Psychophysiology, 52 (2015), 157–168. Wiley Periodicals, Inc. Printed in the USA.
Copyright © 2014 Society for Psychophysiological Research
DOI: 10.1111/psyp.12317

Regression-based estimation of ERP waveforms: I. The rERP framework

NATHANIEL J. SMITH[a] AND MARTA KUTAS[b]

[a]School of Informatics, University of Edinburgh, Edinburgh, Scotland
[b]Departments of Cognitive Science and Neurosciences, University of California, San Diego, San Diego, California, USA

Abstract

ERP averaging is an extraordinarily successful method, but can only be applied to a limited range of experimental designs. We introduce the regression-based rERP framework, which extends ERP averaging to handle arbitrary combinations of categorical and continuous covariates, partial confounding, nonlinear effects, and overlapping responses to distinct events, all within a single unified system. rERPs enable a richer variety of paradigms (including high-*N* naturalistic designs) while preserving the advantages of traditional ERPs. This article provides an accessible introduction to what rERPs are, why they are useful, how they are computed, and when we should expect them to be effective, particularly in cases of partial confounding. A companion article discusses how nonlinear effects and overlap correction can be handled within this framework, as well as practical considerations around baselining, filtering, statistical testing, and artifact rejection. Free software implementing these techniques is available.

Descriptors: Other, Language/Speech, Normal volunteers, EEG/ERP

Electroencephalogram (EEG) recorded at the scalp measures activity from many different parts of the brain, as well as various nonbrain artifacts. Only a small portion of this summed activity reflects processing related to any particular task, and so analyzing and interpreting these data requires a mechanism to isolate the task-related "signal" from the unrelated "noise." One strategy is to estimate the event-related potential (ERP), that is, the portion of the EEG signal that is consistently present across trials and time-locked to some event of interest; the usual method for performing this estimation is to extract time-locked epochs from the continuous EEG signal, align them, and compute their point-by-point average.

This technique has compelling advantages. It produces detailed waveforms that give a millisecond by millisecond trace of how processing evolves at each electrode across each condition, and has a long track record of effectively producing insight into the dynamics of neural processing. The resulting body of literature provides a critical source of comparisons for interpreting new results, and has produced a large stock of carefully characterized components such as the P300, N400, etc., which can be used as dependent measures in further experiments. This success also has led to a family of analogous techniques for analyzing related signals: the event-related magnetic field (ERF), event-related spectral perturbation (ERSP; Makeig, 1993), event-related optical signal (EROS; Gratton & Fabiani, 1998), etc., with similar advantages.

However, these techniques also share the primary disadvantage of traditional ERP estimation, which is that the averaging technique places limitations on experimental design: for optimal results, stimuli must be chosen to fall into a small set of discrete categories, and these categories must be carefully controlled to ensure that all other stimulus properties that might affect neural processing are held constant. Furthermore, if multiple events occur in close temporal proximity, the ERPs time-locked to each will generally overlap in both the EEG signal and the resulting ERP estimates, making it difficult to determine which portions of the final waveform are attributable to which event.

These are not merely theoretical problems. For example, in our subfield of language comprehension, the stimuli—words—necessarily vary along a large number of continuous and confounded dimensions (Cutler, 1981). In this domain, stimuli can never be chosen to fully avoid confounding, and attempting to do so leads to the use of nonrepresentative materials. Handling continuous dimensions requires dichotimization, which wastes data and prevents the use of potentially more powerful continuous parametric designs (Baayen, 2004, 2010; Cohen, 1983). In reading paradigms using rapid serial visual presentation (RSVP), the problem of overlap may motivate the use of presentation rates that are well below natural reading speeds (Dambacher et al., 2012); in auditory speech

This work was supported in part by NIH grant T32-DC000041 to the Center for Research in Language, NIH grant T32-MH20002 to the Institute for Neural Computation, NICHD grant HD22614 to MK, funding from the Army Research Laboratories Cognition & Neuroergonomics Collaborative Technology Alliance, and funding from the European Union's 7th Framework Programme under grant agreement No. 270273 (XPERIENCE). We thank R. H. Baayen, K. A. DeLong, S. Frank, D. Groppe, T. F. Jaeger, R. T. Knight, R. Levy, S. Makeig, S. Narayan, M. Ramscar, M. Steedman, T. Urbach, and two anonymous reviewers for comments and discussion.

Address correspondence to: Nathaniel Smith, University of Edinburgh, Informatics Forum 3.29, 10 Crichton Street, Edinburgh, EH8 9AB, Scotland, UK. E-mail: nathaniel.smith@ed.ac.uk

comprehension, the confounding and overlap problems are even worse, and these difficulties may contribute to a relative paucity of auditory language ERP studies, even though EEG is one of the only available dependent measures for probing the time course of auditory comprehension. In the behavioral literature, recent years have seen an increasing number of studies based on behavioral corpora, which provide a valuable complement to factorial designs by trading off a priori control of confounds to achieve much larger sample sizes and increased naturalism (e.g., Boston, Hale, Kliegl, Patil, & Vasishth, 2008; Demberg & Keller, 2008; Kliegl, Nuthmann, & Engbert, 2006). Such studies can potentially allow for the measurement of detailed quantitative effects that are beyond the reach of smaller, designed studies (New, Ferrand, Pallier, & Brysbaert, 2006; Smith & Levy, 2013), but these analyses rely crucially on statistical methods for post hoc control of confounding, which traditional ERP averaging cannot provide.

The challenge, then, is to preserve the advantages of ERP-like methods while relaxing their limitations. A number of alternatives to ERP averaging have been proposed to solve one or another of these problems, but taking advantage of these newer techniques requires the potential user to understand and navigate a complex set of tradeoffs. "ERP images," for example, allow one to visualize the effect of a continuous covariate such as reaction time on the ERP, but can handle only one such covariate at a time, and have no way to control for confounding (Jung et al., 2001; Lorig & Urbach, 1995). A number of studies have used multiple regression to analyze the average amplitude of the EEG extracted from a window from either single trials (Amsel, 2011; Dambacher, Kliegl, Hofmann, & Jacobs, 2006; Frank, Otten, Galli, & Vigliocco, 2013; Groppe et al., 2010) or single-item ERPs (Laszlo & Federmeier, 2011, 2014); this technique naturally allows post hoc control of multiple simultaneous discrete and continuous covariates, but the need for prespecified analysis windows means that it cannot produce the temporally detailed waveforms that are one of the primary attractions of ERP analysis. A more promising approach is to calculate a separate regression model at each possible latency, similar to the mass univariate techniques used in PET/fMRI analysis, variations of which have been proposed by multiple groups under different names: the event-related regression coefficient (ERRC; Hauk, Davis, Ford, Pulvermüller, & Marslen-Wilson, 2006; Hauk, Pulvermüller, Ford, Marslen-Wilson, & Davis, 2009; Miozzo, Pulvermüller, & Hauk, 2014), general linear model (GLM) analysis (Rousselet, Pernet, Bennett, & Sekuler, 2008; Rousselet et al., 2009, 2010; Pernet, Chauveau, Gaspar, & Rousselet, 2011), correlational analysis (Ettinger, Linzen, & Marantz, 2014; Solomyak & Marantz, 2009, 2010), or no name at all (Amsel, 2011); a closely related proposal is to analyze EEG via nonparametric regression based on generalized additive models (Hendrix, 2009; Hendrix, Bolger, & Baayen, 2014; Kryuchkova, Tucker, Wurm, & Baayen, 2012; Tremblay, 2009; Tremblay & Baayen, 2010). These approaches can naturally handle multiple covariates while still providing information on the time course of effects, but have no provision for handling overlap correction. Meanwhile, the ADJAR technique (Woldorff, 1993) does allow for a limited form of overlap correction, but it requires the user to make complex heuristic judgments, and applies only to classic categorical ERPs. And all of these approaches face the hurdle of convincing potential users to either give up or somehow adapt the large existing comparative literature, "componentology," and store of experience and lore that practitioners have accumulated around ERPs.

In this article and its companion, we introduce a regression-based framework for estimating ERPs—the rERP framework for short. However, our goal is not to give experimenters yet another ERP alternative to choose from. Nor is our goal to get rid of ERPs—quite the opposite. Instead, we start from the well-known observation that averaging is a special case of least squares linear regression, and use this to extend traditional ERP estimation to handle a much broader range of analysis problems in a unified way. The rERP framework provides a single method for estimating ERP waveforms that works whether the design is factorial or continuous or both, whether it is orthogonal or partially confounded, whether the continuous covariates have linear or nonlinear effects, and whether the events of interest produce overlapping ERPs or not. In the simplest case—a categorical design with no overlap correction—then the rERP estimates we obtain will be mathematically identical to those produced by traditional averaging. This makes it trivial to carry over previous ERP results to rERP: all ERPs *are* rERPs.

