Journal of Neuroscience Methods 347 (2021) 108965

Contents lists available at ScienceDirect

Journal of Neuroscience Methods

journal homepage: www.elsevier.com/locate/jneumeth

A user-friendly algorithm for adaptive closed-loop phase-locked stimulation

Cristian Rodriguez Rivero [a,1], Jochen Ditterich [a,b,*]

[a] Center for Neuroscience, University of California, Davis, United States
[b] Dept. of Neurobiology, Physiology & Behavior, University of California, Davis, United States

ARTICLE INFO

Keywords:
Local field potential
Oscillation detection
Phase prediction
Closed-loop phase-locked stimulation

ABSTRACT

Background: Closed-loop phase-locked stimulation experiments are rare due to the unavailability of user-friendly algorithms and devices. Our goal is to provide an algorithm for the detection of oscillatory activity in local field potentials (LFPs) and phase prediction, which is user-friendly and robust to non-stationarities in LFPs of behaving animals.

New method: We propose an algorithm that only requires specification of the frequency range within which oscillatory episodes are tracked. Frequency-specific detection thresholds and filter parameters are adjusted automatically based on the short-time LFP power spectrum. Estimates of instantaneous frequency and instantaneous phase are used for phase extrapolation, taking advantage of Bayesian estimation. We used real LFP signals, recorded from a variety of different species and different brain areas, as well as artificial LFP signals with known properties to assess the detection and prediction performance of our algorithm and three previously published reference algorithms under various conditions.

Results and comparison with existing methods: Our algorithm, while significantly more user-friendly than previous approaches, provides a solid detection and prediction performance over a wide range of realistic conditions and, in many cases, has a longer prediction horizon than the reference algorithms. Due to its ability to adjust to changes in the signal, the algorithm is well-prepared to deal with non-stationarities in oscillation frequency, even in the presence of multiple oscillation components.

Conclusions: We have created a universal algorithm for oscillation detection and phase prediction, which performs well and is user-friendly at the same time, making closed-loop phase-locked stimulation experiments easier to accomplish.

1. Introduction

Synchronized oscillatory activity across ensembles of neurons or brain areas has been proposed to play an important role in essential mechanisms like modulating functional connectivity ("Communication through Coherence" theory) (Fries, 2005) or memory retrieval (Watrous et al., 2013). The experimental evidence supporting these hypotheses, however, is largely correlational (Bosman et al., 2012; Gregoriou et al., 2009; Grothe et al., 2012). Proving a functional coupling between synchronous oscillatory activity and the proposed mechanisms would require a causal manipulation of the timing of neural activity in one neural ensemble relative to ongoing neural activity in another ensemble. In principle, closed-loop phase-locked stimulation should be able to at least inject time-locked artificial neural activity. Theoretical work suggests that phase-locked optogenetic stimulation might also be used to change the phase relationship between ongoing oscillations itself (Witt et al., 2013). Detecting and predicting oscillatory activity is challenging in awake behaving animals due to associated non-stationarities (oscillatory episodes tend to be short, the oscillation frequency can change over time, etc.) (Akam and Kullmann, 2014). Despite a number of algorithms for the detection and prediction of oscillatory activity having been published, some of them quite sophisticated (e.g., (Chen et al., 2013; Rutishauser et al., 2013), experimental studies taking advantage of closed-loop phase-locked stimulation are still rare. A closer look at the existing algorithms suggests that a potential reason could be that they are not particularly user-friendly. For example, the algorithm proposed by Chen et al. (Chen et al., 2013) requires the user to record a sample of the to-be-tracked oscillatory activity and submit it to a potentially lengthy offline optimization process to obtain optimal parameters for an autoregressive model. If the characteristics of the signal change over

* Corresponding author at: Center for Neuroscience, University of California, Davis, United States.
E-mail address: jditterich@ucdavis.edu (J. Ditterich).
[1] Present address: University of Amsterdam, Netherlands.

https://doi.org/10.1016/j.jneumeth.2020.108965
Received 3 June 2020; Received in revised form 28 September 2020; Accepted 29 September 2020
Available online 1 October 2020

time, the determined parameters might no longer be optimal. The algorithm proposed by Rutishauser et al. (Rutishauser et al. (2013)) requires the user to specify an absolute detection threshold for the presence of oscillatory activity as well as a fixed narrow frequency band that is going to be monitored. How these parameters should be chosen is not obvious, and even if the experimenter knew the current oscillation frequency, it could still change over time.

Here we propose a user-friendly algorithm for the detection of oscillatory activity in the local field potential (LFP) and the prediction of oscillation phase for the purpose of closed-loop phase-locked stimulation. The user only has to specify a range of frequencies within which oscillations are to be tracked and a statistical confidence level for the oscillation detection. All other parameters are determined automatically and adjust to the signal. The algorithm is therefore able to handle changes in signal properties like signal amplitude or oscillation frequency. We explain how the algorithm works and demonstrate that its detection and prediction performance can compete with or, in many cases, even outperform a selection of published algorithms. In addition to the already mentioned algorithms by Chen et al. and Rutishauser et al., we also evaluated a more recent algorithm by Mansouri et al. (Mansouri et al. (2017)), which the authors found to be competitive with the Chen et al. algorithm. A brief description of each of these reference algorithms as well as how they were adapted for the comparisons reported here can be found in the Materials and Methods section.

2. Materials and methods

2.1. Algorithm

The overall structure of our algorithm is shown in Fig. 1. The basic processing steps are:

1 Obtain an estimate of the background spectrum
2 Is there any activity in the frequency range of interest, which significantly exceeds the background spectrum? If so, an oscillatory episode is detected.
3 Obtain the range of frequencies exceeding the background spectrum and use it to specify the passband of a zero-phase bandpass filter
4 Apply the bandpass filter and a Hilbert transform to obtain the instantaneous phase
5 Use robust linear regression to obtain the phase at the end of the analysis window, assuming a constant oscillation frequency within the analysis window and avoiding edge artefacts
6 Obtain an estimate of the oscillation frequency from the peak location in the spectrum
7 Use Bayesian estimation to combine the current estimate with a prior distribution of recently observed oscillation frequencies
8 Use the estimated frequency for a linear phase extrapolation

The individual processing steps will now be explained in more detail:

The LFP signal (sampled at 1 kHz) is analyzed in short segments. We considered analysis windows ranging from 100 ms (corresponding to about 5 cycles of the fastest analyzed oscillations near 50 Hz) to 800 ms (corresponding to about 3.5 cycles of the slowest analyzed oscillations near 4.5 Hz). The window is then stepped forward for the next analysis. All of the results reported here are based on a step size of half the window length (with the exception of the detection delay analysis, where we are also considering smaller step sizes). The signal segment was multiplied with a single Slepian taper (which provided overall better results than using a Hamming or Hanning window; data not shown). Fig. 2(A) shows an example of a signal segment before (black) and after applying the taper (blue). We then obtained the power spectrum using a 1024-point FFT. The combination of applying the taper and obtaining the spectrum was achieved by calling the function "mtspectrumc" from the Chronux toolbox (http://chronux.org) with "tapers = [11]" (Mitra and Bokil, 2008). For detecting the presence of oscillations we adopted an idea from BOSC, an established algorithm for offline analysis of oscillatory activity (Hughes et al., 2012). The LFP power spectrum in the absence of any oscillatory activity is expected to resemble a power-law function (colored noise), which corresponds to a line in log f vs. log power space. Oscillatory activity would add a bump on top. Hughes et al. (Hughes et al., 2012) used standard linear regression to estimate the background spectrum. Since our short-time spectra are substantially bumpier than the wavelet-based spectra in BOSC, we use robust linear regression (MATLAB's "robustfit" function), applied to the power spectrum for frequencies ranging from 2 to 100 Hz, to estimate the background spectrum, which is also less prone to be biased by the presence of oscillatory activity. An example of this procedure is shown in Fig. 2(B). Actual power values in the absence of oscillatory activity are expected to be χ^2 distributed around the mean given by the fit (Hughes et al., 2012). The tail of the distribution can therefore be used to define a frequency-dependent detection threshold for oscillatory activity. Power values exceeding this threshold are assumed to reflect oscillations. Determining the threshold location requires picking a statistical confidence level, which has to be provided by the user of the algorithm. After evaluating the algorithm's detection performance for a variety of confidence levels, we settled on a single value of 0.998, which was used for all datasets and all results reported in this manuscript. Since all FFT frequencies in the user-specified range of interest are tested for threshold crossing, we also apply Bonferroni correction. The red line in Fig. 2(B) shows an example of such a frequency-dependent detection threshold.

Power values exceeding the threshold are indicators of possible oscillations. To further reduce spurious threshold crossings, we required at least two neighboring frequencies to exceed threshold, which was interpreted as detecting the presence of oscillatory activity. A group of neighboring frequencies exceeding threshold was treated as a candidate oscillation. In case there was more than one such group in the frequency range of interest, we prioritized the group with the largest number of neighboring frequencies as the most robust oscillation. In case there

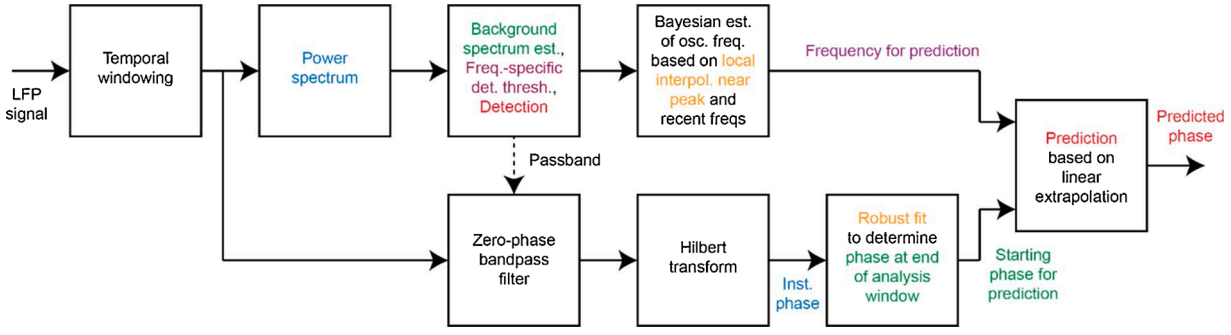

Fig. 1. Overall structure of the proposed algorithm for detecting the presence of oscillatory activity and predicting its phase. The text colors correspond to the colors of the illustrations in Fig. 2.

