figure 8D that quantify decoder performance as a function of the number of electrodes used, we generated a single (SNR) vs. (# of Channels) curve for each random target session included in the study (using closed-loop calibration data from that session, as described above). Each point in the curve was generated by randomly sampling 50 times from the pool of available channels and measuring the SNR of a cross-validated decoder (10-folds), averaging across the re-samplings.

Section 5: Decoding from a simulated ensemble of Poisson-distributed neural features

To make the simulated noise curves in Figure 7C, we simulated 1,000 ensembles of 60 Poisson-distributed neural features whose mean firing rates at each time step were determined as follows:

$$f_{t,i} = (b_{0i} + b_{1i}c_t + b_{2i}|c_t|),$$

where $f_{t,i}$ is the mean rate of the ith feature at time bin t, b_{0i} is the baseline rate, b_{1i} and b_{2i} are tuning parameters, c_t is the (one-dimensional) control vector at time bin t, and $|c_t|$ is the magnitude of the control vector at time step t. The baseline rates were drawn randomly from a uniform distribution between 1 and 20 Hz. The b_{1i} coefficients were drawn from a Gaussian distribution with a mean of 0 Hz and a standard deviation of 5 Hz. If $|b_{1i}|$ was larger than the baseline rate, it was re-drawn (to prevent mean firing rates less than zero). The b_{2i} coefficients were either set equal to zero (model A) or to the baseline rate (model B).

With each ensemble, we first generated a calibration dataset for calibrating the decoding matrix. The calibration dataset consisted of 4,000, 20 ms samples of spike counts drawn from a Poisson distribution with mean rates determined by the above equation. For each time step, the control vector was randomly drawn from a uniform distribution between -1 and 1. To calibrate the decoder, we first centered the spike counts and then used "full OLE" (Chase, Schwartz, and Kass 2009) on the individual time step data to build a decoding matrix to predict the control vectors from the mean-subtracted spike counts.

To measure the decoding noise, a separate dataset was generated where the control vector was varied from -1 to 1 in steps of 1/7; at each step, 1,000 samples of spike count data were generated. First, we subtracted the mean of each feature (as estimated from the calibration dataset) from these spike counts. Then, we applied the decoding matrix to each sample and computed the decoding error (the difference between the decoded vector and the true control vector). The 1,000 estimates of decoding error were then used to estimate the noise standard deviation at that control vector magnitude.

Finally, to generate the curves in Figure 7C, we averaged together the control vector magnitude vs. noise magnitude curves estimated from each of the 1,000 simulated ensembles.

Section 6: Statistical analysis of decoding from an ensemble of Poisson-distributed neural features

The simulation results shown in Figure 7 can be further clarified with a simple mathematical model of linear decoding using neural features that are Poisson distributed. We model each neural feature as follows:

$$f_{t,i} \sim Poisson(b_{0i} + b_{1i}c_t + b_{2i}|c_t|)$$

where $f_{t,i}$ is the spike count of the *i*th feature at time bin *t*, b_{1i} is the baseline mean spike count, b_{1i} and b_{2i} are tuning parameters, c_t is the (one-dimensional) control vector at time bin *t*, and $|c_t|$ is the magnitude of the control vector at time step *t*. We assume that the baseline spike counts are large enough relative to the size of the tuning parameters such that the formula for mean spike count is always above zero.

A linear decoding matrix *D* transforms the observed spike counts from all *N* features into a decoded control vector as follows:

$$u_t = \sum_{i=1}^{N} D_i(f_{t,i} - b_{0i})$$

where D_i is the *i*th element of the decoding matrix and u_t is the decoder output. Note that, as part of the decoding process, the mean spike count for each feature is subtracted from the observed spike count before the decoding matrix is applied. We can now write an expression for the variance of the decoder output as follows:

$$var[u_t] = var[\sum_{i=1}^{N} D_i f_{t,i} - D_i b_{0i}]$$

$$var[u_t] = var[\sum_{i=1}^{N} D_i f_{t,i}]$$

The resultant expression is a sum of independent, Poisson-distributed random variables multiplied by constant coefficients from the decoder. Using the property that the variance is equal to the mean for a Poisson-distributed random variable and making the assumption that each variable is independent, the variance can be expressed as follows:

$$
\begin{aligned}
var[\sum_{i=1}^{N} D_i f_{t,i}] &= \sum_{i=1}^{N} D_i^2 \, (b_{0i} + b_{1i} c_t + b_{2i} |c_t|) \\
&= \sum_{i=1}^{N} D_i^2 \, b_{0i} \\
&+ \sum_{i=1}^{N} D_i^2 \, b_{1i} c_t \\
&+ \sum_{i=1}^{N} D_i^2 \, b_{2i} |c_t|
\end{aligned}
$$

The first term of the final sum describes the "baseline" variance caused by the baseline firing rate of each feature. This term describes signal-independent noise that is not affected by the encoded control vector. The second term describes the change in variance caused by cosine tuning to the encoded control vector. This term will be approximately zero as long as the b_{1i} coefficients are "uniformly distributed" with both positive and negative values. The third term describes the change in variance caused by tuning to control vector *magnitude*. This term may cause an increase in variance with an increase in the control vector magnitude if the b_{2i} coefficients are *not* uniformly distributed and are instead mostly positive. This would cause the decoding noise to have some signal-dependency. However, unless this increase in firing rate is substantial compared to the baseline firing rate, the overall noise will still be predominantly signal-independent.

For example, suppose that tuning to control vector magnitude is 100% of the baseline rate; that is, neurons increase their firing rate by 100% overall when the encoded control vector is at its maximum value (as simulated in Figure 7). In this case, the standard deviation of the decoding noise when the user is holding still (c=0) would be $\sqrt{\sum_{i=1}^{N} D_i^2 \, b_{0i}}$ while the standard deviation of the decoding noise when the user is pushing the cursor at full force would be $\sqrt{2 * \sum_{i=1}^{N} D_i^2 \, b_{0i}} \approx 1.41 \sqrt{\sum_{i=1}^{N} D_i^2 \, b_{0i}}$, only a ~41% increase in noise from baseline. This would appear as weak signal-dependence in the decoding noise. Overall, however, the decoding noise would predominantly be signal-independent.