But rERPs are also flexible enough to handle complex cases where traditional ERP averaging does not apply. In fact, it turns out that every ERP alternative mentioned above, or a close analogue, also falls out naturally as a special case of the rERP approach. Thus, one of our key contributions is to demonstrate that the apparent complexity in this literature is largely illusory: there's no need to separately learn five or more distinct methods, because the rERP framework encompasses and unifies them all. rERPs aren't magic: disentangling confounded covariates, for example, may require more data than would be required if they were orthogonal. But if there is a mild violation of the ERP assumptions—for example, a poorly controlled nuisance variable that is correlated with our variable of interest—then we can statistically control for that nuisance variable while calculating what is otherwise a traditional ERP waveform. And the use of a unified framework allows us to straightforwardly handle even the most complex situations. For example, previously, if one wished to use an ERP image to obtain a detailed picture of a continuous covariate's effect, one could not also correct for overlap, or use multiple regression to disentangle the simultaneous effects of multiple covariates. But in the rERP framework, we can mix and match all the different aspects as appropriate to the situation, and import new ideas from the regression literature as needed. And, because these different options fit together into a single system, it becomes easier to articulate and reason about the trade-offs and relationships between different analytic approaches. Finally, we note that, while we will focus our presentation here exclusively on EEG/ERP analysis, the approach generalizes directly to related modalities, producing rERFs, rERSP, rEROS, etc.

The remainder of this article is structured as follows. We first review the theoretical motivation underlying the traditional averaging approach, and show that the same motivations lead naturally to a specific way of applying least squares regression. Since regression is so well studied, this unlocks a vast literature of tricks and techniques that can then be applied directly to ERP estimation. To aid in making this mapping, we next work through several examples with a simple experimental design, both to illustrate the principles of rERP analysis and to show how the ideas and terminology of ERPs correspond to those used in the regression literature. Finally, we provide a detailed discussion of how regression can (sometimes) disentangle the effects of partially confounded factors, and the trade-offs involved in choosing between the complex designs that our technique makes available. A companion article builds on this foundation to discuss two more sophisticated applications of the framework—the use of spline regression (a generalization of both dichotomization and ERP images) to measure nonlinear effects of continuous predictors, and a technique for correctly estimating ERPs in the presence of overlap—as well as ancillary practical considerations such as baselining, filtering, significance testing, and artifact rejection. A list of free software packages implementing these methods is available at http://vorpus.org/rERP.

ERP Averaging Is Least Squares Regression

ERP analysis starts from the assumption that whenever a particular type of event occurs (e.g., a stimulus appears on a display), then the brain produces a fixed pattern of neural activity—the ERP itself—that is time-locked to that event.[1] Here, we consider how to derive a method for estimating such an ERP from data, starting from first principles, and then show how the same principles lead to rERP analysis. To reduce confusion, in this section we'll be careful to say *ERP estimate* when referring to any estimate based on data, and say *ERP* alone only when referring to the postulated underlying brain activity that we hope our estimates will approximate.

In an ideal world, we could just present our stimulus once, record the resulting brain activity, and be done. But reality, of course, is never that simple. There is always a great deal of other ongoing neural activity that doesn't care about our experimental manipulation at all (or if it does care, then not in a way that matches the assumptions of ERP analysis), which means that what we actually measure on any single trial will be the sum of the ERP activity and this background activity. So, given that we can't measure the ERP directly, how can we estimate it from the data that we can measure?

Instead of trying to estimate the whole waveform at once, we start by working out how to estimate the value of the ERP at one single electrode and latency—for example, 136 ms postevent at electrode Cz. (If we can do this, then we can estimate the rest of the ERP by just repeating our technique at every electrode and latency.) This means the value we're trying to estimate is just a single number, which we call β (pronounced beta). To estimate β, we use the measurements we've made of the scalp potential at 136 ms postevent, at Cz, on many trials—this is a list of numbers, which we call $y_1, y_2, y_3, \ldots, y_n$. And our assumption is that the physical process that produced these numbers was the summation of the true ERP plus each trial's background "noise." So, we can write the relationships between the different numbers involved here as:

$$\begin{aligned} y_1 &= \beta + \text{noise}_1 \\ y_2 &= \beta + \text{noise}_2 \\ &\vdots \\ y_n &= \beta + \text{noise}_n \end{aligned}$$

1. This assumption may or may not be an accurate description of the underlying neural processes in any particular case, and a number of alternative mechanisms have been proposed (Burgess, 2012; Nikulin et al., 2007; Sauseng et al., 2007). For our current purposes, this doesn't really matter; our goal here is to use this assumption to derive a useful method, and the empirical success of the ERP averaging technique demonstrates that this assumption can lead to useful analyses regardless of its objective truth. rERP analysis, being an extension of ERP analysis, starts from the same assumption, and will turn out to have similar properties (i.e., any pattern of neural activity that can be picked up by averaging can also be picked up by rERP, whether or not it arises from a "true" evoked potential).

Or, for short, we write

$$y_i = \beta + \text{noise}_i$$

Notice that on every trial, the value of the noise (at this latency and electrode) is different, but the value of the ERP (at this latency and electrode) is always the same.

At this point, most ERP texts would suggest that we just estimate β by taking the average of the y_i values. But why, mathematically, is that a good idea?

We know the values of y_i, but not β or noise_i. If we knew what noise_i was on each trial, we could solve for β using algebra. Contrariwise, given any estimate of β, we could solve for the estimated pertrial noise: $y_i - \beta = \text{noise}_i$. Because we know neither, we need some other strategy, and the oldest, simplest, and most widely studied strategy for solving such problems is the principle of least squares. This principle says that we should choose our estimate of β to be the number that makes our estimate for the total squared noise,

$$\text{squared noise} = \sum\nolimits_{i=1}^{n} (\text{noise}_i)^2 = \sum\nolimits_{i=1}^{n} (y_i - \beta)^2$$

as small as possible. To minimize this formula, we first take the derivative:

$$\frac{d}{d\beta}\text{squared noise} = \frac{d}{d\beta}\sum\nolimits_{i=1}^{n} (y_i - \beta)^2 = \sum\nolimits_{i=1}^{n} -2(y_i - \beta).$$

Then, we set it equal to zero:

$$\sum\nolimits_{i=1}^{n} -2(y_i - \beta) = 0$$

And finally we solve for β:

$$-2\left(\sum_{i=1}^{n} y_i - \sum_{i=1}^{n} \beta\right) = 0$$

$$\sum_{i=1}^{n} y_i = \sum_{i=1}^{n} \beta$$

$$\sum_{i=1}^{n} y_i = n\beta$$

$$\frac{1}{n}\sum_{i=1}^{n} y_i = \beta$$

Notice that this final formula turns out to be the standard formula for calculating the mean. This means that, according to the least squares principle, the best way to estimate β is to take the average of our measured values, $y_1, \ldots, y_n$. This is the reason why using averaging to estimate ERPs makes sense in the first place.

From Averaging to Regression

So, the traditional averaging method for estimating ERPs can be justified as being the least squares solution to the equation

$$y_i = \beta + \text{noise}_i$$

Now, notice that the above equation is just a simple example of the general least squares linear regression formula:

$$y_i = \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \text{noise}_i$$

What happens if we estimate ERPs using full-fledged linear regression, instead of the simplified version?

Just as before, the y_i values are set to the measured scalp potential at a single electrode, at a single latency, across different time-locked trials. The x_{ji} values (the predictors) are set to indicate various properties of the stimulus presented on trial i, coded numerically. (The original ERP equation that we derived above effectively has a single predictor, x_{1i}, whose value is always 1. We can think of this as a particularly vague property that simply indicates that there was an event.) And, once we've measured the y_i values and specified the x_{ji} values, we again use the principle of least squares to find those values for β_1, β_2, ... that together minimize the total squared noise. Each β value then gives an estimate of some portion of the ERP at this electrode and latency. Alternatively, given the properties of any particular stimulus i, we can compute the sum $\beta_1 x_{1i} + \beta_2 x_{2i} + \cdots$, which we call the model's prediction of the ERP to a stimulus with these properties at this latency.