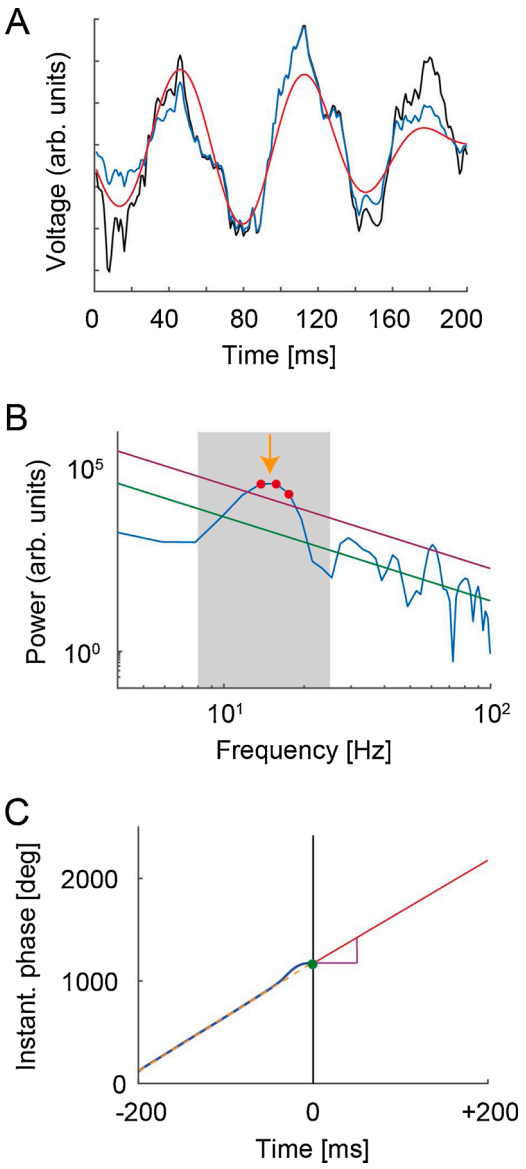

Fig. 2. Illustration of signal processing, oscillation detection, and phase prediction (based on a signal segment taken from real Dataset 1). (A) Windowing and filtering. Signal segment before (black) and after (blue) applying the Slepian taper. The red trace shows the signal after applying the zero-phase bandpass filter. (B) Oscillation detection. The blue trace is the short-time power spectrum. The grey shaded area indicates the frequency range of interest (ROI). The green line is the estimate of the background spectrum based on robust linear regression. The red line is the frequency-dependent detection threshold. The red points mark the frequencies in the ROI, where the power spectrum exceeded the threshold. The orange arrow indicates the estimated oscillation frequency based on a local interpolation of the power spectrum near the peak. (C) Phase prediction. The blue trace is the instantaneous phase in the analysis window. The dashed orange line reflects a robust linear regression. The phase on this line at the end of the analysis window (green dot) is used as the starting phase for the phase prediction. The predicted phase (red line) is a linear extrapolation; the slope is determined by a Bayesian estimate of the oscillation frequency (purple).

were multiple groups with the same number of frequencies, we prioritized the candidate oscillation that exceeded threshold power by the largest amount. The range of frequencies in the selected group was stored for later use in the algorithm.

This frequency range was used to specify the passband of a bandpass filter. The lower cutoff frequency of a 2nd-order Butterworth filter was set to the lowest frequency minus the frequency resolution of the FFT (about 1 Hz), the upper cutoff frequency to the highest frequency plus the frequency resolution of the FFT. The signal segment was then convolved twice with the filter kernel, once in the forward and once in the backward direction using MATLAB's "filtfilt" function, resulting in a zero-phase 4th-order bandpass filter. Thus, the algorithm uses an adaptive filter that passes the selected frequencies exceeding threshold through, without inducing any systematic phase shifts. The red line in Fig. 2(A) shows the result of this filtering process. The filtered signal was then subjected to a Hilbert transform to obtain the instantaneous phase. An example of such an instantaneous phase signal is shown in Fig. 2(C). Since the instantaneous phase is prone to edge artefacts, we used another robust linear regression to obtain a phase estimate based on the assumption of a constant oscillation frequency in the analysis window. Initially, we used the resulting line directly for linearly extrapolating the phase into the future. We also considered a robust 2nd order fit, which would allow the oscillation frequency to change over time, but did not find a clear advantage and therefore stayed with the simpler linear model (data not shown). Thus, our phase prediction is based on the assumption that the oscillation frequency will remain stationary in the very near future.

Overall, however, we observed better prediction performance when estimating the oscillation frequency from the spectrum rather than the slope of the robust fit to the instantaneous phase. Thus, the robust fit is only used to obtain a reliable phase estimate at the end of the analysis window (beginning of the prediction window), which we name φ_0. An estimate of the current oscillation frequency is obtained from the location of the peak in the spectrum in the selected range of frequencies during detection. To be able to obtain estimates in between the discrete FFT frequencies we adopted the Gaussian local approximation approach proposed by (Gasior and Gonzalez, 2004). Assuming that S[0] is the peak power in the discrete spectrum in the selected range, and S[-1] and S[+1] are the power values at the left and right neighboring frequencies, the inferred true peak location (as an index ranging from -1 to +1) is given by

$$k_p = \frac{\ln\left(\frac{S[+1]}{S[-1]}\right)}{2\ln\left(\frac{S[0]^2}{S[+1]\cdot S[-1]}\right)}$$

If $f_{discrete}$ is the frequency associated with the peak in the discrete spectrum, the estimate of the oscillation frequency $\widehat{f}_{osc}$ can be calculated as

$$\widehat{f}_{osc} = f_{discrete} + k_p \cdot \Delta f$$

with Δf being the frequency resolution of the FFT. The variance $\widehat{\sigma}^2_{osc}$ of the local Gaussian interpolation can be determined to be

$$\widehat{\sigma}^2_{osc} = \frac{\Delta f^2}{\ln\left(\frac{S[0]^2}{S[+1]\cdot S[-1]}\right)}$$

which is needed in the next step. While $\widehat{f}_{osc}$ could be used directly for the phase prediction, it is a noisy estimate, and it turns out that the overall prediction quality can be further improved (data not shown) by taking advantage of the oscillation frequency typically showing some continuity over time. This can be exploited to obtain a Bayesian estimate of the oscillation frequency by combining the current frequency estimate with a prior distribution given by recently observed oscillation frequencies. Ideally, the number of recent observations contributing to the prior should be tailored to how (non-)stationary the oscillation frequency is. Since this information is typically not available, we used a fixed number. After initial tests with the most recent 10 and 20 observations, which resulted in comparable prediction performance (data not shown), we settled on the most recent 15 observations as a compromise, which has been used for all results reported in this paper. If we take the

prior to be a normal distribution with mean f_{prior} and variance σ^2_{prior} given by the sample mean and sample variance of the 15 most recently observed oscillation frequencies (estimates resulting from the local Gaussian interpolation near the peak of the spectrum), the a posteriori estimate of the oscillation frequency $\widehat{f}_{Bayes}$ is given by

$$\widehat{f}_{Bayes} = \frac{\widehat{f}_{osc} \cdot \sigma^2_{prior} + f_{prior} \cdot \widehat{\sigma}^2_{osc}}{\sigma^2_{prior} + \widehat{\sigma}^2_{osc}}$$

Thus, phase (in degrees) was predicted to be

$$\varphi(t) = \varphi_0 + \widehat{f}_{Bayes} \cdot t \cdot 360^\circ$$

with time zero corresponding to the end of the analysis window, which is also the beginning of the prediction window.

2.2. Methods for evaluating Algorithm Performance

2.2.1. Datasets

We needed suitable datasets to evaluate the performance of the algorithm. Artificial datasets have the advantage that parameters like oscillation frequency, signal-to-noise ratio (SNR), oscillation episode duration, and non-stationarities in the oscillation frequency can be tightly controlled. Furthermore, the ground truth about the presence/absence of oscillatory activity and the current phase is known. However, the algorithm is supposed to be applied to real LFPs, which might have properties that deviate from the properties of artificial data. Real LFP recordings have the disadvantage though that the ground truth about whether there is currently ongoing oscillatory activity and what the current phase of the oscillation is, is not known. We therefore took a hybrid approach, evaluating the performance of the algorithm using a combination of artificial and real data.

Artificial datasets were constructed by starting with pink noise and embedding oscillatory activity. Oscillation frequencies ranged from 4.5–47 Hz and were either constant, allowed to change between oscillatory episodes (within a particular range, either 10…20 Hz or 20…40 Hz), or even within oscillatory episodes (combination of linear frequency drift with a random slope up to 7.5 Hz/s and random fluctuations according to Brownian motion with a diffusion coefficient of 22.5 Hz^2/s). The SNR ranged from -16 to +12 dB (total signal power vs. total noise power). Oscillatory episodes could be either long (3 s) or short (3–12 cycles). We also considered signals with two simultaneously embedded oscillations at two different frequencies and different SNR (-9 and -2 dB) to test whether the algorithm could track the stronger oscillation component.

