



HAL
open science

CaImAn: An open source tool for scalable Calcium Imaging data Analysis

Andrea Giovannucci, Johannes Friedrich, Pat Gunn, Jérémie Kalfon, Ann Koay, Jiannis Taxidis, Farzaneh Naja, Jeffrey L Gauthier, David W Tank, Dmitri Chklovskii, et al.

► **To cite this version:**

Andrea Giovannucci, Johannes Friedrich, Pat Gunn, Jérémie Kalfon, Ann Koay, et al.. CaImAn: An open source tool for scalable Calcium Imaging data Analysis. 2018. hal-01812108

HAL Id: hal-01812108

<https://hal.science/hal-01812108v1>

Preprint submitted on 11 Jun 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

1 CalmAn: An open source tool for 2 scalable Calcium Imaging data 3 Analysis

4 **Andrea Giovannucci**^{1*}, **Johannes Friedrich**^{1,2}, **Pat Gunn**¹, **J r mie Kalfon**^{3†}, **Sue**
5 **Ann Koay**⁴, **Jiannis Taxis**⁵, **Farzaneh Najafi**⁶, **Jeffrey L. Gauthier**⁴, **Pengcheng**
6 **Zhou**², **David W. Tank**⁴, **Dmitri Chklovskii**¹, **Eftychios A. Pnevmatikakis**^{1*}

***For correspondence:**

agiovannucci@flatironinstitute.org 7
(AG); epnevmatikakis@flatironinstitute.org (EAP)

†JK contributed to this work during
an internship at the Flatiron
Institute

1 Center for Computational Biology, Flatiron Institute, New York, USA; ²Department of
8 Statistics and Center for Theoretical Neuroscience, Columbia University, New York, USA;
9 ³ECE Paris, Paris, France; ⁴Princeton Neuroscience Institute, Princeton University, New
10 Jersey, USA; ⁵Department of Neurology, UCLA, California, USA; ⁶Cold Spring Harbor
11 Laboratory, New York, USA

13 **Abstract** Advances in fluorescence microscopy enable monitoring larger brain areas *in-vivo* with
14 finer time resolution. The resulting data rates require reproducible analysis pipelines that are
15 reliable, fully automated, and scalable to datasets generated over the course of months. Here we
16 present CALMAN, an open-source library for calcium imaging data analysis. CALMAN provides
17 automatic and scalable methods to address problems common to pre-processing, including motion
18 correction, neural activity identification, and registration across different sessions of data collection.
19 It does this while requiring minimal user intervention, with good performance on computers
20 ranging from laptops to high-performance computing clusters. CALMAN is suitable for two-photon
21 and one-photon imaging, and also enables real-time analysis on streaming data. To benchmark the
22 performance of CALMAN we collected a corpus of ground truth annotations from multiple labelers
23 on nine mouse two-photon datasets. We demonstrate that CALMAN achieves near-human
24 performance in detecting locations of active neurons.

26 Introduction

27 Understanding the function of neural circuits is contingent on the ability to accurately record and
28 modulate the activity of large neural populations. Optical methods based on the fluorescence
29 activity of genetically encoded calcium binding indicators (*Chen et al., 2013*) have become a standard
30 tool for this task, due to their ability to monitor *in vivo* targeted neural populations from many
31 different brain areas over extended periods of time (weeks or months). Advances in microscopy
32 techniques facilitate imaging larger brain areas with finer time resolution, producing an ever-
33 increasing amount of data. A typical resonant scanning two-photon microscope produces data at a
34 rate greater than 50GB/Hour¹, a number that can be significantly higher (up to more than 1TB/Hour)
35 with other custom recording technologies (*Sofroniew et al. (2016)*; *Ahrens et al. (2013)*; *Flusberg*
36 *et al. (2008)*; *Cai et al. (2016)*; *Prevedel et al. (2014)*; *Grosenick et al. (2017)*; *Bouchard et al. (2015)*).

37 This increasing availability and volume of calcium imaging data calls for automated analysis
38 methods and reproducible pipelines to extract the relevant information from the recorded movies,
39 i.e., the locations of neurons in the imaged Field of View (FOV) and their activity in terms of raw

¹Calculation performed on a 512x512 FOV imaged at 30Hz producing an unsigned 16-bit integer for each measurement.

40 fluorescence and/or neural activity (spikes). The typical steps arising in the processing pipelines are
41 the following (Fig. 1a): i) Motion correction, where the FOV at each data frame (image or volume)
42 is registered against a template to correct for motion artifacts due to the finite scanning rate and
43 existing brain motion, ii) source extraction where the different active and possibly overlapping
44 sources are extracted and their signals are demixed from each other and from the background
45 neuropil signals (Fig. 1b), and iii) activity deconvolution, where the neural activity of each identified
46 source is deconvolved from the dynamics of the calcium indicator.

47 **Related work**

48 **Source extraction**

49 Some source extraction methods attempt the detection of neurons in static images using supervised
50 or unsupervised learning methods. Examples of unsupervised methods on summary images include
51 graph-cut approaches applied to the correlation image (*Kaifosh et al., 2014; Spaen et al., 2017*),
52 and dictionary learning (*Pachitariu et al., 2013*). Supervised learning methods based on deep
53 neural networks have also been applied to the problem of neuron detection (*Apthorpe et al., 2016;*
54 *Klibisz et al., 2017*). While these methods can be efficient in detecting the locations of neurons, they
55 cannot infer the underlying activity nor do they readily offer ways to deal with the spatial overlap of
56 different components.

57 To extract temporal traces together with the spatial footprints of the components one can use
58 methods that directly represent the full spatio-temporal data in a matrix factorization setup e.g.,
59 independent component analysis (ICA) (*Mukamel et al., 2009*), constrained nonnegative matrix
60 factorization (CNMF) (*Pnevmatikakis et al., 2016*) (and its adaptation to one-photon data (*Zhou*
61 *et al., 2018*)), clustering based approaches (*Pachitariu et al., 2017*), dictionary learning (*Petersen*
62 *et al., 2017*), or active contour models (*Reynolds et al., 2017*). Such spatio-temporal methods are
63 unsupervised, and focus on detecting active neurons by considering the spatio-temporal activity of
64 a component as a contiguous set of pixels within the FOV that are correlated in time. While such
65 methods tend to offer a direct decomposition of the data in a set of sources with activity traces
66 in an unsupervised way, in principle they require processing of the full dataset, and thus can be
67 rendered intractable very quickly. Possible approaches to deal with the data size include distributed
68 processing in High Performance Computing (HPC) clusters (*Freeman et al., 2014*), spatio-temporal
69 decimation (*Friedrich et al., 2017a*), and dimensionality reduction (*Pachitariu et al., 2017*). Recently,
70 *Giovannucci et al. (2017)* prototyped an online algorithm (ONACID), by adapting matrix factorization
71 setups (*Pnevmatikakis et al., 2016; Mairal et al., 2010*), to operate on calcium imaging streaming
72 data and thus natively deal with large data rates.

73 **Deconvolution**

74 For the problem of predicting spikes from fluorescence traces, both supervised and unsupervised
75 methods have been explored. Supervised methods rely on the use of ground truth data to train
76 or fit biophysical or neural network models (*Theis et al., 2016; Speiser et al., 2017*). Unsupervised
77 methods can be either deterministic, such as sparse non-negative deconvolution (*Vogelstein*
78 *et al., 2010; Pnevmatikakis et al., 2016*) that give a single estimate of the deconvolved neural
79 activity, or probabilistic, that aim to also characterize the uncertainty around these estimates
80 (e.g., (*Pnevmatikakis et al., 2013; Deneux et al., 2016*)). A recent community benchmarking effort
81 (*Berens et al., 2017*) characterizes the similarities and differences of various available methods.

82 **CAIMAN**

83 Here we present CAIMAN, an open source suite for the analysis pipeline of both two-photon and one-
84 photon calcium imaging data. CAIMAN includes frameworks for both offline analysis (CAIMAN BATCH)
85 where all the data is processed at once at the end of experiment, and online analysis on streaming
86 data (CAIMAN ONLINE). Moreover, CAIMAN requires very moderate computing infrastructure (e.g., a

87 personal laptop or workstation), thus providing automated, efficient, and reproducible large-scale
88 analysis on commodity hardware.

89 Contributions

90 Our contributions can be roughly grouped in three different directions:

91 **Methods:** CALMAN BATCH improves on the scalability of the source extraction problem by employing
92 a MapReduce framework for parallel processing and memory mapping which allows the
93 analysis of datasets larger than would fit in RAM on most computer systems. It also improves
94 on the qualitative performance by introducing automated routines for component evaluation
95 and classification, better handling of neuropil contamination, and better initialization methods.
96 While these benefits are here presented in the context of the widely used CNMF algorithm
97 of *Pnevmatikakis et al. (2016)*, they are in principle applicable to any matrix factorization
98 approach.