Actually finding the β values that satisfy the least squares principle is somewhat more complicated than just taking the average, but not by much, and there are standard techniques that allow computers to accomplish this quickly and reliably. We then repeat these calculations many times, once for each electrode and latency—the whole process takes a few tens of milliseconds on a modern computer. As we do, we keep the same *x*s—since these represent properties of the event that each trial is time-locked to, which do not vary across electrodes or latencies—but swap out the y values to represent the measurements made at each electrode and latency across our different trials. Finally, we gather up all the computed β_1 values to make one waveform, all the β_2 values to make a second waveform, and so on for all of the βs. The resulting waveforms can then be plotted, smoothed, entered into statistical analysis, have amplitude and latency measures extracted, and generally be treated exactly as if they were ERP waveform estimates obtained via averaging. Likewise, we can combine βs together to compute the predicted waveforms for particular stimuli, and these predictions can also be analyzed like ERP estimates obtained from averaging. To remind ourselves that our waveforms were estimated using regression instead of averaging, we call them *rERPs*.[2]

2. Previous authors have argued for an analogy between regression coefficients and classic ERP averages on the grounds that both can be computed by taking certain weighted sums of the input data (Hauk et al., 2006, 2009; Miozzo et al., 2014). This is true, but it leaves important questions unanswered. There are many ways of weighting the input data so that their sum does not produce any useful value, which means there must be something special about the particular weights that are used in regression and in averaging. What's so special about these weights, and how do regression weights relate to the more familiar averaging weights? If we focus on weighted sums, these questions are difficult to answer because, for regression, the weights have no intuitive relationship to the original experimental design: they are the output of a rather opaque calculation (and in efficient implementations may not be explicitly computed at all). But if we stop worrying about weighted sums and instead observe that at a high level these techniques both select their weights so as to find the unique, best-possible estimate (in the least squares sense) of an underlying ERP signal buried in noise, then the connection between averaging and regression immediately becomes clear, along with the implication that in the cases where both techniques are applicable they will end up using identical weights to produce identical results.

Defining Predictors for rERP Analysis

While rERPs can be treated much like traditional ERP estimates, they do require an important shift in our perspective. Many of us are used to solving data analysis problems by reasoning out which sequence of operations we should apply to our data to achieve our desired result: first dividing into bins, then averaging, then subtracting to create difference waves, etc. When we are then confronted with a new problem (e.g., correcting for a partially confounded variable), it's natural to try to find a solution along similar lines (e.g., estimating some sort of correction factor and then subtracting it out). But in the regression framework, this is not the most productive approach. Instead, we focus on deciding which set of predictors can be used to best characterize our events; the least squares program will then take care of automatically deriving the optimal data processing method from this description. Going from averaging to regression is like going through the looking glass: our standard ERP techniques turn out to have perfect analogues in standard regression techniques, but the terminology and framing are quite different.

This section acts as an introduction to the looking glass world and its correspondences to the familiar ERP world, giving a step-by-step examination of standard ways of setting up rERP predictors to handle factorial, continuous, and combined designs.

We use examples and data drawn from a published experiment by DeLong, Urbach, and Kutas (2005). This is a language comprehension experiment that created contexts in which participants had a graded expectation for either the word *a* or the word *an*, such as *The day was breezy so the boy went outside to fly (**a** kite/**an** airplane).* Thus, our example design has one categorical covariate—word identity, *a* versus *an*—and one continuous covariate—word expectancy, which falls between 0 and 1.

The Traditional ERP as an Intercept Term

The simplest example is the one we have already seen. Suppose we define just a single predictor as

$$x_{1i} = 1$$

In linear regression terminology, this predictor is known as an intercept term. Then, our regression equation is

$$y_i = \beta_1 x_{1i} + \text{noise}_i = \beta_1 + \text{noise}_i$$

and, as we saw above, when we find the least squares solution, β_1 will end up equal to the mean of the y_i values (Figure 1a).

Therefore, this is not only a legitimate method for estimating the activity time-locked to some event, but it produces results that are identical to the conventional averaging technique. However, it is somewhat cumbersome to use, because if we have categorical factors with multiple levels, then it requires us to fit two different models: one on the *a* trials, and another on the *an* trials.

Multiple ERPs Via Dummy Coding

Instead of fitting multiple models, we can estimate both ERPs at once within a single regression model by using a trick known as *dummy coding*, which is one of the standard ways to handle

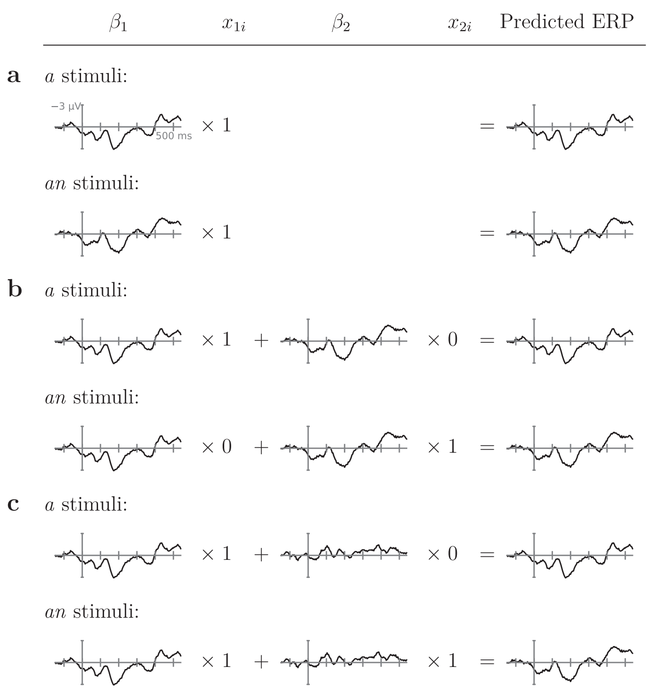

Figure 1. The relationship between *x*s (predictors), β coefficients (rERPs), and predicted ERPs, for various stimuli and models. Waveforms shown are unsmoothed grand-average rERPs fit to data from DeLong et al. (2005). a: Intercept-only models fitted to subsets of the data. b: Dummy coding. c: Treatment coding.

categorical variables within regression models.[3] If we define two different predictors, like so:

$$x_{1i} = \begin{cases} 1, & \text{if stimulus } i \text{ is } a \\ 0, & \text{if stimulus } i \text{ is } an \end{cases}$$

$$x_{2i} = \begin{cases} 0, & \text{if stimulus } i \text{ is } a \\ 1, & \text{if stimulus } i \text{ is } an \end{cases}$$

and then plug them into the standard regression equation:

$$y_i = \beta_1 x_{1i} + \beta_2 x_{2i} + \text{noise}_i$$

then least squares fitting will set β_1 to the average of the *a* trials, and β_2 to the average of the *an* trials (Figure 1b).

It's easy to see why this happens. If on trial *i*, we displayed the word *a*, then x_{1i} is 1 and x_{2i} is 0, so we have

$$y_i = \beta_1 \times 1 + \beta_2 \times 0 + \text{noise}_i = \beta_1 + \text{noise}_i$$

Or, if on trial *i*, we displayed *an*, then x_{1i} is 0 and x_{2i} is 1, so we have

$$y_i = \beta_1 \times 0 + \beta_2 \times 1 + \text{noise}_i = \beta_2 + \text{noise}_i$$

3. This is the default method of coding categorical variables used by SAS, and is also used by default by R for models that do not contain an intercept term.

So effectively we end up fitting two copies of our previous intercept-only model on two different subsets of the data. The only difference from the previous example is that, before, we did this by explicitly dividing our data into subsets; now, we just define our *x*s and the appropriate data splitting happens automatically as a result of the least squares fitting process. A generalized version of this "zero trick" can be used to combine arbitrary regression models into a single fit, and we'll see later that this is useful for several different purposes.

Difference ERPs Via Treatment Coding

However, a more common method for handling categorical variables in regression is by *treatment coding*.[4] This consists of dummy-coding all but one of the levels of our factor (we refer to the level that's left out as the *reference level*), and then adding an intercept term. For example, taking the *a* stimuli as our reference level, we have:

$$x_{1i} = 1, \qquad x_{2i} = \begin{cases} 0, & \text{if stimulus } i \text{ is } a \\ 1, & \text{if stimulus } i \text{ is } an \end{cases}$$

With this coding scheme, least squares fitting will set β_1 to the average of the *a* trials, and β_2 to the difference between the *an* trials and the *a* trials, that is, β_2 will be a conventional difference ERP (Figure 1c). This is why the regression literature calls this *treatment coding*: if you choose a control condition for your reference level, and then apply various "treatments" on top of it, then the resulting βs show you how these treatments change the ERP response versus the control.

This also illustrates a very important aspect of interpreting regression formulas, which is that the fitted value of each β_j depends not just on how we defined the *j*th predictor, but on *all* the predictors, $x_{1i}, \ldots, x_{ni}$. This example and the previous one used the same definition for x_{2i}, and only changed x_{1i}; but the result was that β_1 stayed the same, while β_2 changed completely.[5] Likewise, we saw above that, if all we have is an intercept, then the corresponding β will give the grand mean of all our data, but here it gives the mean of the *a* stimuli only. This behavior can be quite confusing when first encountered.