The real datasets were LFP recordings from different species (humans, nonhuman primates, and rodents) and different brain areas (visual cortex, parietal cortex, hippocampus, and thalamus) and had oscillation frequencies ranging from 6 to 50 Hz. SNR (following the same definition as in the case of the artificial datasets) was typically between -10 and 0 dB. A monkey LFP recording from parietal cortex was provided by the Ditterich Lab (Dataset 1; oscillation frequency: 15 Hz), a monkey LFP recording from the thalamus was contributed by the Usrey Lab (UC Davis; Dataset 2; oscillation frequency: 12 Hz), and rat LFP recordings from the hippocampus were provided by the Gurkoff Lab (UC Davis; Datasets 3 and 4; oscillation frequency: 8 Hz). Human ECoG recordings from visual cortex were from (Hermes et al., 2015) and are available as Supplementary Data (Datasets 5 and 6; oscillation frequencies: 40…50 Hz). Human iEEG recordings from hippocampus were from (Ekstrom et al., 2005) and requested through the Kahana Lab's Cognitive Electrophysiology Data Portal (Datasets 7 through 9; oscillation frequencies: 6…16 Hz). Datasets 8 and 9 were originally the same dataset, which contains two oscillatory components, one at 6 Hz, one at 16 Hz. In Dataset 8 we are testing how well the component with the lower frequency can be tracked (by having extracted reference information for the lower frequency component and setting algorithm parameters such that this component should be evaluated; in this case frequency ranges were set such that they only included one of the oscillation frequencies); in Dataset 9 we are testing how well the component with the higher frequency can be tracked (by having extracted reference information for the higher frequency component and setting algorithm parameters such that this component should be evaluated). Table 1 provides a summary of the real datasets.

2.2.2. Generating reference information for real datasets

While information about the presence/absence of oscillatory activity as well as the oscillation phase at any given time is known a priori for the artificial datasets, such reference information is not available for the real datasets and needs to generated. We used the BOSC algorithm (Hughes et al. (2012)) to determine the presence of oscillatory episodes in the LFP recordings. MATLAB code is available as Supplementary Material to (Whitten et al., 2011). The confidence level was set to 0.95, and oscillatory episodes had to be at least three cycles long to be detected. The background spectrum is estimated using the complete dataset. Since the BOSC algorithm is wavelet-based, it requires a minimum, frequency-dependent signal length. Some of our real datasets contained only one-second-long recording chunks, which are too short for BOSC at low frequencies. We therefore developed an alternative method for obtaining the spectrum based on a windowed 512-point FFT. The window length was chosen to be about 2.5 cycles of the dominant oscillation frequency, and a single Slepian taper was used. The background spectrum was obtained by stepping the window through the dataset without overlaps, the time-resolved spectrum for detecting oscillatory activity was obtained by sliding the window over the signal in 1 ms steps. Once the time-resolved spectrum was computed, the algorithm followed the same detection logic as the wavelet-based BOSC algorithm.

For obtaining the reference phase, for each detected oscillatory episode, we found the frequency with the largest power and defined a zero-phase 4th order bandpass filter with the passband ranging from the next lower to the next higher analyzed frequency. Thus, when using the FFT, the passband was about 4 Hz wide. The wavelet-based approach uses a logarithmic frequency spacing, so the width of the passband was frequency-dependent. The filter was applied to the signal segment corresponding to the oscillatory episode, and a Hilbert transform was used to obtain the instantaneous phase. We only kept the longest continuous phase segment with a positive derivative. Decreasing phase indicates an artefact that sometimes occurs, primarily at the edges.

Table 1
Properties of real LFP datasets. "?" indicates that detailed information was not available.

Dataset #	Species	Brain area	Task	Oscillation frequency near
1	Rhesus monkey	Parietal cortex (LIP)	Perceptual decision-making	15 Hz
2	Rhesus monkey	Thalamus	Visospatial attention	12 Hz
3	Rat	Hippocampus (CA1)	?	8 Hz
4	Rat	Hippocampus	?	8 Hz
5	Human (ECoG)	Visual cortex	Visual stimulation	50 Hz
6	Human (ECoG)	Visual cortex	Visual stimulation	40 Hz
7	Human (ECoG)	?	Virtual navigation	6 Hz
8	Human (ECoG)	?	Virtual navigation	6 Hz and 16 Hz (reference phase extracted for 6 Hz)
9	Human (ECoG)	?	Virtual navigation	6 Hz and 16 Hz (reference phase extracted for 16 Hz)

2.2.3. Evaluation of the detection and prediction performance

Our goal was to come up with simple metrics for quantifying the detection and prediction performance of the algorithms. Each time an analysis is performed, the algorithm has to make a decision about the presence of oscillatory activity. There are two possible outcomes: "oscillation detected" or "no oscillation detected". We compared this against the reference information about whether oscillatory activity was present at the beginning of the prediction window or not. Again, there are two possible states: "oscillation present" or "no oscillation present". We defined the **Detection Performance** (DP; or Detection Accuracy) as the relative frequency of the algorithm making a decision that was consistent with the reference information (i.e., either "oscillation detected" & "oscillation present" or "no oscillation detected" & "no oscillation present"); i.e., it can be calculated as the sum of true positives (TP; correct detections) and true negatives (TN; correct rejections), divided by the total number of detection decisions (DD):

$$DP = \frac{TP + TN}{DD}$$

The two remaining combinations indicate two different types of mistakes:

- False positives: "oscillation detected" & "no oscillation present"
- Misses: "no oscillation detected" & "oscillation present"

To quantify the prediction performance, we analyzed the phase prediction error. Each individual phase prediction can be compared with the reference information, given an oscillation was present. We let the algorithms predict the phase for 800 ms. The reference phase was obtained for the same amount of time, if available, or until the end of the current oscillatory episode if it ended earlier. Since oscillatory episodes can be multiple cycles long, we had to work with the unwrapped phase. As a consequence, predicted phase and reference phase can be offset by multiples of 360°. Thus, we first adjusted the predicted phase to minimize the (absolute) difference at the beginning of the prediction window. The absolute phase error (as a function of time into the future) was obtained by calculating the difference between predicted phase and reference phase and taking the absolute value. Averaging across individual predictions results in the expected absolute phase error. (We averaged for each time point independently, using all available absolute phase errors for that time point.) We defined the **Prediction Performance** (or Prediction Horizon) as the time horizon, for which the expected absolute phase error stayed below a critical value. This value can be chosen arbitrarily. We decided to go with 90°, a quarter of a cycle, reasoning that stimulation that was more than a quarter cycle off target would not really be considered phase-locked anymore. Fig. 3(A) shows an example time course of the expected absolute phase error for our algorithm for one of the artificial datasets (oscillation frequency of 14 Hz, SNR of -2 dB, short oscillatory episodes, analysis window size of 400 ms). To provide more insight into how the expected absolute phase error evolves over time in the case of our algorithm, we extracted two additional critical values: for how long it stayed below 60°, and for how long it stayed below 30°. The Prediction Performance (time horizon below 90°) will be shown as thick solid lines for all algorithms in the prediction performance plots in the *Results* section. The additional time horizons below 60° (dashed thin red line) and below 30° (dotted thin red line) will only be shown for our algorithm.

2.2.4. Measuring the detection delay

To measure how much time typically passes between the onset of oscillatory activity and when it is first detected by our algorithm, we created additional artificial LFP signals with no oscillatory activity at the beginning (only pink noise), but oscillatory activity starting at a random onset time (with a random phase) and then remaining present. We used six different oscillation frequencies (4.5, 9, 14, 22, 33, and 47 Hz), the corresponding optimal analysis windows that will be determined later in this article (800, 400, 400, 200, 200, and 100 ms, respectively), two different SNRs (-2 and +5 dB), and three different window step sizes (50 %, 25 %, and 10 % of the window size). Fig. 3(B) shows an example of a signal with an oscillation frequency of 14 Hz, an SNR of -2 dB, and a window step size of 200 ms (50 % of the analysis window size of 400 ms). We measured the time difference between the end of the analysis window, when the oscillation was first detected, (shown in green in Fig. 3(B)) and the oscillation onset time (dashed black line and black arrow). The procedure was repeated 1000 times for each condition, and the median detection time will be reported in *Results*.

2.2.5. More detailed analysis of the ability to separate two frequency components

To gain more detailed insight into our algorithm's ability to separate two frequency components, we created additional artificial LFP signals with two frequency components embedded in pink noise. The two components were either centered around 11 Hz, with possible absolute frequency differences of 1, 2, 3, 4, or 5 Hz, or centered around 22 Hz, with possible absolute frequency differences of 1, 2, 3, 4, 5, 6, 7, 8, or 9 Hz. The SNR of the stronger component was always -2 dB, the weaker component had an SNR of either -5 or -8 dB. We used the corresponding optimal analysis windows that will be determined later in this article (400 ms for the center frequency of 11 Hz, 200 ms for the center frequency of 22 Hz). The frequency range of interest was set to 6.5…15.5 Hz for the center frequency of 11 Hz and to 15.5…28.5 Hz for the center frequency of 22 Hz. We ran our algorithm on each of these cases until it had made 1000 detections. For each of these detections we determined whether the bandpass filter had been placed such that the frequency of the stronger component was inside the passband of the filter and the frequency of the weaker component was outside the passband of the filter. If so, we called it a successful separation. How the relative frequency of successful separations depended on the different

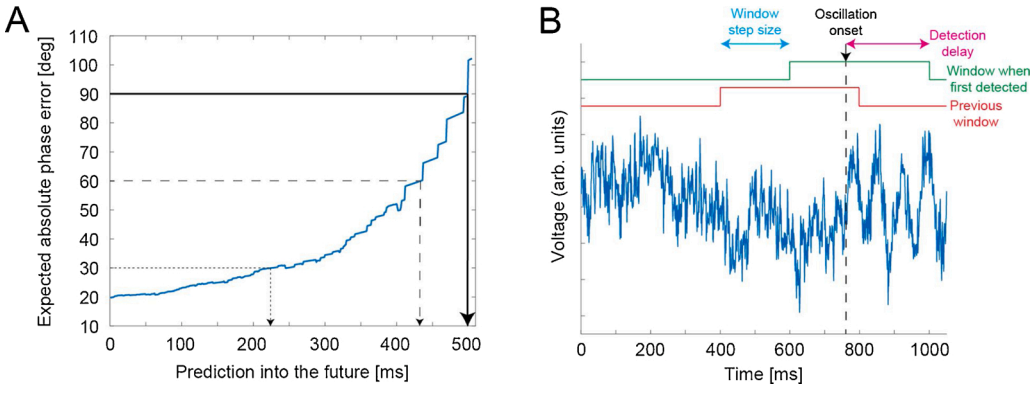

Fig. 3. Analysis methods. (A) Prediction horizon. Example time course of the expected absolute phase error (blue) for our algorithm for an artificial dataset (see text). The thick solid arrow indicates the Prediction Performance (time horizon below 90°), the dashed thin arrow the time horizon below 60°, and the dotted thin arrow the time horizon below 30°. (B) Detection delay. Example artificial LFP signal (blue); the oscillation onset is marked with a dashed black line and a black arrow. The analysis window where the oscillation was first detected is shown in green. The measured detection delay is indicated in magenta.

parameters will be reported in *Results*.

