To the Graduate Council:

I am submitting herewith a dissertation written by Jon Alan Frederick entitled "EEG Coherence and Amplitude Effects of Rhythmic Auditory and Visual Stimulation with an Emphasis in Computational Methods." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Psychology.

Dr. Joel F. Lubar
Major Professor

We have read this dissertation
and recommend its acceptance:

Dr. Debora R. Baldwin

Dr. John W. Lounsbury

Dr. Teresa A. Hutchens

Accepted for the Council:

Dr. Anne Mayhew
Vice Provost and
Dean of Graduate studies

(Original signatures are on file in the Graduate Student Services Office.)

EEG Coherence and Amplitude Effects of Rhythmic Auditory and Visual Stimulation

With an Emphasis in Computational Methods

A Dissertation
Presented for the
Doctor of Philosophy Degree
The University of Tennessee, Knoxville

Jon Alan Frederick
December, 2001

Dedication

This work is dedicated in loving memory to my grandmother, Elizabeth Clara Beaver, whose character, determination, and intelligence inspired the higher education of her children.

Acknowledgments

I am grateful to Dr. Joel Lubar, for providing the freedom, guidance, and the ideal research environment for this work to flourish; to my dissertation committee, Dr. Debora Baldwin, Dr. John Lounsbury, and Dr. Teresa Hutchens, for their advice and encouragement; to my Mom, Betty Windy Boy, M.A., for her excellent example as a scholar and citizen, and for her tireless support and confidence in my achievement; I am especially grateful to Dr. Deanna L. Timmermann for her generous collaboration and support— I am indebted for her years of hard work blazing the trail into this complicated paradigm for our lab; to Efthymios Angelakis, for setting a high standard for industry and persistence in the lab, and for our many collaborations; to Jared Blackburn, for his collaboration and assistance; to Harley Helterhoff, for his excellent consultation as a programmer; to Bob Muenchen, for his excellent statistical advice; to Cindy Ellison, for her assistance organizing the literature cited; to Dr. Rick Doblin, for his wise counsel and support; to Larry Jennings and the NT/Unix Group at UT, for the use of their outstanding facilities and expertise; to Dr. Harold Russell and Synetic Systems for the helpful advice and the PolySync Pro devices used in this research; and to David Joffe and Lexicor corporation for their excellent products, and technical and financial support for our lab.

Abstract

The photic driving response, the effect of a flashing light stimulus on the cortical EEG, has proven to be a sensistive neurometric that varies with differences in perception, mood, and physiological states. The diverse effects of photic stimulation have made commercially available "brainwave syncronizers" popular among consumers and even among some clinicians. It is common in the design of these devices to combine a rhythmic auditory stimulus with the visual stimulus. However, little if any experimental evidence supports the assumption that auditory stimulation enhances the photic driving effect. Therefore, this study compared the amplitude and coherence effects of three stimulation conditions (all sinusoidally-modulated): visual stimulation alone, auditory stimulation alone, and combined auditory and visual stimulation (AVS) at each subject's peak alpha frequency (PAF), in 30 college students, using the standard 10-20 19-electrode montage. An eyes-closed baseline EEG determined each subject's PAF. The three 5-minute stimulation conditions were then administered in a randomized, counterbalanced order, while EEG was recorded. A four minute eyes-closed post-stimulation baseline was recorded after each stimulation condition. Amplitude and coherence values were calculated for 1-2 Hz, 2-4 Hz, 4-8 Hz, 8-12 Hz, 13-21 Hz, and 21-32 Hz, and for the 1.5-Hz band surrounding each subject's PAF. Visual stimulation or combined AVS significantly increased EEG amplitude at the PAF, 8-12 Hz, 13-21 Hz, and 21-32 Hz. Auditory stimulation alone had negligible effects on amplitude, and did not result in significant differences in amplitude between the visual and combined AVS condition. Similarly, the auditory stimulus had only chance-level effects on EEG coherence, while the visual alone and combined AVS conditions evoked significant changes in coherence across the spectrum. However, the interaction between auditory and visual stimulation resulted in significant differences between the visual alone and combined AVS conditions. No residual effects of stimulation on amplitude or coherence were observed in the four minute recordings following each of the three stimulation sessions. Two tables and eight figures are included. The Perl source code for 21 data analysis programs is provided, along with a tutorial chapter explaining how to use these algorithms in one's own research.

Preface

To improve the readability of this dissertation, code examples and suggested command line syntax are written in `Courier` font, and are single-spaced, while the body of this text is written in Times New Roman, and double-spaced. Windows menu and dialog box selections are written in ALL CAPS with colons (':') delimiting subsequent selections. Major chapter subheadings are denoted in **bold.** *Italic* is used for minor subheadings, the names of custom software applications, and to emphasize the introduction of new words or concepts.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter One

Introduction: Physiological basis and clinical applications of the photic driving response

The ability of a flashing light stimulus to evoke EEG rhythms related to the stimulus frequency, has been studied since the early history of electroencephalography (Adrian and Matthews, 1934). Known as the *photic driving response* (PDR), or *steady state visual evoked potential,* this effect is commonly measured in routine clinical EEG examinations, and has been proven useful for investigating neurological disorders (Takahashi, 1987; Coull and Pedley, 1978; Duffy, Iyer, and Surwillo, 1989). Abnormal PDRs have also been observed in patients with Alzheimer disease (Wada, Nanbu, Kikuchi, Koshino, Hashimoto, and Yamaguchi, 1998a; Politoff, Monson, Hass, and Stadter, 1992), schizophrenia (Wada, Nanbu, Kikuchi, Koshino, and Hashimoto, 1998b; Wada, Takizawa, and Yamaguchi, 1985), depression (Jin, Potkin, Sandman and Bunney, 1997) and chemical addictions (Bauer, 1994). Changes from normal PDRs have been reported with the administration of MAO inhibitors (Vogel, Broverman, Klaiber, and Kobayashi, 1974), caffeine (Pollock, Teasdale, Stern and Volavka, 1981), nicotine (Thompson, Tzambazis, Stough, Nagata, and Silberstein, 2000), and antipsychotic medications (Jin, Potkin and Sandman, 1996; Jin, Potkin, Sandman and Bunney, 1999).

The diverse perceptual and emotional effects of photic stimulation (Walter and Walter, 1949; Stwertka, 1993; Gizycki, Jean-Louis, Snyder, Zizi, Green, Giuliano, Spielman, and Taub, 1998), and it's ability to cause seizures in susceptible individuals

(Walter, Dovey and Shipton, 1946; Striano, Meo, Bilo, Ruosi, Soricellis, Estraneo, and Caporella, 1992) have led many to investigate whether rhythmic auditory and visual stimulation (AVS) might also induce clinically beneficial changes in brain activity. In the 1950's and 60's, many studies focused on the ability of AVS to induce relaxation and hypnosis (reviewed in Morse, 1993). Others have reported AVS to be effective for relieving a diversity of pain symptoms (Solomon, 1985; Anderson, 1989; Shealy, Cady, Cox, Liss, Clossen, and Culver, 1990), treating dental anxiety (Morse, 1993), premenstrual syndrome (Noton, 1997), fibromyalgia (Mueller, Donaldson, Nelson & Layman, 2001) and for alleviating the cognitive dysfunctions associated with closed head injury (Montgomery, Ashley, Burns, and Russell, 1994) and strokes (Russell, 1997; Rozelle and Budzinski, 1995). Since the enhancement of beta (13-21 Hz) and inhibition of theta (4-8 Hz) is a goal of EEG biofeedback for the treatment of attention deficit hyperactivity disorder (ADHD; Lubar and Lubar, 1999; Lubar, Swartwood, Swartwood, and O'Donnell, 1995), some have proposed using AVS in neurofeedback as a "priming stimulus" to encourage the endogenous production of desired cortical frequencies, which are then reinforced as the conditioned response. In a study of 25 ADHD children, Patrick (1996) found "photic-driven EEG neurotherapy" effective in improving cognitive, behavioral, and clinical EEG measures in less than half the number of sessions usually required. Meanwhile, Micheletti (1999) found AVS alone effective in improving cognitive and behavioral measures, in a study of 99 ADHD children. Carter and Russell (1993) reported significant improvement in cognitive and behavioral functioning, related to the number of AVS sessions, in learning disabled boys. Joyce and Siever (2000) reported that a 7-week audiovisual stimulation treatment in 8 reading-disabled children, compared to a control group, normalized scores on the Test of Variables of Attention (TOVA), improved scores on the Standardized Test for the Assessment of Reading (STAR), and improved general behavior as noted by teachers and parents.

The most commonly studied PDRs have been the effects of stimulation on alpha (8-13 cycles/second or Hertz, Hz) power over the occipital region. Alpha is the prominent EEG wave form for normal, relaxed adults with eyes closed. Alpha can be substantially reduced in amplitude by eye opening, drowsiness, and moderate to difficult mental tasks. Although alpha is defined as the posterior dominant rhythm, alpha-like rhythms can be recorded from nearly the entire cortical surface, except for regions close to the central motor strip where beta activity (>13) predominates (Nunez, Wingeier, and Silberstein, 2001). Each subject has a dominant or peak alpha frequency (PAF), defined as the frequency of highest amplitude or power in the 8-13 Hz range. The alpha peak usually accompanied by a general side band activation within approximately ± 1 Hz. Lower PAF is often associated with pathology. Higher PAF is associated with higher memory performance (Klimesch, 1997), reading ability (Suldo, 2000), vocabulary, and response control (Angelakis and Lubar, 2001). A common protocol for "EEG-driven phototherapy" involves modifying the PAF by real-time feedback of photic stimulation at a frequency slightly higher or lower than the PAF (Mueller et al., 2001; Rozelle and Budzinski, 1995; Ochs, 1994, 1996).

The photic driving response is most reliable when the stimulus approximates the subject's PAF (Toman, 1941; Townsend, Lubin, and Naitoh, 1975). There is, however, considerable variability in this response between subjects, although repeated testing of individual subjects provides consistent results (Pigeau and Frame 1992; Papakostopolous and Gogolitsyn 1997; Garoutte, Aird, and Correnti, 1958). Significant differences in the response are also seen when using square-wave (“flash,” stroboscobic, or intermittent pulse) light as opposed to sine-wave or sinusoidally modulated light (SML; Townsend et al. 1975). Walter and Walter (1949) used stroboscopic light and found that high alpha subjects, those who already have a strong alpha rhythm in the eyes-closed resting baseline, show little or no photic driving response. However, Pigeau and Frame (1992) observed that the response to SML for high alpha subjects was significantly higher when stimulated at the PAF than at other frequencies, while the response in low alpha subjects did not vary with stimulus frequency. Kawaguchi, Jijiwa, and Watanabe (1993) found that half of their subjects did not did not produce a PDR to stroboscopic stimulation in the alpha bandpass. Rosenfeld, Reinhart, and Srivastava (1997) observed that subjects with a high alpha baseline showed inconsistent effects to 10 Hz square-wave photic stimulus, while low-baseline participants showed more reliable effects. At stimulus frequencies above 18 Hz, however, Van Der Tweel and Verduyn Lunel (1965) found much greater homogeneity between subjects to SML. Interestingly, a 30 Hz stimulus exceeds the temporal resolution of most subjects and is experienced as “fused,” but the rhythmic response can still be detected in the EEG (Regan, 1966). In fact, the modulated light from a 60 Hz refresh rate computer monitor (Lyskov, Ponomarev, Sandstrom, Mild, and Medvedev, 1998) evokes photic driving effects, even though this frequency exceeds the subjective flicker fusion frequency.

An MRI study localized changes in blood flow associated with photic stimulation to the medial-posterior regions of the occipital lobes along the calcarine fissures (Belliveau, Kennedy, McKinstry, Buchbinder, Weisskoff, Cohen, Vevea, Brady, and Rosen, 1991).

Toman (1941) observed that the visual evoked potential resulted from the onset of the light flash, whereas there was no corresponding “offset” effect. Toman followed the stimulus paradigm of Adrian and Matthews (1934), interrupting a beam of light with a motor-driven disc, with sectors cut out from the disc. The optimum period of illumination was found to lie between 15 and 90 degrees, although a stimulus interval of 300 degrees was still effective. Illumination of the entire visual field is important for the effect. A “marked reduction” of the response was observed when either eye, or the lateral or medial half of the visual field, or the macular or peripheral portion of the field, was occluded. Toman speculated that the cortex was a heterogeneous population of elements that were tuned to specific frequencies. Thus, the “entrainment” response is actually a change in the type of neurons responding, rather than a frequency change in individual neurons. He explained the preferential response at the dominant alpha rhythm as a numerical dominance of neurons tuned to that frequency. For isolated flashes of light, or for flash frequencies slower than the response to a single flash, some neurons of all types might respond. Higher frequencies would result in impulses arriving during the refractory period of slower neurons, which would then be removed from the response. Jansen and Brandt (1991) later demonstrated that the amplitude of the visual evoked potential response depended critically on the phase of the ongoing alpha rhythm. The response was maximal for stimuli delivered during a positive-transitioning zero cross (PZC, when the

EEG proceeds from negative to positive voltage across the baseline), and minimal during a negative-transitioning zero cross (NZC). The intrinsic phase of the alpha rhythm can be reset or "entrained" by visual flash stimuli (Brandt, 1997) during the PZC but not during the NZC. Toman had noted that the response begins at small amplitude and attains a maximum amplitude in from 0.3 to 2 sec (after 3 to 20 flashes at 10 Hz). Regan (1966) reported that 12-20 seconds of sinusoidally-modulated light stimulation was required before a steady-state with respect to phase and amplitude is achieved in the response. Thus, neurons undergoing PZC at the stimulus onset, whose natural period of excitability matches the flash frequency (or some multiple or integral fraction of the flash frequency), are the first to be recruited. Possibly, neurons in NZC during the stimulus onset are recruited when their phase spontaneously drifts into a PZC during a subsequent pulse (since the phase of the rhythmic stimulus is stationary and the phase of endogenous rhythms are not).

Walter and Walter (1949) remarked that while it is relatively difficult to observe or control the factors affecting the resting or spontaneous EEG, rhythmic photic stimulation provides a method to evoke specific EEG effects that correlate with changes in perception, mood, and clinical pathological states. In addition to the simple sensation of flicker, Walter and Walter reported, "subjects had experiences in one or more of the following categories: (1) visual sensations with characters not present in the stimulus, that is (a) colour, (b) pattern, or (c) motion; (2) kinesthetic (swaying, spinning, jumping, vertigo) and cutaneous (tingling, prickling) sensations; (3) emotional and abstract experiences: (a) fatigue, (b) confusion, (c) fear, (d) disgust, (e) anger, (f) pleasure, or (g) disturbance of the time sense; (4) organized hallucinations of various types; or (5) clinical psychopathic states and epileptic seizures." In fact, the remarkable, complex visual patterns that can be induced by a rhythmically flashing, uniform field of illumination were reported as early as 1823 by Purkinje (cited in Stwertka, 1993). Walter and Walter believed that the physiological basis of these effects was a "jamming" of the brain's mechanism of interpreting normal sensory signals. That is, the visual system conveys information regarding color, position, shape, and movement by frequency modulation of impulses in the optic nerves and spatial projection on the cortex. Specifically, peripheral receptors change the frequency of action potentials to the cortex in response to spatial and temporal *differences* in stimulus intensity, rather than to absolute levels. Thus, we can read the same page indoors or outdoors, even though the text is actually brighter outdoors than the page is indoors. As Hubel and Weisel (1965; 1968; 1977) later demonstrated, a spatial contrast such as an edge will cause frequency changes in specific areas of visual cortex, while temporal changes in brightness would be represented by frequency modulation in another cortical column of cells. Walter and Walter suggested that the alpha rhythms function as another form of frequency modulation that scans and encodes the anatomical pattern of excitation on the visual cortex into a temporal pattern that is transmitted to other regions of the brain. This system works well with steady or slowly changing illumination, but if a rhythmic stimulus imitates the frequency of impulses in the optic nerves or the endogenous rhythms, the temporal properties of the stimulus signal will be erroneously interpreted by association areas as anatomical properties. Then, if the stimulus signal or one of it's harmonics synchronizes with some other rhythmic mechanism in the association area, the remote mechanism may become jammed just as the visual one is. In the normal brain, something constrains these evoked rhythms to a few circuits, with for the most part only subjective effects. In epileptic patients, this entrainment can spread until the entire brain is resonating in harmonically related modes and a seizure is induced.

Several groups have noted that the response to rhythmic photic stimulation is an interaction between the photic driving response and the general attenuating effect of light stimuli on the spontaneous alpha rhythms. Iwahara, Noguchi, Yang, and Oishi (1974) provided intermittent photic stimulation (IPS; rhythmic flash or square-wave stimulation) to 11 participants at 7.5, 9.2, 11.4, 15, and 22.5 Hz, in random order while recording monopolar referential EEG from a needle electrode at Oz, with a right earlobe reference. Power was analyzed as “integrated millimeters in pen deflection” at 4-6 Hz (theta 1), 6-8 Hz (theta 2), 8-9 Hz (alpha 1), 9-11 Hz (alpha 2), 11-13 Hz (alpha 3), 13-20 Hz (beta 1) and 20-30 Hz (beta 2). Stimulation frequencies outside of each subject’s dominant alpha band had an inhibitory effect on the dominant alpha rhythm that was otherwise independent of frequency. While this article is complicated and obscured by a weak English translation from the Japanese, these authors confidently conclude that the alpha photic driving response is, to a first order approximation, a linear superposition of the specific frequency effect of IPS and it’s nonspecific attenuating effect on the alpha rhythm, and that these two effects are independent. Martini, Venturini, Zapponi, and Loizzo (1979) also found evidence of these two conflicting effects. They delivered IPS to 10 participants at 6, 8, 9, 10, 11, 12, 14, and 16 Hz, in random order, for 35 seconds each.

When individual 1 Hz bands at the stimulus frequency were compared to the corresponding frequency in the baseline, significant power increases were observed at the 6, 8, 9, 12, 14, and 16 Hz bands. A trend toward an increase was observed at 10 and 11 Hz which was not interpreted as significant. A more modern interpretation would be that the significant increases in all the other bands support the likelihood these 10 and 11 Hz effects were real, but it is nonetheless an important observation that they were of lower magnitude and consistency across subjects. In fact, 10, 11, and 12 Hz stimuli had decreased power when a wider 8-12.5 Hz alpha band was evaluated, suggesting that the inhibitory effect across the band as a whole exceeded the driving effect at each specific frequency.

The use of sinusoidally-modulated light (SML) is known to produce a more narrow entrainment response at the frequency of stimulation. Square-wave stimuli (IPS) produce harmonics, subharmonics (Walter and Walter, 1949) and greater side-band activation adjacent the stimulus frequency (Van der Tweel and Verduyn Lunel, 1965). Since Fourier analysis assumes that the raw EEG is composed of an infinite number of sine waves, a square-wave response in the EEG will also produce harmonics at odd integral multiples of fundamental frequency, whereas a perfect sign wave evaluates to only one frequency component. Townsend et al. (1975) noted how flash stimuli clearly produce complex evoked responses in the raw EEG (at O2) and bear little resemblance to the spontaneous EEG waveform, whereas responses to SML were sinusoidal. Townsend et al. also observed that SML had much weaker effects that flash stimuli outside of the range of the subject's dominant alpha rhythm. (This contrasts, however, with Van der Tweel and

Verduyn Lunel's observation that sine wave stimuli as high as 18 Hz tended to be "faithfully reflected" in the response at the central and occipital midline. (Timmermann, Lubar, Rasey, and Frederick (1999) also observed a significant increase in 12-21 Hz and 21-31 Hz power in the averages of 13 subjects who had received SML and sound at twice the dominant alpha frequency). Interestingly, in the Townsend study, whereas SML had weaker effects than flash stimuli outside of the alpha band, it had a greater effect than flash on the dominant alpha rhythm. Townsend et al. also found an important difference between the spontaneous alpha rhythm and the SML-response at the peak alpha frequency. Successive waves of spontaneous alpha decay in amplitude, which is believed to result from fluctuations in the frequency of the spontaneous alpha. There was no such decay in amplitude during SML, it was hypothesized that SML stabilizes the frequency of the spontaneous alpha. Observing this difference in an initial group of subjects motivated Townsend et al. to test the frequency-stabilization hypothesis in a second group of subjects. Five minute eyes-closed baselines at O1 (monopolar linked ears reference) were collected from 18 male subjects, 19-23 years old, and six of these were selected for percent time spontaneous alpha of 20-50%. A second recording from these six subjects was prepared using flash and SML stimuli at each subject's peak alpha frequency. Fast Fourier Transforms then analyzed bands ± 2 Hz from each subject's peak alpha with 1/8 sec resolution (a total of 32 1/8 Hz-wide bands). The average standard deviation in amplitude at the peak alpha frequency between subjects was then compared among the baseline, IPS, and SML conditions. Both t-tests and Wilcoxon signed-rank tests showed a significantly lower standard deviation in the peak alpha frequency under SML, compared to both IPS and baseline (the means were baseline, 0.232 ± 0.137; flash, 0.207 ± 0.149; and SML, 0.045 ± 0.053; $p<.025$ with only 6 subjects). Townsend et al. found several reasons to believe that SML stabilizes the EEG by phase-locking and resonating the spontaneous alpha. First, relatively little amplitude effects were observed from SML stimuli greater than 2 Hz outside of the DA. Second, flash stimuli failed to stabilize the PAF, so frequency stabililization is probably not a result of imposing an frequency-stable evoked response on the spontaneous activity. Finally, the average amplitude of the evoked response abruptly increased to the amplitude of the spontaneous PAF when the SML frequency approached the spontaneous alpha frequency.

Pigeau and Frame (1992) observed similar resonance phenomena at the PAF in high alpha subjects, but not in low alpha subjects. Pigeau and Frame determined the baseline PAF for 16 participants (9 female), and provided 13 SML sessions, each lasting 64 sec, in random order, where the frequency ranged from ± 2 Hz surrounding the PAF in 1/3 Hz intervals. Participants were split into "high alpha" and "low alpha" subjects by rank ordering their baseline 8-13 Hz power at 02 and splitting at the median. (A criticism of this procedure would be that absolute power is influenced by irrelevant factors such as the thickness of the skull; a rank ordering based on relative power would have been more appropriate.) Low alpha subjects were found to have a PAF at a significantly higher frequency than high alpha subjects— 10.3 vs. 9.5 Hz (t-test $p < 0.023$), and a significantly higher variability in the average amplitude of the PAF ($p<.002$). A single FFT was calculated (at O1) from the entire 64 sec of each of the 13 stimulation conditions, allowing for a frequency resolution of 1/64 Hz (there is no discussion of artifact rejection methods or the consequences of artifacts, although the clarity and parsimony of the results provides a strong intuitive support that type I error is not among the consequences). A 4 Hz band centered around each participant's PAF was extracted, and each 1/64 Hz bin, starting from 2 Hz below PAF, was averaged across participants for each SML condition. When amplitude, EEG frequency about the PAF (-2 to 2 Hz), and SML frequency (-2 to 2 Hz) were 3-dimensionally plotted, separately for high and low alpha subjects, the PDR was clearly observable at each 1/3 Hz stimulation interval. However, the high alpha subjects showed a maximal PDR at the PAF, whereas the PDR at the PAF was about average for the low alpha subjects. The amplitude of the PDR decreased with distance from the PAF whereas the frequency response curve for the low alpha subjects was almost flat, with a slight decrease at the highest frequencies. A repeated measures ANOVA, using the Huynh-Feldt epsilon correction for heterogeneity of variance, showed that the response at ± 0.66 Hz around PAF was significant ($p<0.037$) for the high alpha subjects, but a similar trend was not significant ($p<.072$) for the low alpha subjects. To assess whether SML had side-band effects, the power of the PDR at each stimulus frequency was subtracted from the total alpha power from 6-15 Hz during the same stimulus condition (extended from the traditional 8-13 Hz in order to ensure complete coverage of the evoked response). A repeated measures ANOVA showed no significant effect of stimulus frequency on side band activity for the high alpha subjects ($p<0.434$). However, a significant effect of frequency on side band activity was observed for low alpha subjects, with a peak that extended from -.33 Hz to 1.66 Hz about the PAF. Pigeau and Frame explain these differences between high and low alpha subjects in terms of the relationship between (1) alpha generators in the lateral geniculate nucleus (LGN) of the thalamus, which projects to the occipital cortex via the geniculo-calcarine radiation, and (2) cortical neuronal populations which filter stochastic inputs selectively for alpha frequencies. They believed that high alpha subjects have a more strongly coupled thalamic and cortical loop oscillating at each subject's spontaneous PAF. This assumption would explain how high alpha subjects have both a significantly lower variability in the baseline PAF amplitude, as well as a maximal response at the PAF that was lacking in lower alpha subjects. However, the significant variation in side band activity with respect to stimulus frequency in low alpha subjects, shows that the PAF, while weak and unstable in these subjects, nonetheless interacts with SML stimulation. Unlike in high alpha subjects, however, this interaction takes place in adjacent frequencies rather than the frequency of stimulation. Thus, high alpha subjects tend to display resonance in a specific thalamic pacemaker, whereas low alpha subjects show the effects of more diffusely tuned cortical filters.

The combining of a rhythmic auditory stimulus with the light stimulus is common in the design of commercial "brain wave synchronizers" (Morse, 1993). However, there is little if any experimental evidence to support the assumption that auditory stimulation enhances the photic driving effect. To examine the interaction between rhythmic light and sound, we compared the effects of auditory stimulation alone, visual stimulation alone, and combined AVS at 18.5 Hz on EEG amplitude (at the vertex, Cz) in 15 college students (Frederick, Lubar, Rasey, Brim, and Blackburn, 1999). EEG amplitude was evaluated at 4-8 Hz, 13-21 Hz, in a narrower 16-20 Hz band, and at individual half-Hz bands between 16.0 and 20.0 Hz. We found that the combined AVS had no enhancing effect over visual or auditory stimulation alone. In fact, repeated measures ANOVAs found no significant effect of stimulation type at any frequency. However, all three stimulation conditions evoked a significant amplitude increase at 18.5 Hz. While direct comparisons between the stimulation conditions showed no differences, it is worth noting that, if anything, combining auditory with visual stimulation showed a trend toward a diminished response. The 16-20 Hz band increased significantly at $p<.001$ for the auditory stimulus alone, at $p<.011$ for the visual stimulus alone, but marginally failed to reach significance ($p<.051$) for the combined AVS condition. In the half-Hz band at 18.5 Hz, the effect was significant at $p<.003$ for auditory stimulation alone, $p<.007$ for visual stimulation alone, but significant at only $p<.017$ for the combined AVS condition. The effect was significant (at least $p<.05$) for all seven half-Hz bands between 16.0-20.0 Hz in the auditory alone condition, for six of the seven in the visual alone condition, and only for three of the seven in the combined AVS condition. Although only a trend, this finding suggests that combined AVS might have an inhibitory effect on EEG entrainment compared to photic or auditory stimulation alone.

Until recently, a limitation to most EEG studies using photic or audiovisual stimulation has been the examination of a restrictive number of scalp locations and/or frequencies. In fact, AVS affects the EEG beyond the primary sensory cortices and outside of the frequency of stimulation. Using low-frequency theta AVS, Dieter and Weinstein (1995) described significant reduction in “mean activity” (an increase of delta and theta activity) in frontal, central, and parietal regions, in addition to occipital regions.