The key to interpreting the β values produced by these models—and thus to interpreting rERP waveforms in general—is to remember that the least squares fitting process does not care about the β values directly. It only cares about the predicted values (the rightmost column in Figure 1). It will pick whichever β values make these predictions match the data as closely as possible. Since the predictions are created by combining multiple β values together, this means that the chosen β values are not the ones that individually match the data best, but the ones that are most effective at working together. In the treatment coding case, β_1 must work alone to match the *a* stimuli, while for the *an* stimuli, β_1 and β_2 work together; so the most effective teamwork is achieved

4. This is the default method of coding categorical variables in R and SPSS.

5. This is also why we prefer the name treatment coding for this particular coding scheme, even though SPSS and many references refer to it as, simply, dummy coding. Using 0/1 dummy coding for some levels of a factor can produce very different results depending on how other levels are coded, making the name ambiguous; treatment coding refers specifically to this scheme combining an intercept term with dummy coding for all but one level.

when β_1 focuses on matching the *a* stimuli while β_2 focuses on correcting β_1 so that their combination will match the *an* stimuli. And this teamwork will turn out to be the essential feature that allows regression to handle confounding, nonlinear effects, and overlap correction.

But it also creates a potential problem for some coding schemes that might otherwise seem reasonable: those in which several predictors are perfectly collinear, that is, redundant.

Examples of this would be if we accidentally entered the same predictor twice, if we left some cell out of our design, or, less obviously, if we used our original simple dummy coding scheme, but for two different factors at once. For instance, pretend that sometimes in this experiment the critical items were displayed in UPPERCASE. Then, we could define four dummy-coded predictors:

$$x_{1i} = \begin{cases} 1, & \text{if stimulus } i \text{ is } a \\ 0, & \text{if stimulus } i \text{ is } an \end{cases}$$

$$x_{2i} = \begin{cases} 0, & \text{if stimulus } i \text{ is } a \\ 1, & \text{if stimulus } i \text{ is } an \end{cases}$$

$$x_{3i} = \begin{cases} 1, & \text{if stimulus } i \text{ is lowercase} \\ 0, & \text{if stimulus } i \text{ is UPPERCASE} \end{cases}$$

$$x_{4i} = \begin{cases} 0, & \text{if stimulus } i \text{ is lowercase} \\ 1, & \text{if stimulus } i \text{ is UPPERCASE} \end{cases}$$

... but we probably don't want to do this. Suppose that there is some component, say the N1, which is identical between all four conditions. Then, one way the βs could work together to capture this effect would be to say that *a* and *an* are both associated with an N1, and put the N1's deflection into the β_1 and β_2 waveforms. Another way to explain it would be to say that the N1 is triggered by both uppercase and lowercase words, and let the β_3 and β_4 waveforms take care of it. Or maybe *a* and *an* both trigger a positivity during this window, but uppercaseness and lowercaseness both generate an even greater negativity, which cancels it out—that would also be consistent with the data. Because all these combinations of β values ultimately lead to the same predictions, least squares fitting has no way to choose among them.

Some regression software, when confronted with this situation, will respond by silently and semiarbitrarily picking one of the equivalent and equally-best combinations of β values. This can produce valid results, but only if we are careful to remember not to try to interpret the β values directly, and look only at the predicted values and their so-called valid contrasts. This approach is ubiquitous in the fMRI literature using GLM analysis, but for ERP analysis—where the β values are so directly linked to the actual target of the analysis, and where understanding the βs is the simplest method to understand exactly what assumptions the model makes about how the ERP waveform can vary across conditions—we recommend the use of nonredundant models with interpretable βs.

Fortunately, treatment coding always allows categorical variables and their interactions to be straightforwardly coded in a nonredundant way. (In fact, the regression literature usually presents this as a primary motivation for using treatment coding or related schemes.) For example, applying treatment coding to our uppercase/lowercase analysis gives a nonredundant set of predictors:

$$x_{1i} = 1$$

$$x_{2i} = \begin{cases} 0, & \text{if stimulus } i \text{ is } a \\ 1, & \text{if stimulus } i \text{ is } an \end{cases}$$

$$x_{3i} = \begin{cases} 0, & \text{if stimulus } i \text{ is lowercase} \\ 1, & \text{if stimulus } i \text{ is UPPERCASE} \end{cases}$$

$$x_{4i} = \begin{cases} 0, & \text{if stimulus } i \text{ is either lowercase, or } a \\ 1, & \text{if stimulus } i \text{ is an UPPERCASE } AN \end{cases}$$

Here x_{1i} is the intercept, x_{2i} is the treatment coded *a*/*an* factor, x_{3i} is the treatment coded lowercase/UPPERCASE factor, and x_{4i}, if included, represents the word form/letter case interaction. (Notice that $x_{4i} = x_{2i} \times x_{3i}$, which is the general rule for defining interaction predictors.) Here β_1 will estimate the ERP for lowercase *a*, β_2 the difference between lowercase *a* and lowercase *an*, β_3 the difference between lowercase *a* and uppercase *A*, and β_4 will be a difference-of-differences ERP: if we first calculated the difference ERP between uppercase *AN* and uppercase *A*, and then calculated the difference ERP between lowercase *an* and lowercase *a*, then β_4 will be the difference between these two difference ERPs. Notice that this means β_4 will be significantly different from zero exactly when there is a nonadditive interaction between our two factors.

More sophisticated coding schemes are also possible, and well documented by standard references (e.g., Cohen, Cohen, West, & Aiken, 2003). The use of nonredundant coding doesn't in any way alter or limit the set of models we can fit. In general, there are many different but equivalent ways to represent any given linear model; a simple example is shown in Figure 1b versus Figure 1c. Nonredundant codings give us the power to choose the representation we find most interpretable.

Slope rERPs from Numeric Predictors

So far, we've focused on showing the mapping between ERP terminology and regression terminology, but every analysis in Figure 1 could have been done just as well with some combination of averaging and creating difference ERPs. Now, we'll see how the regression approach also allows us to directly analyze a continuous covariate like word expectancy.

The core idea is simple. We define a predictor that indicates the expectancy of each word on some scale: here, this was measured using an offline cloze norming task, and we represent it as a number between 0 and 1, where 0 indicates a word that no norming task participants guessed, and 1 indicates a word that 100% of them guessed. In practice, we'll always include an intercept term as well; otherwise, we're assuming that whenever our predictor is zero, the ERP will be perfectly zero as well. (There may be some cases where this makes sense—perhaps if the predictor indicated visual contrast, then we'd expect no response at all to a zero-contrast display. But such cases are unlikely to occur often in practice.) To look at the expectancy effect, we can define predictors:

$$x_{1i} = 1, \qquad x_{2i} = \text{word expectancy on trial } i.$$

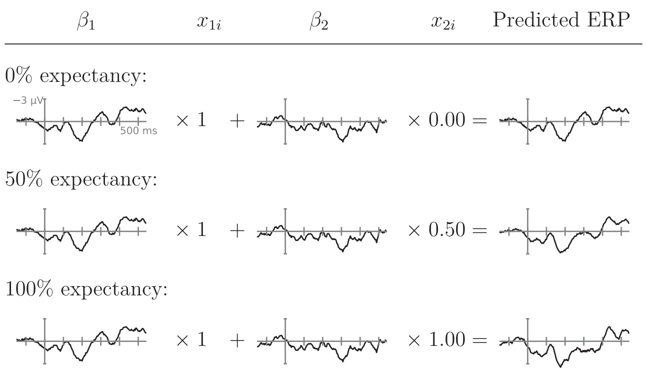

Figure 2. The relationship between xs (predictors), β coefficients (rERPs), and predicted ERPs, for various stimuli in a model containing a single continuous predictor. Waveforms shown are unsmoothed grand-average rERPs fit to data from DeLong et al. (2005).

Now β_1 will give a kind of baseline ERP—the ERP we expect to see for items with $x_{2i} = 0$ (i.e., items that were never guessed in the norming task). β_2 estimates how much this ERP changes with each unit change in expectancy; so going from expectancy 0 to expectancy 0.5 will produce a change of $0.5 \times \beta_2$ in the predicted ERP (Figure 2). β_2 is the slope of the regression line that relates x_{2i} and y_i, so we refer to this as a *slope rERP*.[6]

In Figure 2, notice the positive hump in the β_2 waveform peaking at 300 ms. This indicates an increased positivity for high-expectancy words, which is the same as a negativity to low-expectancy words; we can also see this effect reflected in the predicted ERPs. This is the N400 effect that DeLong et al. (2005) originally reported for these data.