Participant	Condition(s)	Trial Day (Post-Implant Day)	Cursor Gains (WW/s)	Cursor Smoothing (α)	Dwell Times (s)	Target Radii (WW)	Task Visualization	Motor Cue
T6	Mixed Gain & Dwell Time (2 Blocks Each)	727	0.44, 0.77	0.92	0.75, 1.5	0.07, 0.10, 0.13	Cursor	Imagined Index + Thumb
T6	Mixed Gain & Dwell Time (2 Blocks Each)	729	0.51, 0.86	0.92	0.75, 1.5	0.07, 0.10, 0.13	Cursor	Imagined Index + Thumb
T6	Two Gains (5 Blocks Each)	820	0.52, 1.04	0.92	0.75	0.07, 0.10, 0.13	Cursor	Imagined Index + Thumb
T6	Single Gain (10 Blocks)	830	1.09	0.92	0.75	0.07, 0.10, 0.13	Cursor	Imagined Index + Thumb
T6	Single Gain (6 Blocks)	837	3.06	0.92	0.15	0.07, 0.10, 0.13	Cursor	Imagined Index + Thumb
T8	Single Gain (15 Blocks)	101	0.43	0.96	0.5	0.11, 0.14, 0.17	Cursor + Arm	Attempted Arm
T8	Single Gain (16 Blocks)	107	0.74	0.94	0.75	0.10, 0.12, 0.16	Cursor + Arm	Attempted Arm
T8	Single Gain (8 Blocks)	163	1.2	0.94	0.75	0.10, 0.12, 0.16	Cursor + Arm	Attempted Arm
T8	Single Gain (4 Blocks)	179	0.57	0.94	0.75	0.10, 0.12, 0.16	Cursor + Arm	Attempted Arm
T8	Three Gains (2 Blocks Each)	239	0.66, 1.00, 1.33	0.94	0.75	0.10, 0.12, 0.16	Cursor + Arm	Attempted Arm
T8	Two Gains (7 Blocks Each)	274	0.40, 0.64	0.94	0.75	0.10, 0.12, 0.16	Cursor + Arm	Attempted Arm
T8	Single Gain (8 Blocks)	354	0.26	0.96	0.75	0.10, 0.12, 0.16	Cursor + Arm	Attempted Arm

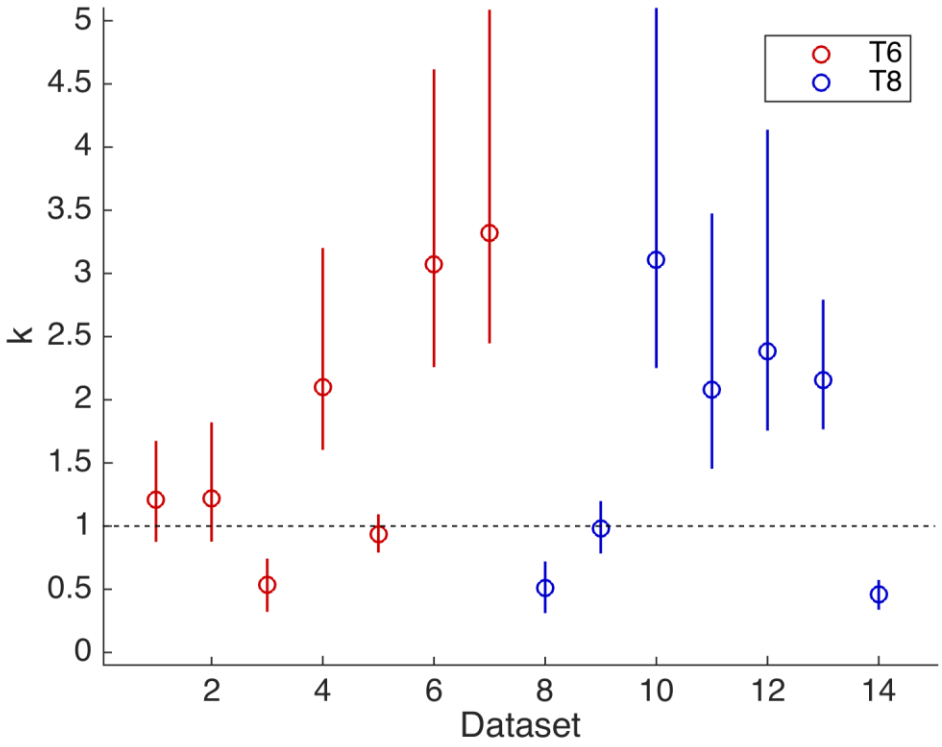
Supplemental Table 1. *A list of all random target sessions included in this paper. WW stands for "workspace width" and refers to the length of a side of the workspace square in which targets randomly appeared (18.5 cm for T6 and 20.5 cm for T8). Note that all target radii are reported as "effective" target radii (target radius plus cursor radius).*

Participant	Conditions	Date (& Post-Implant Day)	Cursor Smoothing (α)	Dwell Time (s)	Target Radius / Target Distance	Task Visualization	Motor Cue
T6	8 Gain Conditions (1 Block Each)	734	0.91	0.15	0.15	Cursor	Imagined Index + Thumb
T6	8 Gain Conditions (1 Block Each)	769	0.91	0.15	0.15	Cursor	Imagined Index + Thumb
T8	4 Gain Conditions (4 Blocks Each)	114	0.94	0.15	0.26	Cursor + Arm	Attempted Arm