2.2.6. Reference algorithm 1: "AR"

The first algorithm we used as a reference for comparing the detection and prediction performance was (Chen et al. (2013)). The MATLAB code is available here: https://github.com/transcranial/cortical-stim. By default, the code uses a fixed analysis window length of one second. We adjusted the code to also work with other window lengths, which was necessary to evaluate the performance at higher oscillation frequencies. The way the algorithm works poses a constraint on the length of the analysis window: it has to be longer than 1.25 times the cycle length at the lowest possible frequency of interest. As a consequence, we imposed this constraint on all compared algorithms and never used analysis window lengths shorter than this limit.

Details about how the algorithm works can be found in (Chen et al., 2013). Briefly, the algorithm obtains an autoregressive (AR) model-based power spectral density for the current signal segment. The existence of a local peak in the power spectrum is interpreted as oscillatory activity being present (detection). An optimal frequency range with high power concentration is used to define the passband of a zero-phase bandpass filter. The central part of the filtered signal (an offline optimization process determines how long this part is; see below) is used to estimate an AR model, which is then used to predict how the time series is expected to continue (until the end of the analysis window and, for the same amount of time, into the prediction window). Only this predicted part is then subjected to a Hilbert transform to obtain the instantaneous phase and the instantaneous frequency at the end of the analysis window (beginning of the prediction window). The predicted phase starts with this instantaneous phase and continues to increase linearly with this instantaneous frequency.

A number of parameters for this algorithm are selected in an offline optimization process based on a sample of the to-be-tracked signal. For each of our datasets, we hand-selected a representative signal segment with clearly present oscillatory activity and subjected it to an optimization process for each analysis window length that was used for this particular dataset.

2.2.7. Reference algorithm 2: "Rutishauser"

The second algorithm that was used as a reference is (Rutishauser et al. (2013)). The MATLAB code is available here: https://github.com/StimOMatic/StimOMatic. In this algorithm, the signal is sent through a fixed zero-phase 8th order bandpass filter. We used a (total) width of the passband of 6 Hz as in the example provided in (Rutishauser et al. (2013)). The StimOMatic code uses a default of 10 Hz, which would not have allowed center frequencies below 5 Hz (see below). The center frequency has to be provided by the user. The filtered signal is submitted to a Hilbert transform to obtain the instantaneous phase as well as the instantaneous power. If the average power during the last 50 ms of the analysis window exceeded a fixed detection threshold (which also has to be provided by the user), an oscillation was detected. The frequency of the oscillation is determined by finding the time difference between successive peaks in the filtered time series. The predicted phase starts with the instantaneous phase at the end of the analysis window and then linearly increases with the oscillation frequency.

For all artificial datasets with fixed oscillation frequencies the bandpass center frequency was set to the actual oscillation frequency. For artificial datasets with varying oscillation frequencies the bandpass center frequency was set to the center of the frequency range. For the real datasets the center frequency was set to the location of the major oscillation-related bump in the overall power spectrum. Since the "Rutishauser" algorithm does not provide a recipe for selecting the detection threshold, we ran the algorithm with 9 different individualized thresholds for each dataset. We first created a histogram of the distribution of power in the frequency band of interest across the whole dataset. An algorithm then selected 9 different values, spanning most of the range of the histogram (from 2/3 of the median power to the midpoint between the median power and the largest observed power). Once the detection results were available for all 9 thresholds, we picked the threshold maximizing the difference between the Detection Performance and the false positive rate. When continuously lowering the detection threshold, the Detection Performance initially increases, but then tends to plateau, whereas the false positive rate starts to increase. Our goal was to catch the sweet spot with nearly optimal Detection Performance and low false positive rate. The results reported for the "Rutishauser" algorithm in this paper are the ones for this particular detection threshold, optimized for each dataset.

2.2.8. Reference algorithm 3: "Mansouri"

The third algorithm that was used as a reference is (Mansouri et al. (2017)). MATLAB code was obtained from the authors. In contrast to the other algorithms, the "Mansouri" algorithm does not have an oscillation detection mechanism. We therefore cannot report a Detection Performance. We had the algorithm always make a prediction and compared it to a reference phase whenever available.

The algorithm uses a 20th order elliptic bandpass filter to extract the frequency range of interest and then performs a high resolution (0.05 Hz) FFT. The frequency with the largest amplitude is selected, and this frequency and the associated phase are used directly for a linear phase extrapolation. Since the algorithm does not make use of a zero-phase filter, the filter delay should be corrected for. The authors' code allowed for a manual correction. We added code to automatically determine and compensate for the filter's phase delay at the selected frequency.

The filter passbands were identical to the frequency ranges of interest used with our own algorithm and picked such that, for artificial datasets with a fixed oscillation frequency, they contained the actual frequency and neighboring frequencies (depending on the actual oscillation frequency between 6 and 16 Hz wide). For artificial datasets with varying oscillation frequency, the frequency range of interest spanned the full range of possible oscillation frequencies, plus neighboring frequencies (between 19 and 29 Hz wide). For the real datasets, we obtained the power spectrum across the whole file and made sure that the major oscillation-related bump was fully included in the frequency range of interest.

2.2.9. Average detection and prediction performance across analysis window sizes

Since it is not a priori clear which analysis window size should be used for a particular dataset, we evaluated the performance for a range of window sizes and report the average performance. The window sizes that were used are: 100, 200, 400, 600, and 800 ms. Whenever possible, all of these window sizes were used with the following exceptions:

- The "AR" algorithm requires a minimum window length depending on the lowest possible oscillation frequency (see "Reference algorithm 1").
- A few of our real datasets consisted of individually recorded trials with a length of one second. We did not use the 800 ms analysis window for these datasets, as only 200 ms would have remained for analyzing the predicted phase.

For a particular dataset, the same set of analysis window sizes was always used across all evaluated algorithms.

2.2.10. Statistical analysis of the detection and prediction performance

To test whether the Detection and Prediction Performance were significantly different across algorithms and whether they were significantly affected by particular parameters like the oscillation frequency or the SNR, we applied mixed-design ANOVAs, with the algorithm as the repeated measures factor, and the parameter on the horizontal axis of a particular plot as the between-groups factor. In case of a significant main

effect of the algorithm, we followed up with a post-hoc test (Tukey's HSD test, a multiple comparison procedure controlling the family-wise error rate) to reveal which pairwise differences were the drivers of the main effect.

3. Results

In this section we will evaluate the oscillation detection and phase prediction performance of our algorithm using both artificial and real LFP datasets and make comparisons with three previously published reference algorithms. First, we will have a look at artificial datasets with different properties, for which the ground truth about the presence of oscillatory activity at any given point in time and the oscillation phase is known.

3.1. Detection and prediction performance across oscillation frequencies

The detection performance for datasets with different oscillation frequencies is shown in Fig. 4(A). Each datapoint represents the average over different datasets with SNRs ranging from -16 to +12 dB and both short and long episode durations. Across the range of studied oscillation frequencies, from 4.5–47 Hz, our algorithm (red) provided a better detection performance than both the Rutishauser (green) and the AR algorithm (blue). The mixed-design ANOVA (see *Methods for evaluating algorithm performance*) revealed a significant main effect of the algorithm ($p = 9{\cdot}10^{-6}$). Tukey's HSD test indicated two significant pairwise differences: Our algorithm detected better than both the AR ($p = 10^{-4}$) and Rutishauser ($p = 0.005$) algorithms. The corresponding prediction performance is shown in Fig. 4(B), thick solid lines. Again, across the range of studied oscillation frequencies, our algorithm (red) provided a better prediction performance than the AR (blue), Rutishauser (green), and Mansouri (cyan) algorithms, with the AR algorithm showing comparable performance at 4.7 Hz, but then falling behind the other algorithms at higher frequencies. The ANOVA indicated a significant main effect of the algorithm ($p < 10^{-6}$) as well as a significant interaction between frequency and algorithm ($p < 10^{-6}$). Tukey's HSD test indicated four significant differences: Our algorithm predicted better than the AR ($p = 10^{-4}$), Rutishauser ($p = 10^{-4}$), and Mansouri ($p = 10^{-4}$) algorithms; the Rutishauser algorithm performed better than the AR algorithm ($p = 0.026$). The dashed and dotted red lines mark the time horizons for the expected absolute phase error staying below 60° and 30°, respectively, to provide additional insight into how the phase error evolves over time (only shown for our algorithm).

