

# Real-time detection of voltage patterns in the brain

Tomas Fiers

Thesis submitted for the degree of  
Master of Science in  
Biomedical Engineering

**Thesis supervisors:**

Prof. dr. ir. A. Bertrand  
Prof. dr. F. Kloosterman

**Assessors:**

Prof. dr. ir. R. Puers  
Dr. eng. J. Couto

**Mentors:**

Ir. J. Wouters  
Dott. D. Ciliberti

© Copyright KU Leuven

Without written permission of the thesis supervisors and the author it is forbidden to reproduce or adapt in any form or by any means any part of this publication. Requests for obtaining the right to reproduce or utilize parts of this publication should be addressed to Faculteit Ingenieurswetenschappen, Kasteelpark Arenberg 1 bus 2200, B-3001 Heverlee, +32-16-321350.

A written permission of the thesis supervisors is also required to use the methods, products, schematics and programmes described in this work for industrial or commercial use, and for submitting this publication in scientific contests.

---

## Abstract – 30 Nov

---

---

# Contents

---

<b>Abbreviations</b>	<b>6</b>
<b>Symbols</b>	<b>7</b>
<b>1 Introduction – 16 Nov [9p]</b>	<b>9</b>
1.1 Closed-loop brain-computer interfaces [1p]	9
1.2 Sharp wave-ripples [6p]	9
1.3 Problem statement [1p]	9
1.4 Thesis overview [1p]	9
<b>2 Single-channel linear filtering – 9 Nov [10p]</b>	<b>10</b>
2.1 Offline labelling of SWR segments [3p]	10
2.2 Quantifying detector performance [4p]	14
2.3 Online band-pass filters [3p]	17
<b>3 Multi-channel linear filtering – 26 Oct [10p]</b>	<b>19</b>
3.1 Data-driven algorithms [1p]	19
3.2 Linear signal-to-noise maximisation [2p]	19
3.3 Result for SWR detection	22
3.4 Combining space and time [1p]	25
3.5 Regularization [1p]	29
3.6 Multiple eigenvectors [2p]	29
3.7 Spectral preprocessing [1p]	29
<b>4 Nonlinear signal detection – 9 Nov [9p]</b>	<b>30</b>
4.1 Recurrent neural networks [2p]	30
4.2 Optimization [3p]	30
4.3 Regularization [2p]	30
4.4 Channels [1p]	30
4.5 Network size [1p]	30
<b>5 Discussion – 23 Nov [4p]</b>	<b>31</b>
5.1 Comparing detectors [2p]	31
5.2 Further work [2p]	31
<b>6 Conclusions – 30 Nov [1p]</b>	<b>32</b>

<i>CONTENTS</i>	5
<b>References</b>	<b>33</b>
<b>Appendices</b>	<b>36</b>

---

# Abbreviations

---

<b>BPF</b>	Band-pass filter. See chapter 2.
<b>CA1</b>	“Cornu Ammonis”, subregion 1. Region in the hippocampus where voltages are recorded from (see ???).
<b>CA3</b>	“Cornu Ammonis”, subregion 3. Region in the hippocampus (see ???). CA3 sends many axons (called “Schafer collaterals”) to CA1.
<b>FIR</b>	Finite impulse response. A linear filter whose output is a convolution of the input signal with some kernel.
<b>GEVal</b>	Generalized eigenvalue. See section 3.2.
<b>GEVec</b>	Generalized eigenvector. See section 3.2.
<b>IIR</b>	Infinite impulse response. A linear filter whose output at each timestep is a function of both input samples and preceding output samples.
<b>IQR</b>	Interquartile range. A measure of the spread of a set of one-dimensional values, that is robust to outliers. Difference between the 75th and the 25th data percentile.
<b>KDE</b>	Kernel density estimate.
<b>LFP</b>	Local field potential. The extracellular electric potential (see ??).
<b>RMS</b>	Root-mean-square. $\sqrt{\langle x_t^2 \rangle}$ for a signal $x_t$ .
<b>RNN</b>	Recurrent neural network. See chapter 4.
<b>SNR</b>	Signal-to-noise ratio. See section 3.2.
<b>SOTA</b>	State of the art. The algorithm currently used for SWR detection, namely an online single channel band-pass filter.
<b>SWR</b>	Sharp wave-ripple. The pattern in the LFP that we want to detect in real-time. See ??.

---

# Symbols

---

## Notation

$y$	Scalars are denoted in lowercase italic.
$\mathbf{z}$	Vectors are denoted in lowercase boldface.
$\mathbf{A}$	Matrices are denoted in uppercase boldface.
$\langle \cdot \rangle$	Time-average of a signal.
$\odot$	Elementwise multiplication. (“Hadamard product”).
$\sigma(\cdot)$	Sigmoid ‘squashing’ function, $\mathbb{R} \rightarrow (0, 1)$ . $\sigma(x) = \frac{1}{1+\exp(-x)}$ .
$\tanh(\cdot)$	Hyperbolic tangent, $\mathbb{R} \rightarrow (-1, 1)$ . $\tanh(x) = 2 \sigma(x) - 1$ .

## Signals

$\mathbf{z}_t$	Digitized LFP sample at discrete time step $t$ . $\mathbf{z}_t \in \mathbb{R}^C$ , with $C$ the number of channels (i.e. the number of electrodes simultaneously recorded from). Input to an SWR detection algorithm.
$o_t$	Output signal of an SWR detection algorithm, $\in \mathbb{R}$ .
$n_t$	‘Envelope’. Transformation of $o_t$ , so that it is constrained to $\mathbb{R}^+$ . Should be high when the corresponding input sample $\mathbf{z}_t$ is part of an SWR segment, and low when it is not. $n_t =  o_t $ for online linear filters; $n_t = \sigma(o_t)$ for the RNN’s of chapter 4.
$y_t$	Binary target signal, used when training data-driven SWR detection algorithms. We define $y_t = 1$ when the corresponding input sample $\mathbf{z}_t$ is part of an SWR segment, and $y_t = 0$ when it is not.

## Measures & parameters

- $T$  Detection threshold applied to the envelope  $n_t$ .  $T \in [\min n_t, \max n_t]$ . Each threshold  $T$  yields a different  $P$ -value,  $R$ -value,  $F_1$ -value, etc.
- $P$  Precision. Also known as positive predictive value. The fraction of correct detections, out of all detections.
- $R$  Recall. Also known as sensitivity, hit rate, or true positive rate. The fraction of detected reference SWR segments, out of all reference SWR segments.
- $F_\beta$  F-score: weighted harmonic mean of recall and precision.  $F_\beta = \frac{(1+\beta^2)PR}{\beta^2P+R}$ . Measures detection performance “for a user who attaches  $\beta$  times as much importance to recall as to precision.” [1]
- $F_1$  F-score where recall and precision are weighted equally.



---

## Introduction – 16 Nov [9p]

---

**1.1 Closed-loop brain-computer interfaces [1p]**

**1.2 Sharp wave-ripples [6p]**

Description [2p]

Scientific importance [2p]

Biophysics [1p]

Closed-loop technology [1p]

**1.3 Problem statement [1p]**

**1.4 Thesis overview [1p]**

---

# Single-channel linear filtering – 9

## Nov [10p]

---

### 2.1 Offline labelling of SWR segments [3p]

To quantify the performance of a sharp wave-ripple (SWR) detection algorithm, we need an evaluation or ‘test’ recording, annotated with the segments of time when actual SWR events were present. This section describes how such annotations can be made, and how our data specifically was annotated.

SWR’s are an empirical phenomenon of hippocampal area CA1, ‘defined’ by what their voltage traces look like. In other words, there is no ground truth available to know when SWR’s occur. Scientists looking to annotate their LFP recordings with SWR segments therefore have to rely either on judgement calls by human labellers, or on an automated, offline SWR detection algorithm. The former could be considered more subjective, and is definitely more labour intensive than the latter – especially if multiple scientists are consulted to obtain a consensus labelling. Most studies use an automated, offline algorithm to detect SWR segments (see appendix A.2 for some examples).

‘Offline’ here means that SWR detection happens after the recording has been completed, and that there are thus no real-time constraints on the detection algorithm. This means that 1) there are no hard bounds on algorithm execution time, and 2) that the algorithm can use information ‘from the future’: when deciding whether a recording sample  $\mathbf{z}_t$  belongs to an SWR segment, it can use samples  $\mathbf{z}_{t_f}$  in its decision that occurred after  $\mathbf{z}_t$  (i.e.  $t_f > t$ ), instead of using only past samples  $\mathbf{z}_{t_p}$  (where  $t_p \leq t$ ).

#### Overview of offline ripple detection

The main steps of the offline SWR detection algorithm that most studies use – such as the ones cited in appendix A.2, and this thesis itself – can be summarized as follows:

1. Use a single channel of input data; namely from an electrode in the pyramidal cell layer of CA1, where the ripple part of SWR’s is most strongly present. (We will denote this voltage signal with  $z_t$ );

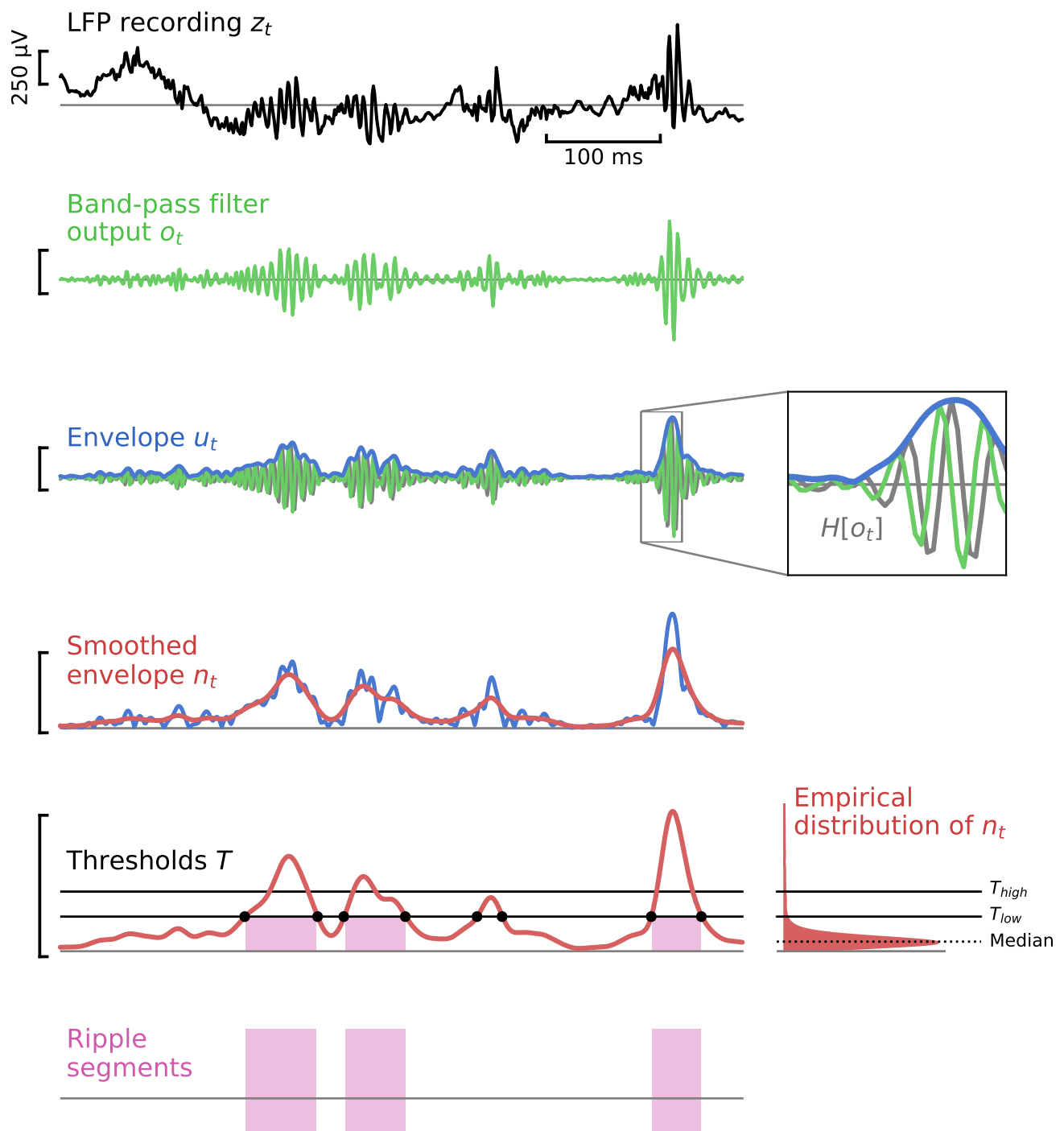


Figure 2.1: **Steps for automated, offline SWR labelling.** See text for details. Each vertical scalebar indicates the same voltage range.  $H[o_t]$  denotes the Hilbert transform of  $o_t$ . Note its phase lag of  $90^\circ$  with respect to  $o_t$ . In the second panel from the bottom, note the two threshold crossings of  $T_{\text{low}}$  (marked with black dots) that did not result in a ripple segment, because the second threshold  $T_{\text{high}}$  was not reached. The distribution of the envelope  $n_t$  was estimated using the entire dataset (and not just the displayed fragment).