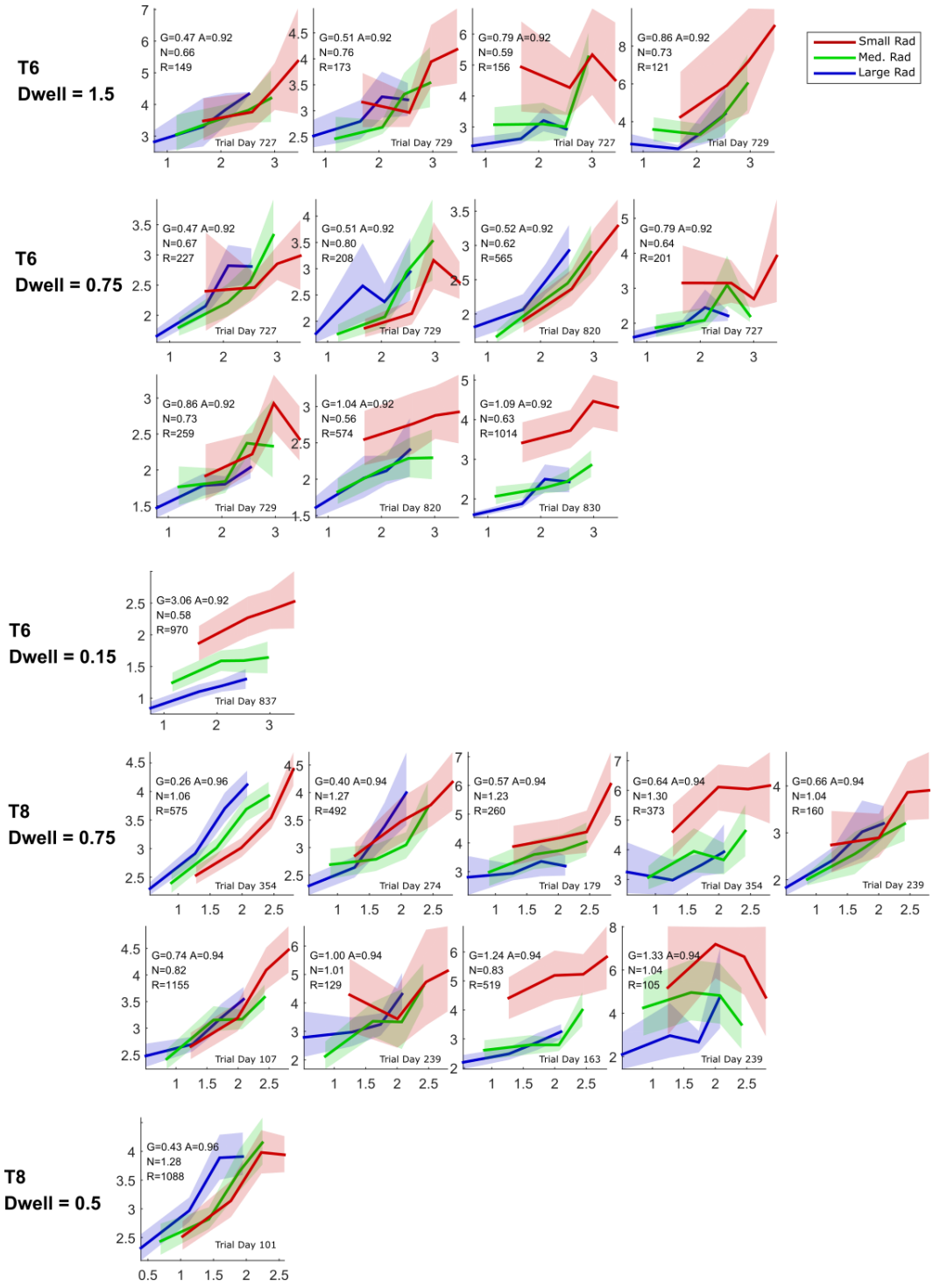
Supplemental Table 2. *Center-out-back sessions included in the paper.*

Task	Scale A	Scale B	Scale C	Scale D	Scale E	Mean Duration of "Ballistic" Component
T6 iBCI	0.1 ± 0.29	0.25 ± 0.30	0.5 ± 0.31	0.75 ± 0.33	1 ± 0.34	300 ms
T8 iBCI	0.1 ± 0.42	0.25 ± 0.43	0.5 ± 0.43	0.75 ± 0.43	1 ± 0.44	300 ms
Schmidt 1979 (Hand Force)	0.1 N ± 0.02	1 N ± 0.075	10 N ± 0.52			"Shots" of pre-programmed force impulses (Figures 2 and 3)
Meyers 1988 (Wrist Rotation)	10 ° ± 1.5	16 ° ± 1.85	25 ° ± 3.2	40 ° ± 4		226 ms (Table 6)
Young 2009 (Stylus Movement, re-analyzed ballistic portion)	3.0 cm ± 0.6	6.1 cm ± 0.85	9.2 cm ± 1.05	12.8 cm ± 1.86		133 ms
Liao 2014 (3D Reaching, re-analyzed ballistic portion)	11.5 cm ± 1.5	15.4 cm ± 1.85	19.9 cm ± 2.1	24 cm ± 2.48		190 ms