Brauchli, Michel, and Zeier (1995) found that both variable and fixed frequency AVS was associated with a reduction in global alpha field power averaged from 17 electrodes distributed across the scalp (FP1 and FP2 not included). However, no systematic study had been conducted of AVS on all 19 locations of the 10/20 International EEG locations over the entire range of EEG frequencies from <1 to 32 Hz. Therefore, we conducted an exploratory study the effects of AVS at the dominant alpha or twice the dominant alpha frequency in 13 college students (Timmermann, Lubar, Rasey, and Frederick, 1999). The two AVS conditions were administered in random order, during two twenty-minute sessions separated by at least two weeks. An eyes-closed baseline was recorded prior to each stimulation session. To determine if AVS had lasting effects on the EEG after stimulation was removed, a second eyes-closed baseline was recorded thirty minutes after the session. Power was analyzed in six bandpasses (delta 1, 0.75-2 Hz, delta 2, 2-4 Hz, theta, 4-8 Hz, alpha, 8-12 Hz, beta 1, 12-21 Hz, and beta 2 21-31 Hz. We found that effects of AVS were widely distributed across the standard 10-20 19-channel montage. Interestingly, AVS at a subject's dominant alpha frequency had no effect in the alpha band, but significantly increased power in the delta 1 (at C3), delta 2 (C3 and Cz), theta (F4 and O2), beta 1 (FP1, F3, Fz, and F4), and beta 2 bands (Fz, Cz, and P3). Only the beta 1 effects (at FP1, F3, and Fz), persisted into the post-stimulation baseline. AVS at twice the dominant alpha frequency significantly increased power in the theta (at T3, T5, F4, F8, C4, T4, P4, and T6), alpha (C3, F8, and C4), beta 1 (FP1, F7, F3, C3, T5, O1, Fz, Cz, Pz, FP2, F4, and C4), and beta 2 (C3, P3, O1, Fz, Cz, and Pz) bands. Twice dominant alpha AVS also significantly decreased power in the delta 1 (at F7, F3, FP2, F4, and T4) and beta 2 (T3) bands. The effects of twice dominant alpha AVS persisted into the post-stimulation baseline only for the increases in the theta (at T5 and T6), alpha (C3, F8, and C4), beta 1 (FP1, F7, F3, C3, T5, O1, Fz, Pz, F4, and C4), and beta 2 (C3, P3, Fz, and Cz) bands. Since sinusoidally modulated light alone, at the dominant alpha frequency, has been demonstrated to produce dramatic driving effects on occipital alpha (Van Der Tweel & Verduyn Lunel, 1965; Townsend et al., 1975), the lack of effect of combined alpha AVS on the alpha band was surprising, and supported the theory that adding a synchronous auditory stimulus attenuates the photic driving response. A coherence analysis of data from this same experiment (presented in chapter 3) lent further support to this theory. Thus, a direct study of the differences between auditory, visual, and combined auditory and visual stimulation on EEG amplitude and coherence (presented in chapter 4) was proposed for this dissertation.

Chapter Two

A tutorial introduction to psychophysiological data analysis with Perl

The analysis of the full richness of a QEEG data set presents methodological challenges which have, in part, contributed to the historical tendency in quantitative EEG research to look at only a very narrow range of variables. For instance, until very recently, there were no popular spreadsheet applications (like Microsoft Excel) capable of opening data sets containing greater than 512 columns of data. Applications driven by graphical user interfaces that require mouse input pose serious limitations to the efficiency of necessary selecting, sorting, transposing, formatting and other operations. While the macro tool in Microsoft Excel allows the user to automate redundant tasks, it became clear during this research that proprietary, graphically-driven data analysis software was often cumbersome and inflexible for the purpose of handling massively multivariate data sets. Therefore, a collection of custom software products was written for this dissertation in Perl. Since the purpose of this written work is to allow others to replicate and extend this research, the following introduction to Perl is presented. Careful study of this introduction, and it's use as a reference, will allow the reader to interpret the code examples which are provided in the methods section, and to adapt them to new data sets. Further, a rudimentary understanding of Perl will help the critical reviewer to evaluate the quality and accuracy of the analytic methods used in this research.

Perl stands for "Practical Extraction and Report Language," and was invented by Larry Wall in 1986. Originally designed as a scripting language, to automate redundant tasks for Unix system administrators, PERL has evolved into a full-featured programming language freely available for nearly every operating system (Wall, Christiansen, and Schwartz, 1996). Perl is a superset of the functions available in the Unix utilities sed and awk, and is syntactically similar to C. Perl is known for being easy to learn and use, with a community of thousands of Usenet group subscribers providing free technical support (comp.lang.sys.perl; Schwartz, Christiansen, and Wall, 1997). Perl syntax is simple and flexible and forgiving of what other languages would consider errors. One of the reasons Perl is popular is that it is free, distributed under the terms of GNU public license. Perl is available for every operating system, so programs transport seamlessly across platforms.

Perl has no built-in limitations. You don't have to predeclare variables as string, integer, floating point or double. Variables just become whatever type you want them to be based upon context, which is one reason it is said that Perl resembles English. Arrays simply grow to whatever length you need them to, limited only by the physical limits of your hardware. Often it is possible to make *implicit references* to *default variables*, just as it is possible to imply the direct or indirect object in English sentences.

Perl is well-documented and supported. If you type man perl on any Unix machine that has PERL installed, there are a number of excellent manual pages. Perl is at the cutting edge of the "open source code movement," so you will find that other Perl programmers are very willing to share their knowledge. *Laziness* is said to be a virtue in a PERL programmer, because an international archive (www.cpan.org) offers thousands of free scripts and modules to accomplish many tasks.

First code example

Since immersion in a language is as important as learning formal principles, this tutorial begins with a program that accomplishes a simple task, and illustrates a number of basic functions, variables and control structures. The following program prompts the user for a file name and a text string, reads the file, and then prints to the screen all lines matching the string.

```
#!/usr/bin/perl print “which file would you like to
search?\n”;
$file = <STDIN>; chomp $file;
open (FH, “<$file”);
print “what pattern do you want to find?\n”;
$pattern = <STDIN>; chomp $pattern;
while ($line=<FH>)      {
     chomp $line;       push(@lines,$line);
}
# all lines in $file are now in @lines
foreach $line2 (@lines) {
     if ($line2 =~ /$pattern/)       {
          print “$line2\n”;
     }
}
```

The first line of every perl program always begins with `#!` and then states the directory path to the perl interpreter. The second line,

```
print “which file would you like to search?\n”;
```

illustrates our first command, print. When this command is executed, it prints the expression in quotation marks to the screen. Notice how each code line or statement in Perl terminates with a semicolon (‘`;`’). Single Perl statements can go on for multiple lines until a semicolon is reached, and single lines can contain multiple, semicolon-delimited, Perl statements. Missing semicolons are a common bug that will keep your programs from compiling.

The next line,

```
$file=<STDIN>; chomp $file;
```

illustrates the use of *scalar variables*. A scalar variable begins with the dollar sign (‘`$`’), and can contain any text, numerical or binary string. Unlike in C, they do not have to be predeclared. Variable names are case sensitive, but basically any combination of letters or numbers following a dollar sign is a valid variable name. The equals sign (‘`=`’) is called the *assignment operator* because it assigns the value on the right to the variable on the left. In this case the value is the standard input, which in this case is whatever the user types on the keyboard before pressing enter. So, this is how you prompt the user for input. `Chomp` is a command that removes the line return from the end of a variable (which the user put there by pressing the “Enter” key). Notice how whitespace (spaces or tabs) are optional between “words” in a Perl “sentence.” Whitespace just makes your code more readable by others and, possibly, by yourself six months in the future.After asking the user for the filename and the pattern to match, the next line,

```
open (FH, “<$file”);
```

opens the file for reading, which is indicated by the arrow pointing to the left, away from the file name. You can also open a file for writing by having the arrow point to the right, toward the file name. This line also introduces another variable type called a *filehandle*. Filehandle names are usually any combination of capital letters (“`FH`” here is an arbitrary choice). The next three lines,

```
while ($line=<FH>)      {
     chomp $line;
      push(@lines,$line);
}
```

is the first example of a *control structure*. A control structure evaluates the truth of whatever is in the parentheses, and in this case, what is in the parentheses is that the filehandle `FH` is being read, one line at a time. As long as the statement remains true (until `FH` runs into the end of the file), this control structure remains true, so the block of code contained in the curly brackets is executed.

There are two commands inside the curly brackets. The first one is `chomp`, which removes the newline character at the end of each `$line`. The next command, `push`, illustrates a third variable type, which is called an *array*, or list variable (arrays begin with an at sign ('`@`'). What is happening here is that as each line is being read from the file specified by the user, the line return is being cut off the end, and then each line is pushed into the array called lines. The next line,

```
# all lines in $file are now in @lines
```

is an example of a *comment*. A comment is a line written into program that is intended to be read only by human beings. Usually, comments provide a direct English play-by-play of what the code is doing, which is helpful for debugging purposes, or for revising code months or years later. The pound sign tells the Perl compiler or interpreter to ignore all of the text that follows until the end-of-line. All lines beginning with the pound sign are comments, with one exception: the first line,

```
#!/usr/bin/perl
```

which states the path to the Perl interpreter.

The next control structure,

```
foreach $line2 (@lines) {
     if ($line2 =~ /$pattern/)       {
          print “$line2\n”;
```

```
        }
}
```

says, “for each element in this list, do whatever is in the curly brackets.” In this case it is taking each line and assigning it to the arbitrarily-named variable `$line2`, and saying if `$line2` matches this pattern, print `$line2` to the screen.

Installing Perl for Windows

To develop the strongest understanding and skill, installing Perl on your desktop machine and trying to run some of these code examples is recommended. Perl for Windows can be downloaded from http://www.activestate.com. To get to the download site from this home page, click on the link to “Active Perl” (the current version is 5.6), and then “Download Now.” Windows 95/98 or NT users will also need to download the Windows Installer available at this site (click on the link for “Windows NT” or “Windows 95/98” after the note for “Windows Users”). The installation file for PERL itself is the link to “Windows Intel” located under “ActivePerl 618.” To clarify any confusion, ActivePerl 618 is Activestate.com’s independent distribution of Larry Wall’s PERL 5.6. Some of these links and directory names may change with upgrades, but readers in the more distant future should still be able to find a valid version of PERL for Windows by browsing activestate.com or by searching the web for “ActivePerl.” By the way, ActivePerl is 8.7 megabytes, so expect to be waiting for a while if you are using a 56K modem. The installation is the same as for any modern Windows software. Save the “Windows Intel” link (the actual current file name behind this link is ActivePerl-5.6.0.618-MSWin32-x86-multi-thread.msi) to your hard drive. Make sure you take note of where you are saving it before you click the SAVE button— the Desktop is recommended. If you are running Windows 95/98 or NT, download and run the Windows Installer first (the current name of this file is InstMsi.exe). Then, simply click on the ActivePerl install file, and follow the instructions provided by the setup program. All of the default setting should work well for this application.

Running a Perl program

Under Windows, executing a Perl program is as easy as clicking on the icon for the program's name. Programs can also be run from the command line by typing

```
perl filename.pl
```

Although a compiler for Perl is available (so you can run .exe versions of your programs on machines that don't have a Perl interpreter installed; see www.indigostar.com), Perl programs generally are text files containing Perl code ending in .pl. In UNIX, you have to make the file executable by changing it's permissions: `chmod u+x filename` or `chmod 755` filename. If you have written a valid path to the Perl interpreter on the first line, you can then just execute the program by entering filename.pl at the command line. If that doesn't work, `perl filename.pl` usually will (note you can omit the first line declaring the path to the Perl interpreter if you want to type the word perl every time you want to run your program).

Two command line options can be helpful for debugging:

```
perl -c filename.pl
```

tests if file compiles successfully without actually running;

```
perl -w filename.pl
```

debug mode, provides extra detail about potential flaws in code.

Scalar variables

To review from the first code example, a scalar variable contains a single element, or string of characters, numbers, or binary data, and can be of any length. Scalar variable names begin with a dollar sign (‘`$`’). Names are case-sensitive (thus, `$var` is not the same variable as `$VAR`). Values assigned to a scalar variable can be any length. The name of a variable can have up to 256 characters. Particular values are assigned with an equals sign (‘`=`’), the assignment operator. Variables will automatically evaluate to a null string or zero numeric value until values are assigned to them. Unlike in C, variable types do not need to be pre-declared as string, floating point, or double, because Perl automatically interprets the variable type from the value assigned to it. This ability to use undeclared variables allows for parsimony and efficiency of style.

To assign a literal string value to a variable, use single quotes:

```
$lab = ‘Brain Research and Neuropsychology Lab’;
```

However, to interpret the value of a variable (while printing or assigning a value to some other variable), is called interpolation, and is accomplished with double quotes:

```
print “My lab is the $lab.”;
# double quotes mean interpolate; prints
# “My lab is the Brain Research and Neuropsychology Lab.”
```

However,

```
print ‘My lab is the $lab.’;
# Single quotes mean literal; prints
# “My lab is the $lab.” to the screen.
```

To concatenate two variables, use a dot (‘`.`’).

```
$var1 = “Hello”;
$var2 = “$var1” . “world!\n”;
Print $var2; # prints “Hello world!”
```

The default variable, ‘`$_`’, is one of the features that allows Perl programs to be written concisely and quickly.

```
print $var foreach $var (@list);   # works
Print $_ foreach $_ (@list);       # also works
print foreach (@list);             # also works
```

Default variables can also make Perl code difficult for novice programmers to understand. As in English, when the direct object of a sentence is occasionally implied and not explicitly stated, the Perl interpreter will assume you know what you’re talking about if you invoke functions without stating which variable to perform these functions on.

Array Variables

Arrays an ordered list of elements, like a prioritized To Do list, or a roster containing the names of students registered for a class. Array names begin with the at sign (‘`@`’). Individual elements of an array are specified by their index number, beginning with 0. For an array named `@array`, the first element could be referred to in scalar context by `$array[0]`. You can refer to the nth element of an array by writing “`$arrayname[n-1]`”. (Just to confuse newcomers, array indexes begin at 0 rather than one. It can be helpful to think of the array index number as the *offset* from the beginning rather as it’s ordinal position.) If an array has 100 elements, the last element is `$array[-1]` or `$array[99]`.

In a scalar context, `@array` is the number of elements in the array;

```
$y = @array;
# scalar context, assigns number of
# elements in @array to $y
```

which can be useful if you want to set up a `while` loop:

```
$y = @array;
$x = 0;
# not required to predeclare $x as zero,
# but it’s more readable.
While ($x<$y)  {
     print “$array[$x]\n”;
     $x++; # increments $x by 1
}
# sequentially prints each element of @array to the screen.
```

Array Input. There are many ways to get data into an array:

(1) use the assignment operator,

```
@frequencies = (“theta”, “alpha”, “beta1”);
```

quoting each element, delimiting with a comma (whitespace optional), or use the quotation operator (‘`qw`’), and spare yourself the formalities:

```
@frequencies = qw(theta alpha beta1);
```

Order can be sorted or reversed:

```
@sorted = sort @frequencies;
# @sorted is now alphabetically sorted: “alpha beta1 theta”

@reversed = reverse @sorted;
# @reversed is now “theta beta1 alpha”
```

(2) The `push` command adds scalar elements to the end of the list: `while($x=<FH>)`

```
     { push (@lines,$x); }
```

(3) The `unshift` command adds scalar elements to the beginning:

```
while($x=<FH>) { unshift (@lines, $x); }
```

Using `unshift` would, of course, result in a page with all the lines in reverse order, which is rarely what you would want.

(4) A quick synonym for

```
while($x=<FH>) { push (@lines,$x); }
```

is

```
@lines = <FH>;
```

(5) Individual array elements can be assigned like scalar variables:

```
$frequency[0] = “theta”;      $frequency[1] = “alpha”;
```

(6) The `split` command divides up the elements of a scalar into an array based on a delimiter:

```
@line = split / /, $lines[0];
```

The delimiter in the `split` statement shown here is a space, but it could be anything. If you don't include the forward slashes with a delimiter between them (followed by a comma), i.e.,

```
@line = split $lines[0];
# same as @line = split / /, $lines[0];
```

Perl will just assume the default delimiter, which is a space. Careful, however, if you have multiple spaces in a row, because the resulting array will include null elements at the boundary between these spaces. If there's a possibility of this, include, somewhere before the split statement, a line of code like this:

```
$lines[0] =~ s/\s+/\s/;
```

which replaces all instances of multiple spaces with a single space (more on pattern matching later).

To review, the syntax of split is

```
split /delimiter/, string ;
```

For example,

```
$line=”Time flies like an arrow\; fruit flies like a
banana.”;
```

```
@time = split /\s+/, $line;
print “$time[5] $time[6]\n”;    # prints “fruit flies”
```

A neat trick, grab only elements 5 and 6 with an *anonymous array*:

```
($word5,$word6) = (split, $line)[5,6];
print “$time[5] $time[6]\n”;    # same result
```

If you leave off the variable during a split, e.g.,

```
@line = split / /;
# splits $_, the default variable, with space as delimiter
```

Perl will assume you mean the default variable.

The most simplistic expression:

```
split;
```

also works. This one-word expression means split the default variable, $_, using space as a delimiter and put the resulting elements into @_, the default array.

Array output. There are equally many methods of getting data out of an array.

(1) By specifying the index:

```
while ($x < @lines) {
print “$lines[$x]\n”;
     $x++;
}
```

The first statement says, while `$x` is less than `@lines`: remember, using `@lines` in a scalar context returns the number of elements in `@lines`; and if this is the first time `$x` has been stated in the program, it’s string value is assumed to be null and it’s numeric value is assumed to be zero... so, on the first iteration, it prints `$lines[0]`, the first element of `@lines`, followed by a line return; and then it increments `$x` by 1. (`$x++` is shorthand for `$x = $x + 1`).

(2) By using `foreach`:

```
foreach (@lines)    {
```

```
          print “$_\n”;
}
```

`Foreach` takes each element of an array, starting with the first or zero-indexed element, assigns it to the default variable (or to `$happy` if you state `foreach $happy (@lines)`), and then allows you to do whatever you want to that variable in between the curly braces. In this example, we are implicitly referring to the default variable in the `foreach` statement, and explicitly referring to it in the `print` statement.

(3) When quoted `“@array”` returns all of the elements in the array separated by a space.

```
@frequencies = qw(theta alpha beta1);
print “@frequencies”;
# prints “theta alpha beta1”. However,
print @frequencies   ;
# prints “3”, the number of elements!
# Be careful to say what you mean.
```

(4) By using `pop` and `shift`:

```
$first = shift @foods;
$last = pop @foods;
```

`Shift` removes the first element of an array and assigns it to a variable (the opposite of `unshift`). `Pop` removes the last element of an array and assigns it to a variable (the opposite of `push`). Note how `pop` and `shift` do double duty, both removing the array element, and making it available to a scalar variable via the assignment operator. Note that sometimes you’ll just use `shift` and `pop` to get rid of array elements (and not use the variable to which those elements are assigned); other times you’ll be more interested in the actual value of the element.

(5) By using `join`:

```
$freqs  = join /:/, @frequencies;
# $freqs is now “theta:alpha:beta1”
```

`Join` is the opposite of `split`. It takes all of the elements of an array and glues them together into a single string, using the delimiter you specify (space by default), and assigns it to the variable you specify (`$_` by default).

Process Input/Output

When running, programs are called processes under the operating system (you can list and kill them with the Task Manager in Windows or with `ps` and `kill` in Unix). We've already seen one way of getting data into a process (prompt the user for input):

```
print “which file would you like to read?\n”
$filename = <STDIN>;
chomp $filename; # get rid of that pesky newline.
```

To build a script that uses the following command line syntax:

```
program.pl pattern filename
```

use the default `@ARGV` array. `@ARGV` is list of words following the name of the program, (or “arguments”) supplied by the user from the command line. Whenever you state anything on the command line after your program name, it's automatically pushed into `@ARGV`, which can be accessed by:

```
$pattern = $ARGV[0]
$filename = $ARGV[1];
```

So, if the first string after your command is the pattern you want to match, and the second string is the file you want to test for the existence of the pattern, the pattern would be stored in `$ARGV[0]` and the file would be stored in

`$ARGV[1]`. Since `@ARGV` is a default variable in PERL, you can open files explicitly:

```
open(FH,”<$ARGV[1]”);
while($var=<FH>)    { print “$var”;  }
close FH;
```

Or, let Perl assume you know what you’re doing:

```
while(<>) {  print;  }
# opens the default file, $ARGV[0], assigns it to the
# default filehandle, assigns each line of $ARGV[0] to the
# default variable, and prints to the default output
# filehandle, STDOUT, the terminal screen.
```

Sometimes you’ll want your script to open and work with multiple files.

```
while (<>){ do...}
```

will open each element of `@ARGV` one at a time as long as each one exists as a file. Be careful, however, because PERL won’t complain if you tell it to open a filename that doesn’t exist. If you want your script to complain in this situation you have to tell it to explicitly:

```
open(FH,”<$ARGV[1]”) or die “Could not open $ARGV[1]: $!
\n”;
```

`$!`, when used in this context, returns the system error string which should tell you why your `open` statement failed.

System calls are another effective method to integrate your program with the outside world. Backticks (‘` `’) allow Perl to *escape* to the command line, run another program, and capture it’s standard output. For example, in Unix:

```
@files = ‘ls‘;
```

The `ls` command, similar to the `dir` command in DOS, returns the names of all files in the working directory. Once in an array named `@files`, they can operated on one by one with other system calls or filehandles.

If all you care about is successfully running another program from your script, and you don't care about it's standard output, `system` is a more efficient use of memory. `System` just executes the expression and returns 1 if successful, 0 if not. For example,

```
system (”mailx -s \”test mailing\” smiile@utk.edu < file”);
```

Executes the Unix `mailx` utility to send a file to `smiile@utk.edu`.

The simplest way to get data out of a Perl program is to use the default or standard output, which prints to the terminal screen:

```
while(<>) {
     print ”$_” if ($_ =~ /$pattern/);
}
```

We've already seen how to use the `print` statement to get data out of a program. It's a basic DOS/Unix trick to redirect that output from the terminal screen into a file using greater than ('>').

```
% program.pl pattern filename > newfile
```

If you don't want to have to type the name of your output file at the command line every time you want to capture your program output to a file, then build it into your program explicitly with an output filehandle.

```
open(OUT,”>output.txt”);
while($_=<>)   {
     print OUT if (/$pattern/);
}
```

Remember that the default filehandle is the standard output, or terminal screen (or whatever file you redirect to). You must explicitly state the name of your outfile handle, in this case `OUT`, after the `print` statement, if you want your output to go somewhere other than the standard output.

First Exercise

Given what you've already been shown, you actually now have the all tools you need to write some useful code. Suppose you have the following SAS output for 1330 variables. Write a Perl program that extracts and prints just the variable name and the p-value of each Signed Rank test, one variable name and p-value per line.

```
The SAS System              23:11 Sunday, February 4, 2001   1
                    The UNIVARIATE Procedure
                   Variable:  C3C4CAL1ALPHA
      Test          -Statistic-       -----p Value------
      Student's t    t  0.095791    Pr > |t|    0.9246
      Sign           M       1.5    Pr >= |M|   0.6776
      Signed Rank    S       5.5    Pr >= |S|   0.8715
```

Think about each step you need to accomplish this, and decide which code to use to execute each step. Go back and review the previous sections to get the exact syntax you need. Want some hints? Well, first you open the file containing all of this SAS output. You could do that either by explicitly opening the file with a filehandle, or by passing the filename to your script as a command line argument, and opening it with the default filehandle. Then, while reading the filehandle, search each line for "`Variable`", and when you get a match, print the next whitepace-delimited string after "`Variable`". (You will need to `split` the line into an array to do this). Whenever you match "`Signed Rank`" print the last string of that line followed by a newline (\"). (One possible solution to this exercise is found in the script, *parse-sas-univariate.pl,* in Chapter 3.)

Note that this problem throws you at least one curveball: Every line of the input above, except for the first line, has an unknown number of space characters before the first word. If you `split` with whitespace as a delimiter, the first few space-delimited elements are null fields! I.e., there is "", followed by at least one whitespace, which would be extracted as a null element into your array. To prevent this happening, you will need a line that says something like,

```
$line =~ s/^\s+//;
```

which gets rid of all the whitespace characters at the beginning of each line (see discussion in **Regular Expressions**, below).

Control Structures

We've already seen some control structures in our examples. When you use a conditional expression such as `if`, `unless`, `while`, or `until`, Perl evaluates the statement in parentheses, and if the statement is true, it carries out the instructions written in the curly brackets.

```
If (some_statement) {
          do something;
          do another something;
} elsif (other_statement){
          do something else;
} else {
          do this only if both statements false;
}
```

Note that for the `if` conditional, you can add as many other alternative conditions as you like with the `elsif` statement, or encompass every other alternative condition with one final `else`. Other conditionals include

```
while (some_statement)   {
          do something;  # until statement becomes false.
}

until (some_statement)   {
          do something;  # wait until statement is true:
                         # the opposite of while
}

unless (some_statement)  {
          do something;  # do only if statement is false:
                         # the opposite of if
}
```

Note that absolutely any truth-test can go in the parentheses: numeric comparisons, tests for the existence or size of a file, or whether a filehandle or socket it is open, even whether a command executes successfully: every command is false it doesn't execute successfully and true if it does. Formally, the nature of truth is that everything is true except 0 and "", the null string. Occasionally, for debugging purposes, it can be useful to replace the `(statement)` evaluated by a control structure with `(1)`, which is always true.

Perl provides the following Boolean operators:

```
&&        And
||        Or
!         Not
```

For example,

```
while ($_=<> && ($x!==12))    {
          print if (/Signed Rank/ || /Student/);
          $x++;
} # print lines containing Signed or Student from
  # the first twelve lines of the standard input.
```

The conjunction ('&&') allows for a convenient shorthand expression for `if()`:

```
do this && do that;
# if it's impossible to "do this," "do this" is false so
# Perl won't even try to "do that," a shorthand for if()
```

which allows for extremely parsimonious expressions like `/Signed Rank/ &&`

```
print((split)[-1]) while (<>);
# while reading each line of the file, if the line contains
# 'Signed Rank', split the line with white space as the
# default delimiter and print the last element, i.e., the
# last word on that line.
```

Perl provides the following comparison operators for evaluating truth-tests:

	Numeric	String
Equal	==	eq
Greater than	>	gt
Less than	<	lt
Greater than or equal	>=	ge
Less than or equal	<=	le
Not equal	!=	ne
Not equal with signed return	<=>	cmp

Given what you've seen so far, can you see what's wrong with the control structure below? (You might want to cover up the answer to see if you can figure it out.)

```
if (($x = 25) && ($y < 25) )  {
          print "$y\n";
          $x++;  }
```

Answer: the statement, "`$x = 25`," is always true because "`=`" is the assignment operator. The comparison operator for tests of numerical equality is "`==`". This is a common mistake among novice programmers.

Perl also supports the standard arithmetic operators:

Plus	+
Minus	-
Divide	/ (floating point mode default)

Multiply	`*`
Exponentiate	`**`
Modulus	`%`

Second Exercise

Suppose you have a data file that has K variables and N observations in a rectangular matrix, thus:

Obs'n, Varname(1), varname(2), ... varname(k)
1, data(1), data(2), ... data (k)
2, data(1), data(2), ... data (k)
...
N, data(1), data(2), ... data(k)

Write a Perl script that finds average for each K across all N. For hints and suggestions, see *median.pl* in chapter 4.

Hashes

Hashes are also known as *associative arrays*, because they associate pairs of elements which are called *keys* and *values*. Hash names begin with the percent sign ('%'). Unlike arrays which are *ordered* lists indexed by *integers*, hashes are *unordered* lists indexed by *keys*. For example:

```
%emails = (Jon => 'smiile@utk.edu',
           AJ  => 'ajw@utk.edu');
print "$emails{AJ}\n";
# prints "ajw@utk.edu";
```

As in array indexing, individual hash elements are referred to with a dollar sign, then the name of the hash, and then, unlike array indexing, the key is placed in curly braces, not square brackets.