99 CALMAN ONLINE improves and extends the ONACID prototype algorithm (*Giovannucci et al.,*
100 *2017*) by introducing, among other advances, new initialization methods and a convolutional
101 neural network (CNN) based approach for detecting new neurons on streaming data. Our
102 analysis on *in vivo* two-photon and light-sheet imaging datasets shows that CALMAN ONLINE
103 approaches human-level performance and enables novel types of closed-loop experiments.
104 Apart from these significant algorithmic improvements CALMAN includes several useful analysis
105 tools such as, a MapReduce and memory-mapping compatible implementation of the
106 CNMF-E algorithm for one-photon microendoscopic data (*Zhou et al., 2018*), a novel efficient
107 algorithm for registration of components across multiple days, and routines for segmentation
108 of structural (static) channel information which can be used for component seeding.

109 **Software:** CALMAN comes as a complete open source software suite implemented in Python, and
110 is already widely used by, and has received contributions from, the community. It contains
111 efficient implementations of the standard analysis pipeline steps (motion correction - source
112 extraction - deconvolution - registration across different sessions), as well as numerous other
113 features. Apart from Python, several of the tools presented here are also available in MATLAB®.

114 **Data:** We benchmark the performance of CALMAN against a previously unreleased corpus of manu-
115 ally annotated data. The corpus consists of 9 mouse *in vivo* two-photon datasets manually
116 annotated by 3-4 independent labelers that were instructed to select active neurons in a
117 principled and consistent way, and who subsequently combined their annotations to create
118 a “consensus” ground truth that is also used to quantify the limits of human performance.
119 The manual annotations are also released to the community providing a valuable tool for
120 benchmarking and training purposes.

121 Paper organization

122 The paper is organized as follows: We first give a brief presentation of the analysis methods and
123 features provided by CALMAN. In the *Results* section we benchmark CALMAN ONLINE and CALMAN
124 BATCH against a corpus of manually annotated data. We apply CALMAN ONLINE to a zebrafish whole
125 brain lightsheet imaging recording, and demonstrate how such large datasets can be processed
126 efficiently in real time. We also present applications of CALMAN BATCH to one-photon data, as well as
127 examples of component registration across multiple days. We conclude by discussing the utility of
128 our tools, the relationship between CALMAN BATCH and CALMAN ONLINE and outline future directions.
129 Detailed descriptions of the introduced methods are presented in *Methods and Materials*.

130 Methods

131 Before presenting the new analysis features introduced with this work, we overview the analysis
132 pipeline that CALMAN uses and builds upon.

133 **Overview of analysis pipeline**

134 The standard analysis pipeline for calcium imaging data used in CALMAN is depicted in Fig. 1a.
135 The data in movie format is first processed to remove motion artifacts. Subsequently the active
136 components (neurons and background) are extracted as individual pairs of a spatial footprint that
137 describes the shape of each component projected to the imaged FOV, and a temporal trace that
138 captures its fluorescence activity (Fig. 1b-d). Finally, the neural activity of each fluorescence trace
139 is deconvolved from the dynamics of the calcium indicator. These operations can be challenging
140 because of limited axial resolution of 2-photon microscopy (or the much larger integration volume
141 in one-photon imaging). This results in spatially overlapping fluorescence from different sources
142 and neuropil activity. Before presenting the new features of CALMAN in more detail, we briefly review
143 how it incorporates existing tools in the pipeline.

144 **Motion Correction**

145 CALMAN uses the NORMCORRE algorithm (*Pnevmatikakis and Giovannucci, 2017*) that corrects non-
146 rigid motion artifacts by estimating motion vectors with subpixel resolution over a set of overlapping
147 patches within the FOV. These estimates are used to infer a smooth motion field within the FOV
148 for each frame. For two-photon imaging data this approach is directly applicable, whereas for
149 one-photon micro-endoscopic data the motion is estimated on high pass spatially filtered data,
150 a necessary operation to remove the smooth background signal and create enhanced spatial
151 landmarks. The inferred motion fields are then applied to the original data frames.

152 **Source Extraction**

153 Source extraction is performed using the constrained non-negative matrix factorization (CNMF)
154 framework of *Pnevmatikakis et al. (2016)* which can extract components with spatial overlapping
155 projections (Fig. 1b). After motion correction the spatio-temporal activity of each source can be
156 expressed as a rank one matrix given by the outer product of two components: a component in
157 space that describes the spatial footprint (location and shape) of each source, and a component
158 in time that describes the activity trace of the source (Fig. 1c). The data can be described by the
159 sum of all the resulting rank one matrices together with an appropriate term for the background
160 and neuropil signal and a noise term (Fig. 1d). For two-photon data the neuropil signal can be
161 modeled as a low rank matrix (*Pnevmatikakis et al., 2016*). For microendoscopic data the larger
162 integration volume leads to more complex background contamination (*Zhou et al., 2018*). Therefore,
163 a more descriptive model is required (see *Methods and Materials (Mathematical model of the CNMF*
164 *framework)* for a mathematical description). CALMAN BATCH embeds these approaches into a general
165 algorithmic framework that enables scalable automated processing with improved results in terms
166 of quality and processing speed.

167 **Deconvolution**

168 Neural activity deconvolution is performed using sparse non-negative deconvolution (*Vogelstein*
169 *et al., 2010; Pnevmatikakis et al., 2016*) and implemented with both the near-online OASIS algo-
170 rithm (*Friedrich et al., 2017b*) and an efficient convex optimization framework (*Pnevmatikakis et al.,*
171 *2016*). The algorithm is competitive to the state of the art according to recent benchmarking studies
172 (*Berens et al., 2017*). Prior to deconvolution, the traces are detrended to remove non-stationary
173 effects, e.g., photo-bleaching.

174 **Online Processing**

175 The three processing steps described above can be implemented in an online fashion on streaming
176 data using the ONACID algorithm (*Giovannucci et al., 2017*). The method builds upon the online
177 dictionary learning framework presented in *Mairal et al. (2010)* for source extraction, by adding the
178 capability of finding new components as they appear and also incorporating the steps of motion
179 correction and deconvolution (Fig. 1e). CALMAN ONLINE extends and improves the ONACID prototype