3.2. Detection and prediction performance across signal-to-noise ratios

The detection performance for datasets with different SNRs is shown in Fig. 5(A). Each datapoint represents the average over different datasets with frequencies ranging from 4.5–47 Hz and all long episode durations (as we had only created short episode duration datasets with two different SNRs in the range that was also observed in the real datasets; see below). Our algorithm (red) showed a small disadvantage at the lowest studied SNR level (-16 dB), but performed better than the AR (blue) and Rutishauser algorithms (green) for the remaining SNR levels (-9 to +12 dB). When analyzing the SNRs of our real LFP datasets, we typically found values between -10 and 0 dB. Our algorithm is therefore expected to show a solid detection performance in the most relevant SNR range for practical applications. According to the ANOVA, there was a significant main effect of SNR ($p < 10^{-6}$), a significant main effect of the algorithm ($p < 10^{-6}$), as well as a significant interaction ($p = 10^{-6}$). The post-hoc test revealed two significant pairwise differences: Our algorithm detected better than both the AR ($p = 10^{-4}$) and Rutishauser ($p = 10^{-4}$) algorithms. The corresponding prediction performance is shown in Fig. 5(B). Across the range of studied SNR levels, our algorithm (red) showed a better prediction performance than the Rutishauser (green), Mansouri (cyan), and AR (blue) algorithms, with the Rutishauser algorithm showing comparable performance at -16 dB, but then starting to fall behind. The saturation of the red curve at 800 ms is a consequence of having limited the phase prediction to 800 ms into the future. The ANOVA indicated significant main effects of both SNR ($p < 10^{-6}$) and the algorithm ($p < 10^{-6}$). According to Tukey's HSD test, there were four significant pairwise differences: Our algorithm predicted better than the AR ($p = 10^{-4}$), Rutishauser ($p = 10^{-4}$), and Mansouri ($p = 10^{-4}$) algorithms; the Mansouri algorithm performed better than the AR algorithm ($p = 0.022$).

3.3. Effect of oscillatory episode duration on detection and prediction performance

Since we had artificial datasets with both long-lasting oscillatory activity (several seconds) as well as shorter oscillatory episodes (3–12 cycles), we were able to analyze the detection and prediction performance for each of these situations separately. Fig. 6(A) shows the detection performance. Each datapoint represents the average over different datasets with frequencies ranging from 4.5–47 Hz and SNRs of either -9 or -2 dB. Our algorithm (red) showed a small disadvantage compared to the Rutishauser algorithm (green) for short oscillatory episodes. One should keep in mind, however, that we ran the Rutishauser algorithm with different threshold levels and then selected an optimized one post-hoc. It is unlikely that such an optimal performance

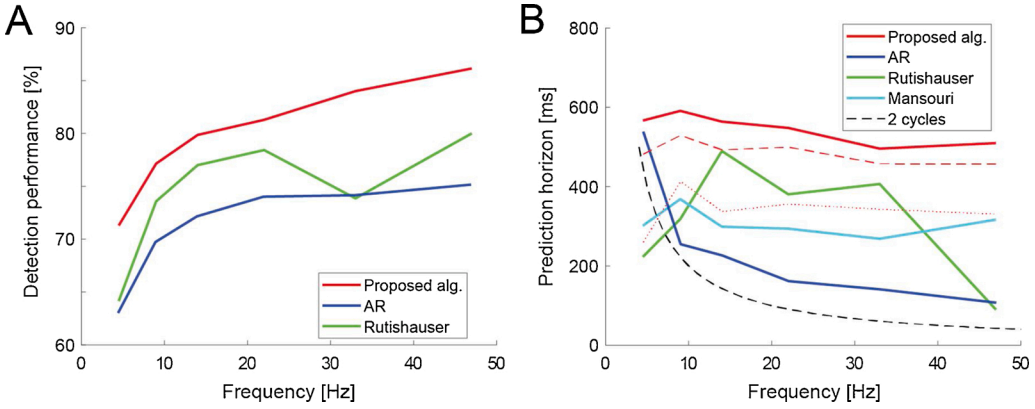

Fig. 4. Artificial datasets with different oscillation frequencies. (A) Detection Performance. (B) Prediction Performance (time horizon below 90°, solid lines). The red dashed and dotted lines additionally mark the time horizons below 60° and 30°, respectively (only for our proposed algorithm). For reference, two cycles are shown as a black dashed line.

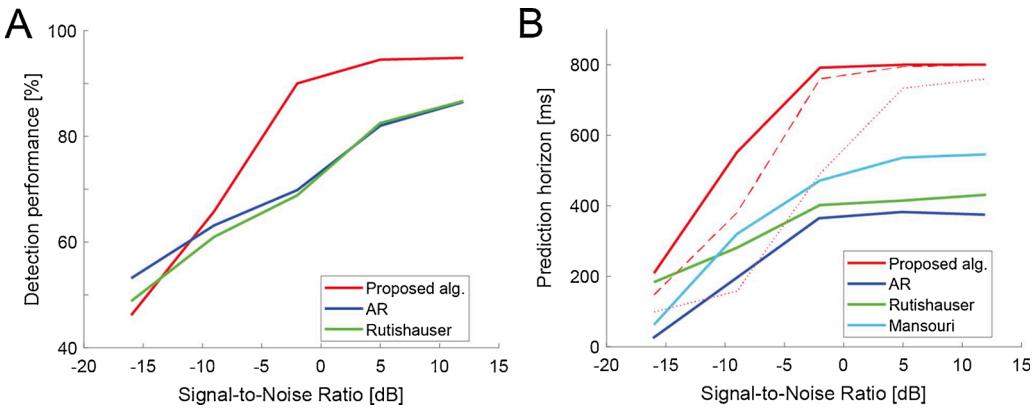

Fig. 5. Artificial datasets with different Signal-to-Noise Ratios. (A) Detection Performance. (B) Prediction Performance.

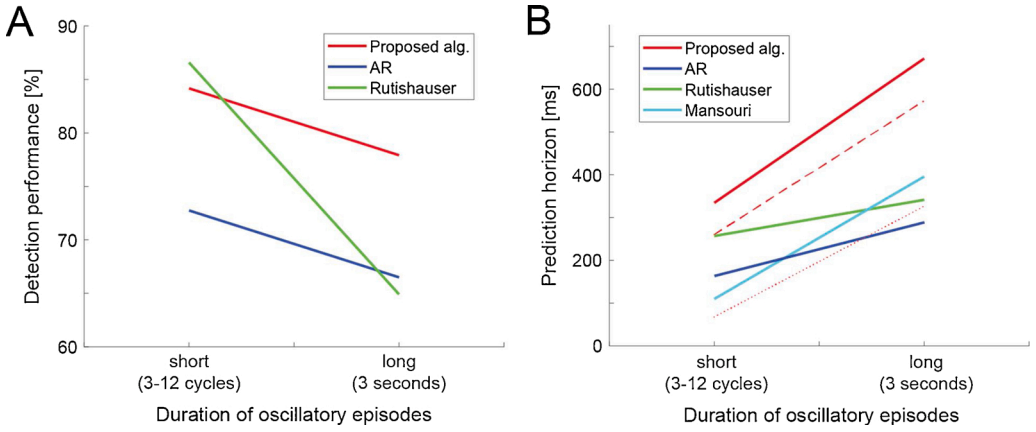

Fig. 6. Artificial datasets with either short or long oscillatory episodes. (A) Detection Performance. (B)Prediction Performance.

could be achieved in a real-world situation, where a human operator would have to pick a particular absolute detection threshold. Our algorithm does not have this issue, as detection thresholds are determined automatically. For long oscillatory episodes, our algorithm (red) performed better than both the AR (blue) and Rutishauser (green) algorithms. The detection performance should not be directly compared between short and long episodes, as the datasets were not matched for the proportion of time when oscillatory activity was present. The significant main effect of episode duration indicated by the ANOVA is therefore not particularly meaningful, but there was also a significant main effect of the algorithm ($p = 10^{-6}$) as well as a significant interaction ($p = 2{\cdot}10^{-4}$). The post-hoc test indicated three pairwise significant differences: Our algorithm detected better than both the AR ($p = 10^{-4}$) and Rutishauser ($p = 0.023$) algorithms; the Rutishauser algorithm performed better than the AR algorithm ($p = 0.005$). The corresponding prediction performance is shown in Fig. 6(B). Here, our algorithm (red) showed better prediction performance than the reference algorithms for both short- and long-lasting oscillatory episodes. According to the ANOVA, there were significant main effects of both episode duration ($p = 5{\cdot}10^{-4}$) and algorithm ($p < 10^{-6}$) as well as a significant interaction ($p = 0.005$). Tukey's HSD test indicated three significant pairwise differences: Our algorithm predicted better than the AR ($p = 2{\cdot}10^{-4}$), Rutishauser ($p = 2{\cdot}10^{-4}$), and Mansouri ($p = 2{\cdot}10^{-4}$) algorithms.

3.4. *Effects of non-stationarities in the oscillation frequency on detection and prediction performance*

To study how the algorithms behave in the presence of non-stationarities in the oscillation frequency, we analyzed datasets with a fixed oscillation frequency throughout (14 Hz in the case of the first analysis, 33 Hz in the second, an SNR of either −9 or −2 dB, and all short episodes), datasets where the oscillation frequency could change from oscillatory episode to oscillatory episode (between 10 and 20 Hz in the first analysis, and between 20 and 40 Hz in the second, an SNR of either −9 or −2 dB, and all short episodes), and datasets where the oscillation frequency was allowed to drift within oscillatory episodes (with the same specifications). We also analyzed the situation of two different oscillation frequencies being present at the same time (within the same frequency limits) and asked how well the algorithms could track the stronger of the two (with an SNR of −2 dB; the weaker component had an SNR of −9 dB).