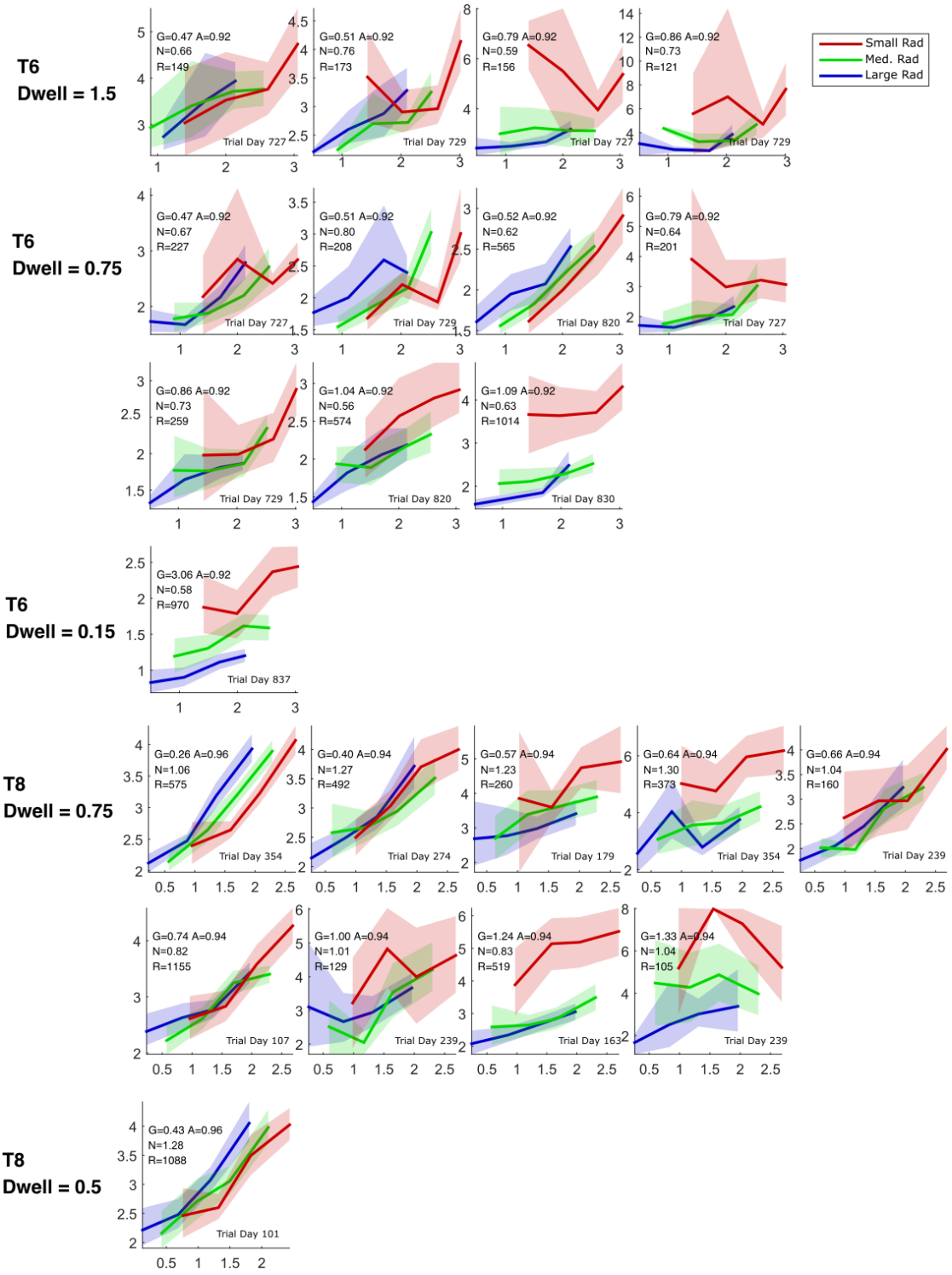
Supplemental Table 3. *Data used to compare the "ballistic" signal to noise ratios of the iBCI to the able-bodied motor system (illustrated in Figure 8). Data is reported in the format [movement amplitude] ± [standard deviation of the movement error].*



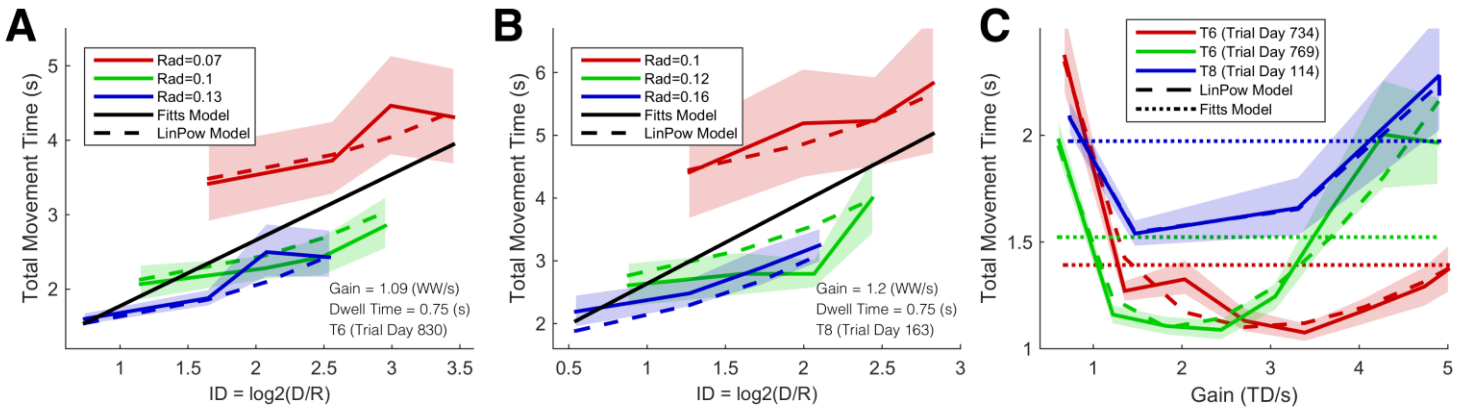
Supplemental Figure 1. *Quantifying the unequal effect of target radius and target distance on iBCI movement times for comparison with previous studies. The degree of asymmetry can be quantified by first fitting the Shannon-Welford movement time model:* $MT = a + b\,log_2(D + 2R) - c\,log_2(2R)$. *The value* $k = c/b$ *then quantifies how much greater an effect target radius has on movement time than target distance does* (Shoemaker et al. 2012; Matlack, Chizeck, and Moritz 2016). *When k=1, the Shannon-Welford model reduces to the Shannon model, a variant of Fitts' law:* $MT = a + b\,log_2(\frac{D}{2R} + 1)$. *Deviations from 1 indicate a departure from Fitts' law. We computed k for each "dataset" (each set of trials corresponding to a unique combination of session date and dwell time setting) and estimated a 95% confidence interval using bootstrapping (we resampled from individual trials 10,000 times). For some datasets, we found very large values of k (several > 3) which significantly exceed the departures from Fitts' law previously reported for able-bodied pointing movements (where k sometimes significantly differed from 1 but did not exceed 1.69) (Shoemaker et al. 2012). In a recent study on iBCI movement times, the value of k reported for the single dataset that was found to be inconsistent with Fitts' law was 1.4, substantially closer to 1 than many of the k values reported here* (Matlack, Chizeck, and Moritz 2016).