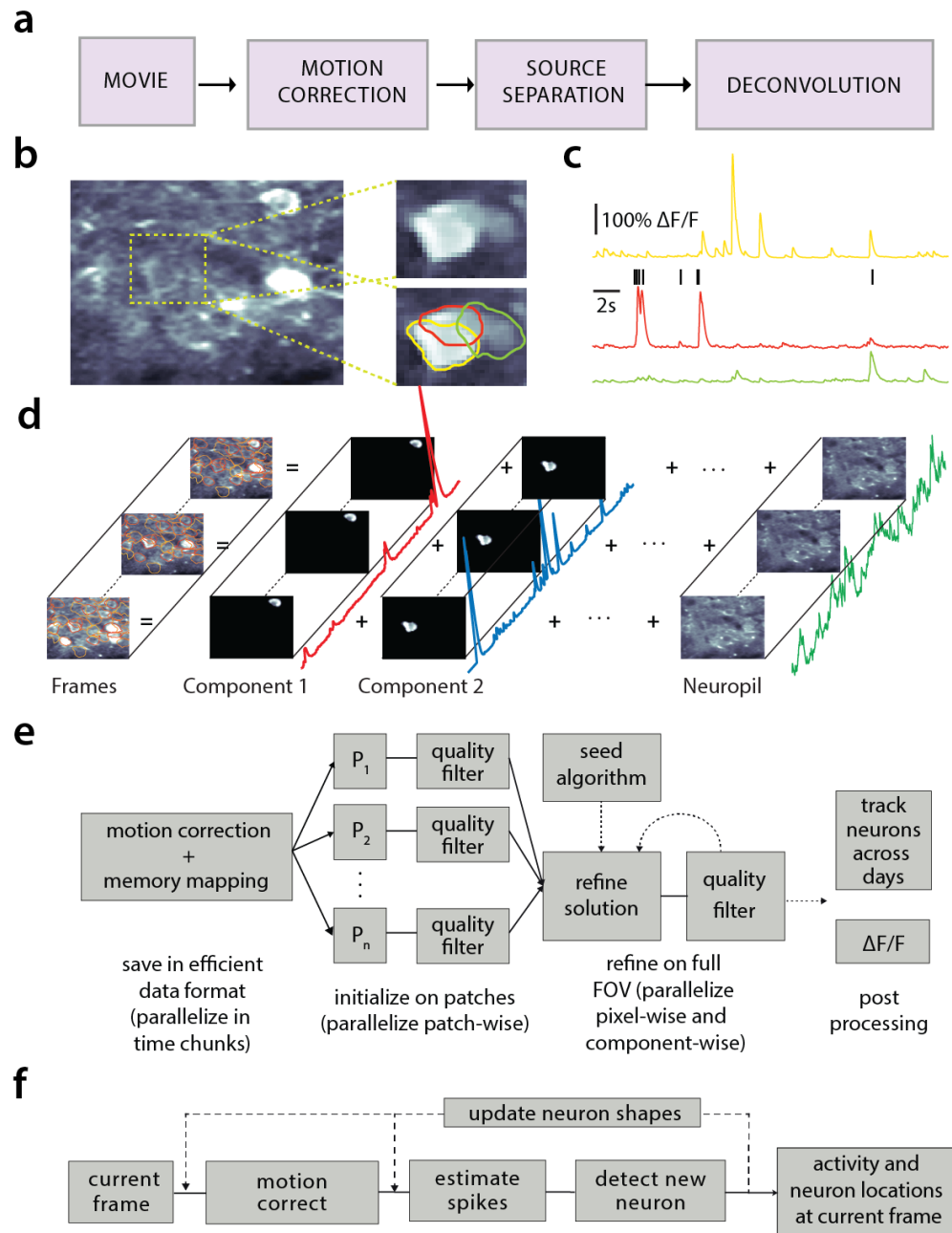


Figure 1. Processing pipeline of CALMAN for calcium imaging data. (a) The typical pre-processing steps include (i) correction for motion artifacts, (ii) extraction of the spatial footprints and fluorescence traces of the imaged components, and (iii) deconvolution of the neural activity from the fluorescence traces. (b) Time average of 2000 frames from a two-photon microscopy dataset (left) and magnified illustration of three overlapping neurons (right), as detected by the CNMF algorithm. (c) Denoised temporal components of the three neurons in (b) as extracted by CNMF and matched by color (in relative fluorescence change, $\Delta F/F$). (d) Intuitive depiction of CNMF. The algorithm represents the movie as the sum of rank-one spatio-temporal components capturing either neurons and processes, plus additional non-sparse low-rank terms for the background fluorescence and neuropil activity. (e) Flow-chart of the CALMAN BATCH processing pipeline. From left to right: Motion correction and generation of a memory efficient data format. Initial estimate of somatic locations in parallel over FOV patches using CNMF. Refinement and merging of extracted components via seeded CNMF. Removal of low quality components. Final domain dependent processing stages. (f) Flow-chart of the CALMAN ONLINE algorithm. After a brief mini-batch initialization phase, each frame is processed in a streaming fashion as it becomes available. From left to right: Correction for motion artifacts. Estimate of activity from existing neurons, identification and incorporation of new neurons. Periodically, the spatial footprints of inferred neurons are updated (dashed lines).

180 algorithm by introducing a number of algorithmic features and a CNN based component detection
181 approach, leading to a major performance improvement.

182 We now present the new methods introduced by CALMAN. More details are given in *Methods and*
183 *Materials* and pseudocode descriptions of the main routines are given in the *Appendix*.

184 **Batch processing of large scale datasets on standalone machines**

185 The batch processing pipeline mentioned above can become a computational bottleneck when
186 tackled without customized solutions. For instance, a naive approach to the problem might have as
187 a first step to load in-memory the full dataset; this approach is non-scalable as datasets typically
188 exceed available RAM (and extra memory is required by any analysis pipeline). To limit memory
189 usage, as well as computation time, CALMAN BATCH relies on a MapReduce approach (*Dean and*
190 *Ghemawat, 2008*). Unlike previous work (*Freeman et al., 2014*), CALMAN BATCH assumes minimal
191 computational infrastructure (up to a standard laptop computer), is not tied to a particular parallel
192 computation framework, and is compatible with HPC scheduling systems like SLURM (*Yoo et al.,*
193 *2003*).

194 Naive implementations of motion correction algorithms need to either load in memory the full
195 dataset or are constrained to process one frame at a time, therefore preventing parallelization. Mo-
196 tion correction is parallelized in CALMAN BATCH without significant memory overhead by processing
197 several temporal chunks of a video data on different CPUs. CALMAN BATCH broadcasts to each CPU a
198 meta-template, which is used to align all the frames in the chunk. Each process writes in parallel to
199 the target file containing motion-corrected data, which is stored in as a memory mapped array. This
200 allows arithmetic operations to be performed against data stored on the hard drive with minimal
201 memory use, and slices of data to be indexed and accessed without loading the full file in memory.
202 More details are given in *Methods and Materials (Memory mapping)*.

203 Similarly, the source extraction problem, especially in the case of detecting cell bodies, is
204 inherently local with a neuron typically appearing in a neighborhood within a small radius from its
205 center of mass (Fig. 2a). Exploiting this locality, CALMAN BATCH splits the FOV into a set of spatially
206 overlapping patches which enables the parallelization of the CNMF (or any other) algorithm to
207 extract the corresponding set of local spatial and temporal components. The user specifies the size
208 of the patch, the amount of overlap between neighboring patches and the initialization parameters
209 for each patch (number of components and rank background for CNMF, stopping criteria for CNMF-
210 E). Subsequently the patches are processed in parallel by the CNMF/CNMF-E algorithm to extract
211 the components and neuropil signals from each patch.

212 Apart from harnessing memory and computational benefits due to parallelization, processing in
213 patches acts indirectly as a dynamic range equalizer and enables CALMAN BATCH to detect neurons
214 across the whole FOV, a feature absent in the original CNMF, where areas with high absolute
215 fluorescence variation tend to be favored. This results in better source extraction performance.
216 After all the patches have been processed, the results are embedded within the FOV (Fig. 2a),
217 and the overlapping regions between neighboring patches are processed so that components
218 corresponding to the same neuron are merged. The process is summarized in algorithmic format in
219 Alg. 1 and more details are given in *Methods and Materials (Combining results from different patches)*.

220 **Initialization Methods**

221 Initialization methods for matrix factorization problems can impact results due to the non-convex
222 nature of their objective function. CALMAN BATCH provides an extension of the GREEDYROI method
223 used in *Pnevmatikakis et al. (2016)*, that detects neurons based on localized spatiotemporal activity.
224 CALMAN BATCH can also be seeded with binary masks that are obtained from different sources, e.g.,
225 through manual annotation or segmentation of structural channel (SEEDEDINITIALIZATION, Alg. 2).
226 More details are given in *Methods and Materials (Initialization strategies)*.