2. Band-pass filter the recording to retain only ‘ripple’ frequencies. (We will denote the filter output as  $o_t$ );
3. Obtain the envelope of the band-pass filtered signal. (We will denote this envelope with  $n_t$ );
4. Calculate a ‘high’ and a ‘low’ threshold ( $T_{\text{high}}$  and  $T_{\text{low}}$ ) to apply to the envelope  $n_t$ , based on summary statistics of  $n_t$  and two custom multipliers ( $\alpha_{\text{high}}$  and  $\alpha_{\text{low}}$ );
5. Define ripple events as times when the envelope crosses the high threshold (i.e.  $n_t > T_{\text{high}}$ );
6. Define the start and end time of each such ripple as the closest times where the envelope falls back below the lower threshold (i.e.  $n_t < T_{\text{low}}$ ).

Figure 2.1 visualizes each step, as applied to a fragment of our dataset. Note that this procedure only detects ripples, and not sharp waves.<sup>1</sup>

### Validating parameter choices

Some of the above steps have free parameters (such as the ripple frequency band used for filtering, or the threshold multipliers  $\alpha_{\text{high}}$  and  $\alpha_{\text{low}}$ ). If not mentioned otherwise, our parameter choices were made as follows.

In a first pass of the offline detection algorithm, ripples were detected using a very broad-band filter and a low detection threshold. This ensured that all ‘true’ SWR’s were included in the detected events set (in addition to many spurious detections).

Next, five neuroscientists were independently asked to decide for each detected event whether it was a sharp wave-ripple or not. This was done through a custom-made web app, an example screen of which is shown in fig. D.1. Note that the labellers could factor multiple recording channels into their decision, including stratum radiatum channels displaying sharp wave activity. Only the events that were labelled as an SWR by at least three neuroscientists were retained.

Finally, when setting the parameters of the eventual offline ripple detection algorithm, the algorithm’s output was compared to the decisions made by the neuroscientists. The parameters were then adjusted until the output of the algorithm matched the neuroscientists’ decisions reasonably well.

The following subsections describe the detection steps in more detail, and compare our choices of methods and parameters to those made in the literature.

### Band-pass filter design

The wideband voltage signal  $x_t$  was band-pass filtered between 100 and 200 Hz, using a linear time-invariant filter with zero output lag.

---

<sup>1</sup> Although interestingly, the sharp wave part of sharp wave-ripples was discovered before the ripple part [2, p. 1].

Many other studies use a higher left bound, of about 140–150 Hz (see table A.1). Using such a high bound in our dataset however resulted in ripples that went undetected (even at low detection thresholds), although they were convincingly marked as SWR’s by the neuroscientists.

The band-pass filter was designed using the windowed-sinc method, with a Kaiser window (using SciPy’s `firwin` and `kaiserord` functions) [3], [4]. The transition width was chosen to be 10% of the bandwidth (i.e. 10 Hz), and the attenuation to be 40 dB, resulting in an FIR filter of order 150 (at a 1000 Hz sampling frequency).

The filter was applied bidirectionally, resulting in a zero-lag output, and a total attenuation of 80 dB. Figure 2.1 shows an example of the filter output in green.

(We cannot easily compare our filter design method with the literature, as most studies do not mention anything about filter design besides the frequency band used).

### Envelope calculation

Next, the instantaneous envelope  $u_t$  of the filter output  $o_t$  was calculated using its Hilbert transform<sup>2</sup>  $H[o_t]$ :

$$u_t = \sqrt{o_t^2 + H[o_t]^2}, \quad (2.1)$$

The Hilbert transform delays each frequency component of a signal by 90° [6] (see the gray signal in fig. 2.1). This means that the envelope of a narrowband signal such as  $o_t$  can be easily obtained as the magnitude of the so called ‘analytic signal’  $o_t + jH[o_t]$ , as in eq. (2.1).<sup>3</sup>

After calculating  $u_t$ , a final, smoothed envelope  $n_t$  was obtained by convolving  $u_t$  with a Gaussian kernel ( $\sigma = 7.5$  ms, support radius of  $4\sigma$ ). Compare the blue ( $u_t$ ) and the red signal ( $n_t$ ) in fig. 2.1.

Other studies (such as [7] and [8]) use a “root-mean-square” approach to calculate the envelope of the filter output  $o_t$ . In these studies, presumably, the squared signal  $o_t^2$  is smoothed using some kernel (of unspecified type and bandwidth) to obtain the “mean-square” signal of which the square root is taken.

<sup>2</sup>To be precise,  $H$  is a discrete approximation to the continuous Hilbert transform  $\mathcal{H}$ . There are multiple ways to make a discrete approximation to  $\mathcal{H}$  [5]. Most of them make use of the discrete Fourier transform, which makes computing  $H$  an efficient operation. We used SciPy’s `hilbert` function (which, confusingly, does not return the Hilbert transform of its input, but rather its analytical signal), and zero-padded the input signal to the nearest power of 2 or 3 to benefit from the speedup brought by the fast Fourier transform algorithm.

<sup>3</sup>To see why, consider a local approximation of the narrowband signal  $o_t$  by a sinusoid  $a \cos \omega t$ . Its Hilbert transform is then  $a \sin \omega t$ . From eq. (2.1), the envelope  $u_t$  will be locally approximated by the magnitude of the original signal:  $u_t = \sqrt{(a \cos \omega t)^2 + (a \sin \omega t)^2} = |a|$ .

### Threshold calculation

The two detection thresholds were calculated as follows:

$$T_{\text{high}} = \alpha_{\text{high}} \times \tilde{n}_t \quad (2.2)$$

$$T_{\text{low}} = \alpha_{\text{low}} \times \tilde{n}_t \quad (2.3)$$

where  $\tilde{n}_t$  denotes the median of the smoothed envelope  $n_t$ . As per the procedure described in “Validating parameter choices”, we set  $\alpha_{\text{high}} = 6.2$  and  $\alpha_{\text{low}} = 3.6$ . At a median envelope magnitude  $\tilde{n}_t = 17.0 \mu\text{V}$ , this results in thresholds  $T_{\text{high}} = 105.4 \mu\text{V}$  and  $T_{\text{low}} = 61.2 \mu\text{V}$ .

Most studies calculate thresholds as follows:  $T_{\text{high}} = \bar{n}_t + \beta_{\text{high}} \times \text{std}(n_t)$ . Here  $\bar{n}_t$  denotes the mean of the envelope,  $\text{std}(n_t)$  denotes its standard deviation, and  $\beta_{\text{high}}$  is a custom multiplier analogous to  $\alpha_{\text{high}}$ . For the non-negative, assymetric distributions of envelope signals, using both a measure of center and a measure of spread to define thresholds seems unnecessary (see the distribution of  $n_t$  in fig. 2.1), which is why we chose to use only one measure (namely the median).

The detection multiplier  $\beta_{\text{high}}$  varies wildly between studies: from 1, over 3, 4, and 5, up until 7 ([7]–[11], respectively).<sup>4</sup> It is clear that such different thresholds will give very different sensitivity-precision trade-offs for SWR detection (the lower thresholds yielding more false positive detections, and the higher thresholds yielding more missed true SWR events).

To compare our thresholds to those in the literature, we calculate the  $\beta$  multipliers corresponding to our chosen  $\alpha$  values. Given that our dataset has a mean envelope magnitude of  $22.3 \mu\text{V}$  and a standard deviation of  $22.9 \mu\text{V}$ , we find  $\beta_{\text{high}} = 3.63$  and  $\beta_{\text{low}} = 1.70$ . These values fall near the center of those reported in the literature (see appendix A.2).

### Segment post-processing

Finally, two add-hoc rules were applied to the automatically detected ripple segments. First, segments with only a small gap between them (of less than 10 ms) were joined together. Then, segments of too short a duration (less than 25 ms) were eliminated.

This step is rarely done in other studies. An exception is e.g. [9], where segments shorter than 15 ms were eliminated.

## 2.2 Quantifying detector performance [4p]

Section 2.1 described how we annotated our dataset with ‘reference’ SWR segments, using an offline algorithm. These reference segments can now be used to benchmark the performance of an online SWR detection algorithm. This section describes how this is done.

---

<sup>4</sup>Threshold multipliers cannot be compared precisely: a different background ‘noise’ level or ripple incidence rate will result in different  $\bar{n}_t$  and  $\text{std}(n_t)$  values. This means that  $\beta_{\text{high}}$  needs to change to maintain an equal threshold  $T_{\text{high}}$ ; which is necessary when the ripples to be detected have not changed their magnitude.

### Generating online detections

Each online SWR detection algorithm studied in this thesis generates an output envelope  $n_t$ , where samples  $n_t$  of larger magnitude denote a higher belief that the corresponding input sample  $z_t$  is part of an SWR event. (See fig. 3.2A for three example output envelopes).

These output envelopes  $n_t$  can be converted to a discrete set of detection times by applying a threshold  $T$  to the envelope. Additionally, we require a certain minimum duration between online detections (which we will call the “lockout time”,  $L$ ). The set  $D$  of detection times  $t_d$  (with  $d = 0, 1, \dots$ ) is then:

$$D = \{ t_d \mid t_d > t_{d-1} + L \text{ and } n_{t_d} > T \}. \quad (2.4)$$

In our analysis, we chose a lockout time  $L$  based on SWR durations. Specifically, we set  $L$  to the 25-th percentile of the durations of all SWR segments detected in our dataset by the offline algorithm, resulting in a lockout time of  $L = 34$  ms.

### Quantifying detector accuracy

In the following, we denote reference SWR segments as the closed intervals  $s_r = [t_r^{\text{start}}, t_r^{\text{end}}]$  (with  $r = 0, 1, \dots$ ). The set of all reference segments is denoted by  $S = \{s_r\}$ .

To benchmark the performance of an online SWR detection algorithm (at a certain threshold  $T$ ), we compare its detections  $D$  with the offline reference segments  $S$ , and divide both sets into two subsets each: namely detected versus undetected reference segments, and correct versus incorrect detections. Formally:

$$D^{\text{correct}} = \{t_d \mid \exists s_r : t_d \in s_r\} \quad (2.5)$$

$$D^{\text{incorrect}} = D \setminus D^{\text{correct}} \quad (2.6)$$

$$S^{\text{detected}} = \{s_r \mid \exists t_d : t_d \in s_r\} \quad (2.7)$$

$$S^{\text{undetected}} = S \setminus S^{\text{detected}}, \quad (2.8)$$

i.e. correct online detections are contained within a reference segment, and detected reference segments contain at least one online detection. Figure 3.2A shows some example correct and incorrect detections (green and red triangles, respectively).

Next, the number of detected reference segments is counted, and compared to the total number of reference segments. This fraction is the recall  $R$  of the SWR detector (also known as sensitivity, hit rate, or true positive rate). Formally:

$$R = \frac{|S^{\text{detected}}|}{|S|} \quad (2.9)$$

Similarly, the number of correct detections is counted, and compared to the total number of detections. This fraction is the precision  $P$  of the SWR detector (also known as the positive predictive value):

$$P = \frac{|D^{\text{correct}}|}{|D|} \quad (2.10)$$

Its opposite is the false discovery rate (FDR), which counts the number of so called “false positive” detections:

$$FDR = 1 - P = \frac{|D^{\text{incorrect}}|}{|D|} \quad (2.11)$$

Each detection threshold yields a different  $(P, R)$ -combination. This is often a trade-off: higher thresholds yield more precise detectors (less false positive detections), at the cost of a lower sensitivity (more missed reference SWR segments). Conversely, lower thresholds generally yield more sensitive but also less precise SWR detectors.