Hash input. There are (at least) three ways to get data into a hash:

(1) Assigning, with commas:

```
%grades= (Jon, A, Harley, C, Marco, B)
# or, “=>” is a more readable synonym for comma:
%grades=(Jon=>A, Harley=>C, Marco=>B);
```

(2) Assign each element in scalar context:

```
$grades{Jon}=A; $grades{Harley}=C; $grades{Marco}=B;
```

If a key already exists, adding it to the hash will overwrite (or “clobber”) the previous value of that key. To prevent this:

```
unless (exists ($emails{$name}))   {
          $emails{$name}=$email;
}

# Or,

if (!emails{$name}) {
          $emails{$name}=$email;
}
```

Hash output. There are at least three ways to get data out of a hash.

(1) Refer to the value with the key:

```
print “$grades{Marco}”;
```

(2) Use each:

```
while (($name,$grade) = each(%grades))  {
          print “$name got a $grade\n”;
}
```

(3) Grab all the keys using keys:

```
print sort keys %grades;
# Just the values
print sort values %grades
```

Note the use of the array operator, sort, which alphanumerically sorts the keys after they have been extracted by keys from `%grades`. Since hashes are *unordered* lists, the order in which data comes out of a hash is not necessarily the order you put the data into the hash. While this might seem pointless and confusing, it actually is a feature that allows for more efficient memory allocation and rapid pattern-match searching across all the keys. For example,

```
if ($hash{key}){
     print $hash{key};
}
```

instantly tells you if a certain key is in the hash. Arrays, on the other hand, require a specific serial position in memory for each element. To test for the existence of a certain element, you have to iterate over all these elements and test each for the pattern-match:

```
foreach $element (@array){
     if ($element =~ /key/)   {
          print $element;
     }
}
```

Finally, the most destructive way to get data out of a hash, `delete`, removes both the specified key and it's corresponding value from the hash.

```
delete $hashname{$key};
```

Regular Expressions

Regular expressions are patterns to be matched against a string. If you are familiar with the pattern matching in Unix, you're in luck. Perl regular expressions are a superset of pattern-matching functions in the Unix utilities `grep`, `sed`, `vi`, and `awk`.

We've already seen:

```
print if (/$pattern/);
```

which is shorthand for:

```
print $var if ($var=~m/$pattern/);
```

`m`, or the *match* operator, is the default pattern matching operator.

Other pattern matching operators and functions include

`s`, the *substitution* operator:

```
$var =~ s/$pattern/$replacementpattern/g;
# “g” modifier means “global”: all occurrences on each line

$var =~ s/$pattern/$replacementpattern/i ;
# “i” modifier means ignore upper vs. lower case
```

“`=~`” is the binding operator. It says, take the pattern on the right, and if it matches the variable on the left, return true if it’s a `m//` type operator, or replace if it’s a `s///` type operator. Again, don’t use the assignment operator (‘`=`’) or the equivalence operator (‘`==`’) when you mean the binding operator (‘`=~`’), or you will spend hours debugging and pulling out your hair.

We’ve already seen `split` and `join`, which are also pattern matching functions:

```
@list = split /$pattern/, $var;
# splits $var into list with $pattern as delimiter

$var = join /delimiter/, @list;
#  joins list into a single variable
```

Metacharacters (‘ \ `|  (  )  [  {  ^  $  *  +  ?  .` ‘), when used in pattern matching, mean something other than their literal denotation, unless you escape them with a backslash (‘`\`’). Backslash means “escape” or literal interpretation of metacharacters:

```
$var =~ s/\|\$/pipe-dollar/;
#means replace ‘|$’ with ‘pipe-dollar
```

Escaping normal alphanumeric characters turns them (some of them) into metacharacters:

\s	white space (tab or space)
\n	line return

\w	any word character [a-zA-Z0-9_]
\W	any nonword character (eq. to [^\w])
\a	beep
\t	tab
\f	formfeed
\r	carriage return
\d	a digit, same as [0-9]
\D	a nondigit, same as [^0-9]
\s	a whitespace character, same as [\t\n\r\f]
\S	a non-whitespace character

"|" means "or"; Parentheses allow grouping:

```
print if (/Dept of (Psychology|Biology)/);
# prints lines containing
# "Dept of Psychology" or "Dept of Biology"
```

"." means "any character"

"*" means any number of the previous character:

```
/Psych.*/   # matches Psychology or Psychiatry
```

"+" means "one or more of the previous character"

```
$line=~s/\s+/\t/g;
# replace one-or-more spaces with a tab
```

"^" means beginning of the line.

"$" means end of the line.

```
s/^\s+//; # gets rid of spaces at beginning of line
```

"[ ]" identify a "character class"

```
s/[A-Ex2]/R/g
# replaces A, B, C, D, E, 2, or x with R.
```

"[^...]" identifies a negative character class. For example,

```
while(<>)       {
     foreach(split /[^\w\@\.\-]/ ) {
          /\@/ && print "$_\n";
     }
}
```

This brief program extracts email addresses from an html file, by splitting each line with every non-valid email character (everything that is not a word character, '@', '.', or '-'). After splitting, the line becomes an anonymous array which is iterated through foreach.

```
/\@/ && print "$_\n"
```

is an efficient use of && as a conditional expression (shorthand for if; see **control structures**)

Parentheses as memory. Using parentheses around all or part of the regular expression to be matched also causes the expression to be remembered for later use. For example,

```
s/C(..)/D\1/;
```

This expression searches for C followed by any two characters, and replaces it with D followed by those same two characters. CAT becomes DAT; COH! becomes DOH! (\1 is the memory buffer that the parenthesized characters are placed into.)

Subroutines

It is entirely possible to write almost any Perl program without using a subroutine. However, sometimes when blocks of code become extremely complex (as when you have conditionals nested to 6 or more levels), or when you want to quickly merge the functions of some old code into your new program, it can be useful to write blocks of code as subroutines, and then simply invoke the subroutine whenever that code is needed. Subroutines are typically written at the end of a program and are created as follows:

```
sub name {  ... }
```

Where `name` is the name of the subroutine and `...` is the subroutine code. Invoking a subroutine involves simply writing an ampersand (‘&’) before the name of a subroutine which is defined elsewhere in the program. An example of a program that uses a subroutine is:

```
print “What’s your name?”;
chomp ($name = <stdin>);
& hello;
sub hello {
     print “Hello, $name!\n”;
}
```

Installing Unix for Windows

Although not essential for running Perl in a Windows environment, UWIN makes life a lot easier when writing Perl on Windows platforms. Go to http://www.research.att.com/~dgk/uwin/ and click on download software. Click for the appropriate license (educational and research use is free), and select the UWIN base, UWIN base update 2.25, and the UWIN perl. Download and install like any windows program. The UWIN k-shell window is then activated by selecting START:UWIN:KSH. Enter `df` from the command line to see the top level of your file hierarchy. UWIN perl is now located in /usr/bin/perl and can be invoked by `#!/usr/bin/perl` on the first line of your script, or `perl` at the command line. However, if you also have Active Perl available. /c/perl/bin/perl is active Perl. I usually will create a symbolic link in /usr/bin/ to help keep them straight:

```
ln -s /usr/bin/activeperl /c/perl/bin/perl
```

Or, you can achieve the same result by using the Windows Explorer to create a shortcut from c:\perl\bin\perl.exe and dragging it over to c:\Program Files\UWIN\usr\bin\ .

ActivePerl gives you more complete control over your Windows operating system. UWIN perl allows you to make Unix system calls. So, be careful which version of Perl you run.

Chapter Three

Pilot Study: EEG coherence effects of audio-visual stimulation (AVS) at dominant and twice dominant alpha frequency

Background

While the amplitude and power effects of AVS have been widely studied, relatively little is known about the effects of AVS on EEG coherence. Coherence is a correlational measure, varying between zero and one, of the variability in phase between two signals over time (Shaw, 1981). This frequency-specific signal correlation suggests the extent to which two regions are cooperating on the same task. Coherence in the eyes-closed baseline reflects the number of synaptic connections between recording sites, and the strength of these connections (Thatcher, 1992). Coherence has been shown to be lower in Alzheimer patients, comatose subjects, and in brain-injured subjects, while it is higher in mentally retarded persons, during sleep, and during epileptic seizures. Between these extremes, "optimal levels" of coherence for normal functioning have been described (Silberstein, 1995). While differences in photic driving of coherence have been described between normal subjects and patients with Alzheimer's disease (Wada et al. 1998a), schizophrenia (Wada et al. 1998b), and between genders (Wada et al., 1996), the effects of combined auditory and visual stimulation on coherence in normal subjects have not been reported. Since AVS devices are commonly referred to as "brain wave synchronizers" (Morse, 1993, Shealy et al., 1990), we hypothesized that AVS would increase coherence at the frequency of stimulation. We assumed that effects would be most prominent over the occipital and temporal leads, which are closest to the primary visual and auditory cortex. Given our previous findings of increased amplitude in multiple frequency bands, we anticipated effects across the coherence spectrum, but did not predict directions of change.

Materials and Methods

This study was a reanalysis, in terms of coherence, of data previously analyzed and reported in terms of power (Timmermann et al., 1999). However, in this study, ten additional participants were added to each stimulation condition. In the original study thirteen participants (7 male and 6 female) were recruited from the undergraduate and graduate populations of the University of Tennessee, Knoxville. Participants ranged in age from 20-30 years, with a mean of 25.5 years. The twenty additional participants (4 male and 16 female) ranged in age from 21-45 years, with a mean of 24.3 years. All participants reported no previous history of epilepsy, learning disabilities, attention deficit disorder, or mental illnesses during personal screening interviews. Participants self-reported that they were free of medication use during the study.

Audio-visual stimulation was provided by a Polysync Pro (Synetic Systems) device. This unit consisted of headphones and a pair of "photoscopic" glasses that were connected to a small, portable unit that was programmed to provide specified levels of visual and auditory stimulation. The glasses had eight light emitting diodes (LEDs), four per side, arranged in a cross pattern. The LEDs were situated approximately 1.5 cm from the eyes, and emitted red light at .166 candle power at the frequencies employed. Audio stimulation consisted of a cycled tone with a pitch of 185 Hz, presented to both ears simultaneously, with a duty cycle of 50% and a loudness level of approximately 77 dB(A scale) for the alpha condition and 81 dB(A scale) for the beta condition. Both auditory and visual stimulation were sinusoidally modulated. The Polysync Pro equipment was tested for and did not produce any detectable electrical interference in our EEG recordings.

The first thirteen participants came to the laboratory on two different occasions for AVS sessions. These sessions were at least two weeks apart to minimize carry-over effects. The presentation of the AVS condition was counter-balanced; during the first AVS session six of the participants experienced alpha stimulation and seven experienced twice-dominant alpha ("beta") stimulation. For the second session, those who experienced alpha stimulation during the first session received beta stimulation, and those who experienced beta stimulation first received alpha stimulation. The additional twenty participants were randomly assigned to either an alpha AVS or beta AVS group, as part of an experiment measuring the effects of multiple sessions of AVS (Timmermann, 1998). Only results from the first session of this experiment are included in this analysis. The two groups were counterbalanced for age and gender, and did not differ significantly with respect to dominant alpha frequency. The mixing of subjects from both repeated measures and independent groups designs in the present study might compromise the validity of some inferences about the differences between the two stimulation conditions. However, both experimental designs have advantages and risks for detecting such differences: repeated measures designs risk random within-subject effects, while independent group designs risk random between-subject effects. We decided that this compromise was worth the increase in statistical power obtained for all other comparisons in this study. The procedure was the same for either stimulation condition. Participants were seated upright in a plastic chair located in a sound-attenuated, dimly lit room for the EEG recordings. The headphones and glasses were placed over the electrode cap. All participants had an eyes-closed baseline EEG recorded for four minutes at the beginning of the session. This baseline recording was analyzed to determine each subject's dominant alpha frequency, defined as the peak power between 8 and 12 Hz at locations Pz and P4 as measured by power spectral analysis, rounded to the nearest 0.5 Hz. Participants were then provided AVS for 20 minutes, with EEG recording occurring simultaneously. For the alpha stimulation condition, each participant was stimulated at their dominant alpha frequency. During the beta stimulation condition each participant received AVS at twice their dominant alpha frequency. Participants were instructed to close their eyes and relax during the 20-minute AVS. Thirty minutes after the stimulation session, a post-session eyes-closed EEG was recorded for 5 min.

A quantitative referential EEG (monopolar montage) was recorded from 19 electrodes following the International 10-20 system for electrode placement, with linked earlobe references. All electrode impedances were below 5 KOhm. Recordings were made using an electrode cap (Electro Cap Inc.). Raw EEG was fed through 19 matched 7P511 pre-amplifiers (Grass Instrument Co.), with bandpass filters set to 0.5-100 Hz. A BMSI 12-bit A to D converter digitized the outputs of the amplifiers. Rhythm software (Stellate Systems) was employed to record the raw EEG. The sampling rate was set at

128 samples per second. Signal aliasing was eliminated by the use of a 16 point FIR (finite impulse response) filter, with a sharp low pass cutoff set at 64 Hz and higher. Coherences were analyzed from the raw EEG off-line on a Pentium 233 processor using Rhythm software. Rhythm analysis employs Hanning windowing and cosine tapering of each selected four-second epoch. Eye-blinks, large eye movements, and all observable muscle artifacts were removed prior to analysis by a visual review of the EEG records.

Results

EEG data were analyzed via Fast Fourier transformation to derive 8 interhemispheric and 55 longitudinal coherence pairings (PAIR; see Table 1; all tables and figures are found in Appendix C) for each stimulation type (STIMTYPE; ALPHA or BETA) in each of six bandpasses (D1, 0.75-2 Hz; D2, 2-4 Hz; TH, 4-8 Hz; AL, 8-12 Hz; B1, 13-21 Hz; or B2, 21-31 Hz). Each 20-minute AVS condition was analyzed in four 5-minute blocks (1, 0-5 min; 2, 5-10 min; 3; 10-15 min; or 4, 15-20 min) to examine changes over the course of stimulation. The post measure ("P") was a 5-minute EEG recording taken one half hour after the AVS session. Thus, there were a total of 4536 combinations of pair, band, stimtype, and time variables. For each subject, 3780 differences from the 756 pre-stimulation baselines were calculated (CODIFs), one for each combination of independent variables.

The Stellate Rhythm software runs with a graphical user interface in DOS, which precludes any method of automating data analysis. Of the 171 possible coherences/frequency band, Rhythm only allows 32 to be calculated per iteration, requiring an impractical and, by modern standards, unnecessary use of human resources. The ".rpt" files containing coherence values thus evaluated were opened as tab-delimited text files in Microsoft Excel. The MACRO recorder and VISUAL BASIC EDITOR under the TOOLS menu in MS Excel allows for significant automation of redundant data importing, parsing, and formatting operations. Any series of mouse or keyboard actions can be captured by selecting TOOLS: MACRO: RECORD (under the TOOLS menu, select MACRO, and then select the RECORD option). The software prompts the user for a name for the macro, and then translates the user's actions into Visual Basic code, which can then be viewed by looking at TOOLS: MACRO: VISUAL BASIC EDITOR. Visual Basic code for a macro can then be edited to write more flexible and powerful applications. Although the initial stages of analyzing these data were performed in Excel, most of these operations have since been replaced by more efficient custom applications written in Perl. The use of Stellate Rhythm and MS Excel for coherence data is no longer recommended by this author. The Lexicor Neurosearch 24 and software are now standard in this laboratory. Therefore, the details of Excel macros will be omitted here, while Perl algorithms for parsing and formatting Lexicor data sets will be described in chapter 4. Some additional Perl scripts originally written for this pilot study, which were upgraded for the follow-up study, will also be omitted here and references made to their most recent version presented in chapter 4. Those which have not changed since the pilot study will be presented here.

SAS proc univariate code. Tests of normality, ANOVAs, and signed rank tests were performed with Statistical Analysis Software (SAS Institute). SAS `proc univariate` was used to perform both normality tests and signed rank tests. The SAS `proc univariate` code, where the independent variables were `STIMTYPE` (`ALPHA` or `BETA`), `SUBJECT` (N=23), `BAND` (`D1, D2, TH, AL, B1, B2`), `PAIR` (for the 63 coherence pairs analyzed), and `TIME` (1, 2, 3, 4, or `P`), where `COHDIF` was the dependent variable, was

```
data stim;
input STIMTYPE $       SUBJECT $ BAND $ PAIR $ TIME $
COHDIF;
cards;
ALPHA   1      AL      C3C4    1       -4.48
/* SAS comments are denoted with backslashes and asterisks
at the beginning and end of each comment. */
/*.... (86938 data lines omitted).... */
BETA    23     TH      T5T6    5       .
run;
proc sort; by stimtype band pair time;
proc univariate normal data=stim;
var cohdif;
by stimtype band pair time; run;
```

Subjects 14, 21, 22, and 23 were missing COHDIF observations for 31 of the coherence pairings. Thus, missing observations were denoted with a dot ('.') as shown for the final observation above. N was adjusted to 19 in these cases.

Shortlines.pl. Since SAS does not allow input files with thousands of columns, the following script revises the input file with a newline character appended after every six space-delimited words:

```
#!/usr/bin/perl
# shortlines.pl, By Jon Frederick
# shortens length of SAS input lines.
while (<>) {
     chomp; @line = split (/\s+/, $_);
     while ($y < @line)  {
          $x = 0;
          while ($x < 6) {
               print "$line[$y] ";
               ++$y; ++$x;
          }
          print “\n”;
          }
$y = 0;
}
```

Parse-sas-univariate.pl. The standard output of this script is redirected into another file with the extension “.sas”, and provided as an argument to SAS from the command line:

```
sas filename.sas
```

which produces an output file with the same filename and the extension “.lst”. With a total of 3780 combinations of independent variables, The output of SAS tended to be thousands of pages, and thus impractical to print or even to read. Thus, the following script, *parse-sas-univariate.pl* extracted the relevant data from the SAS “.lst” file.

```
#!/usr/bin/perl
# parse-sas-univariate.pl, By Jon Frederick
while(<>) {
     if (/STIMTYPE/i && !$seen{$_})      {
          $seen{$_}=1;
          s/\-//g; s/ STIMTYPE=//i;
          s/FREQ=//i; s/BAND=//i; s/PAIR=//i; s/TIME=//i;
# These substitution statements simply search for the
# given patterns and replace them with nothing,
# leaving only the relevant variable information
     chomp;
          print;
     }
     if (/Variable/ && !$seen{$_}) {
# if $_ contains “Variable” and has not yet been seen
          $seen{$_}=1;
# enter it into the %seen hash
          s/\s+Variable/Variable/;
# gets rid of one or more spaces before “Variable”
     ($varname)=(split/\s+/,$_)[1];
# grabs the second (index “1”) space-delimited string
          print “$varname “;
     }
     if (/\s+Mean/&&/Sum Obs/)      {
          s/\s+Mean/Mean/;
          @temp=split/\s+/;
          if ($temp[1]<0)      {
                print "- "; # for negative means
            } else    {
                print "+ "; # for positive means
            }
        }
        if (/Signed/)  {
            s/\s+Signed/Signed/;
            @temp=split;
            $temp[7] =~ s/[\<\>]//;
        # p-values "<.0001" are converted to ".0001"
            print "$temp[7] ";
        }
        if (/Shapiro/) {
            s/\s+Shapiro/Shapiro/;
            @temp=split;
            $temp[6] =~ s/[\<\>]//;
            print "$temp[6] ";
        }
        if (/Kolmo/)   {
            s/\s+Kolmo/Kolmo/;
            @temp=split;
            $temp[6] =~ s/[\<\>]//;
            print "$temp[6]\n";
        }
}
```

The output of this script was 3780 lines, each with the variable name, + if the mean was positive, - if it was negative, and the p values of the signed rank, Shapiro-Wilks, and Kolmogorov-Smirnov statistics. These were sorted separately by the rank of each statistic in MS Excel and the number of significant tests were thus easily counted by visual inspection. This procedure found that 294/3780 (or 7.8 percent) had a significantly non-normal distribution (at $p<0.01$) by the Kolmogorov Smirnov test and 404/3780 (or 11 percent) were significantly non-normal (at $p<0.01$) by the Shapiro-Wilks test. Therefore, it was determined that the underlying distribution was non-normal and all inferential statistics would be performed on ranked data (Conover, 1980).

SAS proc glm code. Repeated measures analysis of variance were performed with SAS `proc glm`:

```
/* data input lines the same as above */
proc glm data=ranked;
class stimtype subject band pair time;
model cohdif = subject band pair time;
random subject;
means stimtype band pair time/ tukey;
run;
```

This analysis found significant main effects of time (F=27.84, p<.0001), electrode pairing (F=41.28, p<.0001), and frequency (F=43.08, p<.0001). Tukey's post-hoc procedure determined (at p<.05) that the overall coherence was higher during the first ten minutes (intervals 1 and 2) and the post-stimulation baseline than during the last ten minutes of stimulation (intervals 3 and 4). Tukey's procedure found three distinct frequency groupings (beta 1& delta 2 > theta & beta 2 > alpha & delta 1). The type of stimulation had no significant main effect (using the more conservative, independent groups ANOVA model; F=3.03, p=.08). However, interactions between between stimulation type and frequency were observed (F=18.23, p<.0001). Alpha stimulation decreased overall coherence, compared to beta stimulation, in the alpha (F=17.18, p<.0001) and theta (F=6.77, p<.0093) bands. Beta stimulation decreased overall coherence, compared to alpha, in the delta 1 (F=15.69, p<.0001) and beta 2 (F=40.43, p<.0001) bands. There were no significant differences of stimulation type in the delta 2 and beta 1 bands.

The number of possible interactions of electrode pairing with other variables was considered too great to be approached practically with ANOVA methods and Tukey's procedure. Thus, the difference from baseline for each pairing location was tested for significance with Wilcoxon's signed rank test (at p<=.01), using the `proc univariate` as for the normality tests above. Since this procedure was performed on 3780 variables (2 stimtypes X 6 frequencies X 5 time-differences X 63 pairing locations), under the null hypothesis, 3780 x .01 = 37.8 false-positive signed rank tests could be expected by random effects alone, whereas 241 positive tests were actually. Although it is highly unlikely that this ratio of signal-to-noise would arise by chance alone in 3780 independent experiments, these variables are highly interdependent. A principal components analysis (code not shown) revealed that at most, 19 independent factors explained 100% of the variance in these data. It is possible that the large number of significant variables is explained, not by the experimental conditions, but by random effects in only one or a few large underlying factors. To test whether this was the case, empirical distribution functions (EDFs) of the number of significant signed rank tests observed when the baseline was randomized with respect to the stimulation conditions, were constructed, resampling each of 3780 sign-rank tests 1000 times with replacement. The rank of the actual observed number among the randomized trials (divided by 1000) was thus it's significance (Edgington, 1987). To preserve the covariance relationships among variables, a single random decision determined whether the sign of the difference from baseline would be reversed for all variables for each subject, a total of 23 decisions per trial.

Avsrandom.pl. The randomizations were performed by the following Perl script, *avsrandom.pl*. Rather than reading the data from a file for each of 1000 iterations, *avsrandom.pl* includes all of the data in a subroutine called `getdata`, which declares all the data as elements in the `@datalines` array. Each of the 23 data lines in @datalines consists of all 3780 variable observations for one subject. *Avsrandom.pl* randomly decides whether to reverse the sign of all 3780 differences from baseline, writes and executes SAS `proc univariate` code with the randomized data, and parses out the sign-rank statistics by invoking *parse-sas-univariate.pl.* It then invokes another Perl script called *grab-srs.pl*, which extracts the significant sign-rank statistics (p<0.01) for each iteration, and wrote them to a temporary file. This temporary file is then read by another Perl script, *countpervariable.pl,* which reads the temporary file from *grab-srs.pl* and reports how many significant signed rank tests were observed for each variable. This output is then appended to `results-$nodename`, for each of the 1000 iterations. The 1000 iterations were distributed among five Sun 300-350 MHz workstations, running SunOS 5.8, with 256 Megs of RAM or greater, for about two hours, and then all five `results-$nodename` files were appended together. Finally, a separate script called *avsedf.pl* reads the results file and reports the relative rank of the real data among the 1000 randomized trials.

```
#!/soft/script/bin/perl
# avsrandom.pl, By Jon Frederick
# randomizes the sign of each difference from # baseline,
# thus shuffling the baseline with each
# stimulation condition. Writes and executes SAS proc
# univariate code and reports the number of significant
# tests per variable category for each of $xxx iterations.
# Requires parse-sas-univariate.pl, grab-srs.pl, and
# countpervariable.pl, and returns a file named
# $nodename-results
$nodename=‘uname -n‘;
# since multiple unix boxes are running this code,
# grab the name of the unix box running this instance
chomp $nodename;
& getvarnames;
& getdata;
srand; # seed the random number generator
‘date >> log-$nodename‘;
while ($xxx<200)    {
#----------------------------------
open(SAS,”>avs$nodename.sas”);
print SAS “data stim\;\n”;
print SAS “input \n”;
$length=0;
foreach $var (@varnames) {
     if ($length<4) {
          print SAS “$var “; $length++;
     } else {
          print SAS “$var\n”; $length=0;
     }
}
print SAS “\n\; cards\;\n”;
$length=0;
foreach $line (@datalines) {
     @data = split/ /,$line;
# commenting out the following lines allows a nonrandom run
     $rand = rand;
# set $rand equal to a random number between zero and one
     if ($rand < .50000) {
          foreach $datum (@data)   {
               $datum = -1*$datum;
          }
     }
     foreach $datum (@data) {
     # shorten input lines so that SAS won’t crash
     if ($length<5) {
               print SAS “$datum “; $length++;
          } else {
               print SAS “$datum\n”; $length=0;
          }
     }
}
print SAS “\n\; run\; \nproc univariate
normal data=stim\; run\;”;
close SAS;
‘sas -memsize 1G avs$nodename.sas‘;
‘parse-sas-univariate.pl avs$nodename.lst|grab-srs.pl >
tmp-$nodename‘;
‘countpervariable.pl tmp-$nodename >> results-$nodename‘;
$xxx++;
} # end while
‘date >> log‘;
#---------------------------------------------
sub getvarnames {
 @varnames = qw(C3C4CAL1ALPHA C3P3CAL1ALPHA C4P4CAL1ALPHA
#                ........
# (hundreds of variable name lines omitted here)
#                ........
T3T5CTH5BETA T4T6CTH5BETA T5T6CTH5BETA);
}
sub getdata {
     @datalines = ("-4.48 -4.14 -3.05
#               ........
# (thousands of data lines omitted here)
#               ........
");
}
```

Grab-srs.pl. Grab-srs.pl is invoked above by backticks-escape to command line in *avsrandom.pl*.

```
#!/usr/perl-5.6/bin/perl5.6.0
# grab-srs.pl, By Jon Frederick, 8/16/01
# extracts sign-rank statistics and varnames
# from parse-sas-univariate output
while(<>)      {
     s/FP/F/g; # lexicor/vbmapper naming convention
     @line = split;
     $stimtype=”BETA”;
     $stimtype=”ALPHA” if($line[2]=~/ALPHA/);
     @a=split//,$line[2]; $freq=”$a[5]$a[6]”;
     $time=”$a[7]”; $pair=”$a[0]$a[1]$a[2]$a[3]”;
     print “$stimtype $freq $pair $time $line[4]\n” if
($line[6]<= .01000) ;
}
```