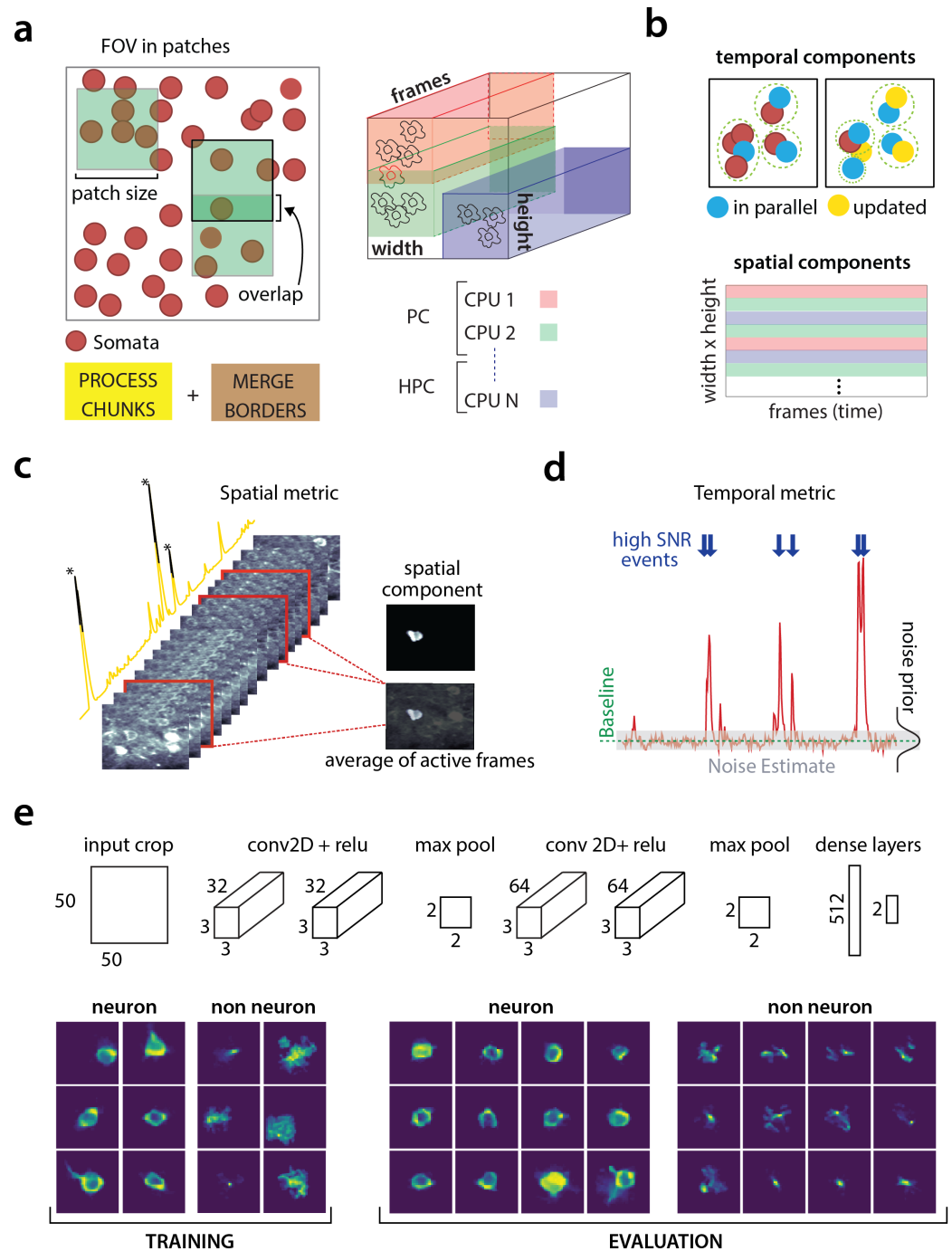


Figure 2. Parallelized processing and component quality assessment for CALMAN BATCH. (a) Illustration of the parallelization approach used by CALMAN BATCH for source extraction. The data movie is partitioned into overlapping sub-tensors, each of which is processed in an embarrassingly parallel fashion using CNMF, either on local cores or across several machines in a HPC. The results are then combined. (b) Refinement after combining the results can also be parallelized both in space and in time. Temporal traces of spatially non-overlapping components can be updated in parallel (top) and the contribution of the spatial footprints for each pixel can be computed in parallel (bottom). Parallelization in combination with memory mapping enable large scale processing with moderate computing infrastructure. (c) Quality assessment in space: The spatial footprint of each real component is correlated with the data averaged over time, after removal of all other activity. (d) Quality assessment in time: A high SNR is typically maintained over the course of a calcium transient. (e) CNN based assessment. *Top*: A 4-layer CNN based classifier is used to classify the spatial footprint of each component into neurons or not. *Bottom*: Positive and negative examples for the CNN classifier, during training (left) and evaluation (right) phase. The CNN classifier can accurately classify shapes and generalizes across datasets from different brain areas.

227 **Automated component evaluation and classification**

228 A common limitation of matrix factorization algorithms is that the number of components that
229 the algorithm seeks during its initialization must be pre-determined by the user. For example,
230 *Pnevmatikakis et al. (2016)* suggest a large number of components which are then heuristically
231 ordered according to their size and activity pattern. When processing large datasets in patches
232 the target number of components is passed on to every patch implicitly assuming a uniform
233 density of (active) neurons within the entire FOV. In general this assumption does not hold and can
234 generate a large number of spurious components. CALMAN introduces tests to assess the quality
235 of the detected components and eliminate possible false positives. These tests are based on the
236 observation that active components are bound to have a distinct localized spatio-temporal signature
237 within the FOV. We present below unsupervised and supervised tests employed by CALMAN for
238 component classification. In CALMAN BATCH, they are initially applied after the processing of each
239 patch is completed, and additionally as a post-processing step after the results from the patches
240 have been merged and refined, whereas in CALMAN ONLINE they are used to screen new candidate
241 components. We briefly present these tests below and refer to *Methods and Materials (Details of*
242 *quality assessment tests)* for more details:

243 **Spatial footprint consistency:** To test whether a detected component is spurious, we correlate
244 the spatial footprint of this component with the average frame of the data, taken over the
245 interval when the component, with no other overlapping component, was active (Fig. 2c).
246 The component is rejected if the correlation coefficient is below a certain threshold θ_{sp} (e.g.,
247 $\theta_{sp} < 0.5$).

248 **Trace SNR:** Similarly, for each component we computed the peak SNR of its temporal trace averaged
249 over the duration of a typical transient (Fig. 2d). The component is rejected if the
250 computed SNR is below a certain threshold θ_{SNR} (e.g., $\theta_{SNR} = 2$).

251 **CNN based classification:** We also trained a 4-layer convolutional neural network (CNN) to classify
252 spatial footprints into true or false components (Fig. 2e), where a true component here
253 corresponds to a spatial footprint that resembles the soma of a neuron. The classifier, named
254 batch classifier, was trained on a small corpus of manually annotated datasets (full description
255 given in section *Benchmarking against ground truth*) and exhibited similar high classification
256 performance on test samples from different datasets.

257 While CALMAN uses the CNMF algorithm, the tests described above can be applied to results obtained
258 from any source extraction algorithm, highlighting the modularity of our tools.

259 **Online analysis with CALMAN ONLINE**

260 CALMAN supports online analysis on streaming data building on the core of the prototype algorithm
261 of *Giovannucci et al. (2017)*, and extending it in terms of qualitative performance and computational
262 efficiency:

263 **Initialization:** Apart from initializing CALMAN ONLINE with CALMAN BATCH on a small time interval,
264 CALMAN ONLINE can also be initialized in a bare form over an even smaller time interval, where
265 only the background components are estimated and all the components are determined during
266 the online analysis. This process, named BAREINITIALIZATION, can be achieved by running
267 the CNMF algorithm (*Pnevmatikakis et al., 2016*) over the small interval to estimate the back-
268 ground components and possibly a small number of components. The SEEDEDINITIALIZATION
269 of Alg. 2 can also be used.

270 **Deconvolution:** Instead of a separate step after demixing as in *Giovannucci et al. (2017)*, decon-
271 volution here can be performed simultaneously with the demixing online, leading to more
272 stable traces especially in cases of low-SNR, as also observed in *Pnevmatikakis et al. (2016)*.
273 Online deconvolution can also be performed for models that assume second order calcium
274 dynamics, bringing the full power of *Friedrich et al. (2017b)* to processing of streaming data.

275 **Epochs:** CALMAN ONLINE supports multiple passes over the data, a process that can detect early
276 activity of neurons that were not picked up during the initial pass, as well as smooth the
277 activity of components that were detected at late stages during the first epoch.

278 **New component detection using a CNN:** To search for new components in a streaming setup,
279 ONACID keeps a buffer of the residual frames, computed by subtracting the activity of already
280 found components and background signals. Candidate components are determined by
281 looking for points of maximum energy in this residual signal, after some smoothing and
282 dynamic range equalization. For each such point identified, a candidate shape and trace are
283 constructed using a rank-1 NMF in a local neighborhood around this point. In its original
284 formulation (*Giovannucci et al., 2017*), the shape of the component was evaluated using the
285 space correlation test described above. Here, we introduce a CNN classifier approach that
286 tests candidate components by examining their spatial footprint as obtained by the average
287 of the residual buffer across time. This online classifier (different from the batch classifier
288 for quality assessment described above), is trained to be strict, minimizing the number of
289 false positive components that enter the online processing pipeline. It can test multiple
290 components in parallel, and it achieves better performance with no hyper-parameter tuning
291 compared to the previous approach. More details on the architecture and training procedure
292 are given in *Methods and Materials (Classification through CNNs)*. The identification of candidate
293 components is further improved by performing spatial high pass filtering on the average
294 residual buffer to enhance its contrast. The new process for detecting neurons is described in
295 Algs. 3 and 4. See Supplemental Movies 1 and 2 on a detailed graphic description of the new
296 component detection step.

297 **Component registration across multiple sessions**

298 CALMAN provides a method to register components from the same FOV across different sessions.
299 The method uses a simple intersection over union metric to calculate the distance between different
300 cells in different sessions and solving a linear assignment problem to perform the registration in
301 a fully automated way (REGISTERPAIR, Alg. 5). To register the components between more than 2
302 sessions (REGISTERMULTI, Alg. 6), we order the sessions chronologically and register the components
303 of the current session against the union of component of all the past sessions aligned to the current
304 FOV. This allows for the tracking of components across multiple sessions without the need of
305 pairwise registration between each pair of sessions. More details as well as discussion of other
306 methods (*Sheintuch et al., 2017*) are given in *Methods and Materials (Component registration)*.

307 **Benchmarking against ground truth**

308 To quantitatively evaluate CALMAN we benchmarked its results against ground truth data.

309 **Creating ground truth data through manual annotation**

310 We collected manual annotations from multiple independent labelers who were instructed to find
311 round or donut shaped² *active* neurons on 9 two-photon *in vivo* mouse brain datasets. The datasets
312 were collected at various labs and from various brain areas (hippocampus, visual cortex, parietal
313 cortex) using several GCaMP variants. A summary of the features of all the annotated datasets is
314 given in Table 2. Details about the annotation procedure are given in *Methods and Materials*.

315 To address human variability in manual annotation each dataset was labeled by 3 or 4 inde-
316 pendent labelers, and the final ground truth dataset was created by having the different labelers
317 reaching a *consensus* over their disagreements (Fig. 3a). The result of this process was defined as
318 ground truth for the evaluation of CALMAN as well as each individual labeler against the consensus
319 (Fig. 3b)³. More details are given in *Methods and Materials (Collection of manual annotations and*

²Since proteins expressing the calcium indicator are confined outside the cell nuclei, neurons will appear as ring shapes, with a dark disk in the center.

³It is possible that this process generated slightly biased results in favor of each individual annotators since the ground truth was always a subset of the union of the individual annotations.