First, we confined the frequencies to be in the range 10…20 Hz. The detection performance is shown in Fig. 7(A). Across the board, the Rutishauser algorithm (green) showed a slightly better detection performance than our algorithm (red), but one should again keep in mind that this is the optimized performance based on a post-hoc selection of the detection threshold. Our algorithm performed much better than the AR algorithm (blue). The ANOVA indicated a significant main effect of the algorithm ($p = 0.002$). The post-hoc test revealed two significant pairwise differences: Both our algorithm ($p = 0.006$) and the Rutishauser algorithm ($p = 0.003$) detected better than the AR algorithm. The corresponding prediction performance is shown in Fig. 7(B). Our algorithm (red) and the Rutishauser algorithm (green) showed a similar performance when the oscillation frequency was completely stationary and when the frequency was allowed to drift within episodes. Our algorithm showed a clear advantage when the oscillation frequency could change across episodes and when two different oscillation frequencies were present at the same time. The advantage results from the passband

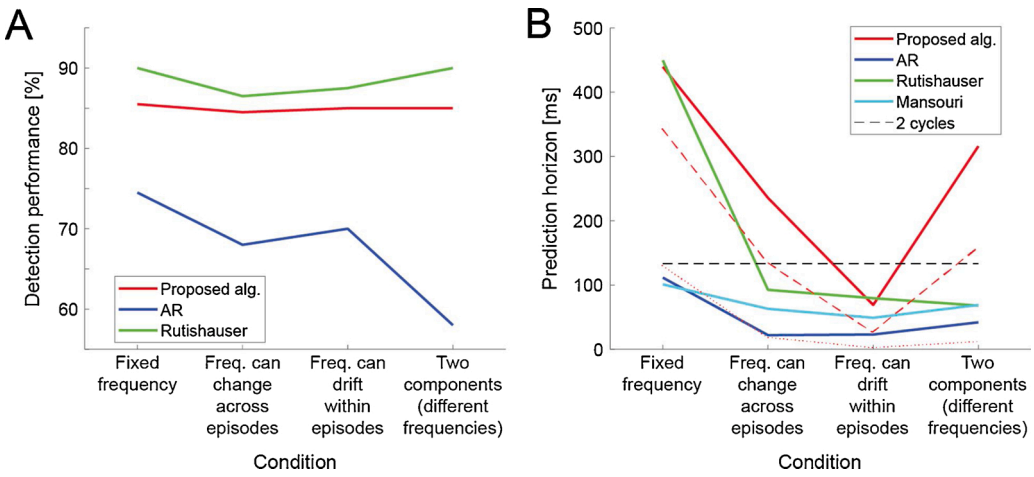

Fig. 7. Artificial datasets with **oscillation frequencies between 10 and 20 Hz** and either fixed oscillation frequencies, frequencies that can change across oscillatory episodes, frequencies that can drift within episodes, or two oscillatory components with different frequencies. (A) Detection Performance. (B)Prediction Performance. For reference, two cycles are shown as a black dashed line.

of the bandpass filter being adjusted automatically based on the currently present signal in our algorithm, compared to the fixed bandpass filter in the Rutishauser algorithm. According to the ANOVA, there were significant main effects of both the non-stationarity condition ($p = 0.027$) and the algorithm ($p = 10^{-6}$) as well as a significant interaction ($p = 10^{-4}$). Tukey's HSD test indicated five significant pairwise differences: Our algorithm predicted better than the AR ($p = 2{\cdot}10^{-4}$), Rutishauser ($p = 0.007$), and Mansouri ($p = 2{\cdot}10^{-4}$) algorithms; the Rutishauser algorithm performed better than both the AR ($p = 3{\cdot}10^{-4}$) and the Mansouri ($p = 4{\cdot}10^{-4}$) algorithms.

Next, we assessed the performance in a higher frequency band. In this case the frequencies were confined to be in the range 20…40 Hz. The detection performance is shown in Fig. 8(A). Similar to what we had seen in Fig. 7(A), the Rutishauser algorithm (green) provided a slightly better detection performance than our algorithm (red), which, in turn, performed much better than the AR algorithm (blue). According to the ANOVA, there were significant main effects of both the non-stationarity condition ($p = 0.033$) and the algorithm ($p < 10^{-6}$) as well as a significant interaction ($p = 0.007$). The post-hoc test indicated three significant pairwise differences: The Rutishauser algorithm detected better than both our algorithm ($p = 0.008$) and the AR algorithm ($p = 2{\cdot}10^{-4}$); our algorithm performed better than the AR algorithm ($p = 2{\cdot}10^{-4}$). The prediction performance can be seen in Fig. 8(B). In this case, the Rutishauser algorithm (green) provided the best performance when the oscillation frequency was completely stationary. It was, however, provided with exact knowledge of the oscillation frequency such that the bandpass filter was centered perfectly on the oscillation frequency. In a real-life situation, the center frequency would have to be set by a human operator, and it is unlikely that this optimal performance could be achieved. In the remaining cases (oscillation frequency can change across episodes or within episodes, two oscillation frequencies being present at the same time), our algorithm (red) provided a clear advantage over all reference algorithms. The ANOVA indicated a significant main effect of the algorithm ($p = 5{\cdot}10^{-5}$) as well as a significant interaction between the non-stationarity condition and the algorithm ($p = 0.010$). Tukey's HSD test revealed four significant pairwise differences: Our algorithm predicted better than both the AR ($p = 2{\cdot}10^{-4}$) and the Mansouri ($p = 4{\cdot}10^{-4}$) algorithms; likewise, the Rutishauser algorithm also performed better than both the AR ($p = 5{\cdot}10^{-4}$) and the Mansouri ($p = 0.003$) algorithms.

3.5. Detection and prediction performance with real LFP datasets

The detection performance across nine real LFP datasets is shown in Fig. 9(A). The Rutishauser algorithm (green) provided the best detection

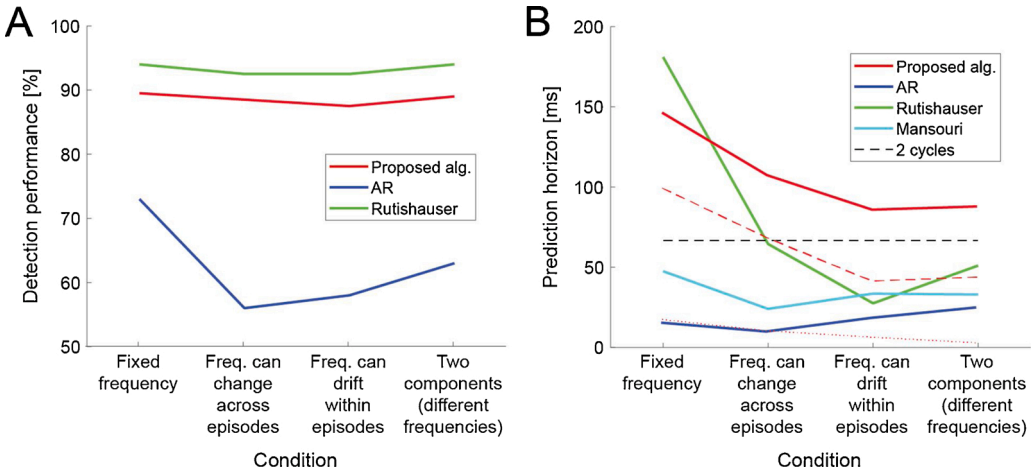

Fig. 8. Artificial datasets with **oscillation frequencies between 20 and 40 Hz** and either fixed oscillation frequencies, frequencies that can change across oscillatory episodes, frequencies that can drift within episodes, or two oscillatory components with different frequencies. (A) Detection Performance. (B)Prediction Performance. For reference, two cycles are shown as a black dashed line.

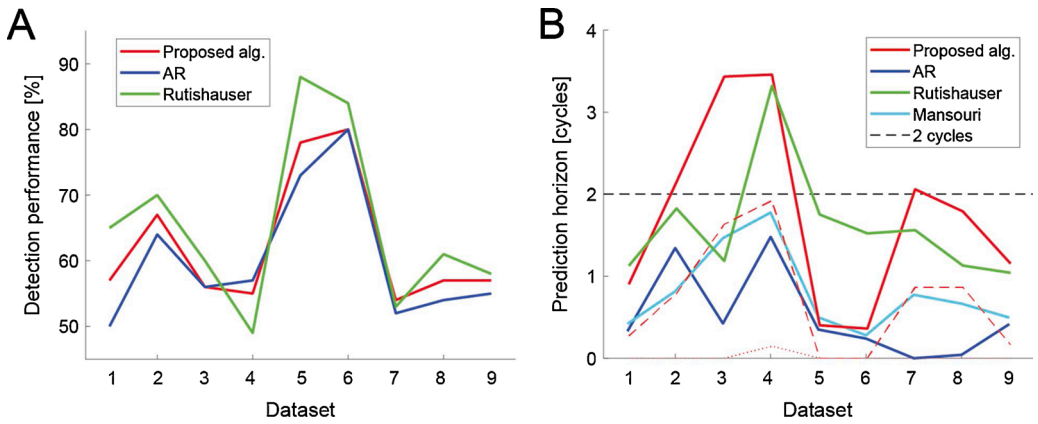

Fig. 9. Real LFP datasets. (A) Detection Performance. (B) Prediction Performance.

performance for seven of these, with an average detection performance of 65 %, followed by our algorithm (red), with an average detection performance of 62 %, and the AR algorithm (blue), with an average detection performance of 60 %. Again, it should be kept in mind that the reported detection performance of the Rutishauser algorithm is the consequence of a post-hoc selection of an optimized detection threshold for each combination of dataset and analysis window size and therefore most likely better than what could be achieved in a real experiment. The ANOVA indicated significant main effects of both the dataset ($p < 10^{-6}$) and the algorithm ($p = 5{\cdot}10^{-4}$) as well as a significant interaction ($p = 0.027$). The post-hoc test revealed two significant pairwise differences: The Rutishauser algorithm detected better than both our algorithm ($p = 0.034$) and the AR algorithm ($p = 3{\cdot}10^{-4}$). The prediction performance is shown in Fig. 9(B), plotted as number of cycles rather than time in ms, due to the large variability of oscillation frequencies across datasets. Our algorithm (red) provided the best performance for six of the nine datasets, the Rutishauser algorithm (green) performed better for the remaining three. Two of the datasets (5 and 6) stand out, because the Rutishauser algorithm was the only one that was able to make accurate phase predictions beyond one cycle. Both datasets were human ECoG recordings from visual cortex with high oscillation frequencies (40–50 Hz). We will return to this point below. According to the ANOVA, there were significant main effects of both the dataset ($p < 10^{-6}$) and the algorithm ($p < 10^{-6}$) as well as a significant interaction ($p < 10^{-6}$). Tukey's HSD test indicated five significant pairwise differences: Our algorithm predicted better than the AR ($p = 2{\cdot}10^{-4}$), Rutishauser ($p = 0.029$), and Mansouri ($p = 2{\cdot}10^{-4}$) algorithms; the Rutishauser algorithm performed better than both the AR ($p = 2{\cdot}10^{-4}$) and the Mansouri ($p = 9{\cdot}10^{-4}$) algorithms.