This is why the performance of a detection algorithm is often given for a *range* of thresholds, instead of for just a single threshold: different users of the SWR detection algorithm may prefer different trade-offs between latency and precision. (One user might require a minimal amount of false detections, for example, while another may want to detect as many SWR events as possible, without minding false detections). When precision and recall are plotted against each other for a range of different thresholds, the result is a so called “*PR-curve*” (see the lower-left panel of fig. 3.3 for an example). Generally, higher quality detectors have *PR*-curves that draw nearer to the top-right  $(1, 1)$ -corner.

The precision  $P$  and recall  $R$  for a given threshold can be summarized in a single number: the so called *F-score*. It is defined as:

$$F_\beta = \frac{(1 + \beta^2)PR}{\beta^2 P + R}. \quad (2.12)$$

It is the mean of recall and precision (more precisely: their harmonic mean, as precision and recall are ratio’s), with recall weighted by  $\beta^2$ . It measures detection performance “for a user who attaches  $\beta$  times as much importance to recall as to precision.” [1]

For example, the  $F_1$ -score weighs recall and precision equally. When we want to summarize the accuracy of a detector in a single number, we will often choose the maximum  $F_1$ -score of the detector (over the entire threshold range) as this number.

A note on terminology: we will refer to the precision and sensitivity performance of a detector often simply as the ‘accuracy’ of the detector. This is a shorthand, and does not refer to the technical definition of accuracy.<sup>5</sup>

### Quantifying detector latency

Besides counting how many SWR segments were detected, we are also interested in how early they were detected. For each detected reference segment  $s_r = [t_r^{\text{start}}, t_r^{\text{end}}]$ , we note the first online detection  $t_d$  contained within it (reference segments may contain multiple online detections).

The absolute detection latency  $AL_r$  of this SWR segment is then defined as:

$$AL_r = t_d - t_r^{\text{start}} \quad (2.13)$$

---

<sup>5</sup>Technically, accuracy is the sum of true positive and true negative cases, divided by the total number of cases. This concept does not readily apply to SWR detection, as there is no straightforward definition of “true negative” cases here. Only false positives ( $D^{\text{incorrect}}$ ), true positives ( $D^{\text{correct}}$  and  $S^{\text{detected}}$ ), and false negatives ( $S^{\text{undetected}}$ ) are clearly defined.



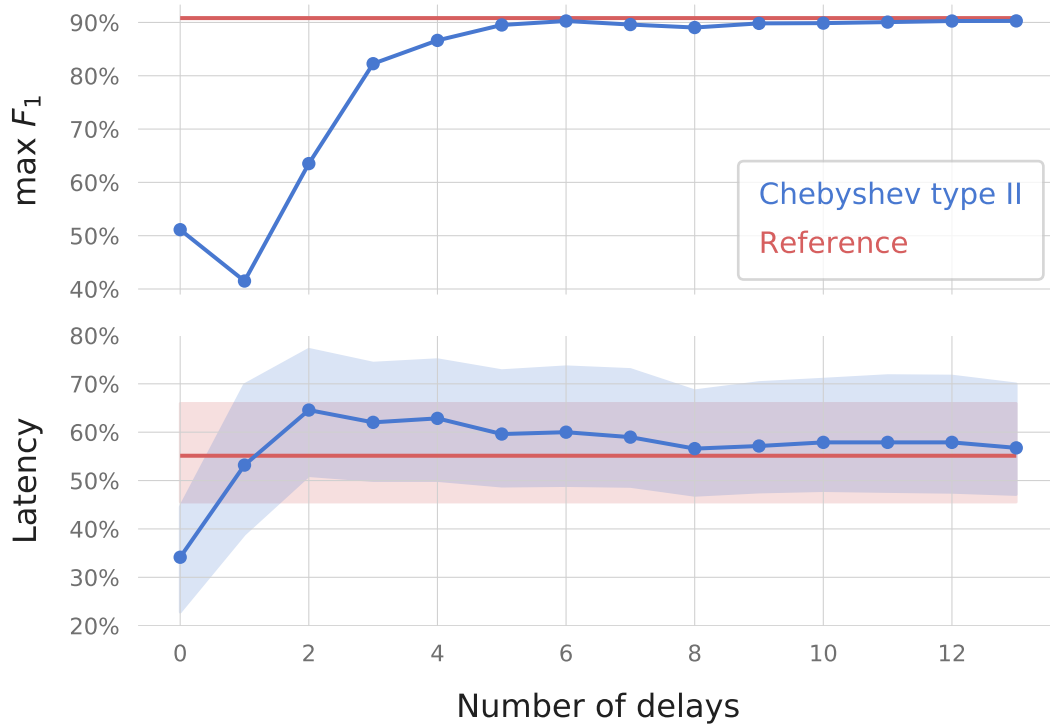


Figure 2.2: **Performance of Chebyshev type 2 IIR-filters**, for different filter orders. The minimum stop-band attenuation was set to 40 dB.

We also define a detection latency,  $RL_r$ , relative to the duration of the SWR segment in question:

$$RL_r = \frac{AL_r}{t_r^{\text{end}} - t_r^{\text{start}}} \quad (2.14)$$

Whereas an SWR detector yields only a single precision and a single recall value for a given threshold, it yields two entire sets of absolute and relative detection latencies (namely one absolute and one relative latency value for each detected reference segment). This is why we often summarize latency performance using the median of one of these sets. When not otherwise specified, the latency distribution (or its median) corresponding to the threshold with maximal  $F_1$ -score is reported.

## 2.3 Online band-pass filters [3p]

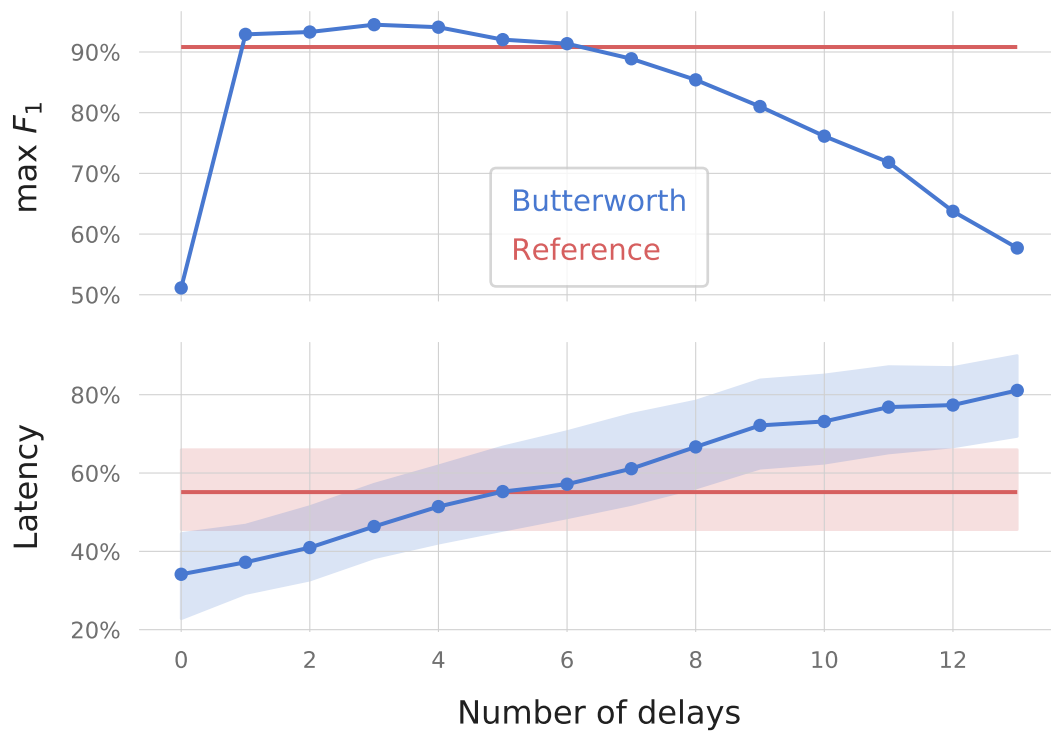


Figure 2.3: Performance of Butterworth IIR-filters, for different filter orders.

---

## Multi-channel linear filtering – 26 Oct [10p]

---

### 3.1 Data-driven algorithms [1p]

In this and the following chapter, we describe *supervised*, or data-driven SWR detection algorithms: they require training data  $\mathbf{z}_t^{\text{train}}$ , and an associated labelling  $y_t^{\text{train}}$  which marks the presence of an SWR event in  $\mathbf{z}_t^{\text{train}}$ , for every discrete time sample  $t$ . We arbitrarily define  $y_t \in \{0, 1\}$ , with  $y_t = 1$  when the corresponding input sample  $\mathbf{z}_t$  is part of an SWR segment, and  $y_t = 0$  when it is not.

The problem of obtaining such a labelling  $y_t$  for some recording data  $\mathbf{z}_t$  is the topic of section 2.1. Training labels can be obtained either by human expert labellers, or by using an automated *offline* SWR detection algorithm, where we assume that the automated labelling corresponds well to a supposed human expert labelling. In this thesis, we use the automated labelling method of section 2.1 to generate target labellings  $y_t^{\text{train}}$ .

Before a supervised algorithm can be used for real-time detection, its parameters have to be ‘tuned’. This is done using a training dataset  $(\mathbf{z}_t^{\text{train}}, y_t^{\text{train}})$ , during the so called *training phase*. Parameters are changed such that the algorithm’s output  $o_t$  for an input  $\mathbf{z}_t^{\text{train}}$  matches the target labelling  $y_t^{\text{train}}$  well. Sections 3.2 and 4.2 describe how this tuning can be done for two concrete detection algorithms.

The hope is that the trained algorithm also performs well on input data  $\mathbf{z}_t^{\text{test}}$  not part of the training set. That is, that the algorithm has good *generalization performance*. When this is not the case and the algorithm is tuned so that it only performs well on the training data, we say that the algorithm has been *overfit*. Often, so called *regularization* methods exist to discourage overfitting on the training data. Some example regularization methods are discussed in sections 3.5 and 4.3.

### 3.2 Linear signal-to-noise maximisation [2p]

In this chapter, we search for a linear combination of channels that yields an output signal  $o_t$  useful for sharp wave-ripple detection. More precisely, we search for a vector

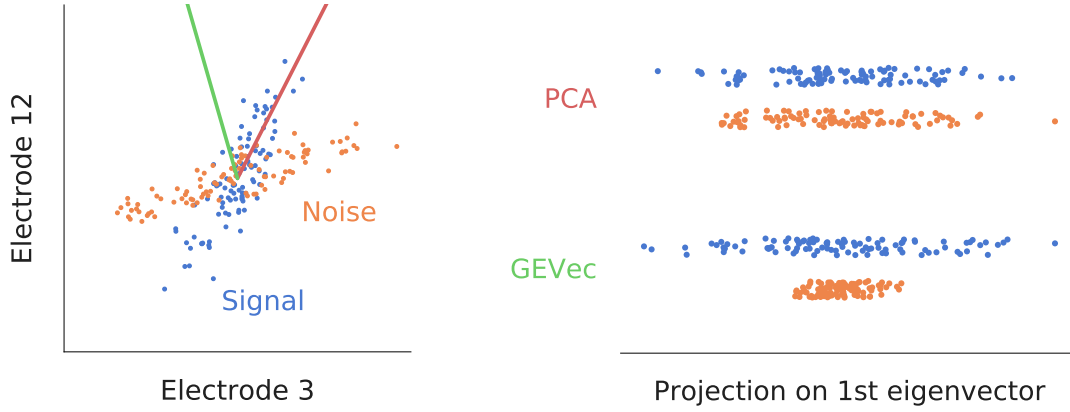


Figure 3.1: **Linear signal-to-noise maximisation.** Toy example to illustrate the generalized eigenvector approach to signal detection. *Left:* multi-channel time-series data plotted in ‘phase space’ (meaning without time axis), with blue dots representing samples where the signal was present, and orange dots representing samples where it was not. Actually toy data drawn from two 2-dimensional Gaussian distributions with different covariance matrices. Red vector: first eigenvector of the signal covariance matrix (also known as the first principal component). Green vector: first generalized eigenvector of the signal and noise covariance matrices. *Right:* Projection of both data sets on both the ordinary eigenvector (“PCA”) and the generalized eigenvector (“GEVec”). The ratio of the projected signal data variance versus the projected noise data variance is maximised for the GEVec case.