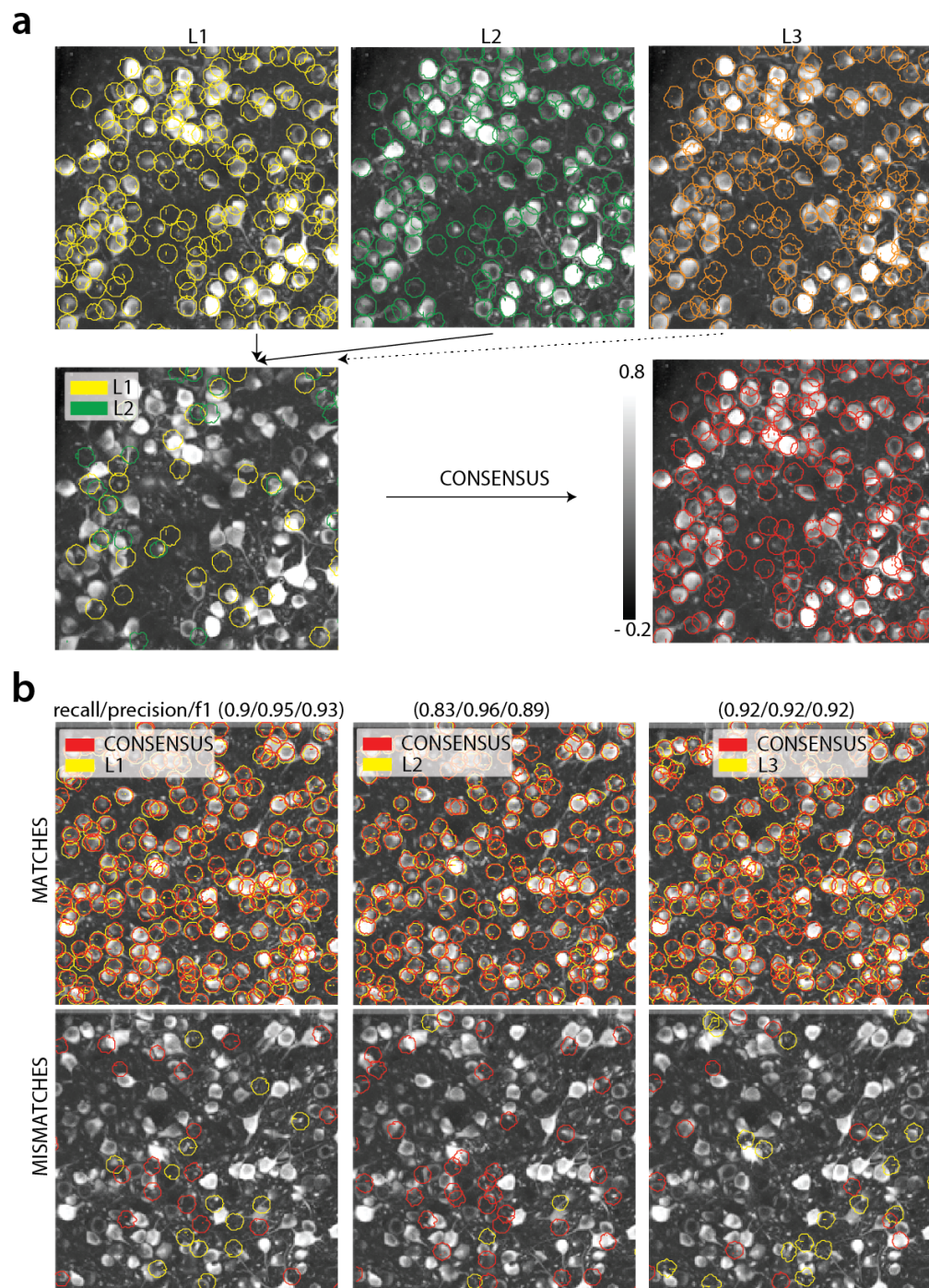


Figure 3. Ground truth generation. (a) *Top*: Individual manual annotations on the dataset N.04.00.t (only part of the FOV is shown) for labelers L1 (left), L2 (middle), L3 (right). *Bottom*: Disagreements between L1 and L2 (left), and ground truth labels (right) after the consensus between all labelers has been reached. In this example, consensus considerably reduced the number of initially selected neurons. (b) Matches (top) and mismatches (bottom) between each individual labeler and consensus ground truth. Red contours on the mismatches panels denote false negative contours, i.e., components in the consensus not selected by the corresponding labeler, whereas yellow contours indicate false positive contours. Performance of each labeler is given in terms of precision/recall and F_1 score and indicates an unexpected level of variability between individual labelers.

320 *ground truth*). We believe that the current database, which will be made publicly available, presents
321 an improvement over the existing neurofinder database (<http://neurofinder.codeneuro.org/>) in
322 several aspects:

323 **Consistency:** The datasets are annotated using exactly the same procedure (see *Methods and*
324 *Materials*), and in all datasets the goal is to detect only active cells. In contrast, the annotation of
325 the various neurofinder datasets is performed either manually or automatically by segmenting
326 an image of a static (structural) indicator. Even though structural indicators could be used for
327 ground truth extraction, the segmentation of such images is not a straightforward problem
328 in the case of dense expression, and the stochastic expression of indicators can lead to
329 mismatches between functional and structural indicators.

330 **Uncertainty quantification:** By employing more than one human labeler we discovered a sur-
331 prising level of disagreement between different annotators (see Table 1, Fig. 3b for details),
332 which renders individual annotations somewhat unreliable for benchmarking purposes, and
333 non-reproducible. The combination of the various annotations leads to more reliable ground
334 truth and also quantifies the limits of human performance.

335 Comparing CALMAN against ground truth

336 To compare CALMAN against the consensus ground truth, the manual annotations were used as
337 binary masks to construct the ground truth spatial and temporal components, using the SEEDINI-
338 TIALIZATION procedure (Alg. 2) of CALMAN BATCH. The set of spatial footprints obtained from CALMAN
339 is registered against the set of ground truth spatial footprints (derived as described above) using
340 the REGISTERPAIR algorithm (Alg. 5) for component registration described above. Performance is
341 then quantified using a precision/recall framework similar to other studies (*Apthorpe et al., 2016*;
342 *Giovannucci et al., 2017*).

343 Software

344 CALMAN is developed by and for the community. Python open source code for all the methods
345 described above is available at <https://github.com/flatironinstitute/CalmAn>. The repository contains
346 documentation, numerous demos, and Jupyter notebook tutorials, as well as visualization tools, and
347 a message/discussion board. The code, which is compatible with Python 2 and Python 3⁴, uses tools
348 from several open source libraries, such as OpenCV (*Bradski, 2000*), scikit-learn (*Pedregosa et al.,*
349 *2011*), and scikit-image (*Van der Walt et al., 2014*). Most routines are also available in MATLAB[®] at
350 <https://github.com/flatironinstitute/CalmAn-MATLAB>.

351 Results

352 Manual annotations show a high degree of variability

353 We compared the performance of each human annotator against a consensus ground truth. The
354 performance was quantified with a precision/recall framework and the results of the performance
355 of each individual labeler against the consensus ground truth for each dataset is given in Table 1.
356 The range of human performance in terms of F_1 score was 0.69-0.94, with average 0.83 ± 0.07 (mean
357 \pm STD). All annotators performed similarly on average (0.83 ± 0.05 , 0.83 ± 0.08 , 0.84 ± 0.06 , 0.85 ± 0.08).
358 We also ensured that the performance of labelers was stable across time (i.e. their learning curve
359 plateaued, data not shown). As shown in Table 1 (see also Fig 4b) the F_1 score was never 1, and in
360 most cases it was less or equal to 0.9, demonstrating significant variability between annotators.
361 Fig. 3 (bottom) shows an example of matches and mismatches between individual labelers and
362 consensus ground truth for dataset K53, where the level of agreement was relatively high. The high
363 degree of variability in human responses indicates the challenging nature of the source extraction
364 problem and raises reproducibility concerns in studies relying heavily on manual ROI selection.

⁴All future development of CALMAN will be in Python 3, eventually rendering it incompatible with Python 2.x.

Table 1. Results of each labeler, CALMAN BATCH and CALMAN ONLINE algorithms against consensus ground truth. Results are given in the form F_1 score (precision, recall), and empty entries correspond to datasets not manually annotated by the specific labeler. In *italics* the datasets used to train the CNN classifiers.