To gain more insight into why the Rutishauser algorithm had a substantially better prediction performance in the case of Datasets 5 and 6, we performed additional analyses. First, we looked at the two components of the detection performance separately. While our algorithm had an average true positive rate (correct detections) of about 6% (fraction of analyzed signal segments that were marked by the offline BOSC analysis as containing oscillatory activity and also marked by our algorithm as an oscillation being present), it was only about 1% in the case of the Rutishauser algorithm. Rutishauser's detection approach was therefore much more conservative (a consequence of finding an optimal detection threshold for each combination of dataset and analysis window size, while also trying to minimize false positives) compared to our algorithm, which also made it miss more signal segments that were marked by BOSC as oscillatory (14 % of all segments) compared to our algorithm (8%). Detecting fewer segments means that these will contain, on average, more oscillatory power. This was confirmed by comparing the average power time courses in the frequency band of interest, as calculated by the Rutishauser algorithm, between Rutishauser's true positives and our true positives. The average power across time was between 37 % and 125 % higher for Rutishauser's true positives compared to ours. As a consequence of the better SNR, the parameters of the oscillation can be estimated more accurately. The difference in the prediction horizons was clearly reflected in how accurately the phase of the oscillation at the end of the analysis window could be estimated. While the average expected absolute phase error at the end of the analysis window across window sizes was not too different in the case of Dataset 5 (77° vs. 82°), it was substantially smaller (49°) for one particular analysis window size (600 ms), which was the only window size where the Rutishauser algorithm had a higher prediction performance than our algorithm, driving the substantially larger average shown in Fig. 9(B). In the case of Dataset 6, the average expected absolute phase error at the end of the analysis window across window sizes was substantially lower for the Rutishauser algorithm (53° vs. 80°), providing it with a higher prediction performance across window sizes. In summary, Rutishauser's better prediction performance in the case of Datasets 5 and 6 resulted from a more conservative detection approach, limiting the detections to stronger oscillations, which could be analyzed more accurately, resulting in a better phase prediction.

3.6. Recommendations for picking a suitable analysis window

The results we have reported so far were an average across a number of studied analysis window sizes. When using our algorithm in a real-world scenario, one would have to select a particular analysis window size to work with. To provide some guidance for this selection process, we analyzed which window size provided the best detection and prediction performance for each dataset, focusing on the, probably more realistic, artificial datasets with shorter oscillatory episodes and the real datasets. In case multiple window sizes provided an identical performance, we used the average of these window sizes. Fig. 10 shows the optimal window sizes as a function of the oscillation frequency, for detection in blue, for prediction in red. The dashed lines are robust fits of the form $\frac{a}{f} + b$, with a and b being free parameters. This diagram suggests that the optimal analysis window size for detection and prediction is not identical: the window should be slightly larger for an optimal prediction compared to optimal detection. One therefore has to find a compromise between best detection and best prediction. Second, the analysis window clearly should be longer for lower oscillation frequencies, and shorter for higher oscillation frequencies. The green solid line illustrates a possible mapping between oscillation frequencies and window sizes, when only considering the window sizes that were used in this study:

- An analysis window size of 800 ms could be used for frequencies up to 7 Hz,
- a window size of 400 ms for higher frequencies up to 15 Hz,
- a window size of 200 ms for higher frequencies up to 40 Hz,

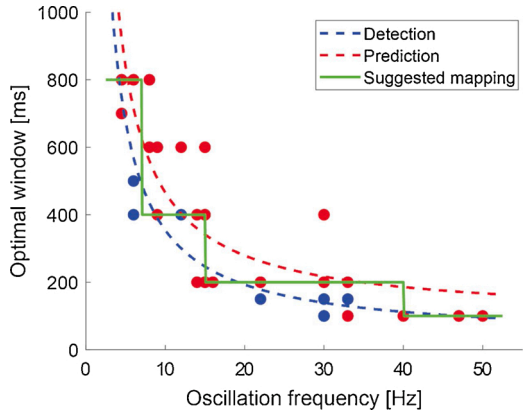

Fig. 10. Finding an optimal analysis window size based on the window sizes providing best Detection (blue dots) and Prediction (red dots) Performance for each real dataset and artificial dataset with short oscillatory episodes. The dashed lines are robust nonlinear regressions. The optimal analysis window for prediction is slightly larger than the one for detection, and one therefore has to find a compromise. The green solid line indicates a suggested mapping between oscillation frequency and analysis window (for the set of window sizes used in this study).

- and a windows size of 100 ms for frequencies above 40 Hz.

As can be seen from the diagram, a slightly longer analysis window is probably advisable for high oscillation frequencies. We have run some experiments with window sizes of 120 and 140 ms, focusing on the datasets with the highest oscillation frequencies, and have seen some advantages over using a 100 ms-long window (data not shown).

3.7. Evaluation of the detection delay

To measure how much time typically passes between the onset of oscillatory activity and when it is first detected by our algorithm, we created additional artificial LFP signals with different oscillation frequencies and SNRs and random oscillation onset times, and used the recommended window sizes from the previous section in combination with different window step sizes (by how much the analysis window is advanced; details can be found in *Measuring the detection delay*). We determined the median time difference between the end of the analysis window, when the oscillation was first detected, and the oscillation onset time (see Fig. 3(B)). When measured in number of oscillation cycles, the detection delay turned out not to change systematically with oscillation frequency. The SNR and window step size, however, had a clear impact on the detection delay. On average, the recommended analysis window was 4.8 cycles long. For an SNR of -2 dB, a value in the range covered by our real LFP signals, the oscillation was typically detected 4.1 (± 0.2; standard error) cycles after its onset when the window step size was 50 %, 3.4 (± 0.1) cycles when the window step size was 25 %, and 3.1 (± 0.1) cycles for 10 %. Thus, although we did not see any major effects on the overall detection and prediction accuracy when increasing the window overlap (reducing the window step size from 50 % to 25 %; data not shown), the detection delay actually can be improved. The detection is also faster when the signal is stronger: with an SNR of +5 dB, the detection delays were 3.1 (± 0.1) cycles, 2.5 (± 0.1) cycles, and 2.1 (± 0.1) cycles (for a window step size of 50 %, 25 %, and 10 %, respectively).

3.8. More in-depth evaluation of the ability to separate two frequency components

We had seen above that our algorithm made better phase predictions than the comparison algorithms in the presence of two frequency components in the signal. This is due to its ability to adjust the bandpass filter based on the current signal properties. To be able to exclude a weaker frequency component, it either needs to remain below the detection threshold, or it needs to be separated from the group of neighboring frequencies comprising the stronger component by at least one frequency bin, whose power does not exceed the detection threshold. To gain more detailed insight into when the algorithm is able so successfully separate two components, we created additional artificial LFP signals with two embedded frequency components centered around different frequencies, with different frequency differences, and with different SNRs (details can be found in *More detailed analysis of the ability to separate two frequency components*). We again used the recommended window sizes and, each time our algorithm detected the presence of oscillatory activity, evaluated whether the bandpass filter had been placed such that the frequency of the stronger component was inside the passband of the filter and the frequency of the weaker component was outside the passband, which we called a successful separation. The relative frequency of successful separations is plotted in Fig. 11. It largely was an approximately linear function of the relative frequency difference of the two components (difference between the two frequencies, divided by the center frequency), but also affected by the absolute frequency difference/center frequency and the relative strength of the two components. The separability improved with increasing relative frequency difference and was also better for the larger center frequency, equivalent to a larger absolute frequency difference, and the more different the SNRs of the two components were. All datapoints were captured well by a function of the form

$$p(successful\ separation) = 1.95 \cdot (\Delta f_{rel} - 0.065) + 0.01 \cdot \Delta f_{abs} \cdot (\Delta SNR - 1.8)$$

limited to the range 0…1 (i.e., negative values have to be converted into zero, values larger than one into one), with Δf_{rel} being the relative frequency difference, Δf_{abs} being the absolute frequency difference (in Hz), and ΔSNR being the difference between the SNR of each component (in dB), which is represented by the lines in the plot. This suggests that the algorithm can start separating frequencies when the relative frequency difference exceeds 6.5 % and when the difference in SNR exceeds 1.8 dB. A relative frequency difference of 25 % results in a chance of a successful frequency separation of about 50 %. Close to perfect separation can be achieved when the relative frequency difference exceeds at least 40 %.