Supplemental Figure 2. *Index of difficulty (ID) vs. movement time plots for every unique combination of gain, dwell time, and session date included in the study (some sessions included multiple conditions with different gains and/or dwell times). Six of these plots were chosen as examples to display in Figure 3. Each set contains data corresponding to a certain participant and dwell time. For each individual plot, gain (G), smoothing parameter α (A), noise standard deviation (N), and number of cursor movements (R) are indicated. Bin edges are linearly spaced (compare to Supplemental Figure 3).*



Supplemental Figure 3. *Same as Supplemental Figure 2 except that logarithmically spaced bin edges were used (bin edges at 0.15, 0.22, 0.34, 0.50, and 0.75 workspace widths). Results are not substantially affected by the binning method (panels that depart from Fitts' law do so under either binning method). We used linearly spaced bin edges for all other figures because it resulted in the most balanced distribution of data points across the bins.*



Supplemental Figure 4. *Examples of how the proposed equation (LinPow) can better explain iBCI movement time data that departs significantly from Fitts' law. (A,B) ID vs. Movement Time curves from two random target sessions. Since LinPow has separate radius and distance terms, there is a separate ID vs. movement time line for each target radius (colored dotted lines). In contrast, Fitts' law can only fit a single line (black solid line). (C) LinPow can also explain how changing the cursor gain results in a U-shaped movement time curve (colored dashed lines, one for each session, show the LinPow model). In contrast, Fitts' law predicts that movement time is invariant to cursor gain (horizontal, colored dotted lines).*

To use the LinPow equation to describe the effect of changing the cursor gain while target distance and radius is held constant, we can divide the target distance and radius by the cursor gain to yield:

$$MT(g) = a + b\frac{D}{g} + c(\frac{R}{g})^{-2}$$

$$MT(g) = a + bg^{-1} + cg^{2},$$

where *D* and *R* have been absorbed into the parameters. The resulting function is a U-shaped function of *g*. In this form, the LinPow equation can be used to predict and understand how changing the cursor gain will affect iBCI performance.

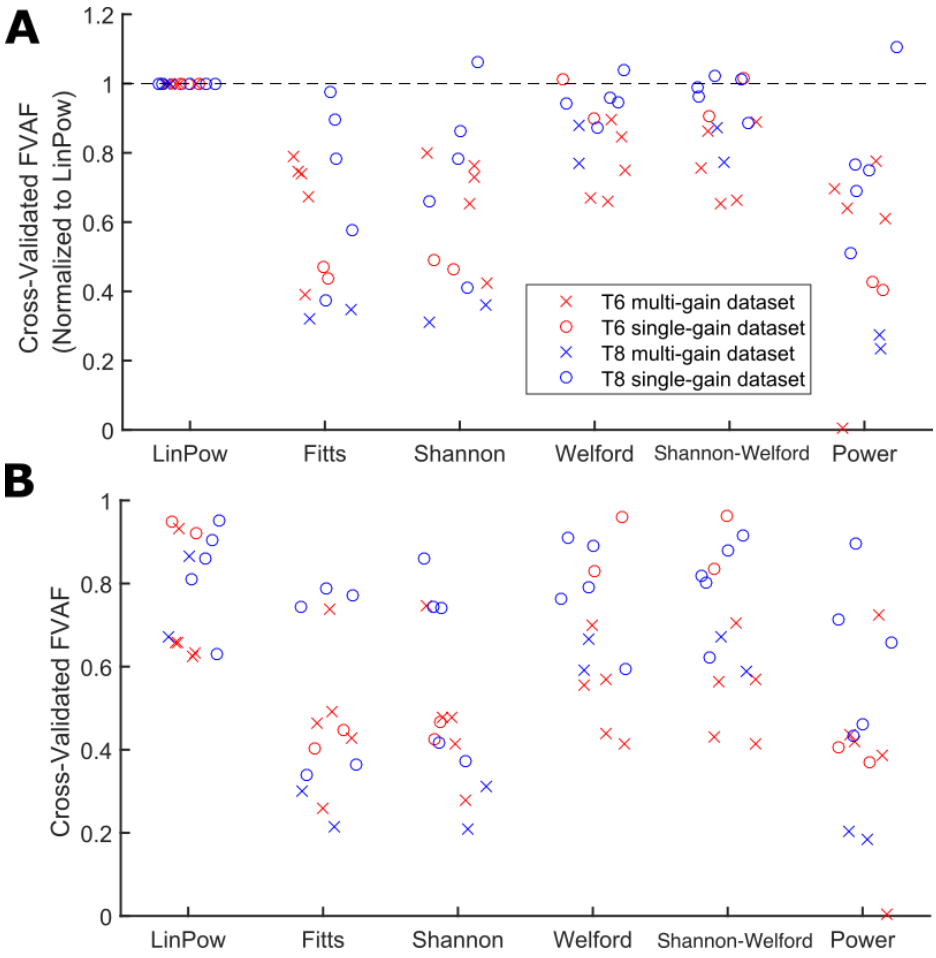
Note that the Welford model, $MT = a + b\log_2(D) - c\log_2(2R)$, and the Shannon-Welford model, $MT = a + b\log_2(D + 2R) - c\log_2(2R)$, also have separate terms for the target distance and radius (Welford, Norris, and Shock 1969; Shoemaker et al. 2012; Matlack, Chizeck, and Moritz 2016). Thus, they also yield a non-constant gain vs. movement time curve. However, this curve is monotonically increasing or decreasing and is not U-shaped. This can be demonstrated for the Welford model as follows (a similar demonstration can be made for the Shannon-Welford model):

$$MT(g) = a + b\log_2(D/g) - c\log_2(2R/g)$$

$$MT(g) = a + b\log_2(D) - b\log_2(g) - c\log_2(2R) + c\log_2(g)$$

$$MT(g) = [a + b\log_2(D) - c\log_2(2R)] + (c - b)\log_2(g).$$

The constant terms that don't depend on gain have been grouped to the left. The term on the right varies as a function of gain but monotonically, either increasing towards infinity if *c* is larger than *b* or otherwise decreasing.