Notice also the similarity between these predictors and the predictors we used for treatment coding above. In both cases, we have an intercept term x_{1i} whose corresponding parameter β_1 gives us a kind of baseline ERP, and then there is an additional term x_{2i} which codes for deviations from this baseline. With treatment coding, this second parameter β_2 measures the difference in scalp voltage between *a* and *an* stimuli, and has units of µV. Here, β_2 measures the difference in scalp voltage corresponding to any one unit change in expectancy, and has units of µV per unit change in expectancy. Our slope rERP β_2 can be viewed as a variant of the familiar difference ERP; what's new is that we allow for stimuli that fall between the two extremes, and which we assume create waveforms that likewise fall between the two extremes.

Putting It All Together

Now that we've seen how to handle both categorical and continuous covariates, we can also combine both in the same model. [7] For a complete analysis of the DeLong et al. (2005) design, we combine the factorial and continuous predictors into a single model:

$$x_{1i} = 1$$

$$x_{2i} = \begin{cases} 0, & \text{if stimulus } i \text{ is } a \\ 1, & \text{if stimulus } i \text{ is } an \end{cases}$$

$$x_{3i} = \text{word expectancy on trial } i.$$

(See Figure 3). The resulting βs have the same interpretation as they did in the previous examples in which these same predictors occurred—β_1 gives the predicted ERP for *a* trials with zero measured expectancy, β_2 gives the difference between *an* trials and *a* trials (this model assumes that this is the same for all expectancies), and β_3 gives the change in the predicted ERP for each unit change in expectancy.

The β waveforms together provide a form of linear decomposition of the underlying ERPs, but should not be confused with techniques like temporal principal component analysis (tPCA; Donchin & Heffley, 1978) or independent component analysis (ICA; Makeig, Bell, Jung, & Sejnowski, 1996) that also produce linear waveform decompositions; these other techniques use very different mechanisms to accomplish a very different purpose. Their goal is to identify and disentangle the underlying functional components or neural source generators that contribute to the scalp EEG. The goal of rERP analysis, by contrast, is to characterize the systematic relationships between event-related brain activity and the properties of time-locking events, and it represents this relationship using decompositions like those shown in Figures 1–3. In the tPCA/ICA literatures, the physical reality of the estimated components is a major concern; in rERP, we choose freely between coding schemes that produce different but equivalent decompositions in order to maximize interpretability. tPCA or ICA alone do not allow us to analyze differences between conditions or different kinds of time-locking events, and thus are commonly used together with ERP averaging; they can just as well be used with rERP instead. These techniques and rERP are thus complementary and compatible.

Now that we've seen how to use both categorical and continuous predictors together within a single model, it should be clear that we can extend our example as far as we like, and combine as many predictors as are appropriate to our experimental design. For example, in our example analysis we might want to also add a term to represent the interaction between the *a*/*an* manipulation and the expectancy manipulation. Such a term would be defined as $x_{4i} = x_{2i} \times x_{3i}$, and if it were included, then β_3 would denote the expectancy effect for *a* stimuli, while β_4 would denote the *difference* between the expectancy effect for *a* stimuli and the expectancy effect for *an* stimuli. Note how similar this is to our earlier discussion of categorical interactions.

In more complex cases, it might be the case that we have some events that have more or different predictors associated with them than others (e.g., in a go/no-go paradigm, go trials will have an associated response time, while no-go trials will not). This is also easily handled. There are two mostly equivalent options. We can either divide these trials into two bins and perform a separate rERP analysis on each bin, or we can reuse the zero trick that we previously used to go from Figure 1a to Figure 1b. In this second approach, the idea is to simply include both sets of predictors into a single model, with the rule that, whenever we have a trial where a predictor doesn't apply, then we set it to zero:

6. Slope rERPs correspond to the ERRCs described by Hauk et al. (2006, 2009).

7. This combined approach has been used to study the EEG response to face recognition (Rousselet et al., 2008, 2009, 2010).

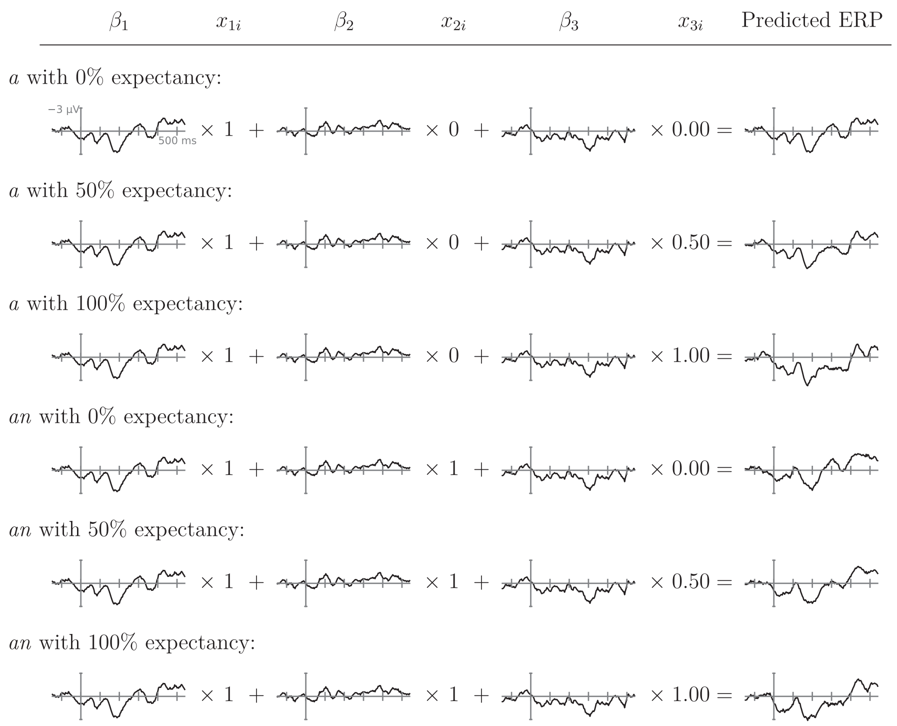

Figure 3. The relationship between xs (predictors), β coefficients (rERPs), and predicted ERPs, for various stimuli in a model containing a combination of factorial and continuous predictors. Waveforms shown are unsmoothed grand-average rERPs fit to data from DeLong et al. (2005).

$$x_{1i} = \begin{cases} 1, & \text{on no-go trials} \\ 0, & \text{on go trials} \end{cases}$$

$$x_{2i} = \begin{cases} 0, & \text{on no-go trials} \\ 1, & \text{on go trials} \end{cases}$$

$$x_{3i} = \begin{cases} 0, & \text{on no-go trials} \\ \mathrm{RT}_i, & \text{on go trials} \end{cases}$$

This ensures that that this predictor's β value will have no effect on our model's prediction for no-go trials. As Figure 1a and 1b illustrate, these two approaches in general give the same result, so one can use whichever seems more convenient. There are, however, some cases where the second approach allows for additional flexibility. For example, if there is one covariate that is shared between our different event types, and we wish to constrain the estimated waveform for this covariate to have the same shape for both types of event (i.e., we want this covariate to have only a main effect, with no interaction with event type), then we can do this by entering only a single predictor for this covariate to be shared by both event types, while simultaneously applying the zero trick to the other nonshared covariates. Another example where the second approach is useful occurs when our different event types occur in close temporal proximity, and we want to use the overlap correction technique described in the companion article to disentangle their effects; in this case, we will need to estimate all of our waveforms simultaneously within a single model, so that each waveform can be corrected for its overlap with the others, and the zero trick makes this possible.

At this point, though, the prospect of models that bristle with dozens of predictors might give us pause. First, as more terms and interactions are added to a model, it may become a challenge to interpret the various β coefficients. The key to achieving clarity in such situations is always to consider how the coefficient under consideration works together with the rest to affect the overall predicted ERP for different stimuli. We find that drawing pictures like those shown in Figures 1–3 often helps.

Second, as our models become more flexible, we may worry that we will be unable to effectively estimate our rERPs from a limited set of data—and the time and cost of recording data are already often a limiting factor on ERP experiments. We now have the tools to run and analyze arbitrary designs, from the simplest to the most complex. But should we? There are trade-offs required to use these models effectively, and the next section gives us the tools to navigate them.

Straight Talk on Collinearity

In technical terms, the fundamental challenge for analyzing data with large numbers of predictors, and those with partially confounded covariates, is collinearity. It is important therefore to understand what this is, and how, precisely, it affects our analyses; but even many textbooks dedicated to regression are somewhat vague on these points, or offer advice targeted at audiences with very different goals than typical scientific research.[8] So, here we summarize the facts and how they apply to rERPs.