Countpervariable.pl. Countpervariable.pl is a script invoked above by backticks-escape to command line in *avsrandom.pl*. Reads output from *grab-srs.pl*.

```
#!/usr/perl-5.6/bin/perl5.6.0
# countpervariable.pl, By Jon Frederick, 8/16/01
# counts the number of significant
# univariate tests observed for each variable
@allvars=qw(ALPHA BETA D1 D2 TH AL B1 B2 1 2 3 4 5 F1O1 O1C3
O1F3 O1F7 O1P3 O1T3 O1T5 F2O2 O2C4 O2F4 O2F8 O2P4 O2T4 O2T6
CZPZ F1PZ F2PZ FZPZ PZF3 PZF4 F1C3 F1CZ F1F3 F1FZ F1P3 F1T5
F2C4 F2CZ F2F4 F2FZ F2P4 F2T6 F1F7 F1T3 F2F8 F2T4 F1F2 F7F8
F3F4 O1O2 T5T6 P3P4 T3T4 C3C4 C3P3 C4P4 F3C3 F3P3 F3T5 F4C4
F4P4 F4T6 F7P3 F7T3 F7T5 F8P4 F8T4 F8T6 FZCZ FZP3 FZP4 T3T5
T4T6);
while(<>) { chomp; push(@lines,$_);}
foreach $var (@allvars)  {
     foreach $line (@lines)   {
          if ($line=~/\s$var\s/ || $line=~/^$var\s/ )  {
               @thisline=split/\s+/,$line;
               $pos{$var}++ if ($thisline[4]=~/\+/);
               $neg{$var}++ if ($thisline[4]=~/\-/);
          }
     }
     $pos{$var} = 0 unless (defined $pos{$var});
     $neg{$var} = 0 unless (defined $neg{$var});
     if ($pos{$var}>$neg{$var})    {
          push(@posvars,"$var $pos{$var} $neg{$var}\n")
     } elsif ($pos{$var}<$neg{$var})    {
          push(@negvars,"$var $pos{$var} $neg{$var}\n")
     } elsif ($pos{$var}==0 && $neg{$var}==0){
          push(@zerovars,"$var $pos{$var} $neg{$var}\n");
     } else {
          push(@equalvars,"$var $pos{$var} $neg{$var}\n");
     }
}
print foreach (@posvars);
print "\n";
print foreach (@negvars);
```

Avsedf.pl. Compares real data file to the empirical distribution function created by *countpervariable.pl*.

```
#!/soft/script/bin/perl
# avsedf.pl, By Jon Frederick, 8/16/01
# finds a p value for each line of a real data
# file by comparing it to an empirical distribution
# function which it generates from a file containing
# randomized trials.
open(REAL,"<$ARGV[0]");
open(RAND,"<$ARGV[1]");
while(<RAND>)  { chomp; push(@lines,$_); }
print "varrule pos  neg  mystat    p    min  max\n";
while(<REAL>)  {
     chomp;
     ($var,$rule,$pos,$neg,$mystat)=(split/\s+/,$_);
#    next if ($var=~/ALL/);
     @varrule=(); foreach $line (@lines){

     ($var2,$rule2,$pos2,$neg2,$mystat2)=(split/\s+/,$line);
```

```
          if ($var=~/^$var2$/&&$rule=~/^$rule2$/) {
               $mystat2=(int($mystat2*1000))/1000;
               push(@varrule,"$mystat2 $var2 $rule2 $pos2
$neg2");
          }
     }
     @ordered=(); $rank=1000;
     foreach $stat (sort {$a<=>$b} @varrule) {
          $p=$rank/1000; $p = (int($p*100000))/100000;
          push(@ordered,"$p $stat"); $rank--;
     }
# @ordered is now the edf lookup table in ascending values
# of $mystat2 and descending values of p.
     ($p,$minmystat2) = (split/\s+/,$ordered[0])[0,1];
     if ($mystat<$minmystat2) {
          $myp = 1; $mymax=$minmystat2; $mymin=0;
     }
     if ($mystat==$minmystat2){
          $pmin = $p; $mymin=$minmystat2;
          $onlyonce=0;
          foreach $thing (@ordered)       {

($mystat2,$p)=(split/\s+/,$thing)[0,1];
               if($mystat>$mystat2)    {
                    if($onlyonce<1) {
                         $pmax = $p; $onlyonce++;
                         $mymax = $mystat2;
                         }
               }
          }
          $myp = ($pmin + $pmax)/2;
     }

     ($p,$maxmystat2) = (split/\s+/,$ordered[-1])[0,1];
     if ($mystat>$maxmystat2) {
           $myp = $p; $mymax=$maxmystat2;
$mymin=$maxmystat2;
     }
     if ($mystat==$maxmystat2){
          $pmin = $p;
          foreach $thing (@ordered)       {
               ($p,$mystat2)=(split/\s+/,$thing)[0,1];
               if ($mystat>$mystat2)   {
     # this will grab the last $p before $mystat = $mystat2
                    $pmax = $p;
                    $mymax=$mystat2;
               }
          }
          $myp = ($pmin + $pmax)/2;
     }
```

```
    if (($minmystat2<$mystat) && ($mystat < $maxmystat2))
    {
         $onlyonce=0;
         foreach $thing (@ordered){
              ($p,$mystat2)=(split/\s+/,$thing)[0,1];
              if ($mystat>$mystat2)    {
                   $pmax = $p;
                   $mymax=$mystat2;
              }
              if($mystat<$mystat2){
                   if($onlyonce<1){
                   # grabs p for the first
                        $pmin = $p; $onlyonce++;
                        $mymin = $mystat2;
                   }
              }
         }
         $myp = ($pmax + $pmin)/2;
    }
    $mystat = (int($mystat*1000))/1000; # 3 sig figs.
    $myp = (int($myp*1000))/1000;
    $_="$var  rule$rule $pos $neg      $mystat";
    print "$_ $myp $mymin    $mymax\n";
}
```

The probability of 241/3780 false-positive signed rank tests at p<=.01 arising by chance was thereby determined to be p<.001. Similar comparisons demonstrated that a significant number of positive tests had been observed in each stimulation type, frequency, and time, with the exception of the post-stimulation baseline (table I).

When random EDFs were generated for each of the 19 electrode locations (entering the name of each of the 19 standard 10-20 locations for `@allvars` in *countpervariable.pl*), the number of positive tests was significant at p<.01 for most, and at p<.05 for all 19 locations. This analysis produced an unexpected finding: 238 effects involved the right side of the brain while only 173 involved the left side. This pattern was true for each homologous pair (F1 vs. F2, F3 vs F4, etc.) except the temporal leads, which were roughly equal. It was then of interest to see how the 63 coherences contributed to this pattern. Seventeen of these comparisons were significant at p<.01 (29 at p<.05). Table I also reports the number of increased vs. decreased coherences observed for each variable. The distribution of the positive vs. negative effects reflected the general findings of the ANOVAs: B1 and D2 had more positive changes than D1, and the first ten minutes had a higher ratio of positive-to-negative changes than the last ten minutes. Alpha and beta stimulation evoked roughly the same ratio of positive-to-negative changes, although alpha stimulation evoked a greater total number of changes. The anatomical distribution of this pattern became clear when plotted on graphical heads, as shown in figures 1 and 2 (plotted with *Vbmapper.pl*; Frederick, 2001; most recent version presented in chapter 4). With few exceptions, the decreasing coherences were longitudinal projections from the occipital leads and Pz. The increasing coherences were always associated with the frontal poles, the interhemispheric pairings, and all the remaining longitudinal projections that did not involve O1, O2, or Pz. An interesting exception was that 4/4 of the significant frontal interhemispheric coherences (FP1FP2 and F7F8) decreased. This tendency toward decreasing frontal interhemispheric coherences became even more apparent at p<=.05. We found that the findings at p<=.05 showed a high degree of consistency with those at p<=.01, so these are also represented (with lighter, less saturated lines) in figures 1 and 2. Most remarkable was the observation that individual coherence pairing locations generally changed in only one direction, regardless of frequency, time, or type of stimulation. For example, the main effect of time appeared to be mediated by the decreasing number of increased frontal intrahemispheric coherences (from 25 in the first five minutes to 8 in the last five minutes), along with the increasing number of decreased posterior intrahemispheric coherences (from 18 to 26).

Discussion

This study has demonstrated a number of anatomical asymmetries in the coherence effects of AVS. AVS decreased intrahemispheric coherences involving the posterior leads Pz, O1 and O2, but increased intrahemispheric coherence frontally and centrally. Meanwhile, the interhemispheric derivations increased posteriorly and at high frequencies, and tended to decrease frontally and at low frequencies.

The overwhelming tendency of all the significant effects for a given electrode pair to go in the same direction independently of frequency, time, or type of stimulation, is another striking asymmetry. The segregation of these positive and negative changes into discrete anatomical compartments suggests that descriptions of "coherence" that omit discussion of location are somewhat oversimplified. These results also demonstrate that the popular naming and marketing of AVS devices as "brain wave synchronizers" is also oversimplified. Indeed, overall coherence in the alpha band was significantly lower under alpha AVS, compared to the beta AVS condition.

The statistical analysis of these data has taken a new approach to the problem of type I error from multiple comparisons. Many previous studies (e.g., Dafters, Duffy, O'Donnell, and Bouqet, 1999; Wada et al., 1998b) have avoided this problem by "not looking" at more than a small representative subset (6 to 10) of the 171 coherences that are possible among 19 electrodes. We believe that this approach has unacceptable consequences for type II error. Limitations to the Stellate Rhythm software made it impractical for us to look at more than 64 coherence pairs, but at this level of resolution, it is apparent that groups of coherences (e.g. frontal longitudinal) can change reliably while individual members of that group might not. Other studies have simply performed hundreds of univariate tests and relied on the intuitive clarity of the overall magnitude and scope of the effect (e.g., Gevins, Morgan, Bressler, Cutillo, White, Illes, Greer, Doyle, and Zeitlin, 1987). We believe that an optimal balance between type I and type II error is achieved not by avoiding or ignoring the problem of multiple comparisons, but by measuring it. When the problem of experimentwise error is framed as a null hypothesis, that "these significant univariate tests arise from random effects," clearly the most direct method of testing this hypothesis is to compare the number of observed effects to the empirical distribution of random effects.

The finding of a highly consistent inhibitory effect in the occipital intrahemispheric coherences stands in contrast with two recent studies of photic driving of coherence. Wada et al. compared the photic driving of coherence between Alzheimer patients and 10 normal participants (mean age 59; Wada et al., 1998a); and between schizophrenic patients and 30 normal participants (mean age 22.3; Wada et al., 1998b). Although comparisons with the baseline were not statistically tested, Wada et al. presented graphs comparing the subject groups showing that 5, 10, and 15 Hz photic stimulation (with no auditory stimulus present) increased or tended to increase all intrahemispheric coherences from the baseline in normal subjects. These included O1P3, O2P4, O1T5, and O2T6 in the first study, and O1C3 and O2C4 in the second study. The observation of significant desynchronizing effects in all of these pairings in the present study suggests that combining auditory with visual stimulation at the same frequency might have an interfering effect rather than a synergistic one, possibly owing to transmission time differences in the auditory and visual systems. We previously reported (Frederick et al., 1999) that combined 18.5 Hz AVS tended to produce a smaller effect on 16-20 Hz amplitude than auditory or visual stimulation alone. Such a view might also explain the lack of a main effect of combined alpha AVS on alpha power reported by Timmermann et al. (1999). The follow-up study in chapter 4 was proposed to directly test the differences between unimodal vs. combined AVS in the full 19-channel montage.

Chapter Four

Effects of auditory stimulation on the photic driving response

Background

Since previous studies had shown that alpha bandpass effects are strongest when the SML stimulus approximated the subject's dominant alpha frequency (Van Der Tweel & Verduyn Lunel, 1965; Townsend et al., 1975), the lack of effect of alpha AVS on the alpha band observed by Timmermann et al. (1999) was unexpected. Timmermann (1998) also replicated this lack of effect in a study of 10 new subjects. Rosenfeld et al. (1997) had also failed to observe a significant alpha driving effect of a 10 Hz stimulus in 9 subjects. Only when subjects were divided into low and high baseline alpha groups did a pattern become evident (low baseline alpha subjects more reliably showed the alpha driving effect). Kawaguchi et al. (1993) also failed to observe an alpha PDR in 6 out of 12 subjects. Nonetheless, since both Rosenfeld et al. and Kawaguchi et al. used a square-wave light stimulus, the lack of an alpha PDR to sinusoidally modulated light and sound in Timmermann et al. (1999) and Timmermann (1998), remains inconsistent with the findings of Townsend et al. and Van der Tweel & Verduyn Lunel, who used SML. Given that the only apparent difference among these studies was the addition of the auditory stimulus to SML in Timmermann et al. (1999) and Timmermann (1998), it was hypothesized for this dissertation that the auditory stimulus had an attenuating effect on the normal photic driving response. A trend toward such an effect had also been observed by Frederick et al. (1999) when comparing 18.5 Hz combined sinusoidal AVS with auditory or visual stimulation alone. The prominent desynchronizing effects of AVS on posterior longitudinal coherence (chapter 3), particularly those coherences between the occipital and temporal regions, lent further convergent support to this theory. Therefore, the following study compared the effects of visual stimulation alone, auditory stimulation alone, and combined auditory and visual stimulation. Given the diverse effects of AVS in terms of location and frequency (Brauchli et al., 1995; Deiter and Weinstein, 1995; Timmermann et al., 1999), the EEG was examined among all 19 channels and in 7 frequency bands from 0.75-32 Hz. It was predicted that SML alone at the dominant alpha frequency would increase alpha amplitude and coherence generally across the scalp, with effects being strongest or most reliable over the primary visual areas (near O1 and O2). It was predicted that auditory alpha stimulation alone would increase amplitude and coherence over the primary auditory areas (near T3 and T4). Since Frederick et al. (1999) had observed a significant increase in 18.5 Hz amplitude at Cz with an 18.5 Hz auditory stimulus, it was predicted that the auditory effects at T3 and T4 would generalize beyond the primary auditory areas, but no predictions were made other than that the effect was likely to be an increase. Combined auditory and visual stimulation was expected to replicate the coherence results obtained in chapter 3, and the amplitude results obtained by Timmermann.

Materials and Methods

Thirty participants (13 male, 17 female) were recruited from the undergraduate and graduate populations of the University of Tennessee, Knoxville. Participants ranged in age from 18-29 years, with a mean of 21 years. All participants were right-handed, and reported no previous history of neurological disorders or current diagnosis of mental illness during personal screening interviews. Participants reported that they were free of medication use during the study.

Audio-visual stimulation was provided by a Polysync Pro (Synetic Systems) device. This unit consisted of headphones and a pair of “photoscopic” glasses that were connected to a small, portable unit that was programmed to provide specified levels of visual and auditory stimulation. The glasses had eight light emitting diodes (LEDs), four per side, arranged in a cross pattern. The LEDs were situated approximately 1.5 cm from the eyes, and emitted red light at 50 percent of the maximum setting. Audio stimulation consisted of a cycled tone with a pitch of 185 Hz, presented to both ears simultaneously, with a duty cycle of 50 percent and a loudness level of 70 percent of maximum. Both auditory and visual stimulation were sinusoidally modulated.

Participants were seated upright in a plastic chair located in a sound-attenuated, dimly lit room for the EEG recordings. The headphones and glasses were placed over the electrode cap. All participants had an eyes-closed baseline EEG recorded for five minutes at the beginning of the session. This baseline recording was analyzed to determine each subject’s dominant alpha frequency, defined as the peak power between 8 and 12 Hz at locations O1 as measured by power spectral analysis, rounded to the nearest 0.5 Hz (using EEG Editor, NovaTech.com). This frequency was programmed into the PolySync Pro, and participants were provided with three five-minute stimulation sessions, consisting of (a) auditory stimulation alone; (b) visual stimulation alone; (c) combined auditory and visual stimulation, where the order of the conditions was randomly assigned in a counterbalanced manner. A four minute eyes-closed post-stimulation baseline was recorded after each stimulation condition. There were, thus, seven recordings: an eyes-closed baseline, three stimulation conditions, and three post-stimulation baselines.

A quantitative referential EEG (monopolar montage) was recorded from 19 electrodes following the International 10-20 system for electrode placement, with linked earlobe references. All electrode impedances were below 5 KOhm. Recordings were made using an electrode cap (Electro Cap Inc.). EEG was recorded at 128 samples/sec with a Lexicor Neurosearch 24 analog to digital system. Data were stored on a Pentium 120 MHz computer. Using V41E software (Lexicor), eye blinks, large eye movements, and all observable muscle artifacts were removed by a visual inspection of the EEG records. Amplitude values were calculated with Lexicor's Exporter software (Lexicor). Spectral correlation, a "coherence-like" measure, was also calculated by Exporter, but was later found to be very different from coherence under certain conditions (see Frederick and Lubar, 2001). Thus, a custom algorithm to calculate coherence was contributed by David Joffe (Lexicor). Both Exporter and the custom algorithm processed digital EEG by Fast Fourier Transformation with cosine tapering and a Hanning window.

Amplitude and coherence values were calculated for the following band passes: delta 1 (1-2 Hz), delta 2 (2-4 Hz), theta (4-8 Hz), alpha (8-12 Hz), beta 1 (13-21Hz), and beta 2 (21-32 Hz activity), and for peak alpha, a 1.5 Hz band surrounding each subject's dominant alpha frequency.

Lexicor Exporter files, by default, are comma-delimited and will include successive frequency variables on successive lines. There is, however, an option to print all variables on one line, one epoch per line, in space-delimited format (recommended). From the Exporter home directory, enter

```
exporter ?
```

at the DOS prompt for details.

*Coherence1.pl***.** I was fortunate to have the collaboration and support of David Joffe, the chief software engineer at Lexicor, who wrote a custom algorithm that computes a real measure of coherence. A comparison of spectral correlation vs. coherence results in this experiment has recently been submitted (Frederick and Lubar, 2001), but is beyond the scope of this dissertation. Joffe provided a program called *compare.exe*, (source code unavailable) which calculates both coherence and spectral correlation for each electrode pair. It takes four arguments at the command line, i.e.,

```
compare.exe inputfile outputfile lowest highest
```

where `inputfile` is a Lexicor ".dat" (proprietary binary format) file, `outputfile` saves the results, and `lowest` and `highest` are numbers demarcating the upper and lower limits of the bandpass to be evaluated. I found that the standard output of the program (printed to the screen during the process) contained all the information that I needed, so I just overwrote the output file with each iteration. My Perl script created an output file by altering the name of each input file (see source code below). The standard output of `compare.exe` appeared as follows:

```
jf08fb3.dat  SRATE:128  EPS_SAVED:58 FGAIN:   32000
F3 FP1  COH:0.876184 SCC:0.883730
C3 FP1  COH:0.446091 SCC:0.646432
```

```
C3 F3  COH:0.725637 SCC:0.804673
P3 FP1  COH:0.107092 SCC:0.455569
P3 F3  COH:0.255948 SCC:0.539272
…<lines omitted>...
EPOCHS PROCESSED:58
```

where the first line states the name of the input file, the sampling rate, the number of unrejected epochs, and the amplifier gain. Each subsequent line contains the electrode pair names, the coherence, and then the spectral correlation. The final line of output repeats how many epochs were processed. Since this output was returned in column format, the following perl script, *coherence1.pl*, extracted the coherences and wrote them to an output file in row format.

```
#!/usr/bin/perl
# coherence1.pl, By Jon Frederick, 8/16/01
opendir(DIR, “.”) or die “can’t opendir: $!”;
while (defined($file = readdir(DIR))) {
     if ($file=~/JF.*DAT/)    {
          push(@files,$file)
     }
}
@files = sort @files;
closedir(DIR);
@freqs = (“0.75 2”, “2 4”, “4 8”, “8 12”, “12 21”, “21 31”);
foreach $file (@files)   {
     $file =~ s/\.DAT//;
     foreach $freq (@freqs)   {
          @data = ‘compare.exe $file temp $freq‘;
          print “calculating $freq coherence for $file\n”;
          $thisline=0; foreach $line (@data) {
               chomp $line;
               if ($thisline < 2 ) {
                    ($f1, $f2)=(split/\s+/,$line)[1,2];
                    $thisline++;
                    next;
               }
               unless ($line=~/EPOCHS/) {
                    ($e1,$e2,$coh)=
                         (split/[:\s+]/,$line)[0,1,4];
                    $freq2 = $freq;
                    $freq2=~s/0\.75 2/D1/;
                    $freq2=~s/2 4/D2/;
                    $freq2=~s/4 8/TH/;
                    $freq2=~s/8 12/AL/;
                    $freq2=~s/12 21/B1/;
                    $freq2=~s/21 31/B2/;
                    push(@vars,"$e1$e2$freq2");
                    push(@cohs,$coh);
               }
          }
     }
     $file =~ s/$/\.COH/; open (OUT,">$file");
     print OUT "@vars\n@cohs";
     @vars=(); @cohs=();
}
```

*Coherence2.pl***.** The code above was completed in an enthusiastic rush to see results. However, it might have been more efficient to implement code for running compare.exe at the dominant alpha frequency (below) along with the six standard frequencies computed above. Thus, *coherence2.pl* picks up the slack, running *compare.exe* to compute dominant alpha, just as *coherence1.pl* runs *compare.exe* to compute coherences for the standard six frequency bands. The dominant alpha bands were constructed in Microsoft Excel from a two-column matrix containing each subject and their peak alpha frequency (PAF). Since PAF is defined the highest 0.5 Hz bin between 8 and 13 Hz (e.g., a PAF of 10 actually refers to the band between 10 and 10.5 Hz with a true mean of 10.25), a 0.75 Hz band around the true mean was constructed by subtracting 0.5 from PA for the lower limit and adding 1.0 for the upper limit. The PAF column was then deleted and the file saved as tab-delimited text, and opened in the Unix text editor vi. Vi provides search and replace functions more powerful in some respects than Microsoft Word's EDIT:REPLACE function. By searching and replacing line returns, commas and quotation marks were added to create the associative array, `%das`, in the code below. (Note that `%das` contains a table of the dominant alpha frequencies for all subjects, data which some readers might find useful and is not provided elsewhere in this manuscript.)

```
#!/usr/bin/perl
# coherence2.pl, By Jon Frederick, 8/16/01
# dominant alpha frequencies for all subjects:
%das = (“03”, “7.5 9”, “04”, “10 11.5”, “05”, “9.5 11”,
“06”, “10 11.5”, “07”, “10 11.5”, “09”, “9 10.5”, “10”, “9
10.5”, “11”, “9 10.5”, “12”, “9 10.5”, “13”, “8.5 10”, “14”,
“9.5 11”, “15”, “10 11.5”, “16”, “8 9.5”, “17”, “10 11.5”,
“18”, “11.5 13”, “19”, “10.5 12”, “20”, “11.5 13”, “22”,
“10.5 12”, “23”, “8.5 10”, “24”, “9.5 11”, “25”, “11.5 13”,
“26”, “9.5 11”, “27”, “11.5 13”, “28”, “8 9.5”, “29”, “10.5
12”, “30”, “10.5 12”, “31”, “9.5 11”, “32”, “10 11.5”, “33”,
“10.5 12”, “34”, “9.5 11”);
opendir(DIR, “.”) or die “can’t opendir: $!”;
while (defined($file = readdir(DIR))) {
     if ($file=~/JF.*DAT/)    {
          push(@files,$file)
     }
}
@files = sort @files;
closedir(DIR);
foreach $file (@files)   {
     $subnumb = $file; $subnumb=~s/JF//;
     $subnumb=~s/(..).*/\1/;
     $file =~ s/\.DAT//;
#    foreach $freq (@freqs)   {
          $freq = $das{$subnumb};
          @data = ‘compare.exe $file temp $freq‘;
          print “calculating $freq coherence for $file\n”;
          $thisline=0; foreach $line (@data) {
               chomp $line;
               if ($thisline < 2 ) {
                    ($f1, $f2)=(split/\s+/,$line)[1,2];
                    $thisline++;
                    next;
               }
               unless ($line=~/EPOCHS/) {
                    ($e1,$e2,$coh)=
                         (split/[:\s+]/,$line)[0,1,4];
                    $freq2 = $freq;
                    $freq2=~s/0\.75 2/D1/;
                    $freq2=~s/2 4/D2/;
                    $freq2=~s/4 8/TH/;
                    $freq2=~s/8 12/AL/;
                    $freq2=~s/12 21/B1/;
                    $freq2=~s/21 31/B2/;
                    push(@vars,”$e1$e2$freq2”);
                    push(@cohs,$coh);
               }
          }
     #}
     $file =~ s/$/\.COH/; open (OUT,”>$file”);
     print OUT “@vars\n@cohs”;
     @vars=(); @cohs=();
}
```

Colcount.pl. For troubleshooting or debugging, it was often necessary or prudent to count the number of data rows and columns in a space-delimited text file. The standard unix word count command,

```
wc filename
```

returns the number of rows. The following script was written to count the number of words (or columns) on each line.

```
#!/usr/bin/perl
# colcount.pl, By Jon Frederick, 8/16/01
# counts number of columns in each row

$file = “$ARGV[0]”;
while (<>) {
     @array = split;
     $count = @array;
     print “$count “;
}
print “\n”;
```

Catcols.pl. Since it is more efficient to analyze all coherence and amplitude data at the same time from the same files, the following script was written to concatenate the coherence and amplitude files together, as separate columns in the same document.

```
#!/usr/bin/perl
# catcols.pl, By Jon Frederick, 8/16/01
# takes two files containing different
# variables for the same set of observations (same time
# series within subjects or same subjects between
# subjects), and merges them. files have identical names,
# ending in “.EXP”, but are located in different
# directories.
$dir1 = “amplitudes”; $dir2 = “coherences”;
$files = ‘ls $dir1/*EXP‘; $files =~ s/$dir1\///g;
@files = split /\n/, $files;
foreach $file (@files)   {
     $x++; print “$x:    $file\n”;
     open(F1,”<$dir1/$file”) or die “couldn’t open
$dir1/$file!\n”;
```

```
     $file =~ s/M\.EXP/C\.EXP/;
     open(F2,”<$dir2/$file”) or die “couldn’t open
$dir2/$file!\n”;
     $file =~ s/M\.EXP/CM\.EXP/;
     open(OUT,”>$file”);
     while(($a=<F1>) && ($b =<F2>)){
          chomp $a;
          print OUT “$a $b”;
     }
}
```

Outliers.pl. Having column-appended all the analytic measures for each epoch for each subject to the same files, it was the necessary to select a measure of central tendency for each subject across all epochs. The mean is commonly derived and reported by popular software packages for this purpose, without first testing for the existence of outliers, which can significantly shift the mean. One method of testing for outliers is to measure for observations greater than three standard deviations from the mean. However, the standard deviation is computed from the uncorrected mean, outliers can result in large standard deviations and thus a failure to detect outliers. *Outliers.pl* tests for observations greater than $x mean absolute deviations (MADs) from the median, a measure of deviation which is not sensitive to the effect it is intended to measure. $x is defined by the user at the command line, usually ranging from 1 to 7 (*Outliers.pl* first transposes the columns of the input file into rows by invoking another script called *trans.pl*, which is described next.)

```
#!/usr/bin/perl
# outliers.pl, By Jon Frederick, 8/16/01
# for each column of data, replaces each data
# point with a dot ‘.’ if greater that 2 abs. devs. from
# median $x is the number of mean absolute deviations
# defining an # outlier, provided by the user $x =
# $ARGV[0];

# transpose all the columns into a bunch of rows.
@samples = ‘trans.pl $ARGV[1]‘;
```

```
# the first @sample is the epoch numbers so they are
# removed:
shift @samples;
foreach $sample (@samples)    {
next if ($x > 0);
     @temp=split /\s+/, $sample;
# THE FOLLOWING ASSUMES FIRST IS VARNAME AND REMOVES IT!
     shift @temp;
     # get rid of missing observations denoted by “.”
     foreach $temp (@temp)    {
          unless ($temp =~ /^\.$/) {
               push (@found, $temp);
          }
     }
     @sorted = sort {$a<=>$b} @found; @found=();
     # print “\@sorted is @sorted\n”;
     $modulus = @sorted % 2;
     # modulus is 0 if @sorted has even, 1 if
     # odd, number of elements
     if ($modulus == 1 ) {
          $index = (@sorted / 2) + 0.5;
          $median = $sorted[$index];
     } elsif ($modulus == 0)  {
          $index = @sorted / 2;
          $median = ($sorted[$index] + $sorted[$index+1])/2;
     }
# now, find the mean absolute deviation from the median
     $sumabsdev = 0;
     foreach $obs (@sorted)   {
          $dev = $obs - $median;
          $absdev = abs $dev;
          $sumabsdev = $sumabsdev + $absdev;
          #print “$absdev “
     }
     #print “\n”;
     #print “sumabsdev is $sumabsdev\n”;
     $mad = $sumabsdev / @sorted;
     $mad = int (1000 * $mad)/1000;
     #print “mad is $mad\n”;
     # replace each $obs with ‘.’ if > $x $mads from $median
     $countoutliers=0;
     foreach $obs (@sorted)   {
          $dev = $obs - $median;
          $absdev = abs $dev;
          if ($absdev > ($x * $mad) )   {
               $obs = ‘.’;
               $countoutliers++;
          }
     }
     $percentout = $countoutliers / @sorted;
     $percentout = int (100 * $percentout)/100;
     push(@revised, “$median $mad $percentout outliers\n”);
     # note, @sorted is in numerical order,
     # not original order of obs.
}
print “@revised”;
```

*Trans.pl***.** Since the standard input method in Unix and Perl reads one row of a file at a time (not one column at a time), to perform mathematical operations on a column of data requires transposing the columns into rows. The following script was written for this purpose.

```
#!/usr/bin/perl
# trans.pl, By Jon Frederick, 8/16/01
# transposes the elements of a matrix.
while (<>) {
     chomp;
     @_ = split;
     push(@array,@_);
     ++$countrows;
}
$countcols = @_ ;
while ($z < $countcols) {
     while ($y < $countrows) {
          print “$array[$x] “;
          $x = $x + $countcols;
          ++$y;
     }
     print “\n”;
$y = 0; ++$z;
$x = $z;
}
```

*Median.pl***.** A large number of outliers was routinely detected using 2, 3, 4, and even 5 mean absolute deviations from the median. Therefore, the mean was considered inaccurate, and the following script was written to extract the median value across all unrejected epochs for each variable for each subject.

```
#!/usr/bin/perl
# median.pl, By Jon Frederick, 8/16/01
# for each column of data, finds the median.
# first, transpose all the columns into a bunch of rows.
@samples = ‘trans.pl $ARGV[0]‘;
foreach $sample (@samples)    {
     @temp=split /\s+/, $sample;
     # ASSUMES FIRST IS VARNAME AND REMOVES IT!
     shift @temp;
     # get rid of missing observations denoted by “.”
     foreach $temp (@temp)    {
#         unless ($temp =~ /^\.$/) {
               push (@found, $temp);
#         }
     }
     @sorted = sort {$a<=>$b} @found; @found=();
     $modulus = @sorted % 2;
     # modulus is 0 if @sorted has even, 1 if odd,
     # number of elements
     if ($modulus == 1 ) {
          $index = (@sorted / 2) + 0.5;
          $median = $sorted[$index];
     } elsif ($modulus == 0)  {
          $index = @sorted / 2;
          $median =($sorted[$index] + $sorted[$index+1])/2;
     }
     print “$median “;
}
print “\n”;
```

Results

This experiment analyzed 1197 coherence variables in each condition, i.e. all 171 combinations of 19 electrodes at delta 1, delta 2, theta, alpha, beta 1, beta 2, and at each subject’s dominant alpha frequency. For amplitude, 133 variables were analyzed, one for each electrode location over each of 7 frequencies. (1197 spectral correlation and 1197 phase variables were also computed, but analysis of these results will be reported elsewhere; see, e.g., Frederick and Lubar, 2001.) There were, for each subject, a total of seven conditions under which EEG was recorded: a pre-stimulation baseline, auditory stimulation alone, visual stimulation alone, combined AVS, and three post-stimulation baselines, each recorded after each of the three stimulation conditions. The median value across all unrejected epochs for each EEG variable was computed for each condition for each subject.