Supplemental Figure 5. *(A) Cross-validated (leave one out) prediction performance of six different movement time equations and fourteen different "datasets" (sets of trials corresponding to a unique session date and dwell time combination). Each symbol corresponds to one dataset. Jitter has been added in the x-direction to aid visualization. Performance is measured by fraction of variance accounted for (FVAF), and is normalized with respect to LinPow (the proposed equation) by dividing by the FVAF of LinPow. Normalization facilitates comparison across datasets with different levels of intrinsic predictability. Values below one indicate worse performance than LinPow. LinPow significantly outperforms the alternative models (t-tests performed on the 14 normalized FVAF scores for each alternative model indicate a mean significantly less than 1 with p<0.01). (B) Same as in(A) except the raw FVAF is reported instead of the normalized version.*

We briefly explain each alternative model here.

The Shannon model modifies Fitts' law based on information-theoretic considerations and can sometimes better describe movement times when D/R is close to zero (MacKenzie 1992). The equation is:

$$MT = a + b\log_2(\frac{D}{2R} + 1)$$

The Shannon model performs very similarly to Fitts' law for our datasets.

Welford's model has the same logarithmic proportionality as Fitts' law, but separates the target distance and radius effects into separate terms (Welford, Norris, and Shock 1969):

$$MT = a + b\log_2(D) - c\log_2(2R).$$

Welford's model does not imply scale invariance. Consequently, it describes the data better than Fitts' law, but still not as well as LinPow, since it retains a logarithmic proportionality.

The "Shannon-Welford" model, a modification of the Welford model (Shoemaker et al. 2012), has recently been used to explain iBCI movement times (Matlack, Chizeck, and Moritz 2016):

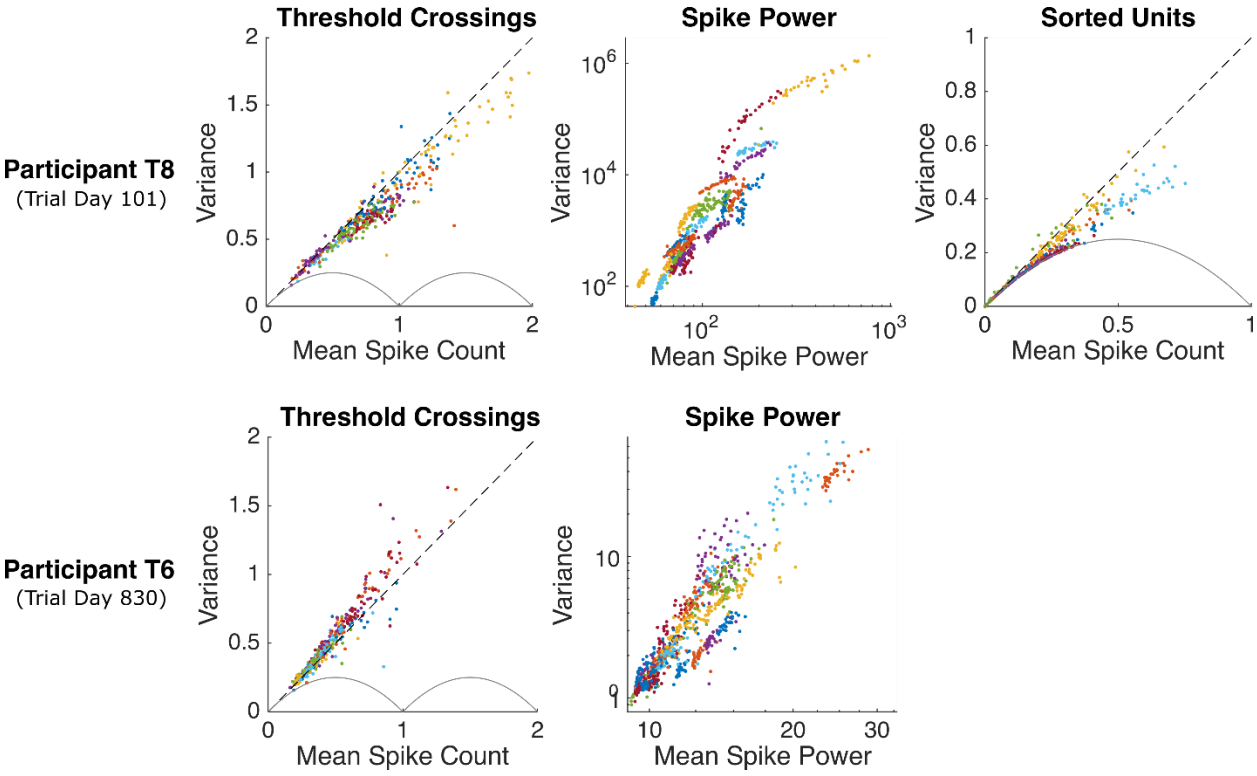
$$MT = a + b\log_2(D + 2R) - c\log_2(2R).$$

The Shannon-Welford model performs similarly to the Welford model and is based on the same basic idea: using separate terms to describe the unequal effects of target distance and radius on movement time.