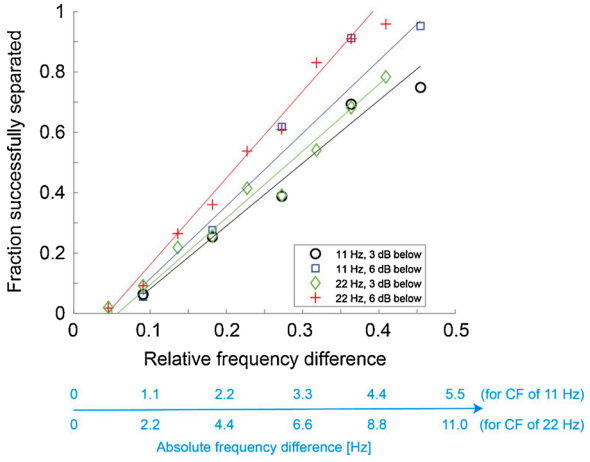

Fig. 11. Ability to separate two frequency components as a function of the relative frequency difference (main horizontal axis), the center frequency (CF; 11 vs. 22 Hz), and the difference in SNR between the two components (3 vs. 6 dB). The cyan auxiliary horizontal axis shows the corresponding absolute frequency differences for each CF.

4. Discussion

4.1. User-friendliness

Whereas previously proposed algorithms were typically difficult to use in a real-world experimental scenario, our algorithm is straightforward to use and does not require the user to specify any parameters they might not be familiar with. For example, the algorithm proposed by Chen et al. (Chen et al. (2013)), referred to as the "AR" algorithm in this study, requires the user to first record a sample of the to-be-tracked oscillatory activity and to submit it to a potentially lengthy offline optimization process, before the parameters for tracking the oscillations online are available and the closed-loop stimulation experiment can be started. In our evaluation, using Chen et al.'s MATLAB code, we have seen runtimes of up to half an hour on current Intel Core and Xeon processors. In contrast, our algorithm does not require the user to perform any preparatory steps. Oscillatory activity can be detected and tracked right away. Similarly, the algorithm proposed by Rutishauser et al. (Rutishauser et al. (2013)), referred to as "Rutishauser" algorithm in this study, requires the user to specify both the oscillation frequency and an absolute detection threshold, which the user probably doesn't know going into an experiment. In contrast, our algorithm requires the user only to specify a range of frequencies, within which oscillations should be tracked, and a statistical confidence level for the detection of oscillatory activity, which we didn't have to change for any of our results. We just left it at its default setting. The actual frequency-dependent detection threshold is then determined automatically by the algorithm, as is the current filter passband for tracking the ongoing oscillatory activity.

4.2. Tracking oscillations with changes in frequency and multiple oscillatory components

In contrast to the Rutishauser algorithm, which uses a fixed bandpass filter, our algorithm chooses the passband of the analysis filter dynamically based on short-term spectral analysis. This allows it to track oscillations with non-stationarities in the oscillation frequency and to selectively process one oscillatory component, while filtering out a weaker neighboring component. A corresponding advantage in the prediction horizon was clearly seen when analyzing artificial datasets with changes in oscillation frequency across oscillatory episodes or with multiple embedded oscillatory components.

4.3. Difficulty predicting the phase of high-frequency oscillations in human ECoG recordings

As mentioned in Results, we noticed that the Rutishauser algorithm was the only one that had a prediction horizon of more than one cycle when applying it to two human ECoG recordings from visual cortex with high oscillation frequencies (40…50 Hz). Our additional analyses indicated that this was primarily the consequence of the Rutishauser algorithm taking a very conservative detection approach, limiting the detection to stronger oscillations, which could be analyzed more accurately, leading to a better phase prediction, but also made it miss more oscillatory episodes. Another contribution is likely made by one specific feature that sets the Rutishauser algorithm apart from all other algorithms, including ours: Due to using a fixed bandpass filter, the Rutishauser algorithm can track the power in a narrow frequency band of interest as a function of time. It can therefore make a decision about the presence of oscillatory activity based on the situation at the end of the analysis window. If oscillatory activity is present there, it is quite likely to continue after the analysis window. In contrast, the other methods primarily use information around the center of the analysis window: the AR algorithm is explicitly designed to base its time series prediction on a signal segment that is centered in the analysis window, our algorithm uses spectral information that is obtained through the application of a taper that puts the largest weight on the center of the analysis window. These methods are therefore more likely to make a prediction based on oscillatory activity that was still strong near the center of the window, but might be about to end, or to miss oscillatory episodes that are just starting towards the end of the analysis window. When comparing the time courses of the power in the frequency band of interest, we noticed that, in the case of 600 ms-long analysis windows, the power peaked at a substantially later point in time for Rutishauser's true positives compared to our true positives. We have some ideas how this problem could be addressed in our algorithm by not scheduling a stimulation when there are indications that the phase prediction is likely to be inaccurate or that the oscillatory episode is about to end, and we plan to implement them in a future, further improved version of our algorithm. Another possible avenue would be to further optimize the choice of the bandpass filter, which could help reduce edge artefacts that are caused by the filtering and therefore potentially improve the phase prediction.

5. Conclusion

We have proposed a user-friendly algorithm for the detection of oscillatory activity in LFP signals and for predicting its phase, as needed for closed-loop, phase-locked stimulation experiments. We have seen that the algorithm is able to provide a robust oscillation detection and phase prediction performance over a wide range of oscillation conditions, which often either rivals or exceeds the performance of the reference algorithms. Only in rare cases it is not competitive, but it provides a clear advantage when oscillation frequencies are non-stationary or when different oscillation components have to be separated. All of this could be achieved with only having to specify a range of frequencies, within which oscillatory activity is tracked. The detection confidence level was set to the same default value across all analyzed datasets, and detection thresholds and filter passbands were chosen automatically by our algorithm.

5.1. Future plans

We are currently in the process of implementing the algorithm for real-time use on embedded hardware, and we are planning to share this design with the research community. How the algorithm can be optimized for real-time implementation and the specifics of mapping it onto particular target hardware will be reported on separately.

CRediT authorship contribution statement

Cristian Rodriguez Rivero: Data curation, Formal analysis, Investigation, Methodology, Software, Writing - review & editing. **Jochen Ditterich:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing - original draft, Writing - review & editing.

Declaration of Competing Interest

The authors report no declarations of interest.

Acknowledgments

This work has been supported by NIH grant R21MH107671 and NSF grant 1631329. The authors would like to thank the laboratories mentioned in the main text for providing LFP recordings and Henry Alitto, Alex Berrian, Naoki Saito, Marty Usrey, and David Weber for helpful discussions.

References

Akam, T., Kullmann, D.M., 2014. Oscillatory multiplexing of population codes for selective communication in the mammalian brain. Nat. Rev. Neurosci. 15, 111–122.

Bosman, C.A., Schoffelen, J.M., Brunet, N., Oostenveld, R., Bastos, A.M., Womelsdorf, T., Rubehn, B., Stieglitz, T., De Weerd, P., Fries, P., 2012. Attentional stimulus selection through selective synchronization between monkey visual areas. Neuron 75, 875–888.

Chen, L.L., Madhavan, R., Rapoport, B.I., Anderson, W.S., 2013. Real-time brain oscillation detection and phase-locked stimulation using autoregressive spectral estimation and time-series forward prediction. IEEE Trans. Biomed. Eng. 60, 753–762.

Ekstrom, A.D., Caplan, J.B., Ho, E., Shattuck, K., Fried, I., Kahana, M.J., 2005. Human hippocampal theta activity during virtual navigation. Hippocampus 15, 881–889.

Fries, P., 2005. A mechanism for cognitive dynamics: neuronal communication through neuronal coherence. Trends Cogn. Sci. 9, 474–480.

Gasior, M., Gonzalez, J.L., 2004. Improving FFT frequency measurement resolution by parabolic and Gaussian spectrum interpolation. AIP Conf. Proc. 732, 276–285.

Gregoriou, G.G., Gotts, S.J., Zhou, H., Desimone, R., 2009. High-frequency, long-range coupling between prefrontal and visual cortex during attention. Science 324, 1207–1210.

Grothe, I., Neitzel, S.D., Mandon, S., Kreiter, A.K., 2012. Switching neuronal inputs by differential modulations of gamma-band phase-coherence. J. Neurosci. 32, 16172–16180.

Hermes, D., Miller, K.J., Wandell, B.A., Winawer, J., 2015. Stimulus dependence of gamma oscillations in human visual cortex. Cereb. Cortex 25, 2951–2959.

Hughes, A.M., Whitten, T.A., Caplan, J.B., Dickson, C.T., 2012. BOSC: a better oscillation detection method, extracts both sustained and transient rhythms from rat hippocampal recordings. Hippocampus 22, 1417–1428.

Mansouri, F., Dunlop, K., Giacobbe, P., Downar, J., Zariffa, J., 2017. A fast EEG forecasting algorithm for phase-locked transcranial electrical stimulation of the human brain. Front. Neurosci. 11, 401.

Mitra, P.P., Bokil, H., 2008. Observed Brain Dynamics. Oxford University Press.

Rutishauser, U., Kotowicz, A., Laurent, G., 2013. A method for closed-loop presentation of sensory stimuli conditional on the internal brain-state of awake animals. J. Neurosci. Methods 215, 139–155.

Watrous, A.J., Tandon, N., Conner, C.R., Pieters, T., Ekstrom, A.D., 2013. Frequency-specific network connectivity increases underlie accurate spatiotemporal memory retrieval. Nat. Neurosci. 16, 349–356.

Whitten, T.A., Hughes, A.M., Dickson, C.T., Caplan, J.B., 2011. A better oscillation detection method robustly extracts EEG rhythms across brain state changes: the human alpha rhythm as a test case. Neuroimage 54, 860–874.

Witt, A., Palmigiano, A., Neef, A., El Hady, A., Wolf, F., Battaglia, D., 2013. Controlling the oscillation phase through precisely timed closed-loop optogenetic stimulation: a computational study. Front. Neural Circuits 7, 49.