Diffsfiles.pl. Differences from the pre-stimulation baseline median were computed for each of the three stimulation conditions, and each of three post-stimulation baseline, using the following program, *diffsfiles.pl.*

```
#!/usr/bin/perl -w
# diffsfiles.pl, By Jon Frederick, 8/16/01
# find the difference from baseline for
# all recording conditions
@files = ‘ls JF*MDN‘;
foreach $file (@files)   {
     chomp $file;
#    next if ($file =~ /B1/);
     open(F, “<$file”) or die “couldn’t open $file”;
     $base = $file; $base =~ s/..CM/B1CM/;
     open(BASELINE, “$base”) or die “couldn’t open $base”;
     print “comparing $file and $base\n”;
     while (<BASELINE>)  {
          chomp; s/^\s+//; s/\s+/ /;
          @values1=split;
     }
     while (<F>)    {
          chomp; s/^\s+//; s/\s+/ /;
                @values2=split;
     }
     $c=0;@diffs=();
     if (@values1 != @values2){ print “error!”; die; }
     while ($c < @values1)    {
          $diffs[$c] = $values2[$c] - $values1[$c];
          $diffs[$c] = (int(1000*$diffs[$c]))/1000;
          ++$c;
     }
     $out = $file; $out =~ s/MDN/DIF/;
     open(OUT,”>$out”) or die “couldn’t open $out!\n”;
     print OUT “@diffs\n”;
}
```

*Assemble.pl***.** A fairly simple program was then used to concatenate all of the differences from baseline into the same file.

```
#!/usr/bin/perl
# assemble.pl, By Jon Frederick, 8/16/01
# puts all the files with names
# matching $pattern into a single file with the
# file name as the first item on each row.
# gets rid of “.” in filename so SAS won’t bark.
$pattern = $ARGV[0];
```

```
@files=‘ls‘;
foreach $file (@files)   {
     chomp $file;
     if ($file =~/$pattern/)  {
          open(INPUT,”<$file”);
          $file =~ s/\.//; print “$file “;
          while(<INPUT>) {
               print;
          }
     }
}
```

*Writeprocuniv.pl***.** The following code takes a “*.all” K x N data file and a “varnames” file with K variable names as input, and writes a “*.sas” SAS proc univariate code file as output.

```
#!/usr/bin/perl
# writeprocuniv.pl, By Jon Frederick, 8/16/01
# adds varnames, and gets rid of epoch
# columns and writes SAS proc univariate code to *.SAS
# output file.
@files = ‘ls *all‘;
foreach $file (@files)   {
     chomp $file; open(IN,”<$file”);
     open(VAR,”<varnames”); open(TMP,”>temp”);
     while (<VAR>)  { print TMP “$_\n”; }
     while (<IN>)   { print TMP; }
     system(“trans.pl temp|grep -v EPOCH|trans.pl > temp2”);
     open(TMP3,”>temp3”);
     open(TMP2,”<temp2”); while(<TMP2>) {
          if ($x<1) { # first line only
               print TMP3 “data a\; input \n $_\; cards;\n”;
               $x++;
          } else    {
               print TMP3;
          }
     }
     print TMP3 “\n\; run\; proc univariate data\=a; run\;”;
     $sasfile=$file; $sasfile=~s/all/sas/;
     system(“shortlines.pl temp3 > $sasfile”);
}
```

*Parseuniv.pl***.** The following script, an upgraded version of *parse-sas-univariate.pl* extracts the significance of the signed rank statistic from the sas output “*.lst” file, and formats it for input to *vbmapper.pl*.

```
#!/usr/bin/perl
# parseuniv.pl, by Jon Frederick
# extracts sign rank statistics from SAS proc univariate
# output
while(<>) {
     if (/Variable/ && !$seen{$_}) {
          $seen{$_}=1;
          s/\s+Variable/Variable/;
          ($varname)=(split/\s+/,$_)[1];
          print "$varname ";
     }
     if (/\s+Mean/&&/Sum Obs/){
          s/\s+Mean/Mean/;
          @temp=split/\s+/;
          if ($temp[1]<0){
               print "-";
          }
     }
     if (/Signed/)  {
          s/\s+Signed/Signed/;
          @temp=split;
          $temp[7] =~ s/[\<\>]//;
          $certainty = 1 - $temp[7];
          print "$certainty\n";
          # Vbmapper plots .99 rather than .01
     }
}
```

*Vbmapper.pl***.** Translates statistical results into graphical maps.

```
#!/usr/bin/activeperl
# vbmapper.pl, By Jon Frederick, 8/16/01
# a graphical user interface that translates
# an input file containing variable names and t-stats (one
# per line) into Visual Basic code that plots coherence,
# asymmetry, or phase maps on heads in a Microsoft Excel
# worksheet. Upgraded to do power and amplitude maps.

# invocations to make the perl2exe compiler work:
#perl2exe_include Tk.pm
#perl2exe_include Tk/FileSelect.pm

use Tk;
use Tk::Text;

#optional commandline-only support written 12-01-00.
if (!$ARGV[0]) {
     $GUI=1;
} else    {
     open(RC, "<vbmapper.rc");
     while(<RC>)    {
          push(@rclines,$_);
     }
     $minstat= $rclines[0]; $maxstat= $rclines[1];
     $minsatur = $rclines[2]; $maxsatur = $rclines[3];
     $freqlist = $rclines[4]; $file = $ARGV[0];
     # $forcediscrete allows for exactly two levels of
     # saturation
     $forcediscrete=$rclines[5];
     chomp($minstat,$maxstat,$minsatur,$maxsatur,$freqlist,$
forcediscrete);
     & write_vbcode ($minstat, $maxstat, $minsatur,
$maxsatur, $freqlist, $file,$forcediscrete);
}
#----------------------------------------------------
opendir(DIR, “./”) if $GUI;
my @files = readdir(DIR) if $GUI;
@files = sort @files if $GUI;
closedir(DIR) if $GUI;
my $main = new MainWindow  if $GUI;

$frame00 = $main-> Frame()->pack( -side => ‘right’)  if
$GUI;
$frame00->Label(-text => ‘Translate Which File?’)->pack  if
$GUI;
my($listbox_1) = $frame00->Scrolled (“Listbox”,
     -takefocus => ‘1’,
        -height => ‘19’,
        -width => ‘20’,
     -selectmode => “single”    )->pack  if $GUI;
$listbox_1 -> insert(‘end’, @files) if $GUI;
$minstat=1  if $GUI; #default value
$frame0 = $main-> Frame(-width => 70)->pack  if $GUI;
$frame1 = $frame0-> Frame()->pack(-fill=>’both’)  if $GUI;
$frame1 -> Label(-text => ‘Enter Minimum Abs. Value of
Statistic To Be Mapped:’
          )->pack(-side=>’left’)  if $GUI;
$entry_1 = $frame1->Entry( -textvariable => \$minstat,
                -width => ‘6’,
               )->pack(-side=>’right’)   if $GUI;
$maxstat=100 if ($GUI);
$frame2 = $frame0-> Frame()->pack(-fill=>’both’)  if $GUI;
$frame2->Label(-text => ‘Enter Max. Abs. Value of Statistic
To Be Mapped:’
                )->pack( -side => ‘left’)  if $GUI;
$entry_1 = $frame2->Entry( -textvariable => \$maxstat,
                -width => ‘6’,
                        )->pack( -side => ‘right’)  if $GUI;
$minsatur=0.9  if $GUI; #default value
$frame3 = $frame0-> Frame()->pack(-fill=>’both’)  if $GUI;
$frame3->Label(-text => ‘Enter Abs. Value of Statistic for
Zero Saturation of Red/Blue:’
```

```
                )->pack(-side=> ‘left’, -fill=>’both’)  if
$GUI;
$entry_1 = $frame3->Entry( -textvariable => \$minsatur,
                         -width => ‘6’,
                        )->pack( -side=> ‘right’, -fill=>
‘both’)  if $GUI;
$maxsatur=3  if $GUI; #default value
$frame4 = $frame0-> Frame()->pack(-fill=>’both’)  if $GUI;
$frame4->Label(-text => ‘Enter Abs. Values of Statistic for
Maximum Saturation of Red/Blue:’
                )->pack( -side => ‘left’)  if $GUI;
$entry_1= $frame4->Entry( -textvariable => \$maxsatur,
                -width => ‘6’,
                        )->pack( -side => ‘right’)  if $GUI;
# no default value for $forcediscrete
$frame4a = $frame0-> Frame()->pack(-fill=>’both’)  if $GUI;
$frame4a->Label(-text => ‘Enter 0-255 to force one discrete
saturation for absvals < maxstat (192 looks good):’
          )->pack( -side => ‘left’)  if $GUI;
$entry_1= $frame4a->Entry( -textvariable => \$forcediscrete,
               -width => ‘6’,
               )->pack( -side => ‘right’)  if $GUI;

$datatype = “Location data (e.g., Power, Amplitude)” if
$GUI;
$frame4b = $frame0-> Frame()->pack(-fill=>’both’) if $GUI;
foreach(“Location data (e.g., Power, Amplitude)”,
“Relationship data (Coherence, Phase)”) {
     $frame4b -> Radiobutton(-text => $_,
          -value => $_,
               -variable => \$datatype, ) ->pack(-side =>
‘left’) if $GUI;
}

$freqlist=”TH,LA,HA,LB,HB,GA” if $GUI; #default value
$frame5 = $frame0-> Frame()->pack(-fill=>’both’)  if $GUI;
$frame5->Label(-text => ‘Enter comma-delimited list of
frequency variables (Default is displayed):’
                )->pack(-side => ‘left’)  if $GUI;
$entry_1 = $main->Entry( -textvariable => \$freqlist,
                         -width => ‘40’, )->pack  if $GUI;
$main->Button(-text => ‘RUN’,
          -command => sub {&write_vbcode_if_file if $GUI;} )
-> pack  if $GUI; my($text_1) = $main->Scrolled (“Text”, -
scrollbars => “se”,
          -height => ‘9’,
          -width  => ‘59’,
     )-> pack (-side => ‘bottom’)  if $GUI;
MainLoop  if $GUI;
#---------------------------
sub write_vbcode_if_file {
     $file = $listbox_1 -> curselection() if $GUI;
     if ($file){
          & write_vbcode if $GUI;
     } else {
          $text_1->delete(‘1.0’,’end’) if $GUI;
          $text_1->insert(‘end’,”You must select a file
before clicking Run!\n”) if $GUI;
     }
} #end sub write_vbcode_if_file
#--------------------------
sub write_vbcode
{
$file = $listbox_1 -> curselection() if $GUI;
$file = $files[$file] if $GUI;
open(F,”<$file”) or die “could not open $file\n”;
$file=~s/[\.\-\_]//g;  # get rid of nonalphanumeric stuff in
filename
my @line;
$text_1 -> delete(‘1.0’,’end’) if $GUI;
$text_1 -> insert(‘end’, “\’ code for $file\n”) if $GUI;
my @freqs=split /\,/, $freqlist;
my ($BG,$RG,$abstest,$one,$two,$three,$freq);
while(<F>)   {# put all lines with freq variables in a
     s/FP/F/g;# list, adding pair naming convention the
     chomp;   # frequency name to the beginning of the line.
     foreach $freq (@freqs)   {
          if (/$freq\b/) {
     # $freq\b means match at end-of-word only
               push(@list,”$freq $_”);
          } # this gets rid of the assumption that the input
     }      # file only contains lines with relevant data.
} close F;    # drop that filehandle out of memory
foreach $freq (@freqs)   {
     @eachfreq = grep {/$freq/} @list;
     #do this before you change 1s&2s
     $freq=~s/1/ONE/; $freq=~s/2/TWO/; $freq=~s/3/THREE/;
     # get rid of D1, B2 etc. for legal Excel VB sub names
     $text_1 -> insert(‘end’, “sub $freq\(\)\n”) if $GUI;
     print “sub $freq\_$file\(\)\n” if (!$GUI);
# since Excel macros are limited to about 400 lines of code,
# a separate macro or “sub” must be made for each frequency
# (max 171 pairs or 342 lines).
     #put all instances of the frequency into a second list
     foreach (@eachfreq) {
          ($pair,$stat) = (split /\s+/, $_)[1,2];
          $absstat = abs($stat);
          push(@unsorted,”$absstat:$stat:$pair”);
     }
     @sorted = sort { $a <=> $b } @unsorted;  @unsorted=();
     foreach(@sorted)    {
          & drawlines_on_head(\$_,\$forcediscrete);
     }
     $text_1 -> insert(‘end’, “Application.Run
\”\’1020.xls\’\!moveheads2\”\n”) if $GUI;
     print “Application.Run \”\’1020.xls\’\!moveheads2\”\n”
if(!$GUI);
     $text_1 -> insert(‘end’, “End Sub\n”) if $GUI;
     print “End Sub\n” if (!$GUI);
} # end writing code foreach frequency
@list = ();
# make it possible to run another file without restarting
# create the main macro that runs all six frequencies
$text_1 -> insert(‘end’, “sub aaaamap\(\)\n”) if $GUI;
print “sub aaaamap_$file\(\)\n” if (!$GUI);
my $frequency;
foreach $frequency (@freqs)   {
     $text_1 -> insert(‘end’, “Application.Run
\”\’1020.xls’!$frequency\”\n”) if $GUI;
     print “Application.Run
\”\’1020.xls’!$frequency\_$file\”\n” if(!$GUI);
}
$text_1 -> insert(‘end’, “Sheets\(\”Sheet4\”\).Select\n”) if
$GUI;
print “Sheets\(\”Sheet4\”\).Select\n” if (!$GUI);
$text_1 -> insert(‘end’, “End Sub\n”) if $GUI;
print “End Sub\n” if (!$GUI);
$code = $text_1 -> get(‘1.0’,’end’) if $GUI;
$text_1 -> clipboardClear if $GUI;
$text_1 -> clipboardAppend($code) if $GUI;
$text_1 -> delete(‘1.0’,’end’) if $GUI;
$text_1 -> insert(‘end’, “Code for drawing head maps for the
file \”$file\”\nhas been copied to the clipboard and is
ready to paste \ninto the Excel Visual Basic Editor.\n”) if
$GUI;
} # end sub write_vbcode
#------------------------------------
sub drawlines_on_head    {
     ($absstat,$stat,$pairing) = split(/:/,$_);
     if ($stat >= $minstat && $stat < $maxsatur)  {
          if ($forcediscrete) {
               $BG=$forcediscrete;
          } else    {
               $BG = int(($stat -$maxsatur)*(-
255/($maxsatur-$minsatur))); # BG is saturation of red
          }
          # 0 (pure red) if $maxsatur stderrs; 255 (white)
if stat is 0
     } elsif ($stat>$maxsatur){
          $BG = 0;
     } elsif ($stat<(-1*$minstat) && $stat>-1*$maxsatur )
     {
          if ($forcediscrete) {
               $RG=$forcediscrete;
          } else    {
               $RG = int(($stat+$maxsatur)*(255/($maxsatur-
$minsatur))); #RG is saturation of blue
          }
     } elsif ($stat<-1*$maxsatur)  {
          $RG = 0;
     }
     # make pairing the first four characters
        ($one,$two,$three,$four) = (split //,
$pairing)[0,1,2,3];
     $pairing = “$one$two$three$four”;
     if($datatype=~/Location/){
          $pairing = “$one$two”;
     }
     if ($datatype=~/Relationship/){
          if ($absstat>$minstat && $absstat<$maxstat) {
               $pairing=~s/C3/CTHREE/;
$pairing=~s/C4/CFOUR/;
               # C3 & C4 not allowed in Excel macro names
               $text_1 -> insert(‘end’, “Application.Run
\”\’1020.xls\’\!$pairing\”\n”) if $GUI;
               print “Application.Run
\”\’1020.xls\’\!$pairing\”\n” if (!$GUI);
               if ($stat>$minstat && $stat<$maxstat)   {
                    $text_1 -> insert(‘end’,
“Selection.ShapeRange.Line.ForeColor.RGB = RGB\(255\, $BG\,
$BG\)\n”) if $GUI;
                    print
“Selection.ShapeRange.Line.ForeColor.RGB = RGB\(255\, $BG\,
$BG\)\n” if (!$GUI);
               } elsif   ($stat<(-1*$minstat & $stat>-
1*$maxstat))   {
                    $text_1 -> insert(‘end’,
“Selection.ShapeRange.Line.ForeColor.RGB = RGB\($RG\, $RG\,
255\)\n”) if $GUI;
                    print
“Selection.ShapeRange.Line.ForeColor.RGB = RGB\($RG\, $RG\,
255\)\n” if (!$GUI);
               }
          }
     } elsif ($datatype=~/Location/)    {
     # support for location data added 06-10-01
# note $pairing is a misnomer here, technically.
          if ($absstat>$minstat && $absstat<$maxstat) {
               $pairing=~s/1/one/; $pairing=~s/2/two/;
               $pairing=~s/3/three/; $pairing=~s/4/four/;
               $pairing=~s/5/five/; $pairing=~s/6/six/;
               $pairing=~s/7/seven/; $pairing=~s/8/eight/;
               # allowable Excel macro names
```

```
                $text_1 -> insert('end', "Application.Run
\"\'1020.xls\'\!$pairing\"\n") if $GUI;
                print "Application.Run
\"\'1020.xls\'\!$pairing\"\n" if (!$GUI);
                if ($stat>$minstat && $stat<$maxstat)   {
                     $text_1 -> insert('end',
"Selection.ShapeRange.Fill.ForeColor.RGB = RGB\(255\, $BG\,
$BG\)\nSelection.ShapeRange.Fill.Solid\n") if $GUI;
                     print
"Selection.ShapeRange.Fill.ForeColor.RGB = RGB\(255\, $BG\,
$BG\)\nSelection.ShapeRange.Fill.Solid\n" if (!$GUI);
                } elsif   ($stat<(-1*$minstat & $stat>-
1*$maxstat))   {
                     $text_1 -> insert('end',
"Selection.ShapeRange.Fill.ForeColor.RGB = RGB\($RG\, $RG\,
255\)\nSelection.ShapeRange.Fill.Solid\n") if $GUI;
                     print
"Selection.ShapeRange.Fill.ForeColor.RGB = RGB\($RG\, $RG\,
255\)\nSelection.ShapeRange.Fill.Solid\n" if (!$GUI);
                }
           }
      }
} # end sub drawlines_on_head
```

Figure 3 shows the significant signed rank tests for differences in EEG amplitude from the pre-stimulation baselines ($p<.05$, light circles, and $p<.01$, dark circles; two-tailed). Very few significant signed rank tests (consistent with type I error) were observed during auditory stimulation or during the post-stimulation baselines. In the auditory alone condition, only 2/133 effects or 1.5% were univariately significant (at $p<.01$), while in the post-stimulation baselines, fewer than 1% were significant. However, during the visual and combined conditions, widespread, significant amplitude increases were observed in the dominant alpha, 8-12 Hz, 13-21 Hz, and 21-32 Hz bands. In the visual alone condition, 46/133 or 36% of the comparisons were significant. In the combined AVS condition 40/133 or 31% of the comparisons were significant.

Figure 4 shows significant signed rank tests ($p<.05$; light lines, and $p<.01$, dark lines; two-tailed) for the effects of stimulation, compared to baseline, on EEG coherence. Note that *vbmapper.pl* plots coherences in order of significance, so that more significant findings appear in the foreground. As was observed with EEG amplitude, EEG coherence effects in the auditory alone condition and the three post-stimulation baseline occurred at a level consistent with type I error. In the auditory condition,13/1197 variables, or 1.1%, were significant at p<.01. In the post-stimulation baselines, fewer than 1% were significant. However, a pattern of univariately significant decreases in the 0.75-2.0 Hz band was observed during auditory stimulation and the post-stimulation baselines, which was similar to 0.75-2.0 Hz effects observed in the visual and combined AVS conditions. Meanwhile, the visual and combined stimulation conditions increased 8-12 Hz and DA coherence frontally, interhemispherically, and in the long range longitudinal projections. Interestingly, prominent decreases in coherence projections from Pz were observed in 8-12 Hz which were not observed, or observed at much lower levels, at the dominant alpha frequency. Meanwhile, short-range temporal and parietal projections from O1 and O2 decreased prominently at the dominant alpha rhythm, but were not observed or observed at much lower levels in the 8-12 Hz band. Numerous decreases in coherence were observed among the posterior short-range longitudinal coherences involving the parietal, temporal, and central leads, which had not been predicted from the pilot study. Widespread increases in coherence were observed in the 13-21 Hz band. In the 21-32 Hz band, the visual and combined conditions both decreased posterior longitudinal coherences, while the visual condition dramatically decreased frontal coherences in a manner that was not observed in the combined AVS condition. Both visual and combined AVS decreased 0.75-2.0 Hz coherence diffusely across the scalp. In the 2.0-4.0 Hz band, both the visual and combined AVS conditions increased short-range frontal longitudinal coherences, while the combined AVS had a dramatically greater effect on decreasing posterior coherences. In the 4.0-8.0 Hz band, the predicted decrease in occipital longitudinal coherences was observed in both, but a much greater tendency to decrease frontal longitudinal coherences was observed in the combined AVS condition.

Although a visual inspection of figure 3 and figure 4 suggests that there were differences among the three stimulation conditions, these were comparisons with the baseline, not between the conditions. To more directly compare the three conditions, and to determine if there were carry-over effects between them, an analysis of variance was performed. The three files containing differences from baseline for 30 subjects for the three stimulation conditions were concatenated into one file, and the subject number, gender, stimulation condition, and time were added to each line with the following anonymous script.

```
#!/usr/bin/perl
while(<>) {
     split;
     @letters = split//,$_[2];
# the second-indexed word on each line is the original
# filename, e.g., JF03FA2CMFPDIF. Splitting on a null
# delimiter allows you to extract the subject number,
# condition, and time from the filename by it’s ordinal
# position.
     print “$letters[2]$letters[3] $letters[5] $letters[6]
$_”;
}
```

With the subject number, condition, and time now listed in their own columns as independent variables, SAS proc glm code was written as follows:

```
Data a;
Input
SUBJ $ COND $ TIME $ FILE $ F1F2CD1 F1F7CD1 …
```

```
…
; run;
proc rank; by subj; run;
proc rank data=a out=ranked; run;
proc glm data=ranked; class SUBJ TIME COND; random SUBJ;
model F1F2CD1 = SUBJ TIME | COND @ 2; run;
```

*Writeprocglm.pl***.** A separate model statement was written for each variable by parsing a file containing the variable names with the following algorithm:

```
#!/usr/bin/perl
# writeprocglm.pl, By Jon Frederick, 8/16/01
open(IN,"<varnames");
while(<IN>)    {
     @vars = split/\s+/,$_;
}
foreach $var (@vars){
print "proc glm data=ranked\; class SUBJ TIME COND\; model
$var = SUBJ TIME \| COND \@ 2\; random SUBJ; run\;\n"
}
```

The code written to the standard output is then stored in a ".sas" file and executed from the Unix or DOS command line, i.e.,

```
sas filename.sas
```

*Parseglm.pl***.** Results from the SAS ".lst" output file were extracted with the following script.

```
#!/usr/bin/perl
# parseglm.pl, By Jon Frederick, 8/16/01
open(T,">glm-time.txt");
open(C,">glm-cond.txt");
open(INT,">glm-interaction.txt");
print T "Var   p\n";
print C "Var   p\n";
print INT "Var p\n";
while (<>){
     if (/Coeff/)   {
          @varname=split/\s+/;
          print T "$varname[6]";
          print C "$varname[6]";
          print INT "$varname[6]   ";
     }
     if (/\sTIME\s/){
          if ($x < 1)    { # don't print Type I SS
```

```
                $x++;
            } else    { # print p for Type III SS
                @F=split /\s+/;
                print T “$F[6]\n”;
                $x=0;
            }
        }
        if (/\sCOND\s/){
            if ($y < 1)    { # don’t print Type I SS
                $y++;
            } else    { # print p for Type III SS
                @F=split /\s+/;
                print C “$F[6]\n”;
                $y=0;
            }
        }
        if (/TIME\*COND/)   {
            if ($z < 1)    { # don’t print Type I SS
                $z++;
            } else    { # print p for Type III SS
                @F=split /\s+/;
                print INT “$F[6]\n”;
                $z=0;
            }
        }
}
```

Vbmapper.pl requires p values converted to (1-p) values, which is accomplished by this anonymous script:

```
while(<>)      {
        chomp;
        split;
        $oneminusp = 1 - $_[1];
        print “$_[0]    $oneminusp\n”;
}
```

Vbmapper.pl was then used to map the amplitude and coherence AVOVA results for effects of condition, time, and the interaction of condition and time (figures 5 and 6). Since the resulting maps often showed more significant effects than could be reliably counted by visual inspection, the number of findings with p<.01 were counted at the command line using standard Unix pattern-matching utility, `grep`, which has the following syntax:

```
grep pattern filename
```

Like the first code example in chapter 2, `grep` searches the input file, `filename`, for all lines matching `pattern`, and prints those lines to the standard output. The standard output of `grep` can then be *piped* using the pipe symbol ('|') to the standard input of `wc`, the Unix word count utility. The output of `wc` is a single line of text containing first, the number of lines in the standard input, second, the number of words in those lines, and third, the number of characters. For example,

```
grep M glm-cond.txt|wc
     133     266    1729
```

reveals that the input file, `glm-cond.txt` (the `parseglm.pl` output file containing variable names and p-values for the ANOVA effects of condition) contains 133 lines matching `M`, the Lexicor filename marker for "magnitude," or amplitude data. Given that 19 locations and 7 frequencies were measured in this experiment, this result from `wc` verifies that no data has been lost in the previous steps. Meanwhile,

```
grep M glm-cond.txt|grep "\.00"|wc
      40      80     520
```

reveals that 40/133 amplitude results were significant at p<.01.