The power law relaxes the logarithmic constraint of the Fitts and Welford models but retains scale invariance (Kvålseth 2013):

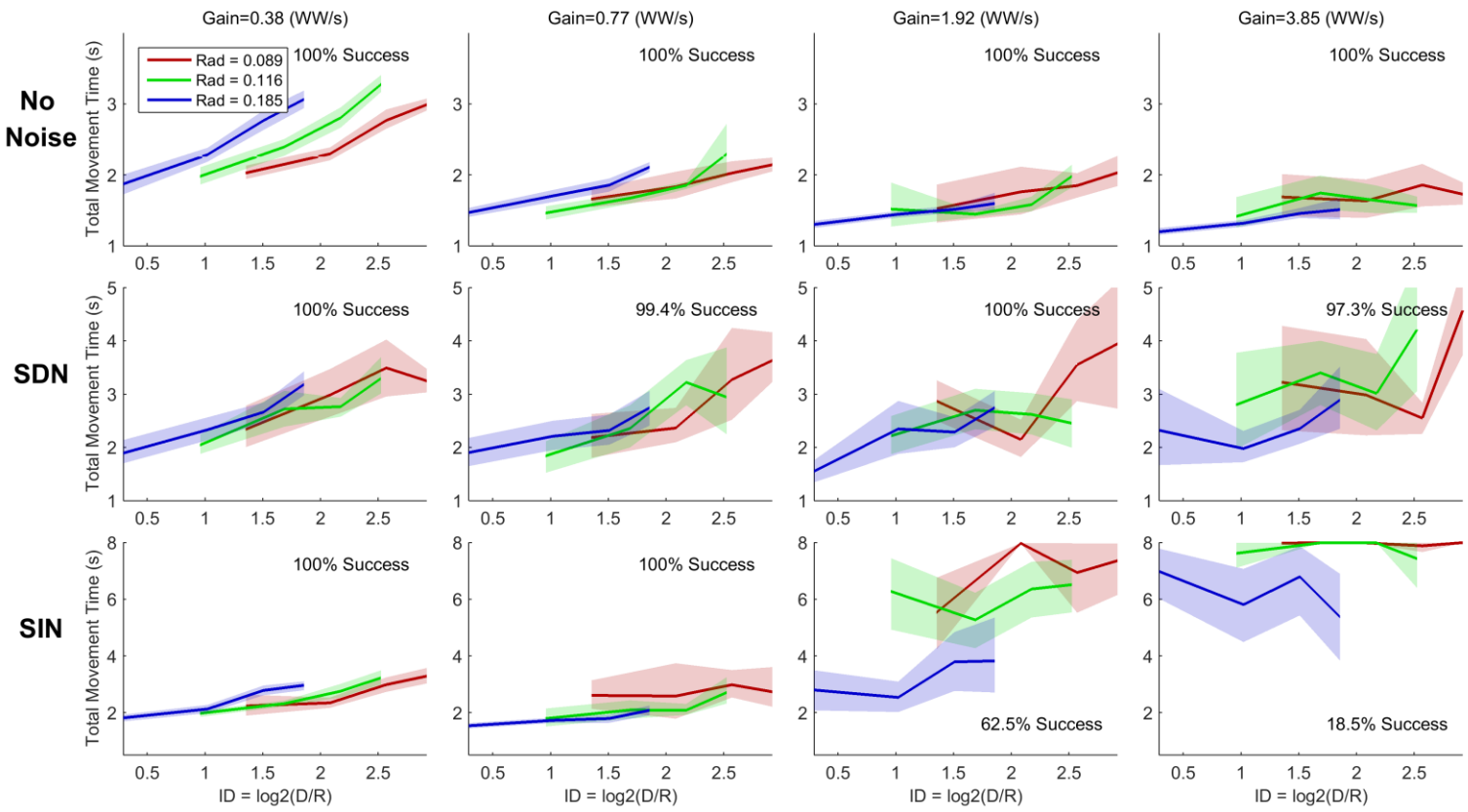
$$MT = a + b\left(\frac{D}{2R}\right)^c.$$

The fact that LinPow outperforms the Welford model and the power law suggest that both the non-logarithmic relationship and the separate *D* and *R* terms are important for describing iBCI movement times.