$\mathbf{w} \in \mathbb{R}^C$  in channel (or electrode) space to project the samples  $\mathbf{z}_t \in \mathbb{R}^C$  on, so that the output signal

$$o_t = \mathbf{w}^T \mathbf{z}_t \quad (3.1)$$

has high variance (or power) during SWR events, and low variance outside them.<sup>1</sup> This principle is illustrated with a two-dimensional toy dataset in fig. 3.1. We can then detect SWR events using threshold crossings of the envelope of  $o_t$ , as discussed in section 2.3.

The next two sections describe how this vector  $\mathbf{w}$  can be found.

### The optimisation problem

Suppose all training samples  $\mathbf{z}_t^{\text{train}}$  are gathered and divided over two data matrices  $\mathbf{S} \in \mathbb{R}^{C \times N_S}$  and  $\mathbf{N} \in \mathbb{R}^{C \times N_N}$ , where  $\mathbf{S}$  (for ‘signal’) contains all  $N_S$  samples of  $\mathbf{z}_t^{\text{train}}$  where an SWR is present, and  $\mathbf{N}$  (for ‘noise’) contains all  $N_N$  other samples. (These matrices can be easily constructed by concatenating segments from  $\mathbf{z}_t^{\text{train}}$ ).

<sup>1</sup>We assume that the input signals are zero-mean, such that the power  $P$  of the output signal equals its variance:  $P_o = \langle o_t^2 \rangle = \langle (o_t - \mu_o)^2 \rangle = \text{Var}(o_t)$  when  $\mu_o = 0$ , which is the case for zero-mean input channels:  $\mu_o = \langle o_t \rangle = \langle \mathbf{w}^T \mathbf{z}_t \rangle = \sum_i w_i \langle z_{t,i} \rangle = 0$  when  $\langle z_{t,i} \rangle = 0$  for all channels  $i$ .

This zero-mean assumption is reasonably well fulfilled for the analysed recording: the sample values of a 10-second moving average of the recording are near zero (median  $-0.02 \mu\text{V}$ , IQR  $0.13 \mu\text{V}$ . In comparison, the RMS-value of the recording is  $210 \mu\text{V}$ ).

Input data that is not zero-mean can be readily transformed to be so (even in an online setting), by subtracting a (moving) average from the input signal.

Equation (3.1) then becomes, in vector notation:

$$\begin{aligned}\mathbf{o}_S &= \mathbf{w}^T \mathbf{S} \\ \mathbf{o}_N &= \mathbf{w}^T \mathbf{N},\end{aligned}$$

where each element of the row vectors  $\mathbf{o}_S$  and  $\mathbf{o}_N$  is a filtered sample of  $\mathbf{S}$  and  $\mathbf{N}$ , respectively. Figure 3.1 (right) shows the distribution of the values in two example data vectors  $\mathbf{o}_S$  and  $\mathbf{o}_N$ .

We want to find the weight vector  $\hat{\mathbf{w}}$  that maximises the variance of  $\mathbf{o}_S$  versus the variance of  $\mathbf{o}_N$ , i.e.

$$\begin{aligned}\hat{\mathbf{w}} &= \arg \max_{\mathbf{w}} \frac{\text{Var}(\mathbf{o}_S)}{\text{Var}(\mathbf{o}_N)} \\ &= \arg \max_{\mathbf{w}} \frac{\frac{1}{N_S} \mathbf{o}_S \mathbf{o}_S^T}{\frac{1}{N_N} \mathbf{o}_N \mathbf{o}_N^T} \\ &= \arg \max_{\mathbf{w}} \frac{\frac{1}{N_S} \mathbf{w}^T \mathbf{S} \mathbf{S}^T \mathbf{w}}{\frac{1}{N_N} \mathbf{w}^T \mathbf{N} \mathbf{N}^T \mathbf{w}}\end{aligned}\tag{3.2}$$

In this last equation, we recognize the empirical covariance matrices  $\mathbf{R}_{SS}$  and  $\mathbf{R}_{NN}$ , which are defined as:

$$\mathbf{R}_{SS} = \frac{1}{N_S} \mathbf{S} \mathbf{S}^T \tag{3.3}$$

$$\mathbf{R}_{NN} = \frac{1}{N_N} \mathbf{N} \mathbf{N}^T \tag{3.4}$$

$\mathbf{R}_{SS} \in \mathbb{R}^{C \times C}$  and  $\mathbf{R}_{NN} \in \mathbb{R}^{C \times C}$  are symmetric matrices, where each diagonal element yields the variance of a channel, and each off-diagonal element yields the covariance between a pair of channels.

The condition for the optimal weight vector, eq. (3.2), is thus equivalent to:

$$\hat{\mathbf{w}} = \arg \max_{\mathbf{w}} \frac{\mathbf{w}^T \mathbf{R}_{SS} \mathbf{w}}{\mathbf{w}^T \mathbf{R}_{NN} \mathbf{w}} \tag{3.5}$$

In appendix C, we show that the solution  $\hat{\mathbf{w}}$  to this optimisation problem is the first so called “generalized eigenvector” of  $(\mathbf{R}_{SS}, \mathbf{R}_{NN})$ .

### The generalized eigenproblem

An arbitrarily scaled vector  $\mathbf{w}_i$  is a so called *generalized eigenvector* (GEVec) for the ordered matrix pair  $(\mathbf{R}_{SS}, \mathbf{R}_{NN})$  when the following holds:

$$\mathbf{R}_{SS} \mathbf{w}_i = \lambda_i \mathbf{R}_{NN} \mathbf{w}_i, \tag{3.6}$$

for some scalar  $\lambda_i$ , which is called the *generalized eigenvalue* (GEVal) corresponding to  $\mathbf{w}_i$ . The largest scalar  $\lambda_1$  for which eq. (3.6) holds is the ‘first’ GEVal, and as mentioned before, the corresponding GEVec  $\mathbf{w}_1$  is the solution  $\hat{\mathbf{w}}$  to eq. (3.5).

Since the 1960's, numerically stable algorithms exist that solve the generalized eigenproblem eq. (3.6) [12]. A specialized algorithm is applicable when the input matrices are symmetric – as is the case for  $\mathbf{R}_{SS}$  and  $\mathbf{R}_{NN}$ . This algorithm (based on a Cholesky factorization and the classical QR-algorithm for ordinary eigenproblems) is implemented in the LAPACK software package (as `ssygv` and `dsygv`), and can be easily applied using e.g. the `eig` function from MATLAB, or the `eigh` function from SciPy's `linalg` module.

### 3.3 Result for SWR detection

We divided the 34-minute long LFP recording into two datasets. The first 60% was used as training data, to calculate the covariance matrices  $\mathbf{R}_{SS}$  and  $\mathbf{R}_{NN}$ , and to calculate from these the optimal linear combination of channels  $\hat{\mathbf{w}}$ , as described in the preceding sections. The remaining 40% was used to evaluate this filter  $\hat{\mathbf{w}}$ , and to compare it to the state-of-the-art method (the single-channel online band-pass filter).<sup>2</sup>

Figure 3.2A shows an excerpt of the test input signal, and the corresponding filter output envelopes (blue for state-of-the art method, orange for GEVec-based multichannel method). Filter outputs  $o_t$  are rectified to obtain envelopes  $n_t = |o_t|$ ). Additional excerpts are shown in fig. D.2. The elements of  $\hat{\mathbf{w}}$  (i.e. the filter weights) are visualized in fig. 3.2B.

It is clear that the GEVec filter output indeed has high power during SWR events, as promised by the theoretical derivation. The filter weights and signal excerpts reveal that the GEVec output is composed mainly of a few channels in the stratum radiatum (channels 4-5-6 here), where they pick up the sharp waves. However, the filter output envelope is also high for sharp wave-like activity on these channels, without or with only very weak ripple activity in the higher channels: see figs. D.2a, D.2b and D.2d. This results in false positive detections.

We also notice some premature detections (figs. D.2a and D.2c), where the sharp wave – and thus also the GEVec filter output – already has high power before the corresponding ripple has started. The GEVec detection then happens before the start of the reference SWR-segment, which is based on ripple power. These early detections thus count (arguably unfairly so) as false positives, under the evaluation scheme that we use.

This effect, where the sharp wave is discernible before the ripple, results in faster detections when the detection *does* fall within the reference segment. When all 468 SWR events from the test set are analysed, we find a large improvement in detection latency: at thresholds where both methods detect 80% of these reference SWR segments, the median absolute detection latency drops from 24 ms for the state-of-the-art online band-pass filter to 12 ms for the GEVec-based filter. The relative detection latency drops by 32.5 percentage points, from 58.1% to 25.6%. Similar latency improvements are found for other recall and precision values: see fig. 3.3.

This strong improvement in detection latency trades off with an increase in false positives, as was already observed qualitatively. At the aforementioned sensitivity of 80%, the

<sup>2</sup>The 60-40 division was chosen arbitrarily – under the constraint that there are sufficient amounts of both training and test data.

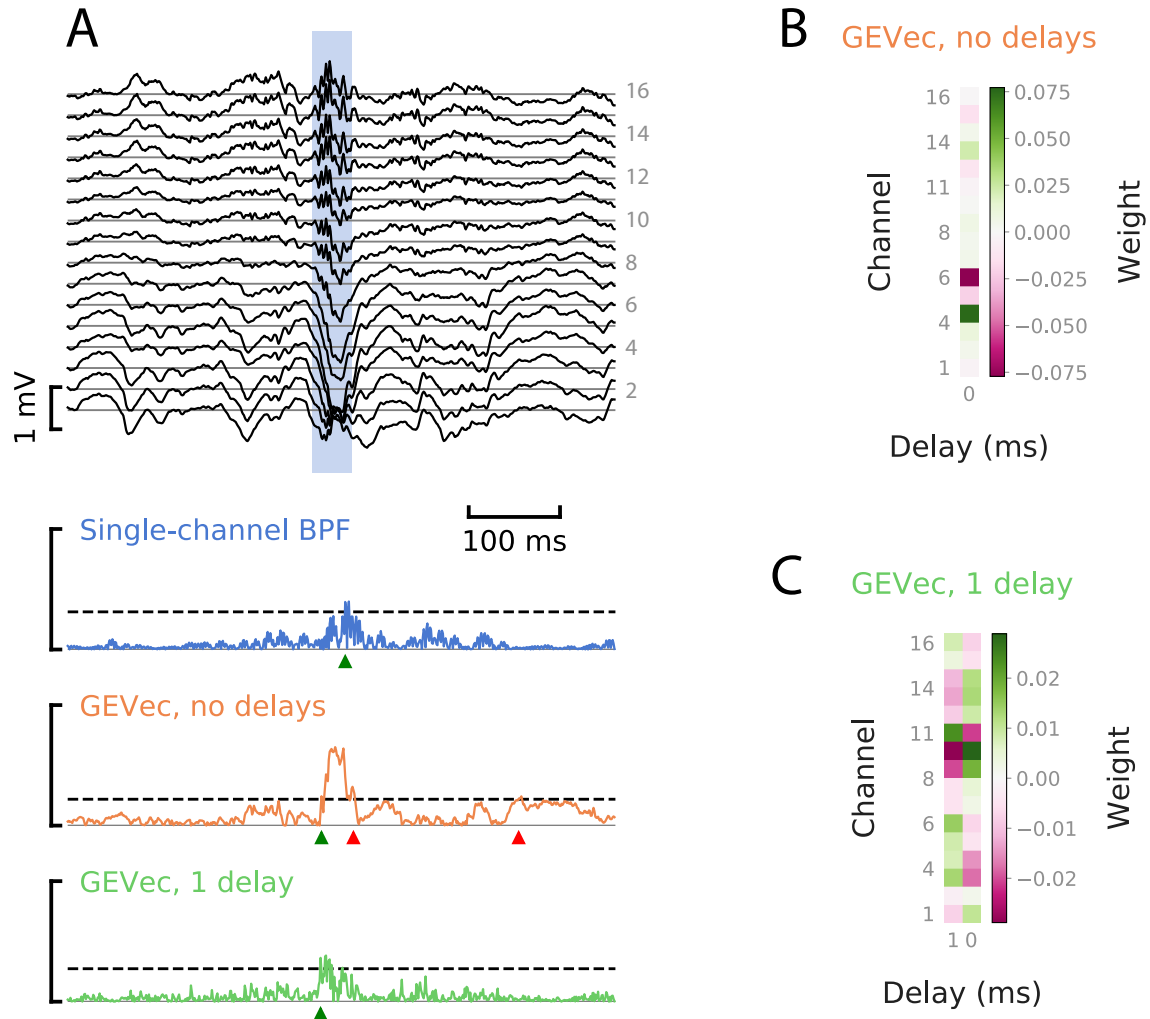


Figure 3.2: **Linear, SNR-maximising combinations of electrodes.**

**A.** Example input and output signals. *Top*: multi-channel LFP,  $\mathbf{z}_t$ . Light-blue vertical band: a reference SWR segment. *Bottom*: output envelopes  $n_t$ , for different filtering algorithms. Dashed horizontal lines: detection thresholds, chosen so that each algorithm reaches a recall value of 80%. Green triangles: correct detections. Red triangles: incorrect detections. Brackets indicate envelope range (min, max) over the entire test set.