ANOVAs detected widespread effect of condition on EEG amplitude in the DA, 8-12 Hz, 13-21 Hz, and 22-31 Hz bands (figure 5, row 1). 40/133 or 30% of ANOVAs for effects of condition were significant at p<.01. ANOVAs found negligible effects of time on amplitude (2/133 significant at p<.01; figure 5, row 2). Interaction effects of time and condition were also at negligible levels (1/133 significant at p<.01; figure 5, row 3).

To further examine the significant effects of condition on amplitude, differences between the three stimulation conditions (combined - auditory, combined - visual, and visual - auditory) were computed and sign-rank tested for significance (figure 5, fourth, fifth and sixth rows). These data show that essentially all of the differences detected by the ANOVA were differences between auditory stimulation and the other two conditions. The fifth row shows that, of the 40 locations where the ANOVAs found a significant effect of condition (at p<.01), only one of these was accounted for by a significant sign-rank difference between the combined AVS and visual alone. Meanwhile, very similar maps resulted from subtracting the auditory alone condition from the combined or visual alone conditions.

The coherence maps (figure 6) showed ANOVA effects of condition in all seven frequency bands (row 1). At p<.01, 161/1197 or 13% of the results were significant. However, only 14/1197 (1.2%) of the effects of time (row 2), and 8/1197 (0.7%) of the interaction effects of time and condition (row 3) were significant, levels consistent with type I error.

Signed rank tests on the differences between the three stimulation conditions showed that the combination of the auditory stimulus with the visual stimulus had complex effects on EEG coherence. While the auditory stimulus alone had negligible effects (figure 4, row 1), subtracting the effects of the visual stimulus from the combined AVS condition (figure 6, row 5) revealed 64/1026 (5.4%) significant differences at p<.01.

Subtracting the effects of the auditory stimulus from the combined AVS condition resulted in maps (figure 6, row 4) with numerous differences from the visual alone condition (figure 4, row 2). In particular, the combination of auditory and visual stimulation had an inhibitory effect on coherence in the 2.0-4.0 Hz and 4.0-8.0 Hz bands, where visual or auditory stimulation alone had negligible effects. Interestingly, in contrast to the hypothesis, the combined AVS condition had less of a desynchronizing effect on posterior longitudinal coherences than the visual alone condition in the 8-12 Hz and dominant alpha bands, particularly those involving T4.

Discussion

This study found that visual stimulation or combined AVS at each subject's dominant alpha frequency significantly increased EEG amplitude at the dominant alpha frequency (± .75 Hz), 8-12 Hz, 13-21 Hz, and 21-32 Hz (figure 3). Auditory stimulation alone at the dominant alpha frequency had negligible effects on amplitude (figure 3), and did not result in significant differences in amplitude between the visual and combined AVS condition (figure 5). Similarly, the auditory stimulus had only chance-level effects on EEG coherence, while the visual stimulus and combined AVS conditions evoked significant changes in coherence across the spectrum (figure 4). However, the interaction between auditory and visual stimulation resulted in significant differences between the visual alone and combined AVS conditions (figure 6). No residual effects of stimulation on amplitude or coherence were observed in the four minute recordings following each of the three stimulation sessions (figures 3 and 4).

These findings clearly refute the hypothesis that auditory stimulation inhibits the photic driving response in EEG amplitude. Thomas Bearden (personal communication, unpublished data) also failed to observe an alpha driving effect of auditory stimulation alone in 10 subjects. Our finding of a widely distributed photic driving response at the dominant alpha rhythm contrasts with the findings of Toman (1941), who noted that while alpha in the eyes-closed resting condition is observed frontally, centrally, and posteriorily (in increasing order), the PDR was rarely seen beyond the occipital lead. This difference could be explained by the enhanced sensitivity of modern QEEG methods, whereas Toman relied upon visual inspection of the raw EEG.

The lack of effect of auditory stimulation alone on EEG amplitude or coherence, and the inhibitory effect of visual stimulation alone short range posterior longitudinal coherences, also clearly refutes the hypothesis that auditory stimulation has any desynchronizing effect. Indeed, a number of short range posterior longitudinal coherences were significantly greater in the combined AVS condition than in the visual alone condition (figure 6, row 5). As a clinical implication, these results suggest that auditory stimulation might not substantially contribute to the clinical effects of AVS. However, an auditory driving effect on EEG power at 18.5 Hz at Cz was observed by Frederick et al. (1999), which suggests that this failure to observe an auditory evoked response might only be a property of stimulation at the dominant alpha frequency. Alpha is, after all, defined as a posterior dominant rhythm associated with the visual cortex.

There were several methodological differences between the present study and those of Timmermann, et al. (1999) or Timmermann (1998) which might account for the differences in the amplitude effects observed in the 8-12 Hz bands between the two studies. First, the measure of central tendency across each recording condition was the median in this study while it was the mean in Timmermann et al. (1999) and Timmermann (1998). Further, individual frequency-by-location differences from baseline were tested with Student's t-test in the Timmermann studies only if differences among the six recording conditions (baseline, times 1-4, and the post-baseline) were detected by an ANOVA procedure. The present study omitted the ANOVA procedure and used the Wilcoxon signed rank procedure for univariate tests. Arguably, a randomization test comparing the number of observed findings in this study to the number expected under the null hypothesis (as in chapter 3) should be conducted. However, the finding of close to one percent of results univariately significant (at $p<.01$) among the auditory alone condition and the post-stimulation baselines, when compared to the 31% and 36% observed in the visual alone and combined AVS conditions, has an intuitive clarity that does not cry out for further analysis. The finding of significant departures from the normal distribution both between and within subjects in the present study suggests that type II error from the use of parametric statistics on data which failed to meet parametric assumptions might account for the lack of effect of alpha stimulation on alpha in the Timmermann studies. (Indeed, the 20 variables that changed under alpha stimulation in Timmermann et al. (1999) were fewer than 4 percent of the 570 time, frequency, and location variables analyzed, a level of significant findings that would be expected from random data, were it not for the fact that all the changes were positive. This is suggestive of the loss of power that is achieved using parametric statistics on nonparametric data.) However, another difference between the two studies was the intensity of the visual stimulus. Timmermann et al. (1999) and Timmermann (1998) set the visual stimulus to it's second lowest setting ("2" of 99 on the PolySync Pro device) whereas the intensity of the visual stimulus was set at "50" in the present study. Toman (1941) had observed that the photic driving response was reliable at stimulus intensities experienced as uncomfortable by some subjects, while a reduction to a more comfortable level “gave a considerable loss in amplitude and regularity of response.” An additional difference between the two studies was the higher subject number. Thirty subjects were measured in this study compared to 13 in Timmermann et al. (1999) and 10 in Timmermann (1998).

Although figure 6 can, with some difficulty, be compared to figure 1, the 171 electrodes analyzed can obscure comparisons between the two studies. To more directly replicate and extend the coherence measurements in the pilot study, figure 7 was prepared. Figure 7 shows coherence effects only for the combined AVS condition at the 63 locations studied in the pilot study, and thus can be directly compared with the first five minutes of figure 1 in chapter 3. Levels of significance from p<.01 to p<.50 are shown because when lower levels of significance are consistent with the higher ones with respect to anatomy and frequency, the significance of each is informally supported.

Replicate-pilot.pl. Since the similarities and differences between the two studies are not always immediately and directly clear by visual inspection of figure 7, the following script, *replicate-pilot.pl,* was prepared. *Replicate-pilot.pl* takes as command line arguments the desired critical value (in 1-p format rather than p format since this is the *vbmapper* convention), and two files each containing 378 lines (63 coherences times the 6 frequencies measured in chapter 3), where the first file contains one variable name and sign-rank significance value per line for the pilot study, and the second file contains one variable name and significance value for the present replication study. *Replicate-pilot.pl* then determines, for each pair of matching variable names, whether they (a) agree— both are significant and change in the same direction from baseline; (b) disagree— both are significant and change in opposite directions; are unreplicated, defined as (c) significant in the pilot study but insignificant in the replication or (d) significant in the replication study by insignificant in the pilot study; or whether (e) both are insignificant.

```
#!/usr/bin/perl
# replicate-pilot.pl, By Jon Frederick, 8/16/01
# measures the number of agreements,
# disagreements, and unreplicated findings by comparing the
# replication study with the pilot study at critical value
# specified by user.
$critval = $ARGV[0];
$pilot = $ARGV[1];
$relication = $ARGV[2];
# critical values are in 1-p format (vbmapper convention)
if(!$ARGV[2])  {
     print "usage:\nreplicate-pilot.pl critval file1
file2\n"; die;
}
open(PILOT,"<$ARGV[1]") or die "can't open $ARGV[1]\n: $!";
while(<PILOT>){ chomp; push(@pilot,$_)}
open(REPLI,"<$ARGV[2]") or die "can't open $ARGV[2]\n: $!";
while(<REPLI>){ chomp; push(@repli,$_)}
foreach $pilot (@pilot)   {   ($pilotvar,$pilotval)   =
(split /\s+/, $pilot);
   @x = split//,$pilotvar;
   foreach $repli (@repli)   {
      ($replivar,$replival) = (split /\s+/,$repli);
# electrode order in each pair-name might not be the same
# between Stellate and Lexicor, so pattern match for both
# orders
      if(($replivar=~ /$x[0]$x[1]$x[2]$x[3]$x[4]$x[5]/) or
($replivar =~ /$x[2]$x[3]$x[0]$x[1]$x[4]$x[5]/))   {
         $abspilotval = abs $pilotval;
         $absreplival = abs $replival;
         $data="$pilotvar $pilotval $replivar $replival";
         if (($abspilotval>=$critval) and
             ($absreplival>=$critval))   {
            if ( (($pilotval>0) and ($replival>0)) or
                 (($pilotval<0) and ($replival<0)) )   {
               push(@agree,"$data");
            } else {
               push(@disagree,"$data");
            }
         } elsif (($abspilotval>=$critval) and
                  ($absreplival<$critval))   {
            push(@onlypilot,"$data");
         } elsif (($abspilotval<=$critval) and
                  ($absreplival>=$critval))   {
            push(@onlyrepli,"$data");
         } else   {
            push(@rejects, "$data");
         }
      }
   }
}
$numagree=@agree; $numdisagree=@disagree;
$numonlypilot=@onlypilot; $numonlyrepli=@onlyrepli;
$numrejects = @rejects;
$total = $numagree + $numdisagree + $numonlypilot +
         $numonlyrepli + $numrejects;
$p = 1 - $ARGV[0];
print "At p<$p, \n$numagree are in agreement between the
two studies, \n$numdisagree have contradictory significant
findings between the two studies,\n$numonlypilot
significant findings were seen only in the pilot study,
\n$numonlyrepli significant findings were seen only in the
relication study, \n$numrejects were insignificant in both
studies, \n$total variables fell into one of these five
categories (should be 378).\n";
```

Replicate-pilot.pl was run at several levels of significance and the standard output redirected to a temporary file, which was opened and re-arranged with MS Excel to produce Table 2. Table two shows the number and proportion of variables agreeing or disagreeing between the present study and the pilot study when the critical value for the signed rank test is adjusted from p<.01 to p<.999. Of particular interest is the critical value p<.2236, the square root of 0.05. Variables agreeing or contradicting at p<.2236 could be regarded as significant at p<.05 if the two independent studies were regarded as a single study. These results illustrate the trade off between type I and type II error that is achieved at varying critical values. Notice how the proportion of values significant in "only pilot" or "only replication" remains relatively constant from p<.05 to p<.50, while the proportion of variables "agreeing" and "disagreeing" between the two studies ascends linearly between these two critical values. The slope of the regression line from p=0 to p<.999 was 1.37 for variables "agreeing," and 3.11 for variables "disagreeing." The r-squared values for these two regression lines were greater than 0.97, indicating a strong linear fit. The dramatically larger rate of increase in disagreements as the critical value is relaxed suggests that they are more likely than the agreements to arise from type I error. The failure of the agreements to plateau or the disagreements to rise more sharply at more relaxed critical values suggests that selecting an optimal critical value is a somewhat arbitrary decision.

To further clarify the nature of the differences between the pilot and replication studies, *replicate-pilot.pl* was modified slightly to print the `@agree`, `@disagree`, `@onlypilot`, and `@onlyrepli` arrays, which were then plotted with *vbmapper* to create figure 8. Figure 8 could be interpreted as supporting the idea that factors in the EEG coherence matrix replicate more reliably than individual coherence variables which reflect these factors. For example, only 11 of the 26 decreases in posterior longitudinal coherence (i.e. from O1, O2, and Pz) that were significant at p<.05 during the first five minutes in the pilot study were replicated at p<.10 (or one-tailed p<.05) in the replication study (figure 8, row 1). However, 23 other posterior longitudinal coherences involving O1, O2, or Pz were decreased at p<.05 in the replication study that were not significantly changed in the pilot study (figure 8, row 3). A more conservative interpretation would throw out all of the replicated posterior longitudinal decreases at p<.05 (except in the beta 2 band where a clear pattern is evident) as random effects of multiple comparisons. The only strong pattern of contradictory findings (at p<.10; figure 8, row 2) was in the delta 1 and delta 2 bands, where longitudinal coherences involving F3, C3, P3, Fz, Cz, and Pz appeared to significantly increase in the pilot study and significantly decrease in the replication study. Very few of the results that were only significant in the pilot or only significant in the replication study, contradict the general pattern of increased frontal and decreased posterior longitudinal coherences, or the pattern of decreased interhemispheric coherences at lower frequencies with increased interhemispheric coherences at higher frequencies.

A tendency for long range posterior longitudinal coherences (O1F7, O1FP1, O1F3, O2F4, O2F8) to increase was observed in the 8-12 Hz and 13-21 Hz bands in the replication which was not seen in the pilot study (figure 8, row 4). However, since no effects at all were observed in these coherences from 8-31 Hz in the pilot study, these findings (with higher subject number) extend rather than refute the original study. These findings tended to agree between the two studies when the critical value was relaxed to.2236, the square root of .05 (i.e., cumulative probability for both studies, p<.05). Similarly, decreases in the longitudinal coherences along the central midline (involving F7, F3, Fz, F4, F8, T3, C3, Cz, C4, T4, T5, P3, P4, and T6) in all bands except the dominant alpha and 13-21 Hz, where generally no effect had been observed in the pilot study with smaller subject number. Frontal coherences with T5, P3, and P4 decreased in the theta band in the replication, whereas no effect had been detected in the pilot study.

Shorter-range frontal coherences increased as they had in the pilot study, except in the 4-8 Hz band where no effects were detected. The slight tendency for frontal interhemispheric coherences to decrease was not replicated. However, the stronger tendency for interhemispheric coherences to decrease at lower frequencies and increase at higher frequencies was replicated. Thus, some effects were observed in either study that might not have been detected in the other study, but very few contradictory results were obtained.

Literature Cited

Adrian, E. D., & Matthews, B. H. C. (1934). The Berger rhythm: potential changes from the occipital lobes in man. *Brain*, 57: 355-384.

Anderson, D. J. (1989). The treatment of migraine with variable frequency photostimulation. *Headache*, 29: 154-155.

Angelakis, E. & Lubar, J.F. (2001). The role of peak alpha frequency in reading ability. *Proc Assoc Applied Psychophysiol Biofeedback.* AAPB, Wheat Ridge, CO.

Aranibar, A. & Pfurtscheller, G. (1978). On and off effects in the background EEG activity during one-second photic stimulation. *Electroencephalogr Clin Neurophysiol,* 44: 307-316.

Bauer, L.O. (1994). Photic driving of EEG alpha activity in recovering cocaine-dependent and alcohol-dependent patients. *The American J On Addictions,* 3(1): 49-57.

Brandt, M. E. (1997). Visual and auditory evoked phase resetting of the alpha EEG. *Int J Psychophysiol* 26: 285-298.

Brauchli, P., Michel, C. M., & Zeier, H. (1995). Electrocortical, autonomic, and subjective responses to rhythmic audio-visual stimulation. *Int J Psychophysiol,* 19: 53-66.

Carter, J. L., & Russell, H. L. (1993). A pilot investigation of auditory and visual entrainment of brainwave activity in learning-disabled boys. *Texas Researcher,* 4: 65-73.

Conover, W.J. (1980). *Practical Nonparametric Statistics*. New York: John Wiley & Sons.

Coull, Bruce M. & Pedley, Timothy A. (1978). Intermittent photic stimulation. Clinical usefulness of non-conclusive responses. *Electroencephalogr Clin Neurophysiol,* 44: 353-363.

Dafters, R.I., Duffy, F., O'Donnell, P.J. & Bouqet, C. (1999). Level of use of 3,4-methylenedioxymethamphetamine (MDMA or Ecstasy) in humans correlates with EEG power and coherence. *Psychopharmacology*, 145(1): 82-90.

Dieter, J. N. I., & Weinstein, J. A. (1995). The effects of variable frequency photo-stimulation goggles on EEG and subjective conscious state. *J Mental Imagery,* 19: 77-90.

Duffy, F. H., Iyer, V. G., & Surwillo, W. W. (1989). *Clinical and Electroencephalography and Topographic Brain Mapping..* New York: Spinger-Verlag.

Edgington, E. S. (1987). *Randomization Tests.* 2nd ed. New York: M. Dekker.

Frederick, J.A. (2001). Technical Note: Open source method of graphical QEEG analysis using PERL and Visual Basic. *J Neurother,* 4(4): 63-70.

Frederick, J. A., and Lubar, J. F. (2001). Differences between coherence and spectral correlation during auditory and visual stimulation at the dominant alpha frequency. *Conference Abstract Submitted.*

Frederick, J.A., Lubar, J.F., Rasey, H.W., Brim, S.A., and Blackburn, J. (1999). Effects of 18.5 Hz auditory and visual stimulation on EEG amplitude at the vertex. *J Neurother,* 3 (3-4): 23-28.

Garoutte, B., Aird, R. B., & Correnti, S. (1958). Studies on the cerebral pacemaker: Some relationships between driving and driven rhythms in intermittent photic stimulation. *Electroencephalogr Clin Neurophysiol,* 10: 770-771.

Gevins, A. S., Morgan, N.H., Bressler, S.L., Cutillo, B.A., White, R.M., Illes, J., Greer, D.S., Doyle, J.C., & Zeitlin, G.M. (1987). Human neuroelectric patterns predict performance accuracy. *Science,* 235: 580-585.

Gizycki, H., Jean-Louis, G., Snyder, M., Zizi, F., Green, H., Giuliano, V., Spielman, A., & Taub, H. (1998). The effects of photic driving on mood states. *J Psychosomatic Research,* 44(5): 599-604.

Hubel, DH, and Wiesel, TN (1965). Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. *J Neurophysiol* 28:229-89.

Hubel, DH, and Wiesel, TN (1968). Receptive fields and functional architecture of the monkey striate cortex. *J Physiol* 195:215-43.

Hubel, DH, and Wiesel, TN (1977). Functional architecture of macaque monkey visual cortex." *Proc Royal Soc London* 198:1-59.

Iwahara, S., Noguchi, S., Yang, K., & Oishi, H. (1974). Frequency specific and nonspecific effects of flickering lights upon electrical activity of the human occiput. *Japanese Psychological Research,* 16: 1-7.

Jansen, B. H., Brandt, M. E. (1991). The effect of the phase of prestimulus alpha activity on the averaged visual evoked response. *Electroencephalogr Clin Neurophysiol* 80(4): 241-50.

Jin, Y., Potkin, S. G., & Sandman, C. (1995). Clozapine increases EEG photic driving in clinical responders. *Schizophr Bull* 21(2): 263-8.

Jin, Y., Potkin, S. G., Sandman, C., & Bunney, W. (1997). Electroencephalographic photic driving in patients with schizophrenia and depression. *Society of Biological Psych,* 41: 496-499.

Joyce, Michael, Siever, Dave (2000). Audio-visual Entrainment Program as a Treatment for Behavior Disorders in a School Setting. *J Neurother,* 4(2): 9-25.

Kawaguchi, T., Jijiwa, H., & Watanabe, S. (1993). The dynamics of phase relationships of alpha waves during photic driving. *Electroencephalogr Clin Neurophysiol,* 87: 88-96.

Klimesch, W. (1997). EEG-alpha rhythms and memory processes. *Int J Psychophysiol* 26: 319-340.

Lubar, J.F., and Lubar, J.O. (1999). Neurofeedback assessment and treatment for attention deficit/hyperactivity disorders. In Evans, J.R., and Abarbanel, A., (Eds). *Introduction to quantitative EEG and neurofeedback* (pp. 103-143). San Diego, CA, US: Academic Press, Inc.

Lubar, J. F., Swartwood, M. O., Swartwood, J. N., & O'Donnell, P. (1995). Evaluation of the effectiveness of EEG neurofeedback training for ADHD in a clinical setting as measured by changes in T.O.V.A. scores, behavioral ratings, and WISC-R performance. *Biofeedback and Self-Regulation,* 20: 83-99.

Martini, P. S., Venturini, R., Zapponi, G. A., & Loizzo, A. (1979). Interaction between intermittent photic stimulation and auditory stimulation on the human EEG. *Neuropsychobiology*, 5: 201-206.

Micheletti, L.S., (1999). The use of auditory and visual stimulation for the treatment of Attention Deficit Hyperactivity Disorder in children. (ritalin, adderall). *Dissertation Abstracts International: Section B: The Sciences and Engineering* 60(6-B): 2953.

Montgomery, D. D., Ashley, E., Burns, W. J., & Russell, H. L. (1994). Clinical outcome of a single case study of EEG entrainment for closed head injury. *Proc Assoc Applied Psychophysiol Biofeedback,* 25: 82-83.

Morse, D. R. (1993). Brain wave synchronizers: A review of their stress reduction effects and clinical studies assessed by questionnaire, galvanic skin resistance, pulse rate, saliva, and electroencephalograph. *Stress Medicine,* 9: 111-126.

Mueller, H.H., Donaldson, C.C.S., Nelson, D.V., & Layman, M. (2001). Treatment of fibromyalgia incorporating EEG-driven stimulation. *J Clinical Psychology* 57(7): 933-952.

Noton, D. (1997). PMS, EEG, and photic stimulation. *J Neurotherapy.* 2(2): 8-13.

Nunez, P.L., Wingeier, B.M., & Silberstein, R. (2001). Spatio-temporal structures of human alpha rhythms: theory, microcurrent sources, multiscale measurements, and global binding of local networks. *Human Brain Mapping* 13: 125-164.

Ochs, L. (1994) Method for evaluating and treating an individual with electroencephalographic disentrainment feedback. United States Patent Number 5,365,939.

Ochs, L. (1996). New light on lights, sound, and the brain. *Megabrain Report,* 2(3): 48-52.

Patrick, G. J. (1996). Improved neuronal regulation in ADHD: An application of 15 sessions of photic-driven EEG neurotherapy. *J Neurotherapy,* 1: 27-36.

Pigeau, R. A., Frame, A. M. (1992). Steady-state visual evoked responses in high and low alpha subjects. *Electroencephalography and Clin Neurophysiol*, 84: 101-109.

Politoff, A.L., Monson, N., Hass, P., Stadter, R. (1992). Decreased alpha bandwidth responsiveness to photic driving in Alzheimer disease. *Electroencephalogr Clin Neurophysiol,* 82: 45-52.

Pollock, V. E., Teasdale, T., Stern, J., & Volavka, J. (1981). Effects of caffeine on resting EEG and response to sine wave modulated light. *Electroencephalogr Clin Neurophysiol,* 51*: 470-476.*

Regan, D. (1966). Some characteristics of average steady-state and transient responses evoked by modulated light. *Electroencephalogr Clin Neurophysiol* 20: 238-248.

Rosenfeld, J. P., Reinhart, A. M., & Srivastava, S. (1997). The effects of alpha (10 Hz) and beta (22 Hz) “entrainment” stimulation on the alpha and beta EEG bands: Individual differences are critical to prediction of effects. *J Applied Psychophysiol Biofeedback,* 22: 3-20.

Rozelle, G.R. and Budzynski, T.H. (1995). Neurotherapy for stroke rehabilitation: A case study. *Biofeedback and Self-Regulation,* 20, 211-228.

Russell, H. L. (1997). Intellectual, auditory, and photic stimulation and changes in functioning in children and adults. *Biofeedback,* 16-17, 24, 23.

Schwartz, R., Christiansen, T., & Wall, L. (1997). *Learning Perl.* Sebastopol, CA: O'Reilly & Associates.

Shaw, J. C. (1981). An introduction to the coherence function and its use in EEG signal analysis. *J Med Eng Tech*, 5(6): 279-288.

Shealy, C. N., Cady, R. K., Cox, R. H., Liss, S., Clossen, W. & Culver Veehoff, D. (1990). *Brain Wave Synchronization (Photo-stimulation) with the Shealy RelaxMate ™.* Shealy Institute for Comprehensive Health Care: Springfield, MO.

Silberstein, R. B. (1995). Neuromodulation of neocortical dynamics. In P. L. Nunez (Ed.), *Neocortical Dynamics and Human EEG Rhythms* (pp. 591-627). New York: Oxford University Press.

Solomon, G. D. (1985). Slow wave photic stimulation in the treatment of headache-a preliminary study. *Headache,* 25: 444-446.

Striano, S., Meo, R., Bilo, L., Ruosi, P., Soricellis, M., Estraneo, A., Caporella, A. (1992). The use of EEG activating procedures in epileptology. *Acta Neurologica,* 14(4-6): 275-289.

Stwertka, S. A. (1993). The stroboscopic patterns as dissipative structures. *Neuroscience and Biobehavioral Reviews,* 17: 69-78.

Suldo, S. (2000). Quantitative EEG research with precociously reading children: the importance of peak alpha frequency (conference abstract). *J Neurother,* 4(4): *In Press.*

Takahashi, T. (1987). Activation methods; in Niedermeyer E., Lopez da Silva, F. (eds): *Electroencephalography, Basic Principles, Clinical Applications and Related Fields,* ed 2. Baltimore, Urban & Scharzenberg 1987, pp 209-227.