Name	L1	L2	L3	L4	CALMAN BATCH	CALMAN ONLINE
<i>N.03.00.t</i>	X	0.90 (0.88,0.92)	0.85 (0.78,0.93)	0.78 (0.73,0.83)	0.78 (0.77,0.79)	0.76 (0.77,0.75)
<i>N.04.00.t</i>	X	0.69 (0.54,0.97)	0.75 (0.61,0.97)	0.87 (0.78,0.98)	0.67 (0.62,0.72)	0.68 (0.65,0.71)
<i>N.02.00</i>	0.89 (0.86,0.93)	0.87 (0.88,0.85)	0.84 (0.92,0.77)	0.82 (1.00,0.70)	0.79 (0.8,0.77)	0.77 (0.79,0.76)
<i>N.00.00</i>	X	0.92 (0.93,0.91)	0.83 (0.86,0.80)	0.87 (0.96,0.80)	0.72 (0.83,0.64)	0.72 (0.83,0.64)
<i>N.01.01</i>	0.80 (0.95,0.69)	0.89 (0.96,0.83)	0.78 (0.73,0.84)	0.75 (0.80,0.70)	0.77 (0.88,0.69)	0.73 (0.78,0.68)
YST	0.78 (0.76,0.81)	0.90 (0.85,0.97)	0.82 (0.75,0.92)	0.79 (0.96,0.67)	0.76 (0.9,0.66)	0.78 (0.76,0.81)
K53	0.89 (0.96,0.83)	0.92 (0.92,0.92)	0.93 (0.95,0.91)	0.83 (1.00,0.72)	0.77 (0.83,0.72)	0.82 (0.80,0.83)
J115	X	0.93 (0.94,0.91)	0.94 (0.95,0.93)	0.83 (1.00,0.71)	0.77 (0.9,0.68)	0.81 (0.75,0.88)
J123	0.85 (0.96,0.76)	0.83 (0.73,0.96)	0.90 (0.91,0.90)	0.91 (0.92,0.89)	0.68 (0.94,0.51)	0.80 (0.82,0.79)

365 CALMAN BATCH and CALMAN ONLINE detect neurons with near-human accuracy

366 We first benchmarked CALMAN BATCH and CALMAN ONLINE against consensus ground truth for the
 367 task of identifying neurons locations and their spatial footprints, using the same precision recall
 368 framework (Table 1). Fig. 4a shows an example dataset (K53) along with neuron-wise matches
 369 and mismatches between CALMAN BATCH and consensus ground truth (top) and CALMAN ONLINE vs
 370 consensus ground truth (bottom).

371 The results indicate a similar performance between CALMAN BATCH and CALMAN ONLINE; CALMAN
 372 BATCH has F_1 scores in the range 0.68-0.79 and average performance 0.75 ± 0.04 (mean \pm STD). On the
 373 other hand CALMAN ONLINE had F_1 scores in the range 0.68-0.82 and average performance 0.76 ± 0.04 .
 374 While the two algorithms performed similarly on average, CALMAN BATCH tends to perform better for
 375 shorter datasets whereas online processing tends to lead to better results for longer datasets (see
 376 Table 2 for characteristics of the various datasets). CALMAN approaches but is in most cases below
 377 the accuracy levels of human annotators (Fig. 4b). This can be attributed to a number of reasons:
 378 First, to demonstrate the generality and ease of use of our tools, the results presented here are
 379 obtained by running CALMAN BATCH and CALMAN ONLINE with *exactly* the same parameters for each
 380 dataset (see *Methods and Materials (Implementation details)*): fine-tuning to each individual dataset
 381 can significantly increase performance. Second, CNMF detects active components regardless of
 382 their shape, and can detect non-somatic structures with significant transients. While non-somatic
 383 components can be filtered out to some extent using the CNN classifier, their existence degrades
 384 performance compared to the ground truth that consists only of neurons. Lastly, the ground truth
 385 is by construction a subset of the union of all individual annotations, which can bias upwards the
 386 scores of individual labelers.

387 Neurons with higher SNR transients are detected more accurately

388 While CALMAN ONLINE had balanced performance with respect to precision and recall (mean precision
 389 0.77 ± 0.05 , mean recall 0.76 ± 0.07), CALMAN BATCH showed significantly higher precision than recall
 390 (mean precision 0.83 ± 0.09 , mean recall 0.69 ± 0.08). We looked into this behavior, by analyzing
 391 CALMAN BATCH performance as a function of the SNR of the inferred and ground truth traces
 392 (Fig. 4c-d). The SNR measure of a trace corresponds to the peak-SNR averaged over the length of a
 393 typical trace (see *Methods and Materials (Detecting fluorescence traces with high SNR)*). An example is
 394 shown in Fig. 4c where the scatter plot of SNR between matched ground truth and inferred traces is
 395 shown (false negative/positive components are shown along the x- and y- axis, respectively). To
 396 evaluate the performance we computed a precision metric as the fraction of inferred components

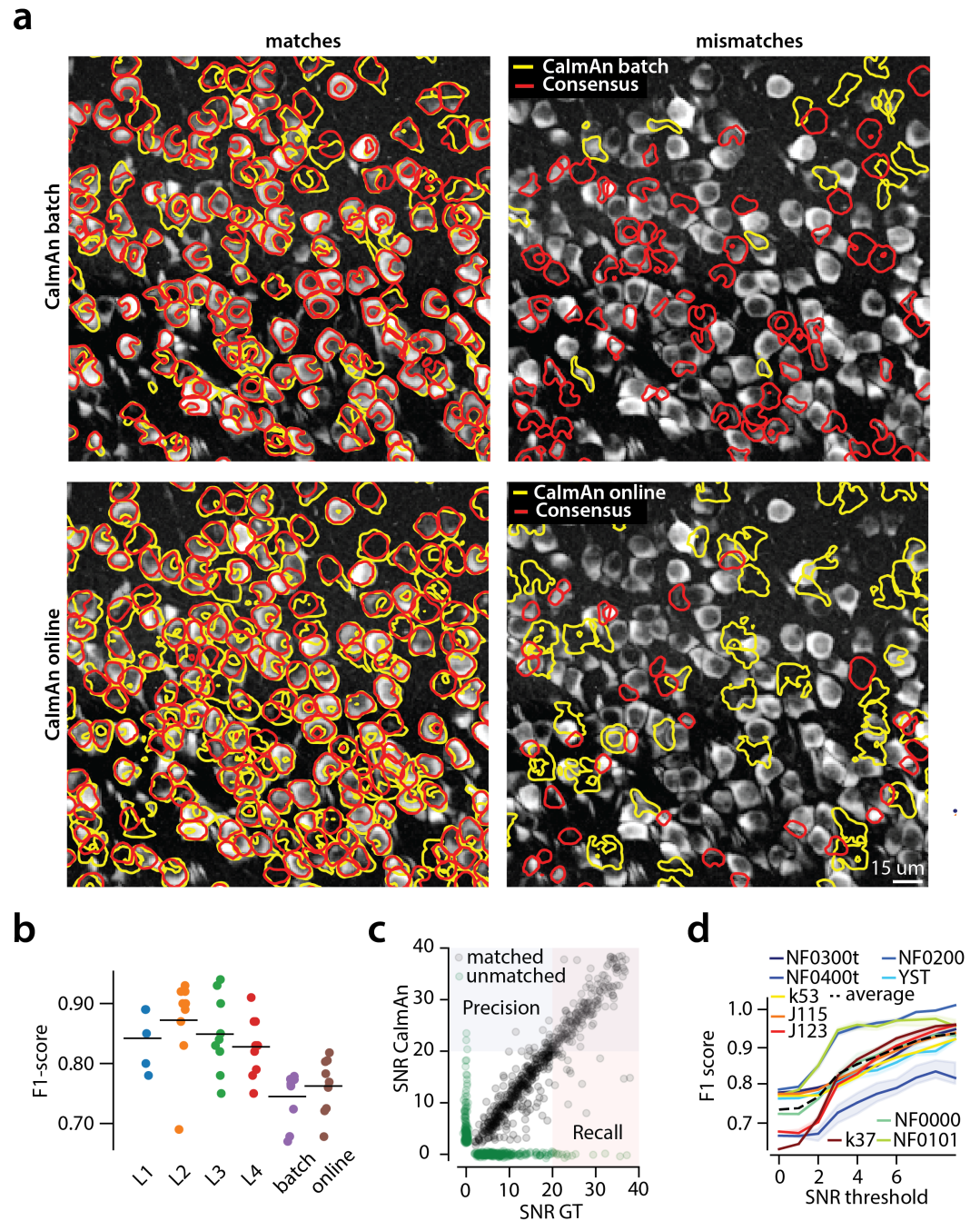


Figure 4. Evaluation of CALMAN performance against manually annotated data. (a) Comparison of CALMAN BATCH (top) and CALMAN ONLINE (bottom) when benchmarked against consensus ground truth for dataset k53. For a portion of the FOV, correlation image overlaid with matches (left panels, true positives red for consensus ground truth, yellow for CALMAN) and mismatches (right panels, red for false negatives, yellow for false positives). (b) Performance of CALMAN BATCH, CALMAN ONLINE and all labelers (L1, L2, L3, L4) for all 9 datasets in terms of F_1 score. CALMAN BATCH and CALMAN ONLINE reach near-human accuracy for neuron detection. Complete results with precision and recall for each dataset are given in Table 1. (c-d) Performance of CALMAN BATCH increases with peak SNR. (c) Example of scatter plot between SNRs of matched traces between CALMAN BATCH and ground truth for dataset k53. False negative/positive pairs are plotted in green along the x- and y-axes respectively, perturbed as a point cloud to illustrate the density. Most false positive/negative predictions occur at low SNR values. Shaded areas represent thresholds above which components are considered for matching (blue for CALMAN BATCH selected components and red for GT selected components) (d) F_1 score and upper/lower bounds for all datasets as a function of various peak SNR thresholds. Performance increases significantly for neurons with high peak SNR traces (see text for definition of metrics and the bounds).