First, what is collinearity? The term is commonly used to refer to two very distinct phenomena. Perfect collinearity is when we have predictors that are perfectly redundant with each other—they make exactly the same predictions in different ways—and was

8. Fox (2008) is a refreshing exception, and one that this discussion is heavily indebted to.

discussed above as a motivation for treatment coding. This is an important concept to understand, but its effects are straightforward and avoiding it is also straightforward. So our discussion here will instead focus on the more subtle problem of partial collinearity, also known as partial confounding or nonorthogonality, which refers to the situation where some of our predictors are correlated, but not identical. Sometimes the word collinearity is reserved for "severe" cases, where the degree of correlation among predictors exceeds some threshold. However, such thresholds are essentially arbitrary; our discussion here applies equally to both mild and severe cases.

The most important thing to know about partial collinearity is that it does not violate any of the assumptions underlying least squares regression. If we have a partially collinear design, then regression is an appropriate method for analyzing our data, and the effects of this collinearity can be precisely characterized by mathematical theory. This theory tells us that, when partial collinearity is involved, there are two things we need to watch out for: whether we have enough predictors, and whether we have enough data.

It's easy to understand the problem that arises from leaving out relevant predictors. Figures 2 and 3 show two estimates of the slope rERP β for expectancy: one from a model that includes the *a*/*an* predictor (Figure 3), and one from a model that leaves it out (Figure 2). For the DeLong et al. (2005) data, the two slope estimates are nearly (though not quite) identical. This is typical of (near-)orthogonal experiments like this one, which was carefully designed to ensure that the *a* stimuli and *an* stimuli were balanced with respect to expectancy. In orthogonal designs, leaving out a covariate doesn't change our estimates for the remaining covariates. But imagine instead a different, less careful version of this experiment, where the *an* stimuli had, on average, lower expectancy than the *a* stimuli. Then, the best way for the expectancy-only model to capture the patterns in the data would be for it to attribute some of the categorical *an* effect to expectancy. In statistical terminology, our analysis would become inconsistent, meaning that no matter how much data we had, we would be doomed to getting a misleading result. But the combined model is different: here, any patterns associated with the word *an* can be attributed to it directly. The wonderful thing about regression is that the least squares fitting process selects whichever β coefficients are best at working together to explain the patterns in the data, and, ultimately, the covariates that are actually responsible for a pattern are always the best way to explain it. As long as we include those covariates in our model, and have enough data, then regression will work out the correct waveforms even if our predictors are confounded.

But do we have enough data? Collinearity also affects the noise in our estimates and how much data we need, and the key to understanding this is to examine variance inflation factors, or VIFs. Each VIF measures how the sampling variance of one estimated β coefficient is affected by the presence of collinearity, and is equal to $1/(1 - R^2)$. The R^2 here is not taken from the model we actually want to fit, but instead measures how much of the variance in this predictor can be explained by the other predictors in our model. So if we have a model with just two predictors, and the correlation between these two predictors is $r = 0.5$, then the VIFs will be $1/(1 - 0.5^2) = 1.33$. This has a very straightforward interpretation: what it means is, if we know that normally we would need 60 trials to get an acceptably accurate estimate of the rERP for a single predictor in an orthogonal design, then we need $60 \times 1.33 = 80$ trials to get equally accurate estimates of the rERPs to these two predictors (O'Brien, 2007). Put another way: each trial in this design is worth three quarters of a trial in a fully orthogonal design, at least with respect to these predictors.

Examining the formula for VIFs tells us several things, and the first is quite surprising: if we add additional uncorrelated predictors to our model, then this has absolutely no effect on the noisiness of our estimates. Any predictor in an orthogonal design has a VIF of 1, which means that, so long as our covariates are uncorrelated, we can estimate a dozen parameters just as well as we can estimate one of them, from the same amount of data—though in practice we may be hard pressed to find a dozen uncorrelated predictors. Notice also that if we have two predictors that are correlated with each other but both are uncorrelated with a third, then we will need more data to estimate the effect of the first two predictors, but their presence in the model will not affect estimates for the third predictor. Collinearity is not contagious: it affects only the variables that are collinear, and only to the extent of their individual collinearity.

The second thing the VIF formula tells us is that VIFs > 1 travel in packs: correlation always goes (at least) two ways, and correlation is what causes increased VIFs, so whenever one predictor has a VIF > 1, others will as well. This is closely related to the underlying cause of variance inflation. Recall how, when we had two identical predictors, the model might know that to match the data it needed $\beta_1 + \beta_2 = 1$, but could not distinguish between an option like $\beta_1 = 0$, $\beta_2 = 1$ and one like $\beta_1 = 100$, $\beta_2 = -99$. When our predictors are not identical, but merely similar, then our model can distinguish these cases in principle—but given finite data, it may struggle; these two options will make very similar predictions, and depending on the noise, setting $\beta_1 = 100$, $\beta_2 = -99$ may seem unreasonably attractive. Technically, this struggle manifests as an increased sampling variance for all the involved βs. This is the extra sampling variance measured by the VIFs, and it manifests in a distinctive way. In orthogonal designs, the background noise corrupts our estimates for each β independently: β_1 might be erroneously high while β_2 is erroneously low, or vice versa, or both might be erroneously high or both might be erroneously low—there's no correlation between the noise in β_1 and the noise in β_2. Designs containing collinearity are different: the same noise affects multiple β values simultaneously, and can produce strange, structured behavior, where, for example, in the most common manifestation, every time β_1 is estimated erroneously high, β_2 will be estimated erroneously low, and vice versa. And, because collinearity effectively increases the amount of noise in our estimates, if we don't have enough data then we should expect to sometimes see β_1 estimated *very* high ($\beta_1 = 100$), while β_2 is estimated *very* low ($\beta_2 = -99$). This is fully expected within the theory of linear regression, and accounted for in ordinary statistical tests. If one participant's estimates have $\beta_1 = 100$, $\beta_2 = -99$, then the next will have $\beta_1 = -99$, $\beta_2 = 100$, and our statistics will correctly conclude that these estimates are too noisy to indicate anything real. But it is important to know about this behavior in order to understand what's happening when we see it.

The third thing the VIF formula tells us is that, while correlations in our design do increase the amount of data we require, it may be feasible to disentangle the effect of even highly correlated predictors. If two covariates are correlated at the level of $r = 0.7$, then we need to collect double the usual amount of data to distinguish them—difficult, but perhaps doable. In worse cases, the use of regression also gives the option of increasing N by using relatively unstructured, corpus-style data sets. These will have higher collinearity than designed experiments, but this can be more than offset by the increased amount of data. In language comprehension research, for example, modern behavioral corpus studies routinely have Ns measured in the tens of thousands (e.g., Demberg & Keller, 2008; New et al., 2006; Smith & Levy, 2013). Collinearity can still become a problem as the number of predictors increases (each one adding just a bit to the R^2 for all the others), and it is generally wise to compute the VIF while planning an experiment rather than waiting until the analysis stage, but the use of regression substantially increases the range of what is possible.

Even so, there will remain cases where the collinearity is too high, and we cannot gather enough data to achieve tight estimates of our β parameters. Would this, then, be a good time to toss out some predictors and simplify our model?

This temptation should generally be resisted, and now that we understand the theory of collinearity, we can see why. There are two cases to consider. First, if the predictors with high VIFs are included as controls in our model, but are not actually the target of our scientific questions, then collinearity is not a problem. For example, Hauk et al. (2009) wished to include bigram and trigram letter frequencies as controls in their analysis. These covariates are highly correlated with each other. To reduce this collinearity, they used PCA to produce a single new predictor that was roughly the average of the two original frequency predictors, and entered that instead as their control. We have no reason to think that this choice had any substantive effect on their results, but it was unnecessary and potentially harmful; they would have done better to skip it and include the original predictors directly. It's true that rERPs for highly collinear control factors will be very noisy and should not be trusted, but if we aren't going to interpret them anyway, then this is not problematic. As we've seen above, this untrustworthiness affects only the highly collinear predictors. If our controls are strongly correlated with each other, but only weakly correlated with the predictors that are the actual targets of our analysis, then we can leave in all our control predictors without worrying that their instability will infect our other βs. And while the βs estimated for these predictors will be noisy and uninterpretable, their presence will provide a valuable control on the βs we do care about, reducing the chance of finding inconsistent, spurious results due to confounding.

The other problematic case is when one or more of the predictors that are the target of our study themselves have VIFs that are too high to achieve usable estimates. This is the case in which it will be most tempting to remove some controls "to increase power."

But if the presence of those controls means that we don't have enough data to estimate our effect, then we don't have enough data to estimate our effect. Technically speaking, removing those controls would give us more statistical power (i.e., our VIFs would go down and our effective data set size would increase). But as we saw above, if we remove the wrong predictors, then even an infinite amount of power will only let us more precisely produce the wrong answer. In this case, we specifically lose the ability to distinguish between those controls and the covariate we care about. When we remove a control variable, what we're assuming is that we know a priori that it has no effect; but if we know that, then why did we include it in the first place? And if it does have an effect, then that power we are gaining may be the power to misattribute this effect to the covariate that we do care about. Better to compute our VIFs before we start gathering data, and if they show that we will have a problem, figure out then how to add more trials, improve our data collection, reduce the collinearity in our stimuli, or reframe our theoretical question.