**B.** Generalized eigenvector  $\hat{\mathbf{w}}$  (i.e. the weights of the multichannel filter), for a purely spatial filter.

**C.** Generalized eigenvector  $\hat{\mathbf{w}}$  for a spatiotemporal filter with a one-sample delay.

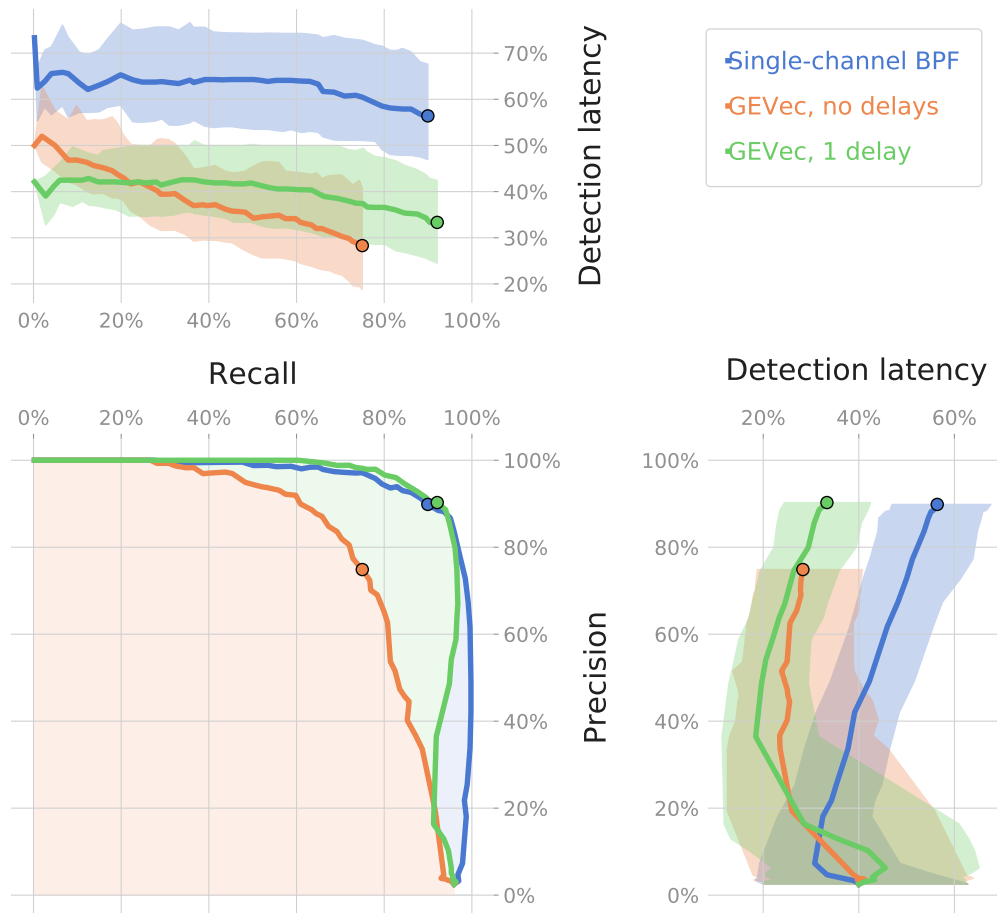


Figure 3.3: **Sensitivity, precision & latency tradeoffs**, for different linear filters & for a range of thresholds.

Each threshold setting for an algorithm corresponds to a point on the precision-recall curve in the bottom-left panel, and to a distribution of relative detection latencies. The median and interquartile range of this distribution are plotted in the top-left or bottom-right panel (as a point of the bold curve, and a slice of the shaded band, respectively).

These latency distribution plots are divided over two panels so that their entire range can be clearly visualized, both for the low recall – high precision regime as for the high recall – low precision regime. The cutoff is made at the point where recall equals precision (AKA the  $\max F_1$  point), marked with shaded black circles.



state-of-the-art method has a precision of 94%, whereas the GEVec-based method has a precision of only 63% (i.e. more than a third of detected events are classified as false positives). This strong decrease in precision is true over the entire  $PR$ -curve: see fig. 3.3.

### 3.4 Combining space and time [1p]

It is hardly surprising that the GEVec-based algorithm as described above cannot discern ripple activity (which is by definition a temporal pattern), as the algorithm is a purely *spatial* filter: at each timestep  $t$ , only current information from the different channels is used in calculating the output  $o_t$ , without incorporating temporal information from previous timesteps  $t_p < t$ .

The GEVec method can be easily adapted to also incorporate temporal information however, by defining a vector  $\mathbf{z}_t^{\text{stack}} \in \mathbb{R}^{CP}$  which consists of stacked sample vectors (each consisting of  $C$  channels) from  $P$  different timesteps  $t_p \leq t$ . The linear weights  $\mathbf{w}^{\text{stack}} \in \mathbb{R}^{CP}$  used to obtain the output signal  $o_t = (\mathbf{w}^{\text{stack}})^T \mathbf{z}_t^{\text{stack}}$  are then calculated analogously to the purely spatial filter, i.e. as the first generalized eigenvector of the ordered pair  $(\mathbf{R}_{SS}^{\text{stack}}, \mathbf{R}_{NN}^{\text{stack}})$ , with both covariance matrices  $\in \mathbb{R}^{CP \times CP}$ .

Adding just one such delayed time step (i.e.  $P = 2$ ) yields a major performance improvement (see fig. 3.3): the precision-recall curve shoots up to (and even slightly exceeds) the  $PR$ -curve of the state-of-the-art algorithm, while the latency improvements of the ‘no delay’ GEVec algorithm are mostly retained: at a sensitivity of 80%, the median absolute latency for the one delay GEVec filter is 15 ms, which is 9 ms faster than the state-of-the-art method (and 3 ms slower than the no delay GEVec filter). The relative latency is 36.6%: 21.5 percentage-points lower than the state-of-the-art (and 11 pp. higher than the no delay GEVec filter).

At this 80% recall mark, the one delay GEVec filter attains a precision of 97%, a 3% increase over the state-of-the-art. The full precision-recall-latency tradeoff and algorithm comparison is shown in fig. 3.3. Note that for very low thresholds, the  $PR$ -curve of the one-delay GEVec method is no longer concave: decreasing the threshold further yields more (not less) missed reference SWR segments. Figure 3.2C shows the GEVec  $\mathbf{w}^{\text{stack}}$ . Figure 3.2A and fig. D.2 show example output envelopes (green traces).

These results, particularly the visualized weights in fig. 3.2C, indicate that this one-delay GEVec filter utilizes spatiotemporal information about both the sharp wave and the ripple.

#### Choosing the number of delays

Adding more delays (fig. 3.4) improves detection accuracy even further – up to a peak  $\max F_1$  of 93% at about eleven delays. (This corresponds to eleven milliseconds, or approximately half a ripple phase). Detection latency also increases with increasing number of delays, but only slightly, always staying well below the state-of-the-art latency. Like the  $\max F_1$  score, latency stagnates after about eleven delays. Further, we note that the latency distributions of the GEVec-based detectors have a lower spread than those of the state-of-the-art detector.

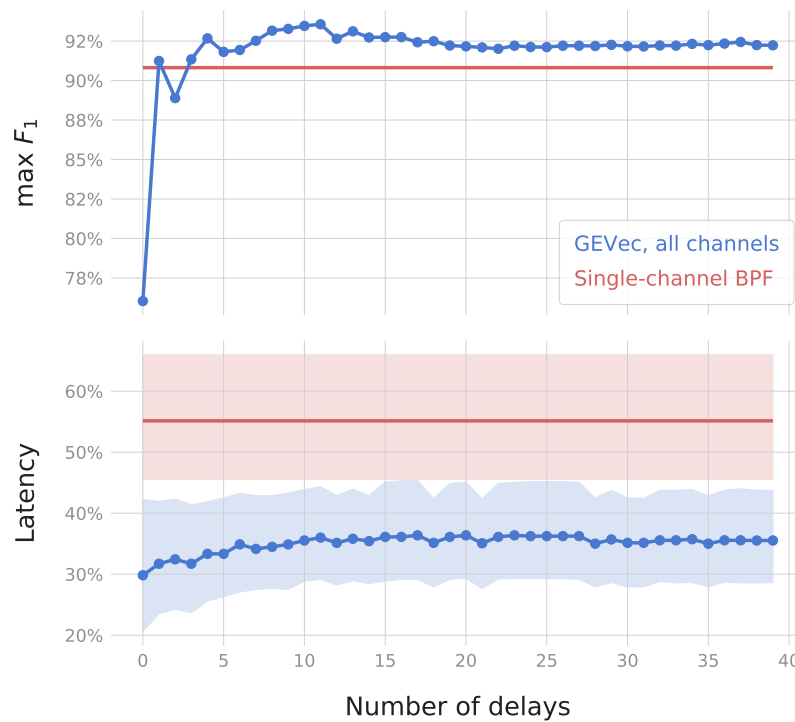


Figure 3.4: **Performance of the GEVec-based SWR detector, for different delay line lengths.** At the chosen 1000 Hz sampling rate, each delay corresponds to 1 ms. The red baseline is the state-of-the-art SWR detector. Detection latency is specified as a fraction of the duration of the corresponding SWR event, and is evaluated at the threshold where each detector reaches its maximum  $F_1$ -score. In the latency panel, bold lines and shaded areas indicate the median and the interquartile range of the latency distributions, respectively.

There is thus a slight tradeoff to be made when choosing the number of delays for a GEVec-based detector: using more delays yields detectors that are more accurate, but also slightly slower. Given that the decrease in speed is minor (about five percentage-points), it is reasonable to choose the amount of delays that maximizes detection accuracy. In this analysis, this optimal point is reached at an eleven milliseconds-long delay line.

### Selecting channels

Many CA1 LFP recordings are not made with multichannel probes, but rather with one or more tetrodes. It is therefore relevant to ask how GEVec-based detectors perform on these types of recordings. We can approximate this setting with our current multichannel probe recording, by only using one or a few channels.

Each such selection of input channels, in combination with a certain number of delays, yields a different GEVec-based SWR detector. We evaluate and compare these detectors using the test data set, analyzing both accuracy (fig. 3.5) and latency (fig. 3.6).