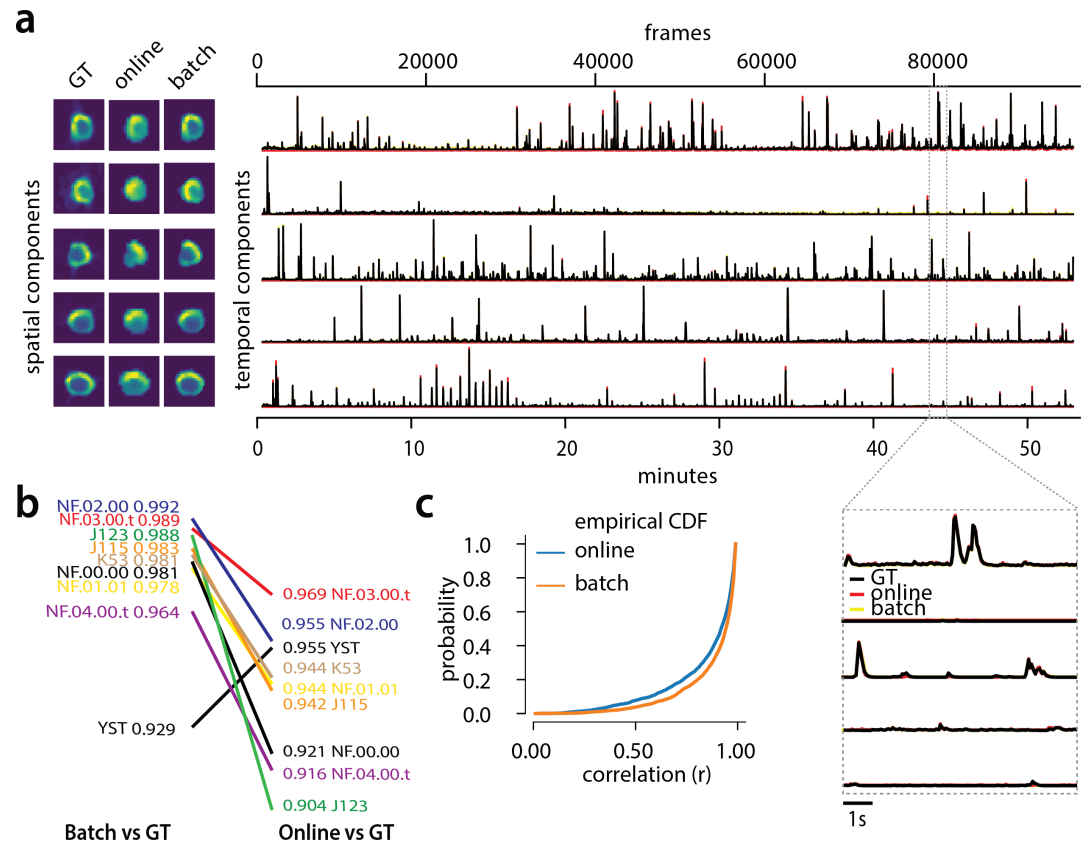


Figure 5. Evaluation of CALMAN extracted traces against traces derived from ground truth. (a) Examples of shapes (left) and traces (right) are shown for five matched components extracted from dataset K53 for consensus ground truth (GT, black), CALMAN BATCH (yellow) and CALMAN ONLINE (red) algorithms. The dashed gray portion of the traces is also shown magnified (bottom-right). Spatial footprints and traces for ground truth are obtained by seeding CALMAN with the consensus binary masks. The traces extracted from both versions of CALMAN match closely the ground truth traces. (b) Slope graph for the average correlation coefficient for matches between ground truth and CALMAN BATCH, and between ground truth and CALMAN ONLINE. Batch processing produces traces that match more closely the traces extracted from the ground truth data. (c) Empirical cumulative distribution functions of correlation coefficients aggregated over all the tested datasets. Both distributions exhibit a sharp derivative close 1 (last bin), with the batch approach giving better results.

397 above a certain SNR threshold that are matched with a ground truth component (Fig. 4c, shaded
 398 blue). Similarly we computed a recall metric as the fraction of ground truth components above
 399 a SNR threshold that are detected by CALMAN BATCH (Fig. 4c, shaded red), and an F_1 score as the
 400 harmonic mean of the two (Fig. 4d). The results indicate that the performance significantly grows as
 401 a function of the SNR for all datasets considered, growing on average from 0.73 when all neurons
 402 are considered to 0.92 when only neurons with traces having $\text{SNR} \geq 9$ are considered (Fig. 4d)⁵.

403 CALMAN reproduces the ground truth traces with high fidelity

404 Testing the quality of the inferred traces is a more challenging task due to the complete lack
 405 of ground truth data in the context of large scale *in vivo* recordings. As mentioned above, we
 406 considered as ground truth the traces obtained by running the CNMF algorithm seeded with the

⁵These precision and recall metrics are computed on different sets of neurons, and therefore strictly speaking one cannot combine them to form an F_1 score. However, they can be bound from above by being evaluated on the set of matched and non-matched components where at least one trace is above the threshold (union of blue and pink zones in Fig. 4c) or below by considering only matched and non-matched components where both ground truth and inferred traces have SNR above the threshold (intersection of blue and pink zones in Fig. 4c). In practice these bounds were very tight for all but one dataset (Fig. 4d). More details can be found in *Methods and Materials (Performance quantification as a function of SNR)*.

407 binary masks obtained by consensus ground truth procedure. After alignment of the ground truth
408 with the results of CALMAN, the matched traces were compared both for CALMAN BATCH and for
409 CALMAN ONLINE. Fig. 5a, shows an example of 5 of these traces for the dataset K53, showing very
410 similar behavior of the traces in these three different cases.

411 To quantify the similarity we computed the correlation coefficients of the traces (ground truth vs
412 CALMAN BATCH, and ground truth vs CALMAN ONLINE) for all the 9 datasets (Fig. 5b-c). Results indicated
413 that for all but one dataset (Fig. 5b) CALMAN BATCH reproduced the traces with higher fidelity, and
414 in all cases the mean correlation coefficients was higher than 0.9, and the empirical histogram
415 of correlation coefficients peaked at the maximum bin 0.99-1 (Fig. 5c). The results indicate that
416 the batch approach extracts traces closer to the ground truth traces. This can be attributed to
417 a number of reasons: By processing all the time points simultaneously, the batch approach can
418 smooth the trace estimation over the entire time interval as opposed to the online approach where
419 at each timestep only the information up to that point is considered. Moreover, CALMAN ONLINE
420 might not detect a neuron until it becomes strongly active. This neuron's activity before detection is
421 unknown and has a default value of zero, resulting in a lower correlation coefficient. While this can
422 be ameliorated to a great extent with additional passes over the data, the results indicate trade-offs
423 between using the online and offline versions of CALMAN.

424 **Online analysis of a whole brain zebrafish dataset**

425 We tested CALMAN ONLINE with a 380GB whole brain dataset of larval zebrafish (*Danio rerio*) acquired
426 with a light-sheet microscope (*Kawashima et al., 2016*). The imaged transgenic fish (Tg(elavl3:H2B-
427 GCaMP6f)jf7) expressed the genetically encoded calcium indicator GCaMP6f in almost all neuronal
428 nuclei. Data from 45 planes (FOV 820x410 μm^2 , spaced at 5.5 μm intervals along the dorso-ventral
429 axis) was collected at 1Hz for 30 minutes (for details about preparation, equipment and experiment
430 refer to *Kawashima et al. (2016)*). With the goal of simulating real-time analysis of the data, we run
431 all the 45 planes in parallel on a computing cluster with 9 nodes (each node is equipped with 24
432 CPUs and 128-256 GB RAM). Data was not stored locally in each machine but directly accessed from
433 a network drive.

434 The algorithm was initialized with CALMAN BATCH run on 200 initial frames and looking for 500
435 components. The small number of frames (1885) and the large FOV size (2048 \times 1188 pixels) for this
436 dataset motivated this choice of increased number of components during initialization. In Fig. 6 we
437 report the results of the analysis for plane number 11 of 45. For plane 11, CALMAN ONLINE found
438 1524 neurons after processing 1685 frames. Since no ground truth was available for this dataset,
439 it was only possible to evaluate the performance of this algorithm by visual inspection. CALMAN
440 ONLINE identified all the neurons with a clear footprint in the underlying correlation image (higher
441 SNR, Fig. 6a) and missed a small number of the fainter ones (low SNR). By visual inspection of
442 the components the authors could find very few false positives. Given that the parameters were
443 not tuned and that the classifier was not trained on zebrafish neurons, we hypothesize that the
444 algorithm is biased towards a high precision result. Spatial components displayed the expected
445 morphological features of neurons (Fig. 6b-c). Considering all the planes (Figs 6e and 11) CALMAN
446 ONLINE was able to identify in a single pass of the data a total of 66108 neurons. See Supplemental
447 Movie 3 for a summary across all planes. The analysis was performed in 21 minutes, with the first
448 3 minutes allocated to the initialization and the remaining 18 to process the rest of the data in
449 streaming mode (and in parallel for each plane). This demonstrates the ability of CALMAN ONLINE
450 to process large amounts of data in real-time (see also Fig. 8 for a discussion of computational
451 performance).