Thatcher, R. W. (1992). Cyclic cortical reorganization during early childhood. *Brain and Cognition,* 20: 24-50.

Thompson, J. C., Tzambazis, K., Stough, C., Nagata, K., & Silberstein, R. B. (2000). Effects of nicotine on the 13 Hz steady-state visually evoked potential. *Clin Neurophysiol, 111(9): 1589-1595.*

Timmermann, D.L. (1998). *An Assessment of the Effects of Multi-session Audio-visual Stimulation on Cognitive Measures and the Cortical EEG.* Dissertation in Experimental Psychology: University of Tennessee.

Timmermann, D., Lubar, J.F., Rasey, H.W., and Frederick, J.A. (1999). Effects of dominant and twice-dominant alpha audiovisual stimulation on the cortical EEG. *Int J Psychophysiol,* 32: 55-61.

Toman, J. (1941). Flicker potentials and the alpha rhythm in man. *J Neurophysiol*, 4: 51-61.

Townsend, R. E., Lubin, A., & Naitoh, P. (1975). Stabilization of alpha frequency by sinusoidally modulated light. *Electroencephalogr Clin Neurophysiol,* 39: 515-518.

Van der Twill, L. H., & Verduyn Lunel, H. F. E. (1965). Human visual responses to sinusoidally modulated light. *Electroencephalogr Clin Neurophysiol,* 18: 587-598.

Vogel, W., Broverman, D. M., Klaiber, E. L., & Kobayashi, Y. (1974). EEG driving responses as a function of monoamine oxidase. *Electroencephalogr and Clin Neurophysiol, 36: 205-207.*

Wada, Y., Nanbu, Y., Kadoshima, R., Jiang, Z.Y. (1996). Interhemispheric EEG coherence during photic stimulation: Sex differences in normal young adults. *Int J Psychophysiol,* 22(1-2): 45-51.

Wada, Y., Nanbu, Y., Kikuchi, M, Koshino, Y., Hashimoto, T., and Yamaguchi, N. (1998a). Abnormal functional connectivity in Alzheimer's disease: Intrahemispheric EEG coherence during rest and photic stimulation. *European Archives of Psychiatry and Clinical Neuroscience,* 248(4): 203-208.

Wada, Y., Nanbu, Y., Kikuchi, M, Koshino, Y., and Hashimoto, T. (1998b). Aberrant functional organization in schizophrenia: Analysis of EEG coherence during rest and photic stimulation in drug-naive patients. *Neuropsychobiology,* 38(2): 63-69.

Wada, Y., Takizawa, Y., and Yamaguchi, N. (1995). Abnormal photic driving responses in never-medicated schizophrenia patients. *Schizophr Bull* 21: 111-115.

Wall, L., Christiansen, T., & Schwartz, R. (1996). *Programming Perl*. Sebastopol, CA: O'Reilly & Associates.

Walter, W. G., Dovey, V. J., & Shipton, H. (1946). Analysis of electrical response of human cortex to photic stimulation. *Nature*, 158: 340-541.

Walter, V.J., and Walter, W.G. (1949). The central effects of rhythmic sensory stimulation. *Electroencephalogr Clin Neurophysiol,* 1: 57-86.

Appendices

Appendix A: Institutional Review Board Form B

Form B

IRB#__________

Date received in ORC_________

THE UNIVERSITY OF TENNESSEE, KNOXVILLE

I. Identification of Project: Date: March 1, 2001

Project Director:	Joel F. Lubar, Ph.D. Department of Psychology 974-3360 or 974-3222
Co-Directors:	Jon A. Frederick, M.S. Department of Psychology 974-0464 or 406-5694
Title of Project:	Effects of Audiovisual Stimulation On the Cortical EEG
Department:	Psychology
Starting Date:	Upon IRB approval
Completion Date:	March 2002
External Funding Agency:	n.a.
Grant Submission Deadline:	n.a.

II. Objective of Project:

Since the early history of electroencephalography (EEG), it has been known that a flashing light stimulus can induce changes in the EEG at frequencies corresponding to the frequency of stimulation. Recent studies in our lab have demonstrated that combined auditory and visual stimulation can activate the EEG at a diverse range of frequencies, beyond the frequency of stimulation (Timmermann, D., Lubar, J.F., Rasey, H.W., and Frederick, J.A., 1999. Effects of dominant and twice-dominant alpha audiovisual stimulation on the cortical EEG. International Journal of Psychophysiology, 32:55-61.) Since abnormal EEG activity has been associated with a number of psychological and neurological disorders, some investigators have researched the ability of rhythmic auditory and visual stimulation (AVS) to evoke clinically beneficial changes in the brain. However, most often, the effects measured in these studies have been clinical, not electrophysiological, so the underlying neurological basis of these effects is not well-understood. A number of recent studies in our lab, and in Peter Rosenfeld's lab at Northwestern University, have suggested that combining auditory and visual stimulation at the same frequency has an interfering rather than a synergistic effect on evoked responses in EEG amplitude and coherence (Frederick, J.A., Lubar, J.F., Rasey, H.W., Brim, S.A., and Blackburn, J., 1999. Effects of 18.5 Hz auditory and visual stimulation on EEG amplitude at the vertex. J. Neurother., 3(3-4), 23-28). However, this hypothesis has not yet been tested directly in a comprehensive manner. This unresolved question is important clinically because combining auditory with visual stimulation is a common therapeutic practice. A conclusive demonstration of whether combined AVS has a synergistic or an interfering effect would be of immediate interest to many neuropsychologists in clinical practice.

III. Description of Research Participants:

This project will seek to recruit 30 adults (ages 18 - 30) from the University of Tennessee, Knoxville. It is believed that spoken announcements in classes taught by members of our lab, offering extra credit for participation, should be sufficient for this purpose. Participants should be of normal intelligence, with no known history of learning disabilities, neurological difficulties, siezure activity, or attention deficit disorder. There will be no monetary incentives to participate in this study.

IV. Methods and Procedures:

All participants will receive 18 minutes of stimulation at 18 Hz using a Proteus (Synetic Systems) device. First, a four-minute baseline EEG will be recorded. Then, each participant will receive the following three 6-minute stimulation conditions, randomly assigned to each participant in a counter-balanced order: (1) auditory stimulation alone; (2) visual stimulation alone; and (3) combined AVS. A four-minute eyes-closed post-stimulation baseline will be recorded after each of these conditions.

All procedures will take place in the Brain Research and Neuropsychology Lab at A305 Walters Life Sciences at the University of Tennessee. Participants will be seated in a comfortable chair with ambient lighting.

The Proteus device includes headphones and a pair of photoscopic glasses that are connected to a small, portable unit that is programmed to provide specified levels of visual and auditory stimulation. Our lab has used similar stimulation protocols with Synetic Systems devices in several approved studies in the past six years.

A quantitative 19 channel EEG or "brain map" will be used to provide percentage and power information within the following encephalographic domains: delta 1 (1 - 2 Hz), delta 2 (2 - 4 Hz), theta (4 - 8 Hz), alpha (8 - 12 Hz), beta 1 (13 - 21Hz), and beta 2 (21 - 32 Hz activity).

The EEG will be administered following the International 10-20 system for electrode placement and will include the central locations. Hookups will be made using an electrode cap (Electro Cap Inc.) and Electrogel conductive cream (Weaver and Co.). Subjects will have the electrode cap with 19 sensors placed on their heads. Electrode gel is applied to each sensor by a small tube inserted through the sensor; the gel forms a conductive pathway between the sensor and the scalp. There is no significant discomfort with this procedure either in the preparation or the wearing of the cap during the testing. Two earclip electrodes will be placed on the earlobe after a light cleaning with Omniprep solution which removes skin oil and allows for good sensor contact.

Following data collection, the cap will be removed, and the electrode gel, which is very similar to hair mousse, will be wiped off with a tissue or paper towel. All creams and gels used during this evaluation are hypo-allergenic, with no known risk of irritation. Our laboratory has used this method of application in previously approved projects since 1990.

V. Specific Risk and Protection Measures:

There is minimal risk to participants in this study. Confidentiality will be maintained with respect to the data collected on individuals, and the procedures themselves are not harmful. An standard EEG “optisolator” built into the amplifier converts the physiological signal from the participant into light, and then back into an electrical signal, which serves to physically isolate the participant from the electrical equipment. All electrical equipment is grounded as an additional measure of protection.

Participants will be informed that they are free to stop the procedure and withdraw their participation at any time without penalty.

Each participant will be assigned a number which alone is used to identify all materials. In addition, group or numbered data will be presented in any future publications or conference presentations. All material containing names of participants will be stored in a locked file cabinet in the Brain Research and Neuropsychology Laboratory, Walter's Life Sciences Building, room A305. Only those individuals listed as directors or co-directors in this form will have access to these files.

VI. Benefits vs. Risks:

The benefits of this study include the clinical implications of understanding the difference between unimodal vs. bimodal stimulation. Also, our previous studies of AVS have produced data which have been valuable for the development of statistical and graphical methods that we have published or submitted. These methods themselves might be of benefit to the broader community of scientists and clinicians who routinely measure EEG. Benefits also include the possibility of receiving extra credit for participation if offered by an instructor.

The risk to either physical or psychological well being of any participant participating in this study is minimal. The procedures described within this form have been used in previous studies in our lab and posed no risk to participants.

VII. Methods of obtaining "Informed Consent" from participants:

All participants will be required to read and sign the appropriate informed consent form prior to any participation in the study.

Participants will be told that participation in this study is strictly voluntary, and that the information collected will be kept confidential. Participants will be informed that the data gathered during this experiment will potentially be shared professionally, but will include numerical coding to prevent identification. Participants will be provided detailed information concerning the evaluation procedures that will be used.

All participants will be free to withdraw from the study at any time.

VIII. Qualifications of the Investigators:

Dr. Joel Lubar is a Full Professor in the Department of Psychology and has over 35 years of experience working in the field of neuroscience. He is a licensed Psychologist within the State of Tennessee, with the designation of "Health Service Provider."

Jon Frederick is a doctoral candidate in the Experimental Psychology Department. He received his M.S. degree in Neuroscience from the University of Michigan in 1995. He has five years of practice administering EEGs.

IX. Adequacy of Facilities to Support Research:

The data for this study will be collected within the Brain Research and Neuropsychology Laboratory. Walter's Life Sciences Building, University of Tennessee. Equipment to be used is owned by either Dr. Joel Lubar or the University of Tennessee. All instrumentation and test materials to be used in this study are directly comparable to materials used in hospitals and clinical settings.

X. Responsibility of Project Director:

By the compliance with the policies established by the University of Tennessee, Knoxville, Committee on Research Participation, the project director subscribes to the principles stated in "The Belmont Report" and standards of professional ethics in all research, development, and related activities involving human subjects under the auspices of the University of Tennessee, Knoxville.

a. Approval will be obtained from the University Committee prior to instituting any change in the research project.

b. Development of any unexpected risks will be reported to the University Committee.

c. Signed consent statements will be kept in a locked file cabinet in the Brain Research and Neuropsychology Laboratory, Walter's Life Sciences Building, Room A305, for the duration of the project and for at least three years thereafter.

XI. Signatures

Project Director: ________________________
Joel F. Lubar

Date: __________________

Co-Director: ________________________
Jon A. Frederick

Date: __________________

XII. DEPARTMENT REVIEW AND APPROVAL

The application described above has been reviewed by the IRB departmental review committee and has been approved. The DRC further recommends that this application be reviewed as:

[ ] Expedited Review -- Category(ies):
____

OR

[ ] Full IRB Review

Chair, DRC ______________________________

Signature _______________________________ Date ____________

Department Head ______________________________

Signature_______________________________ Date _____________

Protocol sent to Research Compliance Services Section for final approval on (Date) ________________

Approved:
Research Compliance Services Section
Office of Research
404 Andy Holt Tower

Signature_______________________________ Date _____________

Appendix B: Informed Consent Form

Consent Form

Objective of Project: This study is designed to evaluate the usefulness of auditory and visual stimulation (AVS) or entrainment devices for driving electroencephalograph (EEG) activity. AVS or entrainment consists of presenting visual (flashing lights) and/or auditory (tones) stimulation at certain frequencies.

Amount of Time Required: About one hour will be required of each participant for administering the EEG and providing the AVS.

Project Summary: For this research project, the following data will be collected:

1. Baseline: a four-minute eyes-closed baseline of EEG.

2. Three six-minute stimulation conditions, administered in random order: (1) auditory stimulation alone; (2) visual stimulation alone; (3) combined auditory and visual stimulation.

3. Three four-minute eyes-closed baseline EEGs will be recorded after each stimulation condition.

All EEG recording will be administered by following the International 10-20 system for electrode placement. Hookups will be made using an electrode cap and Electrogel conductive cream. Participants will have the electrode cap with 19 sensors placed on their heads. Electrode gel is applied to each sensor by a small tube inserted through the sensor; the gel forms a conductive pathway between the sensor and the scalp. There is no significant discomfort with this procedure either in the preparation or the wearing of the cap during the testing. Two earclip electrodes will be placed on the earlobe after a light cleaning with Omniprep solution which removes skin oil and allows for good sensor contact. All creams and gels used during this evaluation are hypo-allergenic, with no known risk of irritation. AVS will be provided by using a Polysync Pro stimulation device. This includes headphones and a pair of photoscopic glasses that are connected to a small, portable unit that is programmed to provide specified levels of visual (flashing lights) and auditory (tones) stimulation. Participants will sit in a dimly lit room with eyes closed while photosonic stimulation is presented. There is no significant discomfort with this procedure.

____
Initials of participant

Informed Consent Form, Page Two

If you have any questions regarding the research study, please feel free to ask them. Any future questions may be addressed to

Jon Frederick, M.S.
Department of Psychology
University of Tennessee, Knoxville
974-0464 or 406-5694

or

Dr. Joel Lubar
Department of Psychology
University of Tennessee, Knoxville
974-3222 or 974-3360

Statement of Consent:

I certify that I have read and fully understand the procedures contained within this form and agree to participate in the research described herein. My participation is given voluntarily and without coercion or undue influence. I understand that I may discontinue participation at any time; however, documentation of participation only be provided upon successful completion of the recording session. I understand that my course instructor (not the study investigators) decides whether extra credit will be provided for my participation in this experiment.

______________________	________________	_____
Signature of Participant	Name of Participant	Date
______________________	________________	_____
Signature of Researcher	Name of Researcher	Date

Appendix C: Tables and Figures

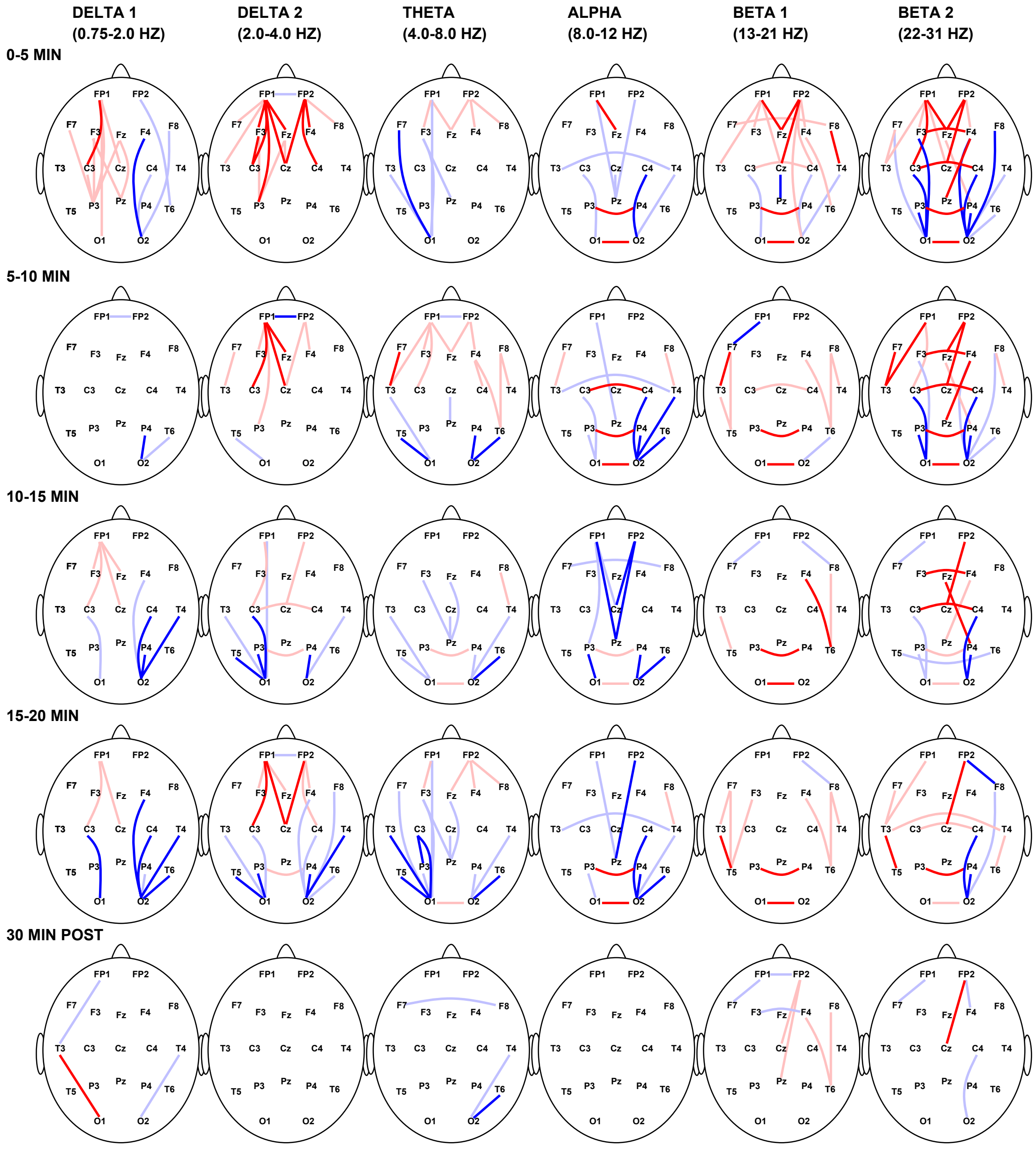

Figure 1. Effect of dominant alpha stimulation on EEG coherence. Dark red lines, increases at p<=.01; light red lines, increases at p<=.05; dark blue lines, decreases at p<=.01; light blue lines, decreases at p<=.05.

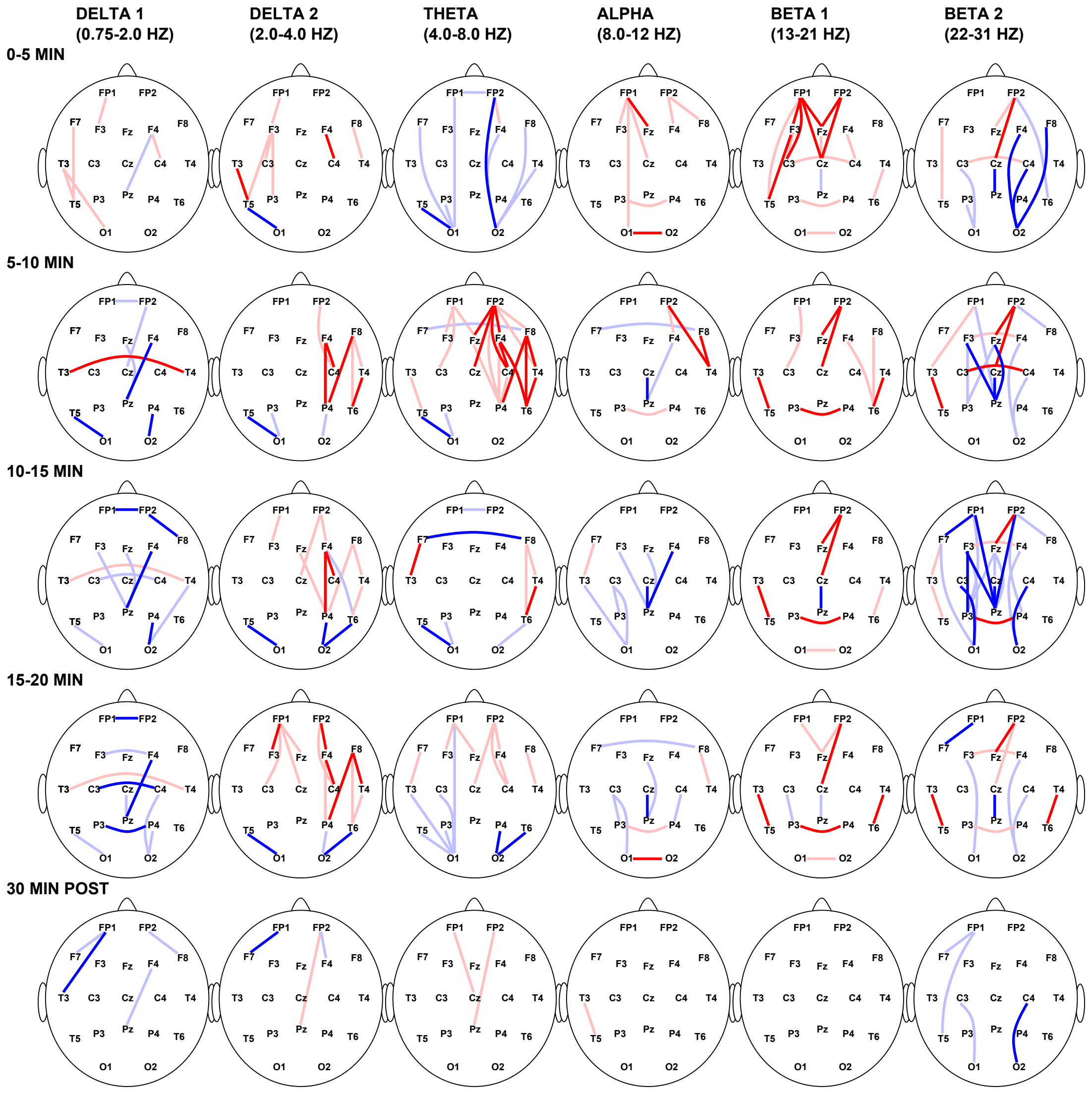

Figure 2. Effect of twice-dominant alpha stimulation on EEG coherence. Dark red lines, increases at p<=.01; light red lines, increases at p<=.05; dark blue lines, decreases at p<=.01; light blue lines, decreases at p<=.05.

Table 1. Number of significant Wilcoxon sign-rank tests (p<=.01) for each variable counted across all other variables. Abbreviations: p, rank of count (divided by 1000) among 1000 randomized trials; pos, number of increases; neg, number of decreases, tot, total, null, average number of false positive tests observed in 1000 random trials; SNR, signal-to-noise ratio, i.e. tot/null.

var	*p*	*pos*	*neg*	*tot*	*null*	*SNR*	*var*	*p*	*pos*	*neg*	*tot*	*null*	*SNR*
ALL	0.001	128	113	241	41.7	5.8	**F4C4**	0.006	5	0	5	0.7	7.2
							F4P4	0.123	2	0	2	0.7	2.9
ALPHA	0.001	67	63	130	21.2	6.1	**F4T6**	0.111	2	0	2	0.7	3.0
BETA	0.001	61	50	111	20.5	5.4	**F7T3**	0.025	4	0	4	0.7	5.8
							F8P4	0.045	3	0	3	0.7	4.5
D1	0.007	3	22	25	7.1	3.5	**F8T4**	0.022	4	0	4	0.6	6.3
D2	0.001	28	17	45	7.1	6.3	**F8T6**	0.275	1	0	1	0.7	1.5
TH	0.003	13	18	31	6.8	4.6	**FZP4**	0.311	1	0	1	0.7	1.4
AL	0.006	13	19	32	7.0	4.5	**C3C4**	0.006	5	1	6	0.7	9.0
B1	0.001	35	3	38	6.8	5.6	**P3P4**	0.001	14	1	15	0.7	22.7
B2	0.001	36	34	70	6.8	10.2	**T3T4**	0.262	1	0	1	0.6	1.6
							T3T5	0.003	8	0	8	0.7	12.3
0-5 min	0.001	40	18	58	8.6	6.7	**T4T6**	0.007	6	0	6	0.7	9.1
5-10 mi	0.001	45	23	68	8.5	8.0	**O1O2**	0.001	11	0	11	0.7	16.9
10-15 m	0.001	17	36	53	8.5	6.2	**F1F2**	0.047	0	3	3	0.6	5.3
15-20 m	0.001	24	32	56	8.6	6.5	**F1F7**	0.019	0	4	4	0.6	6.6
POST	0.545	2	4	6	7.5	0.8	**F1PZ**	0.108	0	2	2	0.7	3.0
							F2F8	0.116	0	2	2	0.7	3.0
F1	0.002	22	10	32	7.3	4.4	**F2O2**	0.297	0	1	1	0.7	1.5
F2	0.001	31	10	41	7.3	5.7	**F2PZ**	0.039	0	3	3	0.7	4.5
F3	0.038	8	4	12	4.7	2.6	**F3P3**	0.291	0	1	1	0.7	1.5
F4	0.001	17	8	25	4.7	5.3	**F7F8**	0.3	0	1	1	0.7	1.4
F7	0.032	4	6	10	4.0	2.5	**FZPZ**	0.278	0	1	1	0.6	1.6
F8	0.008	8	5	13	4.0	3.3	**C3P3**	0.25	0	1	1	0.6	1.7
FZ	0.001	18	1	19	4.1	4.6	**CZPZ**	0.001	0	9	9	0.6	14.5
C3	0.001	12	8	20	3.2	6.2	**O1C3**	0.009	0	6	6	0.6	9.4
C4	0.001	12	13	25	3.3	7.6	**O1F3**	0.278	0	1	1	0.6	1.6
CZ	0.001	19	10	29	2.6	11.2	**O1F7**	0.276	0	1	1	0.6	1.6
T3	0.008	15	2	17	3.3	5.2	**O1P3**	0.02	0	5	5	0.6	7.8
T4	0.001	12	4	16	3.2	5.0	**O1T5**	0.001	0	12	12	0.7	16.9
P3	0.001	15	8	23	4.6	5.0	**O2C4**	0.001	0	12	12	0.7	18.2
P4	0.001	20	17	37	4.7	7.8	**O2F4**	0.016	0	4	4	0.7	6.1
PZ	0.001	2	21	23	4.0	5.8	**O2F8**	0.104	0	2	2	0.7	3.1
T5	0.001	9	12	21	4.1	5.2	**O2P4**	0.001	0	16	16	0.7	22.9
T6	0.001	9	11	20	4.0	5.0	**O2T4**	0.019	0	4	4	0.6	6.6
O1	0.001	12	26	38	5.3	7.2	**O2T6**	0.001	0	11	11	0.6	17.7
O2	0.001	11	50	61	5.3	11.6	**PZF3**	0.1	0	2	2	0.6	3.2
							PZF4	0.003	2	4	6	0.7	8.6
F1C3	0.003	6	0	6	0.6	9.7	**F1T3**	0.119	1	1	2	0.7	2.9
F1CZ	0.021	4	0	4	0.7	6.0	**O1T3**	0.106	1	1	2	0.6	3.2
F1F3	0.05	3	0	3	0.7	4.2							
F1FZ	0.003	7	0	7	0.7	10.1	**C4P4**	0.575	0	0	0	0.6	0.0
F1P3	0.284	1	0	1	0.7	1.5	**F1O1**	0.575	0	0	0	0.7	0.0
F2C4	0.109	2	0	2	0.6	3.2	**F1T5**	0.575	0	0	0	0.7	0.0
F2CZ	0.001	15	1	16	0.7	24.6	**F2P4**	0.575	0	0	0	0.7	0.0
F2F4	0.043	3	0	3	0.7	4.6	**F2T6**	0.575	0	0	0	0.7	0.0
F2FZ	0.001	10	0	10	0.7	14.9	**F7P3**	0.575	0	0	0	0.7	0.0
F2T4	0.27	1	0	1	0.7	1.5	**F7T5**	0.575	0	0	0	0.7	0.0
F3C3	0.298	1	0	1	0.7	1.5	**FZCZ**	0.575	0	0	0	0.7	0.0
F3F4	0.043	3	0	3	0.6	4.8	**FZP3**	0.575	0	0	0	0.7	0.0
F3T5	0.29	1	0	1	0.7	1.5	**T5T6**	0.575	0	0	0	0.7	0.0