We note some general trends. Using no extra delays results in low accuracy detectors, no matter which channels are included (top row of fig. 3.5). For most input channel

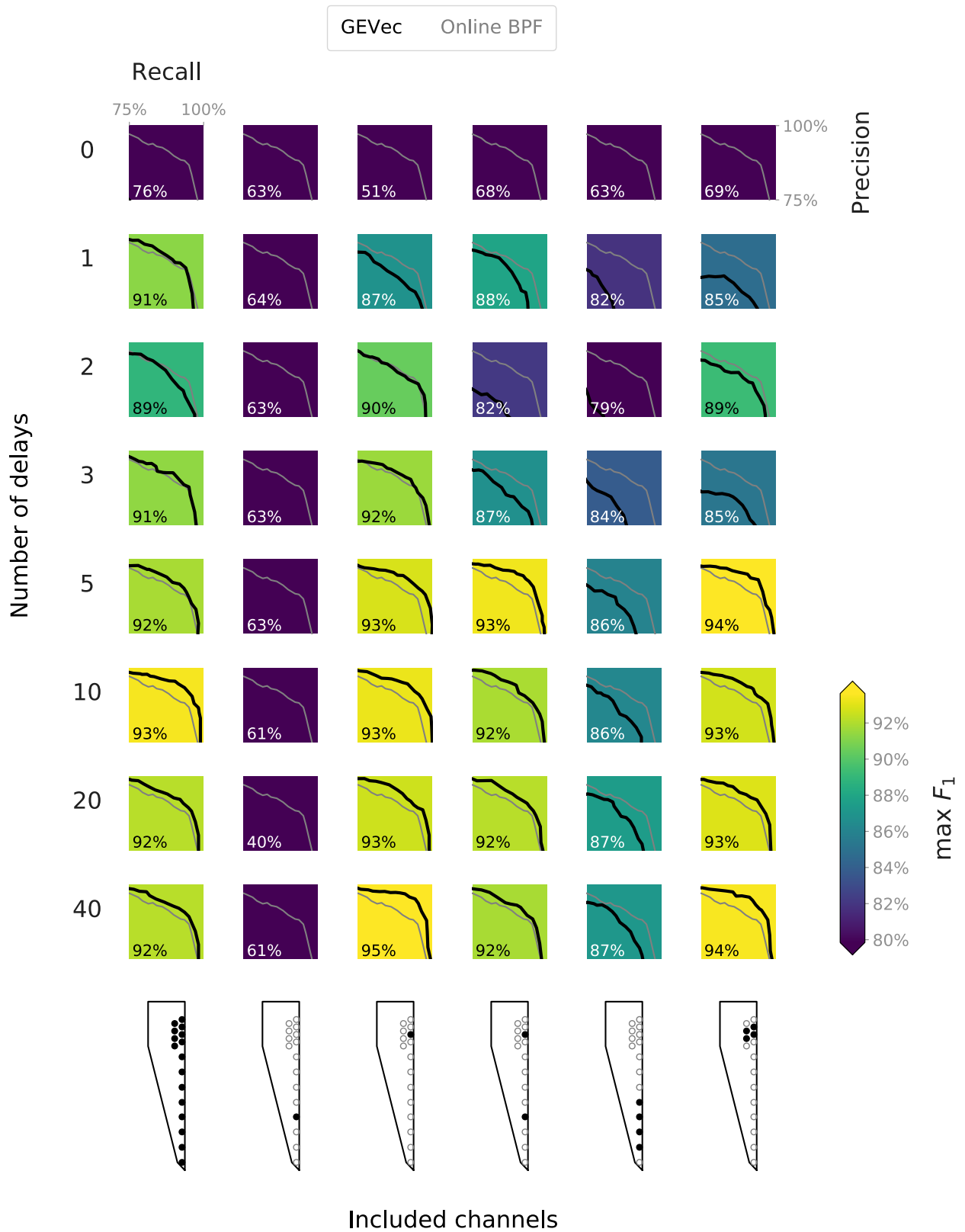


Figure 3.5: **Accuracy of GEVec-based SWR detectors**, for different combinations of active input channels and number of delays. Active input channels are marked in solid black on a schematic of the probe tip. (Height of schematic: 687  $\mu\text{m}$ ). The percentage and background color of each panel indicate the maximum  $F_1$  value obtained for that GEVec-based SWR detector. Each panel includes the same baseline from the state-of-the-art SWR detector, in gray.



Figure 3.6: **Latency of GEVec-based SWR detectors.** See fig. 3.5 for legend.

combinations, the tradeoff in number of delays observed in the previous section is preserved: using more delays increases accuracy (up to a certain point), while also increasing detection latency.

The weight visualizations of fig. 3.2 show a relatively high weight for the stratum radiatum / sharp wave channels. Using only one such channel as input results in low accuracy detectors however, no matter how many delays are used (second column of figs. 3.5 and 3.6). Using four such channels (second to last column) results in detectors with an accuracy approaching, but not reaching, that of the state-of-the-art ripple-based detector when a sufficient number of delays is used. Interestingly, this decent accuracy is reached despite none of these four channels (numbers 2 to 5) displaying any noticeable ripple activity (fig. 3.2A).

Conversely, using one channel from the pyramidal cell layer (where ripples are observed) yields GEVec-based detectors on par with those that use all channels as input, both regarding accuracy and latency (third and first column of figs. 3.5 and 3.6).

This one channel GEVec-based detector thus also outperforms the “state-of-the-art” detector based on a band-pass filter. The state-of-the-art detector also uses only one channel as input, and is also a linear filter. This suggests that either 1) finding linear filter weights through ‘machine learning’ (the GEVec technique) yields better results than manual filter design; or 2) that the band-pass filter used as “state-of-the-art” detector was not designed optimally in the first place.

Adding a stratum radiatum / sharp wave channel to the single pyramidal cell layer channel (fourth column of figs. 3.5 and 3.6), or using a cluster of channels in the pyramidal cell layer (last column) does not improve performance over the one channel case.

### **3.5 Regularization [1p]**

### **3.6 Multiple eigenvectors [2p]**

### **3.7 Spectral preprocessing [1p]**

## Nonlinear signal detection – 9 Nov [9p]

---

- 4.1 Recurrent neural networks [2p]
- 4.2 Optimization [3p]
- 4.3 Regularization [2p]
- 4.4 Channels [1p]
- 4.5 Network size [1p]

---

## Discussion – 23 Nov [4p]

---

5.1 Comparing detectors [2p]

5.2 Further work [2p]

---

## Conclusions – 30 Nov [1p]

---



---

## References

---

- [1] C. J. Van Rijsbergen. *Information Retrieval*. 2nd edition. Newton, MA, USA: Butterworth-Heinemann, 1979. URL: <http://www.dcs.gla.ac.uk/Keith/Preface.html>.
- [2] György Buzsáki. “Hippocampal Sharp Wave-ripple: A Cognitive Biomarker for Episodic Memory and Planning”. In: *Hippocampus* 25.10 (Sept. 26, 2015), pp. 1073–1188. ISSN: 1050-9631. DOI: 10.1002/hipo.22488. URL: <http://onlinelibrary.wiley.com/doi/full/10.1002/hipo.22488> (visited on 04/23/2018).
- [3] Tom Roelandts. *How to Create a Configurable Filter Using a Kaiser Window*. Dec. 26, 2016. URL: <https://tomroelandts.com/articles/how-to-create-a-configurable-filter-using-a-kaiser-window> (visited on 11/25/2018).
- [4] Eric Jones, Travis Oliphant, Pearu Peterson, et al. “SciPy: Open Source Scientific Tools for Python”. In: (2018). URL: <http://www.scipy.org/>.
- [5] Alan V. Oppenheim and Ronald W. Schaffer. *Discrete-Time Signal Processing*. 3rd. Upper Saddle River, NJ, USA: Prentice Hall Press, 2009. ISBN: 978-0-13-198842-2.
- [6] Richard G. Lyons. *Understanding Digital Signal Processing*. Pearson Education, Nov. 1, 2010. 1166 pp. ISBN: 978-0-13-702852-8.
- [7] Zoltán Nádasdy, Hajime Hirase, András Czurkó, Jozsef Csicsvari, and György Buzsáki. “Replay and Time Compression of Recurring Spike Sequences in the Hippocampus”. In: *Journal of Neuroscience* 19.21 (Nov. 1, 1999), pp. 9497–9507. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.19-21-09497.1999. pmid: 10531452. URL: <http://www.jneurosci.org/content/19/21/9497> (visited on 08/02/2018).
- [8] Jozsef Csicsvari, Hajime Hirase, Akira Mamiya, and György Buzsáki. “Ensemble Patterns of Hippocampal CA3-CA1 Neurons during Sharp Wave-Associated Population Events”. In: *Neuron* 28.2 (Nov. 1, 2000), pp. 585–594. ISSN: 0896-6273. DOI: 10.1016/S0896-6273(00)00135-5. URL: <http://www.sciencedirect.com/science/article/pii/S0896627300001355> (visited on 08/01/2018).
- [9] Shayok Dutta, Etienne Ackermann, and Caleb Kemere. “Analysis of an Open Source, Closed-Loop, Realtime System for Hippocampal Sharp-Wave Ripple Disruption”. In: *bioRxiv* (Apr. 11, 2018). DOI: 10.1101/298661. URL: <https://www.biorxiv.org/content/early/2018/04/11/298661> (visited on 05/30/2018).

- [10] Christoph J. Behrens, Leander P. van den Boom, Livia de Hoz, Alon Friedman, and Uwe Heinemann. “Induction of Sharp Wave–Ripple Complexes *in Vitro* and Reorganization of Hippocampal Networks”. In: *Nature Neuroscience* 8.11 (Nov. 2005), pp. 1560–1567. ISSN: 1546-1726. DOI: 10.1038/nn1571. URL: <https://www.nature.com/articles/nn1571> (visited on 07/31/2018).
- [11] Josef H. L. P. Sadowski, Matthew W. Jones, and Jack R. Mellor. “Sharp-Wave Ripples Orchestrate the Induction of Synaptic Plasticity during Reactivation of Place Cell Firing Patterns in the Hippocampus”. In: *Cell Reports* 14.8 (Mar. 1, 2016), pp. 1916–1929. ISSN: 2211-1247. DOI: 10.1016/j.celrep.2016.01.061. URL: <http://www.sciencedirect.com/science/article/pii/S2211124716300390> (visited on 06/17/2018).
- [12] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Fourth edition. The Johns Hopkins University Press, 2013.
- [13] John O’Keefe. “Hippocampal Neurophysiology in the Behaving Animal”. In: *The Hippocampus Book*. Oxford University Press, 2007.
- [14] Valérie Ego-Stengel and Matthew A Wilson. “Disruption of Ripple-Associated Hippocampal Activity during Rest Impairs Spatial Learning in the Rat”. In: *Hippocampus* 20.1 (2009), pp. 1–10. URL: <http://onlinelibrary.wiley.com.kuleuven.ezproxy.kuleuven.be/doi/10.1002/hipo.20707/full>.
- [15] Laura Lee Colgin. “Rhythms of the Hippocampal Network”. In: *Nature Reviews Neuroscience* 17.4 (2016), p. 239.
- [16] Howard Eichenbaum. “Prefrontal–Hippocampal Interactions in Episodic Memory”. In: *Nature Reviews Neuroscience* 18.9 (2017), p. 547.
- [17] H Freyja Ólafsdóttir, Daniel Bush, and Caswell Barry. “The Role of Hippocampal Replay in Memory and Planning”. In: *Current Biology* 28.1 (2018), R37–R50.
- [18] Jozsef Csicsvari, Hajime Hirase, András Czurkó, Akira Mamiya, and György Buzsáki. “Oscillatory Coupling of Hippocampal Pyramidal Cells and Interneurons in the Behaving Rat”. In: *Journal of Neuroscience* 19.1 (Jan. 1, 1999), pp. 274–287. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.19-01-00274.1999. pmid: 9870957. URL: <http://www.jneurosci.org/content/19/1/274> (visited on 11/13/2018).
- [19] Jozsef Csicsvari, Hajime Hirase, András Czurkó, Akira Mamiya, and György Buzsáki. “Fast Network Oscillations in the Hippocampal CA1 Region of the Behaving Rat”. In: *The Journal of Neuroscience* 19.16 (Aug. 15, 1999), RC20–RC20. ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.19-16-j0001.1999. URL: <http://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.19-16-j0001.1999> (visited on 11/13/2018).
- [20] Sen Cheng and Loren M. Frank. “New Experiences Enhance Coordinated Neural Activity in the Hippocampus”. In: *Neuron* 57.2 (Jan. 24, 2008), pp. 303–313. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2007.11.035. pmid: 18215626. URL: [https://www.cell.com/neuron/abstract/S0896-6273\(07\)01030-6](https://www.cell.com/neuron/abstract/S0896-6273(07)01030-6) (visited on 11/13/2018).