That said, it may still be possible to persuade a data set to answer some particular questions, even if it cannot accurately estimate the full effect of every covariate we care about. Suppose we do want to analyze both letter bigram frequency and letter trigram frequency and get something interpretable. We have a few options. If both come out significant when entered alone, but neither is significant when entered together, then it means that at least one of these predictors matters, but more data are needed to distinguish their effects.[9] This may be enough to answer our scientific question, or at least lay the ground for further studies. Or, if we have multiple measures that we believe provide redundant estimates of the same underlying construct (e.g., if participants filled out two different handedness surveys, or if we estimated word frequency from two different corpora), and for purposes of our present analysis we do not care about the differences between them, then it might make sense to average them together, or use some more principled dimensionality reduction technique to discard theoretically uninteresting measurement noise.

Alternatively, while it's difficult to legitimately reduce noise at the analysis stage, it is possible to move the noise around by changing our coding scheme. For example, treatment coding βs give difference waves instead of simple ERPs, which may be less noisy—subtracting two waves will tend to cancel out noise, just as averaging does—but are useful for different purposes. Just as with dummy/treatment coding, these kinds of coding changes will not affect the overall model fit, or the model's predicted ERPs. Indeed, it is often simpler to compare predicted ERPs directly rather than spend time constructing elaborate new coding schemes.

One kind of recoding that in particular is hard to justify is the use of PCA for the sole purpose of orthogonalizing predictors before entering them into regression (e.g., Hauk et al., 2006; Rousselet et al., 2009). Like other kinds of recoding, this has no effect on model fit overall; the only motivation for doing it is if we want to interpret our individual βs, but are prevented by collinearity. Orthogonalization replaces our original βs with a new set that have reduced VIFs and thus are less noisy; but in the process it always makes individual βs less interpretable, by replacing them with strange and arbitrary linear combinations of our original βs. The procedure is therefore self-defeating.

The best rule of thumb seems to be, enter all the predictors that may be relevant, report the full list of predictors that were entered along with details of any transformations and coding schemes, and then test and report whichever waveforms and contrasts are of scientific interest.

β versus R^2, t, and p

In addition to the β values we have focused on so far, there are a number of other statistics that can be extracted from a regression model; most notably the coefficient of determination R^2 (and its close cousin, the correlation coefficient r), and t and p values for individual βs. Signal-to-noise ratio is another similar statistic. Within EEG/MEG (magnetoencephalogram) analysis, differences

9. Stepwise regression and other variable selection techniques will respond to an ambiguous situation like this by picking one of the predictors at random, and give no indication that they have done so. Thus, if a variable selection method picked bigram frequency over trigram frequency, this would not provide any reliable evidence that bigrams were more important than trigrams; it might have effectively been chosen by the flip of a coin. Such techniques are useful, for example, for developing medical screening models where the goal is to achieve high predictive accuracy while measuring an economical set of variables, and it doesn't matter whether the items that end up being measured are the most direct reflections of the underlying process. But it makes them inappropriate for scientific research where we are interested in whether we can be certain that some factor has an effect above and beyond any others with which it may be confounded.

in these metrics have been used to infer differences in ERP/ERF activity between scalp locations (DeLong et al., 2005; Rousselet et al., 2008, 2010), between latencies (Amsel, 2011; Ettinger et al., 2014; Hauk et al., 2006; Laszlo & Federmeier, 2014; Miozzo et al., 2014; Rousselet et al., 2008, 2009, 2010; Solomyak & Marantz, 2009, 2010), and between different participant populations (Rousselet et al., 2009, 2010). In analyses of PET/fMRI data, *t* and *p* values are often plotted instead of β coefficients. Nonetheless, we would argue that for ERP/ERF research, such comparisons should generally use β values instead.

The fundamental difference between βs and these other statistics is that the β values, which are the direct analogue to traditional ERP/ERF estimates, are designed to estimate the event-related signal itself; the other statistics give some kind of estimate of the relative strength of the signal versus the noise. This makes them useful as a measure of how confident we should be in the existence of an effect, but less appropriate for comparisons. Usually our ultimate question is not, How does our confidence in an effect's existence vary across latencies/locations/populations? but rather, How does the effect itself vary between latencies/locations/populations? We should expect these questions to have different answers, at least in some circumstances; the EEG/MEG background noise may well vary between scalp locations (e.g., due to topographic variation in muscle artifact or alpha power), latencies (due to "induced" power changes), and populations (perhaps recordings from our clinical population will be noisier). A difference in (estimated) β values reflects an (estimated) difference in the ERP/ERF itself; a difference in these other statistics might be caused by either a difference in the ERP/ERF or by a difference in task-unrelated noise. Therefore, β values seem like the best main statistic for most purposes; the other statistics are primarily useful as supplementary guides to interpreting the β values and their statistical significance.

This is particularly true when it comes to measuring the scalp topography of an effect. The physics of volume conduction mean that it is not interesting to ask which subset of electrodes some activity projects to; if it exists at all, then it projects to all of them, and the question is, How strongly? β values are linearly related to scalp potential, and thus their variation across the scalp provides a direct estimate of the forward projection constants which are the basis of source localization (Hauk et al., 2006, 2009; Miozzo et al., 2014). This means that, when using scalp maps to visualize component topography, rERP/rERF βs (or linear combinations of βs representing specific contrasts or predictions) are usually the most appropriate values to plot. It also means that rERP/rERF βs can be used directly with existing localization software; the same is not true of the other statistics discussed in this section. Indeed, because multiple-regression-based rERPs/rERFs allow for potentially finer-grained decompositions of EEG/MEG data into functionally relevant waveforms, they may provide a better basis for localization than traditional ERPs/ERFs.

Conclusion

The rERP framework extends classic ERP averaging to handle a much wider range of situations in a unified way. Compared to averaging, it preserves both the results of previous studies—all ERPs (estimated by averaging) are rERPs—and the fundamental theoretical approach of using a least squares method to reconstruct a purported waveform that varies by condition and is hidden in background noise—all rERPs are (estimates of) ERPs. By rephrasing ERP estimation as a regression problem, rERPs allow us to import a rich variety of techniques from this well developed literature, together with the mathematical tools to understand when and how they should be used.

This article has presented the core conceptual framework, explaining what rERPs are, how they relate to traditional ERP averaging, how they can be set up in practice, and how to navigate the inherent trade-offs involved in choosing a regression design. A companion article further shows how nonlinear regression and overlap correction can be handled within this framework, and discusses practical aspects of integrating rERP estimation into an analysis pipeline including baselining, filtering, significance testing, and artifact rejection.

A list of free software packages implementing these methods is available at http://vorpus.org/rERP.

References

Amsel, B. D. (2011). Tracking real-time neural activation of conceptual knowledge using single-trial event-related potentials. *Neuropsychologia*, *49*, 970–983. doi: 10.1016/j.neuropsychologia.2011.01.003

Baayen, R. H. (2004). Statistics in psycholinguistics: A critique of some current gold standards. *Mental Lexicon Working Papers*, *1*, 1–47.

Baayen, R. H. (2010). A real experiment is a factorial experiment? *Mental Lexicon*, *5*, 149–157.

Boston, M. F., Hale, J., Kliegl, R., Patil, U., & Vasishth, S. (2008). Parsing costs as predictors of reading difficulty: An evaluation using the Potsdam Sentence Corpus. *Journal of Eye Movement Research*, *2*, 1–12.

Burgess, A. P. (2012). Towards a unified understanding of event-related changes in the EEG: The firefly model of synchronization through cross-frequency phase modulation. *PLoS One*, *7*, e45630. doi: 10.1371/journal.pone.0045630

Cohen, J. (1983). The cost of dichotomization. *Applied Psychological Measurement*, *7*, 249–253.

Cohen, J., Cohen, P., West, S. G., & Aiken, L. S. (2003). *Applied multiple regression/correlation analysis for the behavioral sciences* (3rd ed.). Mahwah, NJ: Lawrence Erlbaum Associates.