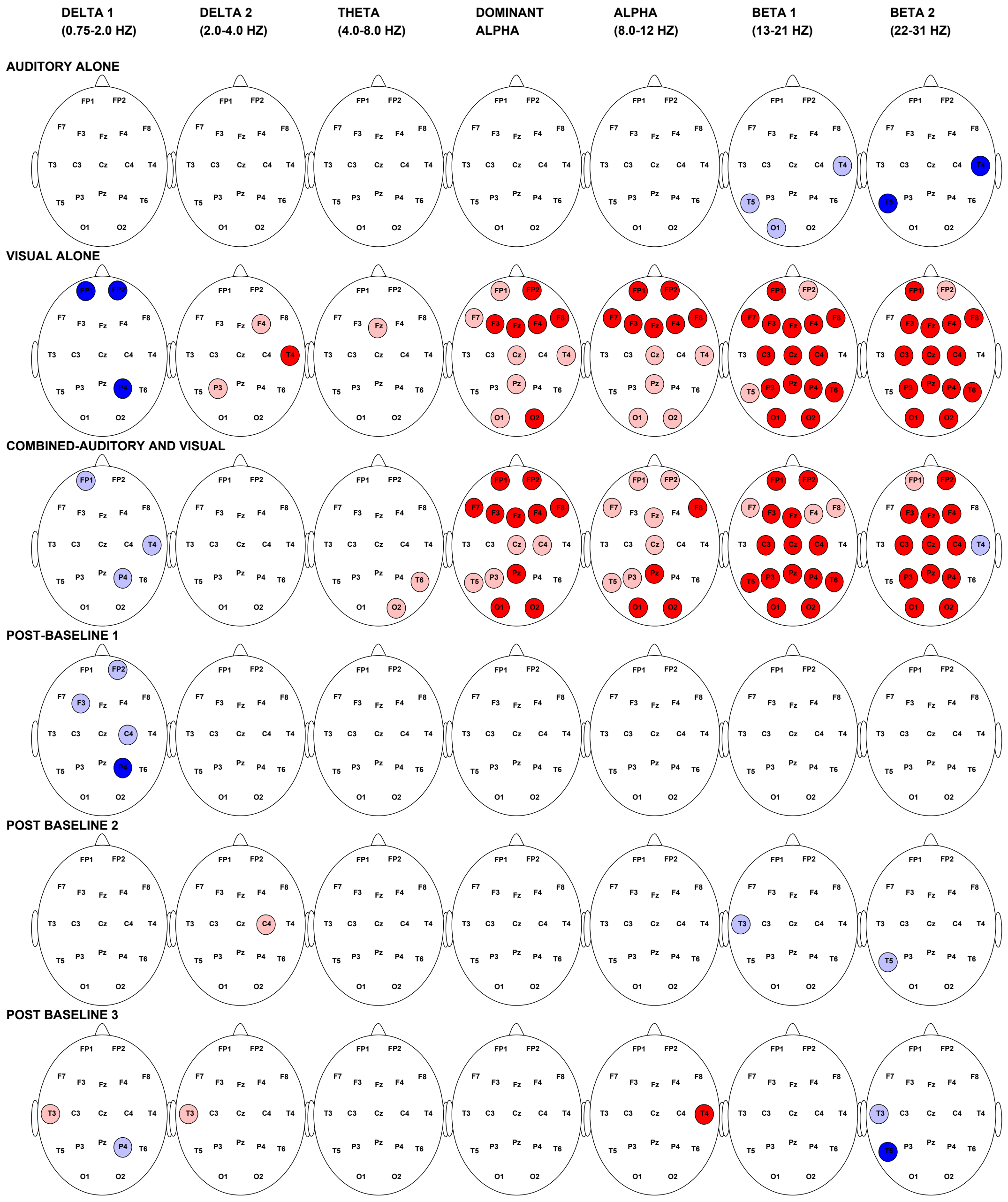

Figure 3. Effect of dominant alpha stimulation on EEG amplitude. Dark red circles, increases at p<=.01; light red circles, increases at p<=.05; dark blue circles, decreases at p<=.01; light blue circles, decreases at p<=.05.

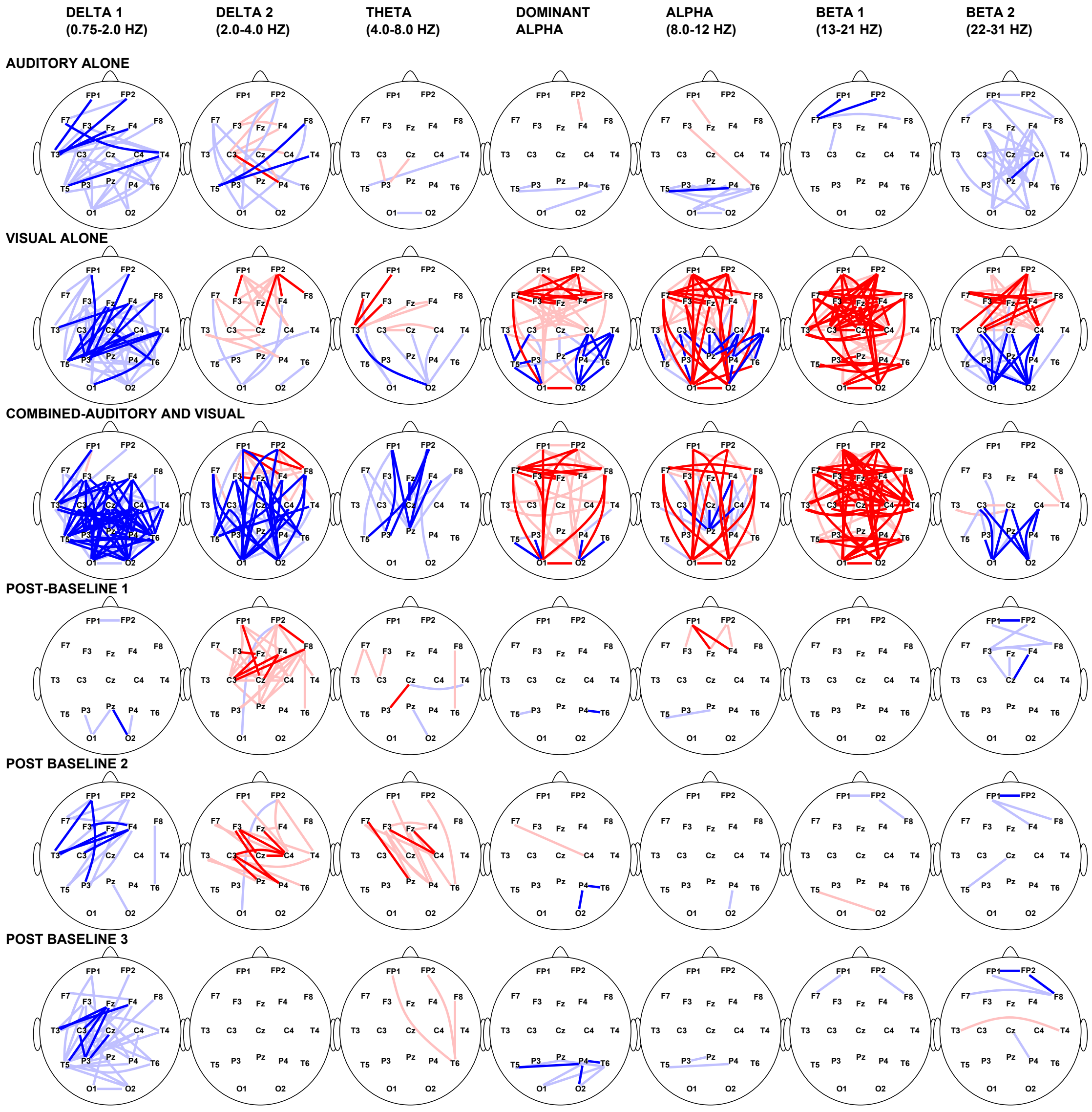

Figure 4. Effect of dominant alpha stimulation on EEG coherence. Dark red lines, increases at p<=.01; light red lines, increases at p<=.05; dark blue lines, decreases at p<=.01; light blue lines, decreases at p<=.05.

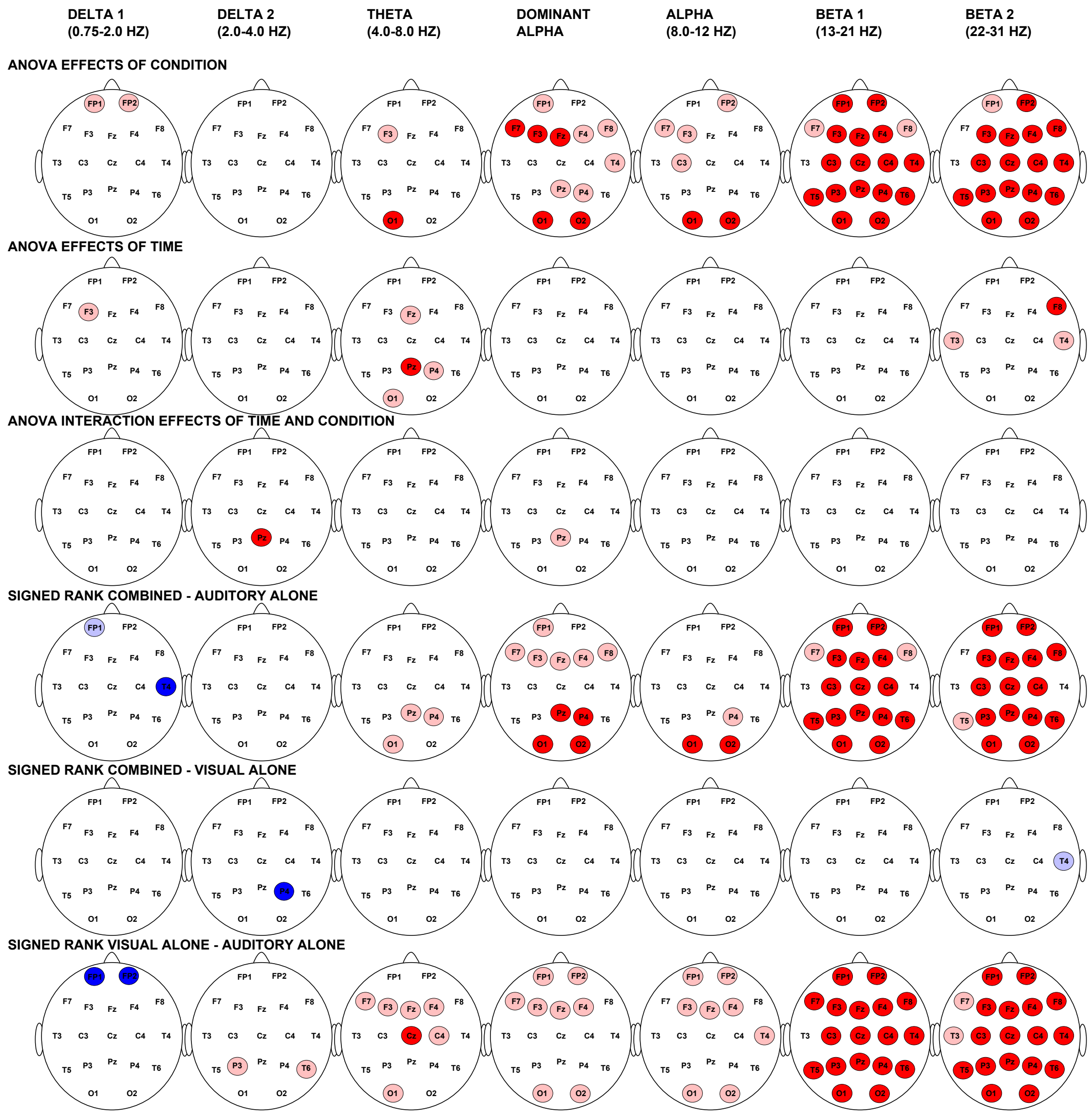

Figure 5. Effects of time and condition on EEG amplitude response to dominant alpha stimulation. For ANOVAs, dark red circles indicate effects significant at p<.01; light red circles, effects significant at p<.05. For signed rank tests, dark red circles indicate increases at p<=.01; light red circles, increases at p<=.05; dark blue circles, decreases at p<=.01; light blue circles, decreases at p<=.05.

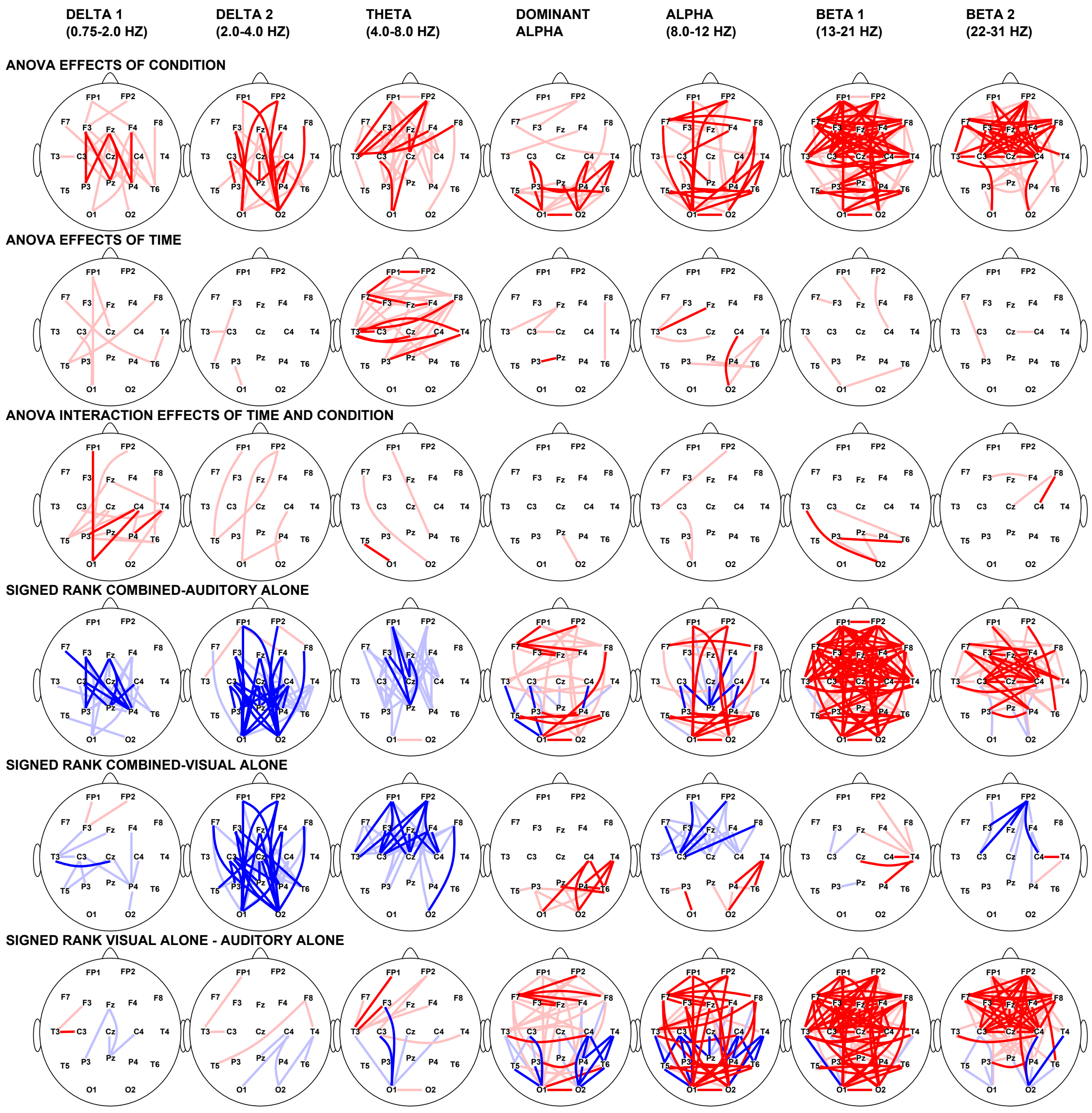

Figure 6. Effects of time and condition on EEG coherence response to dominant alpha stimulation. For ANOVAs, dark red lines indicate effects significant at p<.01; light red lines, effects significant at p<.05. For signed rank tests, dark red lines indicate increases at p<=.01; light red lines, increases at p<=.05; dark blue lines, decreases at p<=.01; light blue lines, decreases at p<=.05.

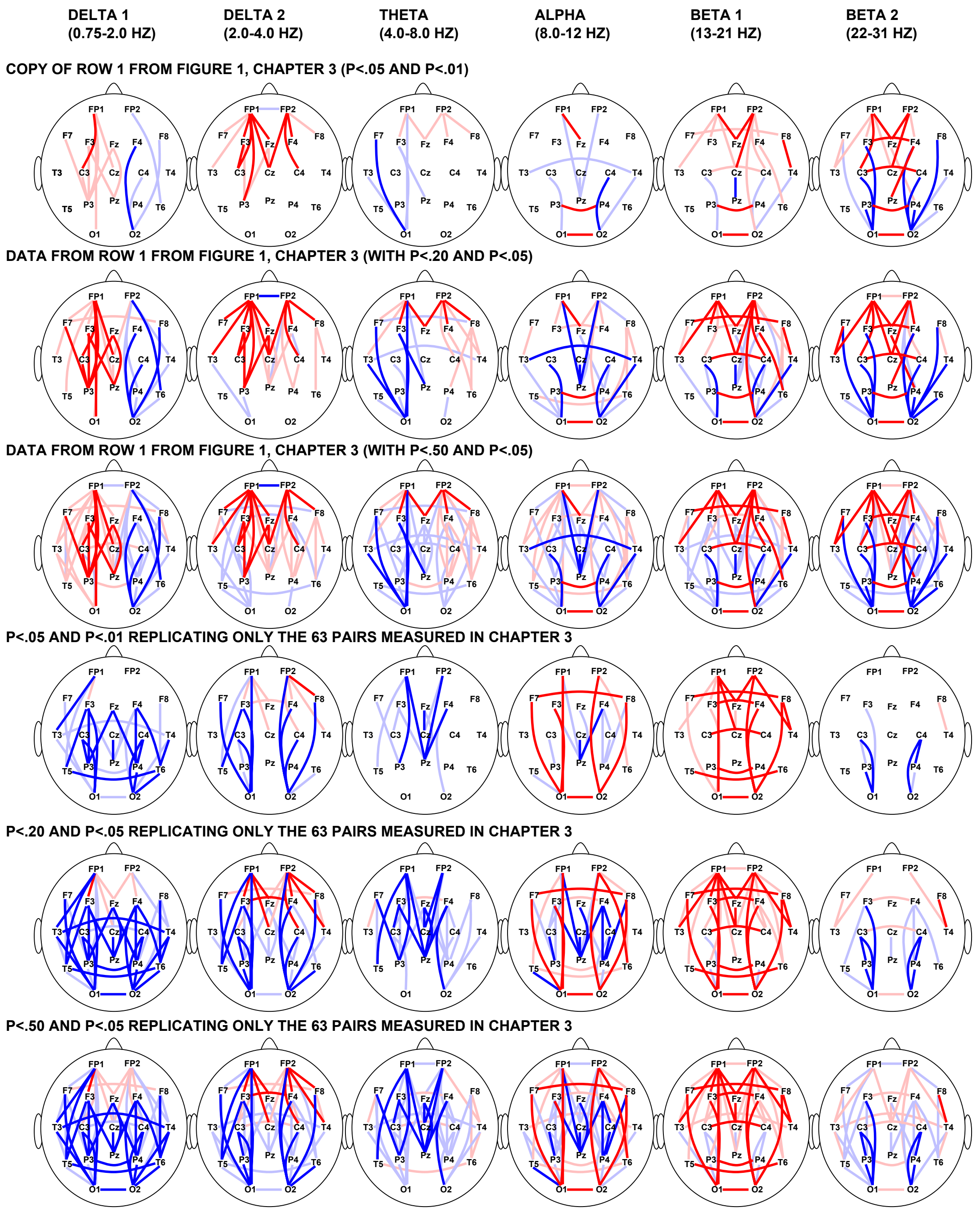

Figure 7. Effects of combined AVS at the dominant alpha frequency on EEG coherence. First three rows are data from the pilot study; bottom three rows are data from the replication study. Dark red lines, increases at p<=.01; light red lines, increases at p<=.05; dark blue lines, decreases at p<=.01; light blue lines, decreases at p<=.05.

Table 2. Contrasting number and proportion of observations between the present study and the pilot study in chapter 3. Variables "agree" when they were significant in both studies and change in the same direction. Variables "disagree" when they were significant in both studies and change in opposite directions. Variables were counted as "only pilot" or "only replication" if they were only significant in one stu and not the other. "Insignificant" variables were not significant in either study at the critical values shown.

number of obs.	agree	disagree	only pilot	only replication	insignificant
p<0.01	9	0	30	51	288
p<0.05	28	5	61	77	207
p<0.1	48	10	71	98	151
p<.2236	79	24	84	111	80
p<0.5	154	63	62	77	22
p<0.999	255	121	1	1	0
proportion of obs.					
p<0.01	0.02	0.00	0.08	0.13	0.76
p<0.05	0.07	0.01	0.16	0.20	0.55
p<0.1	0.13	0.03	0.19	0.26	0.40
p<.2236	0.21	0.06	0.22	0.29	0.21
p<0.5	0.41	0.17	0.16	0.20	0.06
p<0.999	0.67	0.32	0.00	0.00	0.00

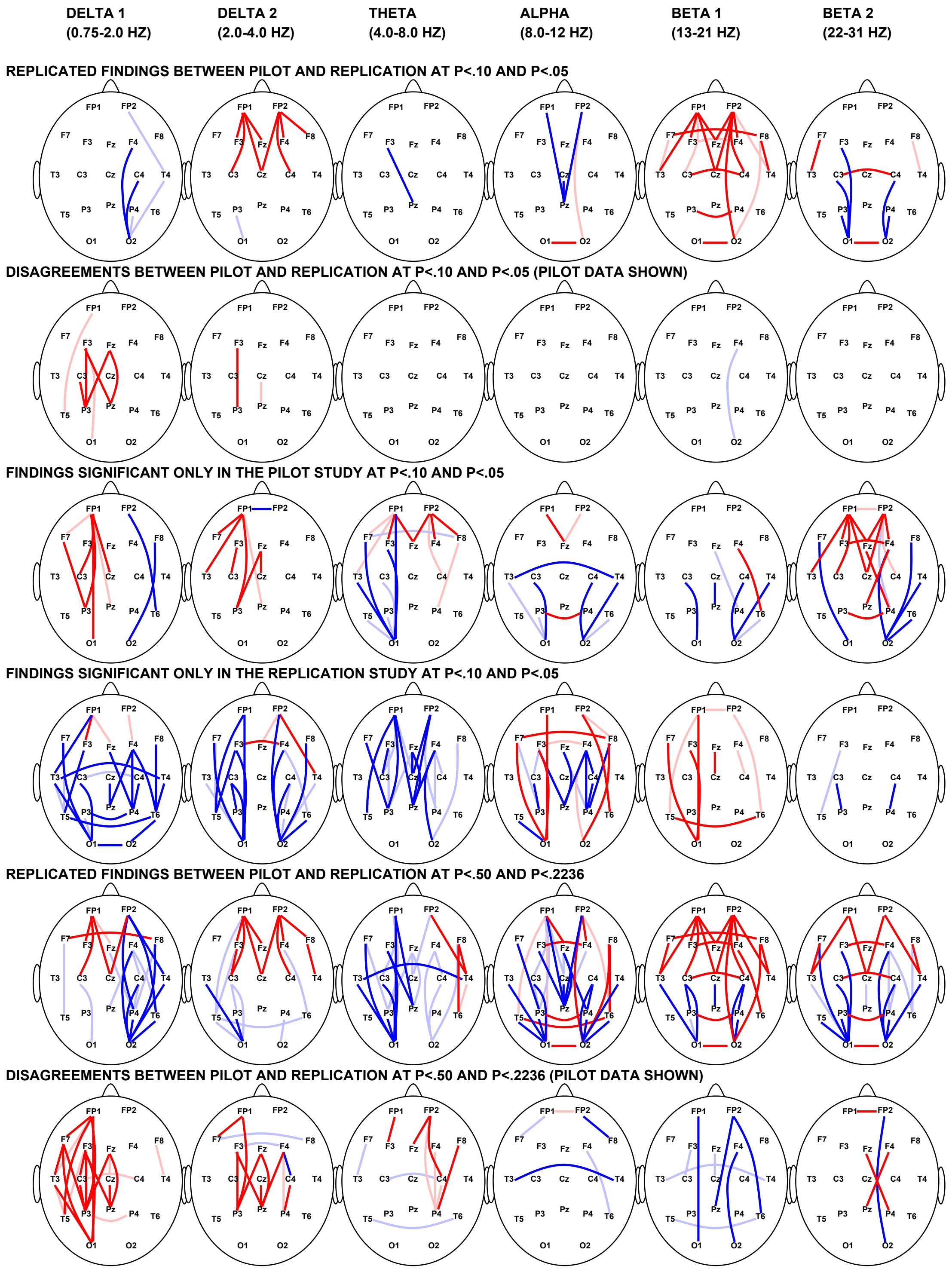
DELTA 1 (0.75-2.0 HZ)
DELTA 2 (2.0-4.0 HZ)
THETA (4.0-8.0 HZ)
ALPHA (8.0-12 HZ)
BETA 1 (13-21 HZ)
BETA 2 (22-31 HZ)
REPLICATED FINDINGS BETWEEN PILOT AND REPLICATION AT P<.10 AND P<.05
DISAGREEMENTS BETWEEN PILOT AND REPLICATION AT P<.10 AND P<.05 (PILOT DATA SHOWN)
FINDINGS SIGNIFICANT ONLY IN THE PILOT STUDY AT P<.10 AND P<.05
FINDINGS SIGNIFICANT ONLY IN THE REPLICATION STUDY AT P<.10 AND P<.05
REPLICATED FINDINGS BETWEEN PILOT AND REPLICATION AT P<.50 AND P<.2236
DISAGREEMENTS BETWEEN PILOT AND REPLICATION AT P<.50 AND P<.2236 (PILOT DATA SHOWN)
FP1
FP2
F7
F3
Fz
F4
F8
T3
C3
Cz
C4
T4
T5
P3
Pz
P4
T6
O1
O2

Figure 8. Effects of combined AVS at the dominant alpha frequency on EEG coherence. Comparison between the pilot and replication studies. Dark red lines, increases at p<=.01; light red lines, increases at p<=.05; dark blue lines, decreases at p<=.01; light blue lines, decreases at p<=.05 (two-tailed tests; divide p by two for one-tailed tests).

Vita

Jon Alan Frederick was born in Grand Rapids, MI in 1967. He graduated from the University of Utah in 1992 with a bachelor's in Philosophy and Biology, and a minor in chemistry. He obtained his master's degree in Neuroscience at the University of Michigan in 1995, emphasizing studies in psychopharmacology. After completing this dissertation, Jon Frederick will work as a post-doctoral fellow in the Neurosignal Analysis Laboratory, Center for Computational Biosciences, University of Texas Health Sciences Center.