- [21] Caleb Kemere, Margaret F. Carr, Mattias P. Karlsson, and Loren M. Frank. “Rapid and Continuous Modulation of Hippocampal Network State during Exploration of New Places”. In: *PLOS ONE* 8.9 (2-Sep-2013), e73114. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0073114. URL: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0073114> (visited on 11/13/2018).
- [22] Lloyd N. Trefethen and David III Bau. *Numerical Linear Algebra*. Siam, 1997.

---

# Appendices

---

<b>A</b>	<b>SWR detection in the literature</b>	<b>37</b>
A.1	Frequency bands . . . . .	37
A.2	Detection algorithms . . . . .	39
	Nádasdy et al. 1999 . . . . .	39
	Csicsvari et al. 2000 . . . . .	39
	Behrens et al. 2005 . . . . .	39
	Sadowski et al. 2016 . . . . .	39
	Dutta et al. 2018 . . . . .	39
<b>B</b>	<b>Data description</b>	<b>41</b>
<b>C</b>	<b>Generalized eigenvectors maximize signal-to-noise</b>	<b>42</b>
	Theorem . . . . .	42
	Proof . . . . .	42
<b>D</b>	<b>Supplemental figures</b>	<b>45</b>

## *Appendix A*

---

# SWR detection in the literature

---

### A.1 Frequency bands

Source	Theta (Hz)	High gamma (Hz)	Ripple (Hz)
Nádasdy et al. 1999 [7]			150 – 250
Csicsvari et al. 2000 [8]			80 – 250
Behrens et al. 2005 [10]			40 – 400
O’Keefe 2007 [13]	6 – 12	30 – 100	100 – 200
Ego-Stengel et al. 2009 [14]			100 – 400
Buzsáki 2015 [2]	6 – 10	100+	110 – 200
Colgin 2016 [15]	6 – 12	60 – 100	150 – 200
Sadowski et al. 2016 [11]			120 – 250
Eichenbaum 2017 [16]	4 – 12	80 – 140	
Dutta et al. 2018 [9]			150 – 250
Ólafsdóttir et al. 2018 [17]	6 – 12		140 – 250
fklab	6 – 12	60 – 140	140 – 225
L2 recording	5 – 10		100 – 200

Table A.1: **Frequency bands of hippocampal LFP events**, according to different sources (both original research papers and literature reviews). When the source does not make a distinction between high and low gamma, the full gamma range is given. Note that Behrens et al. considered artificially induced SWR’s in ex-vivo hippocampus slices. ‘fklab’ refers to the default frequency bands used in the data-analysis software used in the Kloosterman lab. The last row refers to the dataset analysed in this thesis.

## A.2 Detection algorithms

We gathered a representative sample of research papers that use automated offline sharp wave-ripple detection. This appendix quotes these papers; specifically their description of the SWR detection procedure. Emphasis is added.

### Nádasdy et al. 1999

“For the extraction of sharp-wave (SPW) ripple events during sleep, the wide-band recorded data were bandpass filtered digitally (**150–250 Hz**). The power (**root mean square**) of the filtered signal was calculated, and the beginning, peak, and end of individual ripple episodes were determined. The threshold for ripple detection was set to **7 SDs above the background mean** power (Csicsvari et al., 1999 [18]).” [7]

### Csicsvari et al. 2000

“*Detection of SPW-Associated Fast Ripples*: The procedures described here were identical to those described earlier (Csicsvari et al., 1999b [19]). The wide-band (1–5 kHz) recorded data was digitally band-pass filtered (**80–250 Hz**), and the power (**root-mean-square**) of the filtered signal was calculated for each electrode. The mean and standard deviation (SD) of the power signal were calculated to determine the detection threshold. Oscillatory epochs with a power of **one or more SD above the mean** were detected. The beginning and the end of oscillatory epochs were marked at points where the power fell below **0.5 SD**. Theta periods, detected by using the theta-delta power ratio (Csicsvari et al., 1999a [18]), were excluded from the analysis.” [8]

### Behrens et al. 2005

“For ripple detection, raw data were filtered with a Spike 2 software band-pass filter of **40–400 Hz** (threshold: **4–6** times the s.d. of **eventless baseline noise**). For sharp wave detection, recordings were low-pass filtered at 20 Hz.” [10]

### Sadowski et al. 2016

“Ripples were detected offline in the LFP recorded on one CA1 channel. Raw LFP signal was filtered between **120 and 250 Hz**, and deflections in the ripple **power envelope** greater than **5 SDs from the mean** were classified as ripple events. Ripple start times were defined locally as when ripple power exceeded **2 SDs**. Samples of raw LFP and detected ripple times were compared manually to verify detection fidelity.” [11]

### Dutta et al. 2018

“Post-recording, ripple events were defined on tetrodes that displayed characteristics of the CA1 area of the hippocampus. Specifically, the recorded LFP in one of the channels of the selected tetrode (same one subject to online detection for our realtime analysis) first had a digital reference subtracted away. This signal was then LFP band filtered with a 400 Hz low-pass infinite impulse response (IIR) filter (from TRODES). Afterwards the signal was decimated and ripple band filtered (**150–250 Hz**) with a **25 tap** finite impulse response (**FIR**) filter. Ripple band filtering was done using a forward and a time-reversed

path, resulting in a net **zero group delay** (time shift from filtering). The instantaneous power of the ripple band filtered signal was then calculated via a **Hilbert Transform** and further **smoothened with a Gaussian kernel** with a **4 ms standard deviation**. Ripple events were detected as times when z-score of the smoothened power signal exceeded a threshold of **3 z-units** for **at least 15 ms**. The canonical ripple epochs were defined as the time points from which the processed signal **returned down to the mean before and after threshold crossings** [20], [21].

In cases when multiple electrodes (typically channels on different tetrodes) are available for ripple detection, a different canonical definition is required. Ripples were initially defined using as above for each electrode. A canonical multichannel ripple was defined as one which is simultaneously detected on each electrode (two in our analysis). The multichannel ripple epoch is defined as the union of the detected single-channel ripple epochs, i.e., the start of the earliest ripple detected and to end with bound of last ripple detected. As such, we obtain a conservative ripple detection latency estimate while covering the entire span of the time the LFP is in a high ripple band power state. We reanalyzed our data with the canonical ripples being defined on different channels and tetrodes with a 300 tap bandpass FIR filter allowing 1% “ripple” in the passband with 30 dB suppression in the stopband but our results and subsequent conclusions remained consistent.” [9]



*Appendix B*

---

## Data description

---

## Generalized eigenvectors maximize signal-to-noise

---

In this appendix, we show that the weight vector that maximizes the ratio of signal to noise variance (i.e. the solution  $\hat{\mathbf{w}}$  to eq. (3.5)) is equivalent to the first generalized eigenvector  $\mathbf{w}_1$  of the ordered pair of covariance matrices  $(\mathbf{R}_{SS}, \mathbf{R}_{NN})$  (as defined in section 3.2).

This ratio of variances in eq. (3.5) is a quotient of quadratic forms, namely the so called “generalized Rayleigh quotient” of  $(\mathbf{R}_{SS}, \mathbf{R}_{NN})$ .

Formally, the generalized Rayleigh quotient of a non-zero vector  $\mathbf{w} \in \mathbb{R}^N$  and the ordered, symmetric matrix pair  $(\mathbf{A}, \mathbf{B})$  is the scalar  $r(\mathbf{w})$  defined as:

$$r(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{A} \mathbf{w}}{\mathbf{w}^T \mathbf{B} \mathbf{w}} \quad (\text{C.1})$$

We must then prove the following:

### Theorem

The generalized eigenvector  $\mathbf{w}_1$  corresponding to the largest generalized eigenvalue  $\lambda_1$  of  $(\mathbf{A}, \mathbf{B})$ , is also the vector  $\hat{\mathbf{w}}$  that maximises the generalized Rayleigh quotient  $r(\mathbf{w})$  of  $(\mathbf{A}, \mathbf{B})$ .

### Proof

As a first step, we will show that if  $\hat{\mathbf{w}}$  is the maximum of  $r(\mathbf{w})$ , that it is indeed an eigenvector of  $(\mathbf{A}, \mathbf{B})$ . In the second step, we will show that the largest eigenvalue  $\lambda_1$  of  $(\mathbf{A}, \mathbf{B})$  corresponds to the maximum of  $r(\mathbf{w})$ .

If  $\hat{\mathbf{w}}$  is a maximum of  $r(\mathbf{w})$ , then

$$\nabla r(\hat{\mathbf{w}}) = \mathbf{0}. \quad (\text{C.2})$$

Working out the partial derivatives that comprise the gradient of  $r(\mathbf{w})$ , we find:

$$\nabla r(\mathbf{w}) = \frac{2\mathbf{A}\mathbf{w}(\mathbf{w}^T\mathbf{B}\mathbf{w}) - 2\mathbf{B}\mathbf{w}(\mathbf{w}^T\mathbf{A}\mathbf{w})}{(\mathbf{w}^T\mathbf{B}\mathbf{w})^2}$$

With eq. (C.2), we then have the following condition for our maximising vector  $\hat{\mathbf{w}}$ :

$$2\mathbf{A}\hat{\mathbf{w}}(\hat{\mathbf{w}}^T\mathbf{B}\hat{\mathbf{w}}) = 2\mathbf{B}\hat{\mathbf{w}}(\hat{\mathbf{w}}^T\mathbf{A}\hat{\mathbf{w}})$$

or

$$\mathbf{A}\hat{\mathbf{w}} = \frac{\hat{\mathbf{w}}^T\mathbf{A}\hat{\mathbf{w}}}{\hat{\mathbf{w}}^T\mathbf{B}\hat{\mathbf{w}}} \mathbf{B}\hat{\mathbf{w}}$$

$$\mathbf{A}\hat{\mathbf{w}} = r(\hat{\mathbf{w}}) \mathbf{B}\hat{\mathbf{w}}$$

This is the generalized eigenvalue/eigenvector definition (eq. (3.6)) for  $\mathbf{w}_i = \hat{\mathbf{w}}$  and  $\lambda_i = r(\hat{\mathbf{w}})$ .

We have thus shown that if  $\hat{\mathbf{w}}$  is a maximum of  $r(\mathbf{w})$ , that it is an eigenvector of  $(\mathbf{A}, \mathbf{B})$ , with  $r(\hat{\mathbf{w}})$  its corresponding eigenvalue.

As the second step, we now show that  $r(\hat{\mathbf{w}})$  is the *largest* eigenvalue of  $(\mathbf{A}, \mathbf{B})$ . We follow the reasoning of Trefethen and Bau, who prove a related result for the ordinary Rayleigh quotient [22, p. 204].

We will rewrite the generalized Rayleigh quotient  $r(\mathbf{w})$  by writing the arbitrary vector  $\mathbf{w}$  as a linear combination of the generalized eigenvectors  $\mathbf{w}_i$  of  $(\mathbf{A}, \mathbf{B})$ :  $\mathbf{w} = \sum_i c_i \mathbf{w}_i$ . Then:

$$\begin{aligned} r(\mathbf{w}) &= \frac{(\sum_i c_i \mathbf{w}_i)^T \mathbf{A} (\sum_i c_i \mathbf{w}_i)}{(\sum_i c_i \mathbf{w}_i)^T \mathbf{B} (\sum_i c_i \mathbf{w}_i)} \\ &= \frac{\sum_i c_i^2 \mathbf{w}_i^T \mathbf{A} \mathbf{w}_i}{\sum_i c_i^2 \mathbf{w}_i^T \mathbf{B} \mathbf{w}_i} \\ &= \frac{\sum_i c_i^2 \lambda_i \mathbf{w}_i^T \mathbf{B} \mathbf{w}_i}{\sum_i c_i^2 \mathbf{w}_i^T \mathbf{B} \mathbf{w}_i}. \end{aligned}$$

Generalized eigenvectors are defined up to a scaling factor. We may therefore define our  $\mathbf{w}_i$  to be scaled such that  $\mathbf{w}_i^T \mathbf{B} \mathbf{w}_i = 1$ . We then have:

$$r(\mathbf{w}) = \frac{\sum_i c_i^2 \lambda_i}{\sum_i c_i^2}.$$

Each generalized Rayleigh quotient is thus a convex combination of generalized eigenvalues  $\lambda_i$ . The maximum of a convex combination of one-dimensional points is obtained in the largest of these points. If  $\lambda_1$  is thus the largest generalized eigenvalue of  $(\mathbf{A}, \mathbf{B})$ , then  $\max r(\mathbf{w}) = \lambda_1$ .