Cutler, A. (1981). Making up materials is a confounded nuisance, or: Will we able to run any psycholinguistic experiments at all in 1990? *Cognition*, *10*, 65–70. doi: 10.1016/0010-0277(81)90026-3

Dambacher, M., Dimigen, O., Braun, M., Wille, K., Jacobs, A. M., & Kliegl, R. (2012). Stimulus onset asynchrony and the timeline of word recognition: Event-related potentials during sentence reading. *Neuropsychologia*, *50*, 1852–1870. doi: 10.1016/j.neuropsychologia.2012.04.011

Dambacher, M., Kliegl, R., Hofmann, M., & Jacobs, A. M. (2006). Frequency and predictability effects on event-related potentials during reading. *Brain Research*, *1084*, 89–103. doi: 10.1016/j.brainres.2006.02.010

DeLong, K. A., Urbach, T. P., & Kutas, M. (2005). Probabilistic word pre-activation during language comprehension inferred from electrical brain activity. *Nature Neuroscience*, *8*, 1117–1121.

Demberg, V., & Keller, F. (2008). Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. *Cognition*, *109*, 193–210.

Donchin, E., & Heffley, E. F. (1978). Multivariate analysis of event-related potential data: A tutorial review. In D. Otto (Ed.), *Multidisciplinary perspectives in event-related brain potential research* (pp. 555–572). Washington, DC: U.S. Government Printing Office.

Ettinger, A., Linzen, T., & Marantz, A. (2014). The role of morphology in phoneme prediction: Evidence from MEG. *Brain and Language*, *129*, 14–23. doi: 10.1016/j.bandl.2013.11.004

Fox, J. (2008). *Applied regression analysis and generalized linear models* (2nd ed.). Thousand Oaks, California: SAGE.

Frank, S. L., Otten, L. J., Galli, G., & Vigliocco, G. (2013). Word surprisal predicts N400 amplitude during reading. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (pp. 878–883). Sofia, Bulgaria.

Gratton, G., & Fabiani, M. (1998). Dynamic brain imaging: Event-related optical signal (EROS) measures of the time course and localization of cognitive-related activity. *Psychonomic Bulletin & Review*, *5*, 535–563. doi: 10.3758/BF03208834

Groppe, D. M., Choi, M., Huang, T., Schilz, J., Topkins, B., Urbach, T. P., & Kutas, M. (2010). The phonemic restoration effect reveals pre-N400 effect of supportive sentence context in speech perception. *Brain Research*, *1361*, 54–66. doi: 10.1016/j.brainres.2010.09.003

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

Hauk, O., Pulvermüller, F., Ford, M., Marslen-Wilson, W. D., & Davis, M. H. (2009). Can I have a quick word? Early electrophysiological manifestations of psycholinguistic processes revealed by event-related regression analysis of the EEG. *Biological Psychology*, *80*, 64–74. doi: 10.1016/j.biopsycho.2008.04.015

Hendrix, P. (2009). Electrophysiological effects in language production: A picture naming study using generalized additive modeling (Unpublished Master's thesis). Radboud University, Nijmegen, The Netherlands.

Hendrix, P., Bolger, P., & Baayen, H. (2014). Distinct ERP signatures of word frequency, phrase frequency, and prototypicality in speech production. Manuscript submitted for publication.

Jung, T.-P., Makeig, S., Westerfield, M., Townsend, J., Courchesne, E., & Sejnowski, T. J. (2001). Analysis and visualization of single-trial event-related potentials. *Human Brain Mapping*, *14*, 166–185. doi: 10.1002/hbm.1050

Kliegl, R., Nuthmann, A., & Engbert, R. (2006). Tracking the mind during reading: The influence of past, present, and future words on fixation durations. *Journal of Experimental Psychology: General*, *135*, 12–35.

Kryuchkova, T., Tucker, B. V., Wurm, L., & Baayen, R. H. (2012). Danger and usefulness in auditory lexical processing: Evidence from electroencephalography. *Brain and Language*, *122*, 81–91.

Laszlo, S., & Federmeier, K. D. (2011). The N400 as a snapshot of interactive processing: Evidence from regression analyses of orthographic neighbor and lexical associate effects. *Psychophysiology*, *48*, 176–186. doi: 10.1111/j.1469-8986.2010.01058.x

Laszlo, S., & Federmeier, K. D. (2014). Never seem to find the time: Evaluating the physiological time course of visual word recognition with regression analysis of single item event-related potentials. *Language, Cognition and Neuroscience*, *29*, 642–661. doi: 10.1080/01690965.2013.866259

Lorig, T. S., & Urbach, T. P. (1995). Event-related potential analysis using Mathematica. *Behavior Research Methods, Instruments, & Computers*, *27*, 358–366. doi: 10.3758/BF03200430

Makeig, S. (1993). Auditory event-related dynamics of the EEG spectrum and effects of exposure to tones. *Electroencephalography and Clinical Neurophysiology*, *86*, 283–293.

Makeig, S., Bell, A. J., Jung, T.-P., & Sejnowski, T. J. (1996). Independent component analysis of electroencephalographic data. In *Advances in neural information processing systems 8*. Cambridge, MA: MIT Press.

Miozzo, M., Pulvermüller, F., & Hauk, O. (2014). Early parallel activation of semantics and phonology in picture naming: Evidence from a multiple linear regression MEG study. *Cerebral Cortex*. doi: 10.1093/cercor/bhu137 Retrieved from http://cercor.oxfordjournals.org/content/early/2014/07/08/cercor.bhu137

New, B., Ferrand, L., Pallier, C., & Brysbaert, M. (2006). Reexamining the word length effect in visual word recognition: New evidence from the English lexicon project. *Psychonomic Bulletin & Review*, *13*, 45–52. doi: 10.3758/BF03193811

Nikulin, V. V., Linkenkaer-Hansen, K., Nolte, G., Lemm, S., Müller, K. R., Ilmoniemi, R. J., & Curio, G. (2007). A novel mechanism for evoked responses in the human brain. *European Journal of Neuroscience*, *25*, 3146–3154. doi: 10.1111/j.1460-9568.2007.05553.x

O'Brien, R. M. (2007). A caution regarding rules of thumb for variance inflation factors. *Quality & Quantity*, *41*, 673–690. doi: 10.1007/s11135-006-9018-6

Pernet, C. R., Chauveau, N., Gaspar, C., & Rousselet, G. A. (2011). LIMO EEG: A toolbox for hierarchical LInear MOdeling of ElectroEncephaloGraphic data. *Computational Intelligence and Neuroscience, 2011*. doi: 10.1155/2011/831409

Rousselet, G. A., Gaspar, C. M., Pernet, C. R., Husk, J. S., Bennett, P. J., & Sekuler, A. B. (2010). Healthy aging delays scalp EEG sensitivity to noise in a face discrimination task. *Frontiers in Psychology, 1*, article 19. doi: 10.3389/fpsyg.2010.00019

Rousselet, G. A., Husk, J. S., Pernet, C. R., Gaspar, C. M., Bennett, P. J., & Sekuler, A. B. (2009). Age-related delay in information accrual for faces: Evidence from a parametric, single-trial EEG approach. *BMC Neuroscience*, *10*, 114. doi: 10.1186/1471-2202-10-114

Rousselet, G. A., Pernet, C. R., Bennett, P. J., & Sekuler, A. B. (2008). Parametric study of EEG sensitivity to phase noise during face processing. *BMC Neuroscience*, *9*, 98. doi: 10.1186/1471-2202-9-98

Sauseng, P., Klimesch, W., Gruber, W., Hanslmayr, S., Freunberger, R., & Doppelmayr, M. (2007). Are event-related potential components generated by phase resetting of brain oscillations? A critical discussion. *Neuroscience*, *146*, 1435–1444. doi: 10.1016/j.neuroscience.2007.03.014

Smith, N. J., & Levy, R. (2013). The effect of word predictability on reading time is logarithmic. *Cognition*, *128*, 302–319.

Solomyak, O., & Marantz, A. (2009). Lexical access in early stages of visual word processing: A single-trial correlational MEG study of heteronym recognition. *Brain and Language*, *108*, 191–196. doi: 10.1016/j.bandl.2008.09.004

Solomyak, O., & Marantz, A. (2010). Evidence for early morphological decomposition in visual word recognition. *Journal of Cognitive Neuroscience*, *22*, 2042–2057.

Tremblay, A. (2009). Processing advantages of lexical bundles: Evidence from self-paced reading, word and sentence recall, and free recall with event-related brain potential recordings (Unpublished doctoral dissertation). University of Alberta.

Tremblay, A., & Baayen, R. H. (2010). Holistic processing of regular four-word sequences: A behavioral and ERP study of the effects of structure, frequency, and probability on immediate free recall. In D. Wood (Ed.), *Perspectives on formulaic language: Acquisition and communication* (pp. 151–173). London, UK: Continuum.

Woldorff, M. G. (1993). Distortion of ERP averages due to overlap from temporally adjacent ERPs: analysis and correction. *Psychophysiology*, *30*, 98–119. doi: 10.1111/j.1469-8986.1993.tb03209.x

(Received November 19, 2013; Accepted July 3, 2014)