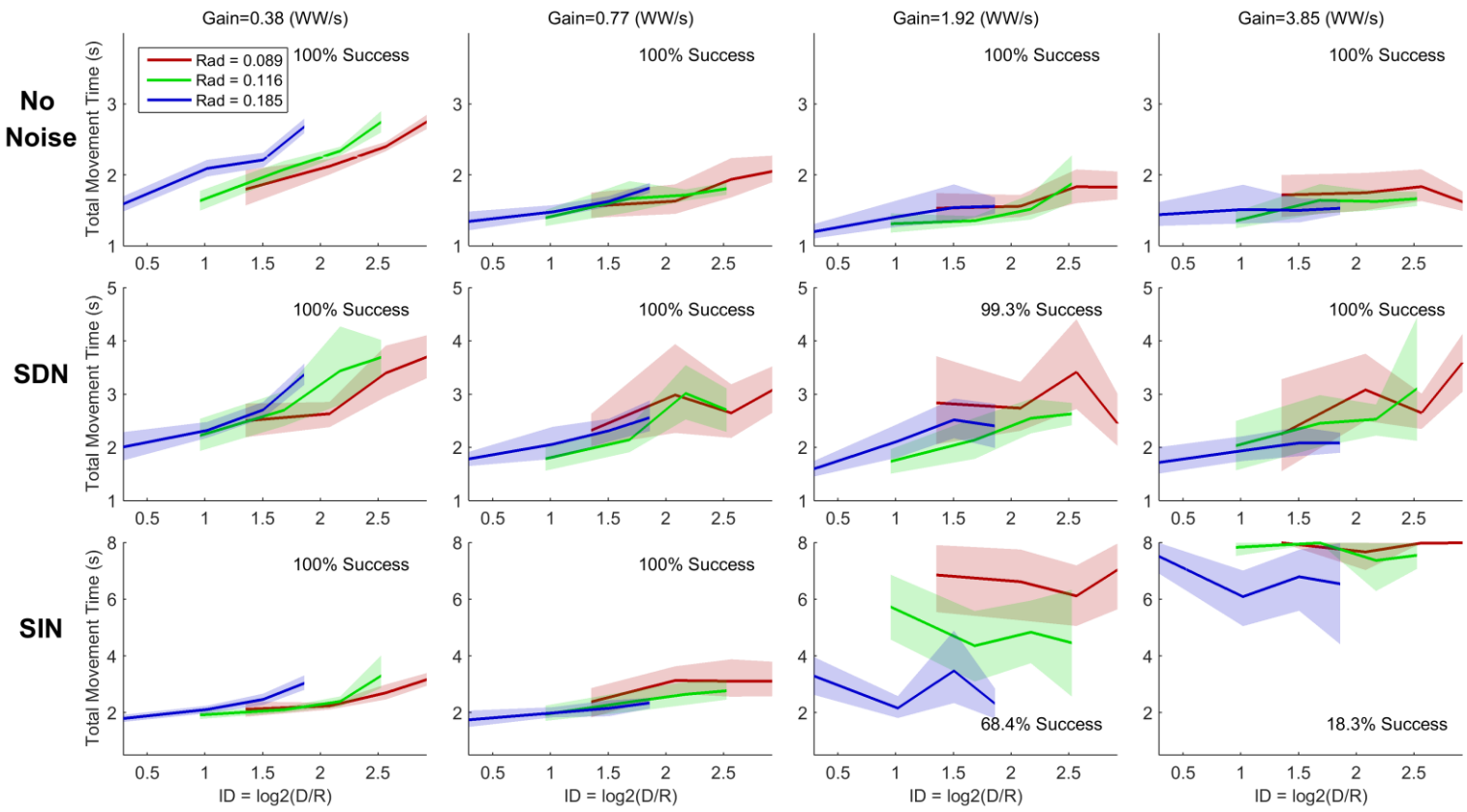


Supplemental Figure 6. *The variance of the neural features we used for decoding (threshold crossing rates and spike power) increased with their mean as expected from previous studies on the firing rates of single neurons. Data is from a single session with participant T8 and T6. Dots of the same color are from the same feature. The gray lines in the threshold crossing and sorted unit panels indicate the minimum possible variance for a given mean (since the spike counts are integers they must have a non-zero variance to obtain non-integer values; de Ruyter van Steveninck et al. 1997). No sorted units are shown for T6 since few isolatable single units were present on the array at that time (trial day 830).*

To make this figure, we analyzed the random target datasets and grouped trials together according to target direction and distance. We used 8 direction categories (grouping trials where the target appeared at an angle from the cursor of 0° to 45°, 45° to 90°, etc.) and 4 distance categories (evenly spaced from the minimum to maximum distance in the dataset). This yielded 8 x 4 = 32 groups of trials. Within each group, we took the neural activity from between 300 and 700 ms after target appearance (using 20 ms bins) and computed the mean and variance of this data, concatenating all samples together within a group. This yielded 32 estimates of mean and variance for each neural feature, which we plotted against each other for each feature that met a significance threshold (cross-validated R2>0.01 when fitting a linear tuning model to the modeled control vector).



Supplemental Figure 7. *Movement time vs. ID curves for able-bodied volunteer B completing the joystick experiment.*



Supplemental Figure 8. *Movement time vs. ID curves for able-bodied volunteer C completing the joystick experiment.*

References

Chase, Steven M, Andrew B Schwartz, and Robert E Kass. 2009. “Bias, Optimal Linear Estimation, and the Differences between Open-Loop Simulation and Closed-Loop Performance of Spiking-Based Brain-Computer Interface Algorithms.” *Neural Networks: The Official Journal of the International Neural Network Society* 22 (9): 1203–13. doi:10.1016/j.neunet.2009.05.005.

Kvålseth, Tarald O. 2013. “An Alternative to Fitts’ Law.” *Bulletin of the Psychonomic Society* 16 (5): 371–73. doi:10.3758/BF03329568.

Liao, James Y., and Robert F. Kirsch. 2014. “Characterizing and Predicting Submovements during Human Three-Dimensional Arm Reaches.” *PLOS ONE* 9 (7): e103387. doi:10.1371/journal.pone.0103387.

MacKenzie, I. Scott. 1992. “Fitts’ Law As a Research and Design Tool in Human-Computer Interaction.” *Hum.-Comput. Interact.* 7 (1): 91–139. doi:10.1207/s15327051hci0701_3.

Matlack, C., H. Chizeck, and C. T. Moritz. 2016. “Empirical Movement Models for Brain Computer Interfaces.” *IEEE Transactions on Neural Systems and Rehabilitation Engineering* PP (99): 1–1. doi:10.1109/TNSRE.2016.2584101.

Meyer, D. E., R. A. Abrams, S. Kornblum, C. E. Wright, and J. E. Smith. 1988. “Optimality in Human Motor Performance: Ideal Control of Rapid Aimed Movements.” *Psychological Review* 95 (3): 340–70.

Ruyter van Steveninck, R. R. de, G. D. Lewen, S. P. Strong, R. Koberle, and W. Bialek. 1997. “Reproducibility and Variability in Neural Spike Trains.” *Science (New York, N.Y.)* 275 (5307): 1805–8.

Schmidt, R. A., H. Zelaznik, B. Hawkins, J. S. Frank, and J. T. Quinn. 1979. “Motor-Output Variability: A Theory for the Accuracy of Rapid Motor Acts.” *Psychological Review* 47 (5): 415–51.

Shoemaker, Garth, Takayuki Tsukitani, Yoshifumi Kitamura, and Kellogg S. Booth. 2012. “Two-Part Models Capture the Impact of Gain on Pointing Performance.” *ACM Trans. Comput.-Hum. Interact.* 19 (4): 28:1–28:34. doi:10.1145/2395131.2395135.

Welford, A. T., A. H. Norris, and N. W. Shock. 1969. “Speed and Accuracy of Movement and Their Changes with Age.” *Acta Psychologica* 30: 3–15. doi:10.1016/0001-6918(69)90034-1.

Willett, Francis R., Chethan Pandarinath, Beata Jarosiewicz, Brian A. Murphy, William D. Memberg, Christine H. Blabe, Jad Saab, et al. 2017. “Feedback Control Policies Employed by People Using Intracortical Brain–computer Interfaces.” *Journal of Neural Engineering* 14 (1): 16001. doi:10.1088/1741-2560/14/1/016001.

Young, Scott J., Jay Pratt, and Tom Chau. 2009. “Target-Directed Movements at a Comfortable Pace: Movement Duration and Fitts’s Law.” *Journal of Motor Behavior* 41 (4): 339–46. doi:10.3200/JMBR.41.4.339-346.