We have thus shown that  $\arg \max r(\mathbf{w}) = \mathbf{w}_1$ , where  $\mathbf{w}_1$  is an eigenvector of  $(\mathbf{A}, \mathbf{B})$ , and that its corresponding eigenvalue  $\lambda_1 = \max r(\mathbf{w})$  is the largest of the eigenvalues of  $(\mathbf{A}, \mathbf{B})$ .

□

*Appendix D*

---

## Supplemental figures

---

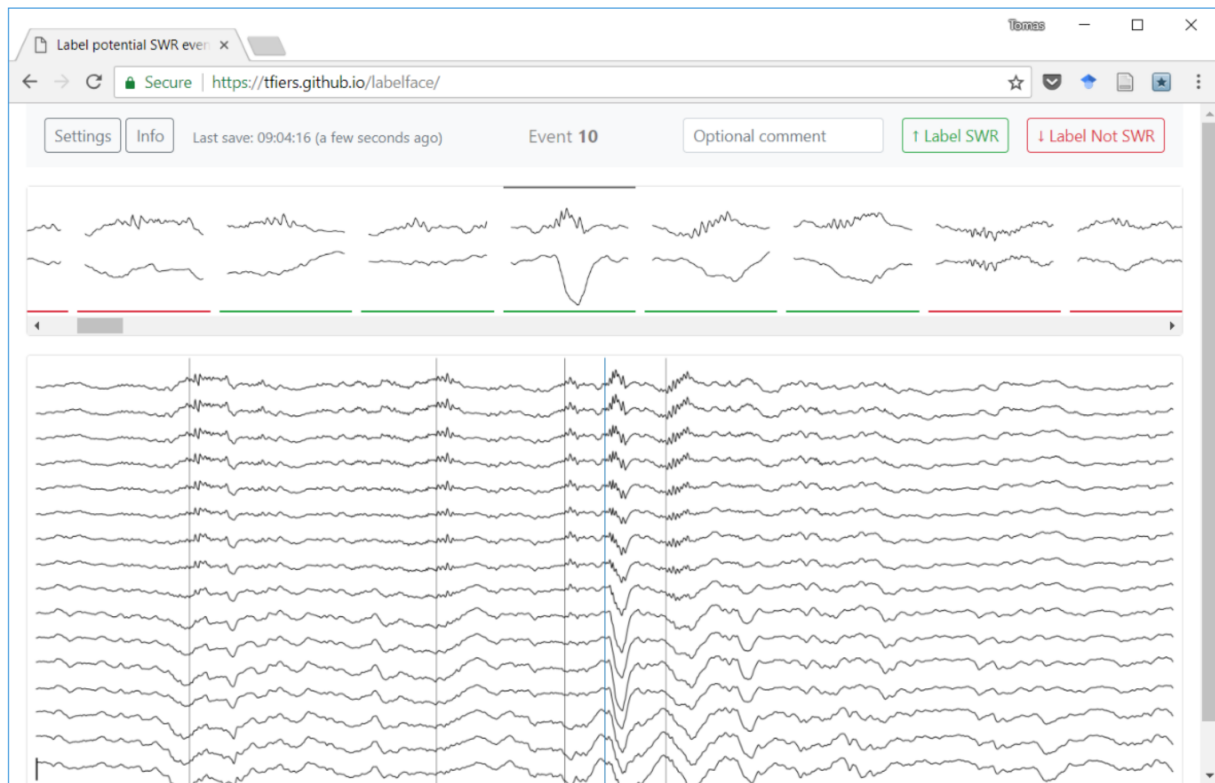
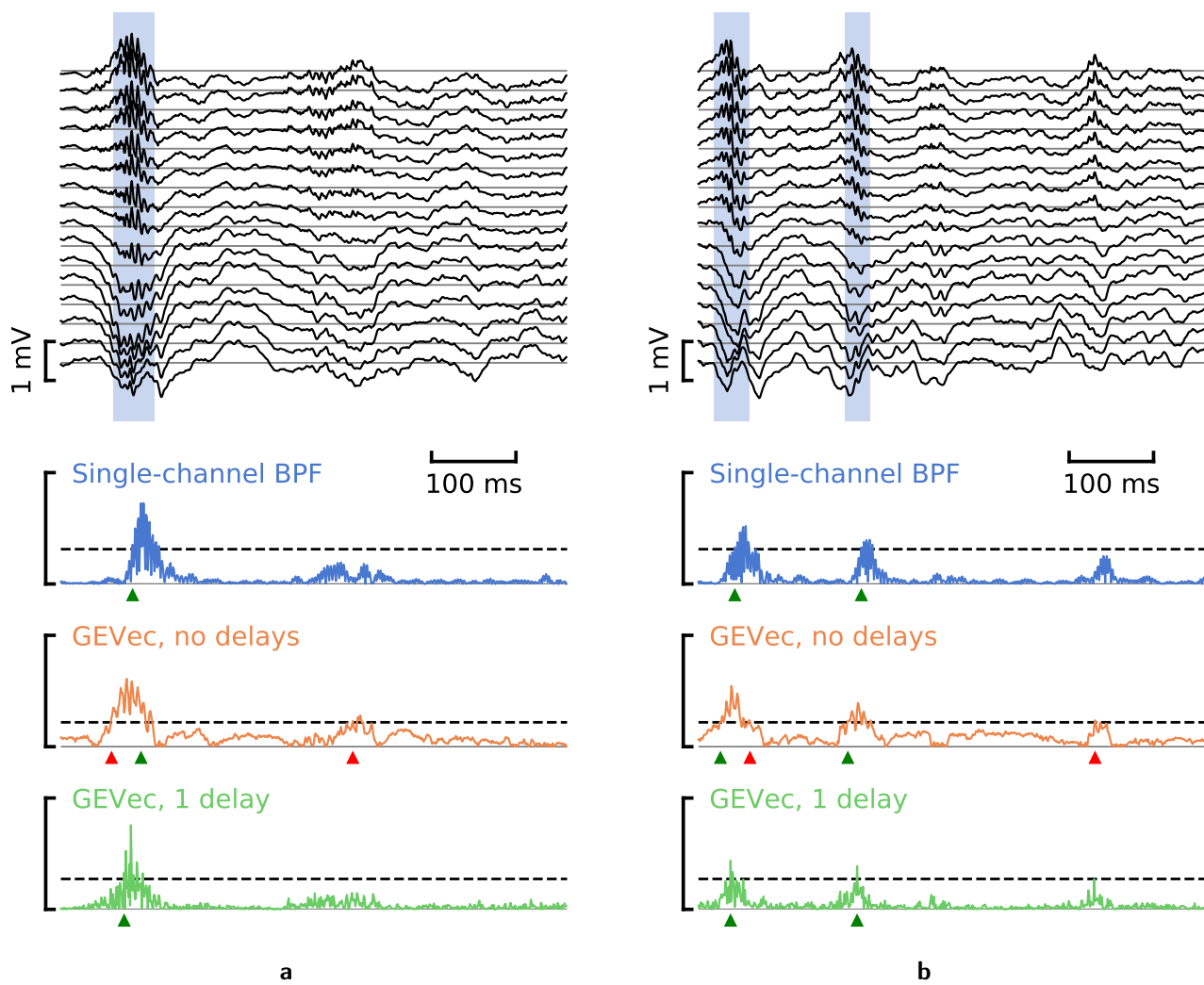


Figure D.1: **User interface for SWR labelling.** Each event in the list at the top is represented by two voltage traces: one from an electrode in the pyramidal cell layer (the top trace), and one from an electrode in the stratum radiatum. The large plot at the bottom gives more comprehensive view of the current event: all 16 channels are plotted (instead of only two), and the plot ranges from 1000 ms before to 1000 ms after the event. In this large plot, the blue vertical line marks the currently active event. The grey vertical lines correspond to other detected events. The scalebar at the beginning of this plot represents 1 mV. The source code for this labelling web app is available at <https://github.com/tfiers/labelface>.



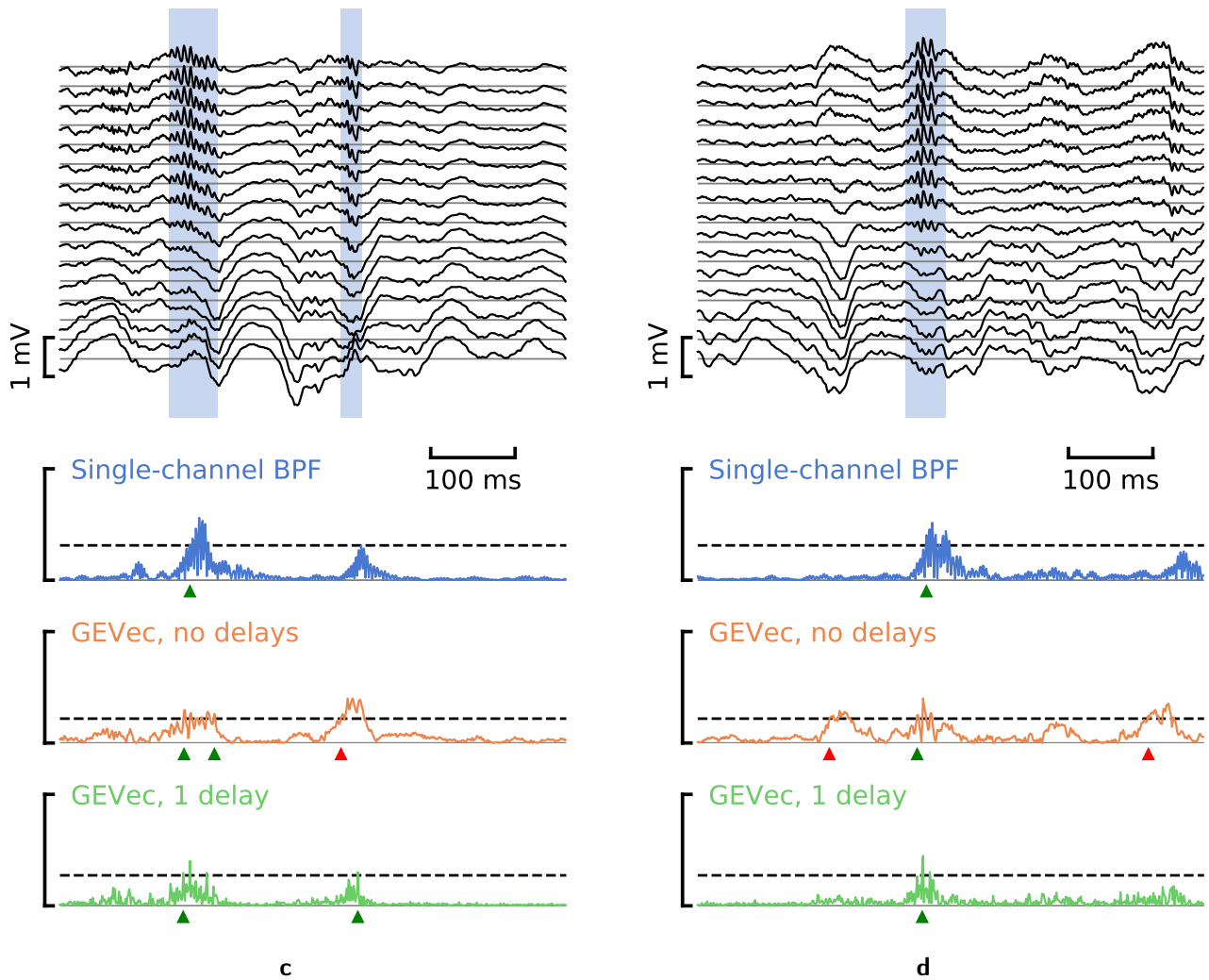


Figure D.2: Extracts from input data and corresponding linear filter output envelopes. See fig. 3.2 for legend.