452 **Analyzing 1p microendoscopic data using CALMAN**

453 We tested the CNMF-E implementation of CALMAN BATCH on *in vivo* microendoscopic data from
454 mouse dorsal striatum, with neurons expressing GCaMP6f. 6000 frames were acquired at 30
455 frames per second while the mouse was freely moving in an open field arena (for further details

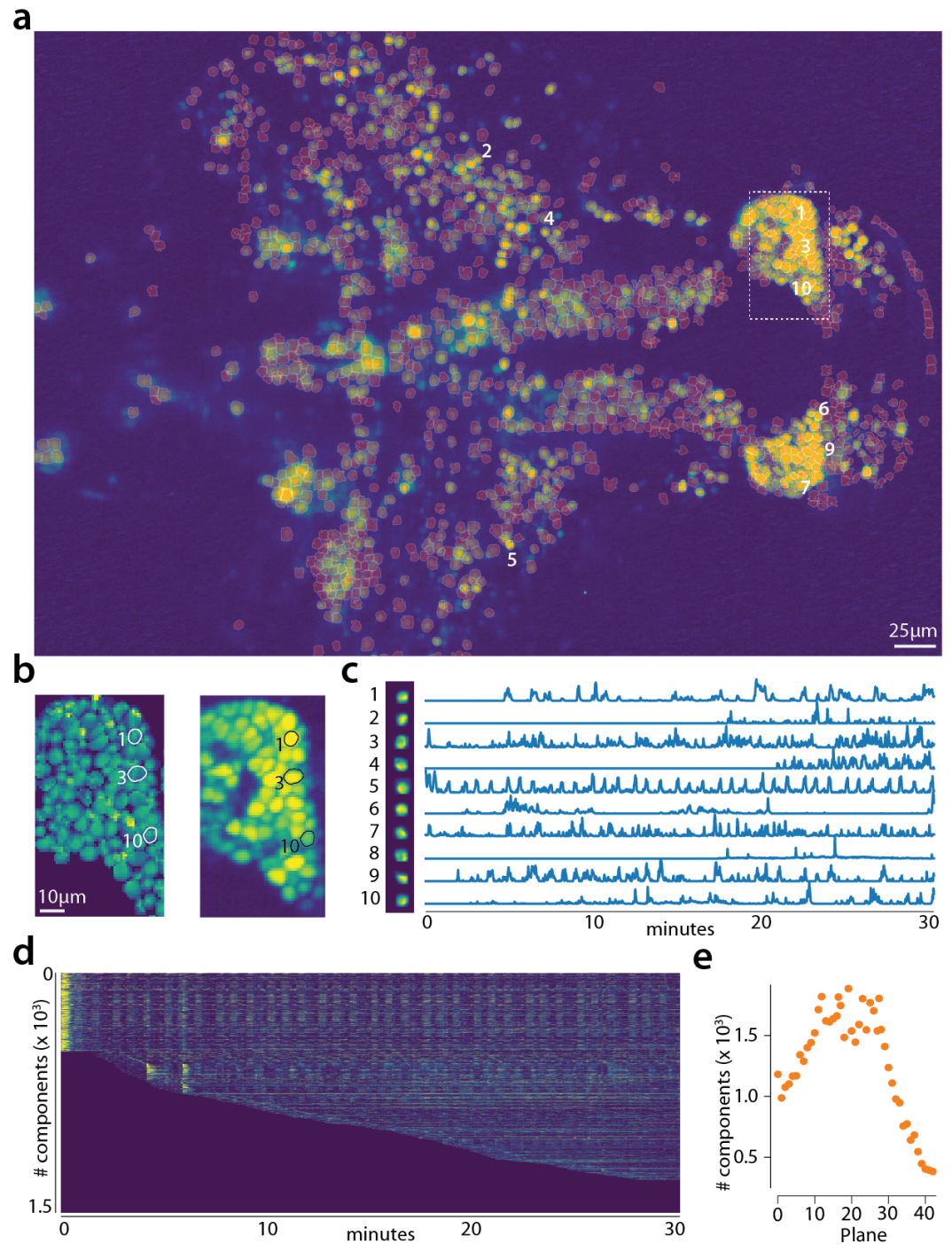


Figure 6. Online analysis of a 30 min long whole brain recording of the zebrafish brain. (a) Correlation image overlaid with the spatial components found by the algorithm (portion of plane 11 out of 45 planes in total). (b) Left: Spatial footprints found in the dashed region in (a), contours represent neurons displayed in (c). Right: Correlation image for the same region. (c) Spatial (left) and Temporal (right) components associated to the ten example neurons marked in panel (a). (d) Temporal traces for all the neurons found in the FOV in (a), the initialization on the first 200 frames contained 500 neurons (present since time 0). (e) Number of neurons found per plane (See also Supplementary Fig. 11 for a summary of the results from all planes).

456 refer to *Zhou et al. (2018)*). In Fig. 7 we report the results of the analysis using CALMAN BATCH with
457 patches and compare to the results of the MATLAB[®] implementation of *Zhou et al. (2018)*. Both
458 implementations detect similar components (Fig. 7a) with an F_1 -score of 0.89. 573 neurons were
459 found by both implementations. 106 and 31 additional components were detected by *Zhou et al.*
460 *(2018)* and CALMAN BATCH respectively. The median correlation between the temporal traces of
461 neurons detected by both implementations was 0.86. Similar results were also obtained by running
462 CALMAN without patches. Ten example temporal traces are plotted in Fig. 7b.

463 **Computational performance of CALMAN**

464 We examined the performance of CALMAN in terms of processing time for the various analyzed
465 datasets presented above (Fig. 8). The processing time discussed here excludes motion correction,
466 which is highly efficient and primarily depends on the level of the FOV discretization for non-rigid
467 motion correction (*Pnevmatikakis and Giovannucci, 2017*). For CALMAN BATCH, each dataset was
468 analyzed using three different computing architectures: i) a single laptop (MacBook Pro) with 8
469 CPUs and 16GB of RAM (blue in Fig. 8a), ii) a linux-based workstation (CentOS) with 24 CPUs and
470 128GB of RAM (magenta), and iii) a linux-based HPC cluster (CentOS) where 112 CPUs (4 nodes, 28
471 CPUs each) were allocated for the processing task (yellow). Fig. 8a shows the processing of CALMAN
472 BATCH as a function of dataset size on the 5 longest datasets, whose size exceeded 8GB, on log-log
473 plot.

474 Results show that, as expected, employing more processing power results in faster processing.
475 CALMAN BATCH on a HPC cluster processes data faster than acquisition time (Fig. 8a) even for very
476 large datasets. Processing of an hour long dataset was feasible within 3 hours on a single laptop,
477 even though the dataset has size multiple times the available RAM memory. Here, acquisition time
478 is defined as number of frames times imaging rate, computed based on the assumption of imaging
479 a FOV discretized over a 512×512 grid at a 30Hz rate (a typical two-photon imaging setup with
480 resonant scanning microscopes), and a representation of the measurements using single precision
481 arithmetic, which is the minimum precision required for standard algebraic processing. These
482 assumptions lead to a data rate of ~ 105 GB/hour. In general the performance scales linearly with
483 the number of frames (and hence, the size of the dataset), but a dependence is also observed with
484 respect to the number of components. The majority of the time (Fig. 8b-left) the majority of the
485 time required for CALMAN BATCH processing is taken by CNMF algorithmic processing either during
486 the initialization in patches (orange bar) or during merging and refining the results of the individual
487 patches (green bar).

488 Fig. 8a also shows the speed performance of CALMAN ONLINE (red markers). Because of the
489 low memory requirements of the streaming algorithm, this performance only mildly depends on
490 the computing infrastructure allowing for near real-time processing speeds on a standard laptop
491 (Fig. 8a). As discussed in *Giovannucci et al. (2017)* processing time of CALMAN ONLINE depends
492 primarily on i) the computational cost of tracking the temporal activity of discovered neurons, ii)
493 the cost of detecting and incorporating new neurons, and iii) the cost of periodic updates of spatial
494 footprints. Fig. 8b-right shows that the two first steps, which are required for each frame, can
495 be done in real-time. In Fig. 8c the cost per frame is plotted for the analysis of the whole brain
496 zebrafish recording. The lower imaging rate (1Hz) allows for the tracking of neural activity to be
497 done with computational cost significantly lower than the 1 second between volume imaging time
498 (Fig. 8c), even in the presence of a large number of components (typically more than 1000 per plane,
499 Fig. 6) and the significantly larger FOV (2048×1188 pixels). As expected the cost of updating spatial
500 footprints can be significantly larger if done simultaneously for all components (Fig. 8c, bottom).
501 However, the average cost of updating a single spatial footprint is roughly 8ms, enabling real-time
502 processing *for each* frame, when this step is evenly distributed among different frames/volumes, or
503 is performed by a parallel independent process (*Giovannucci et al., 2017*).

504 The cost of processing 1p data in CALMAN BATCH using the CNMF-E algorithm (*Zhou et al., 2018*)
505 is shown (Fig. 8d) for the workstation hardware. Splitting in patches and processing in parallel can

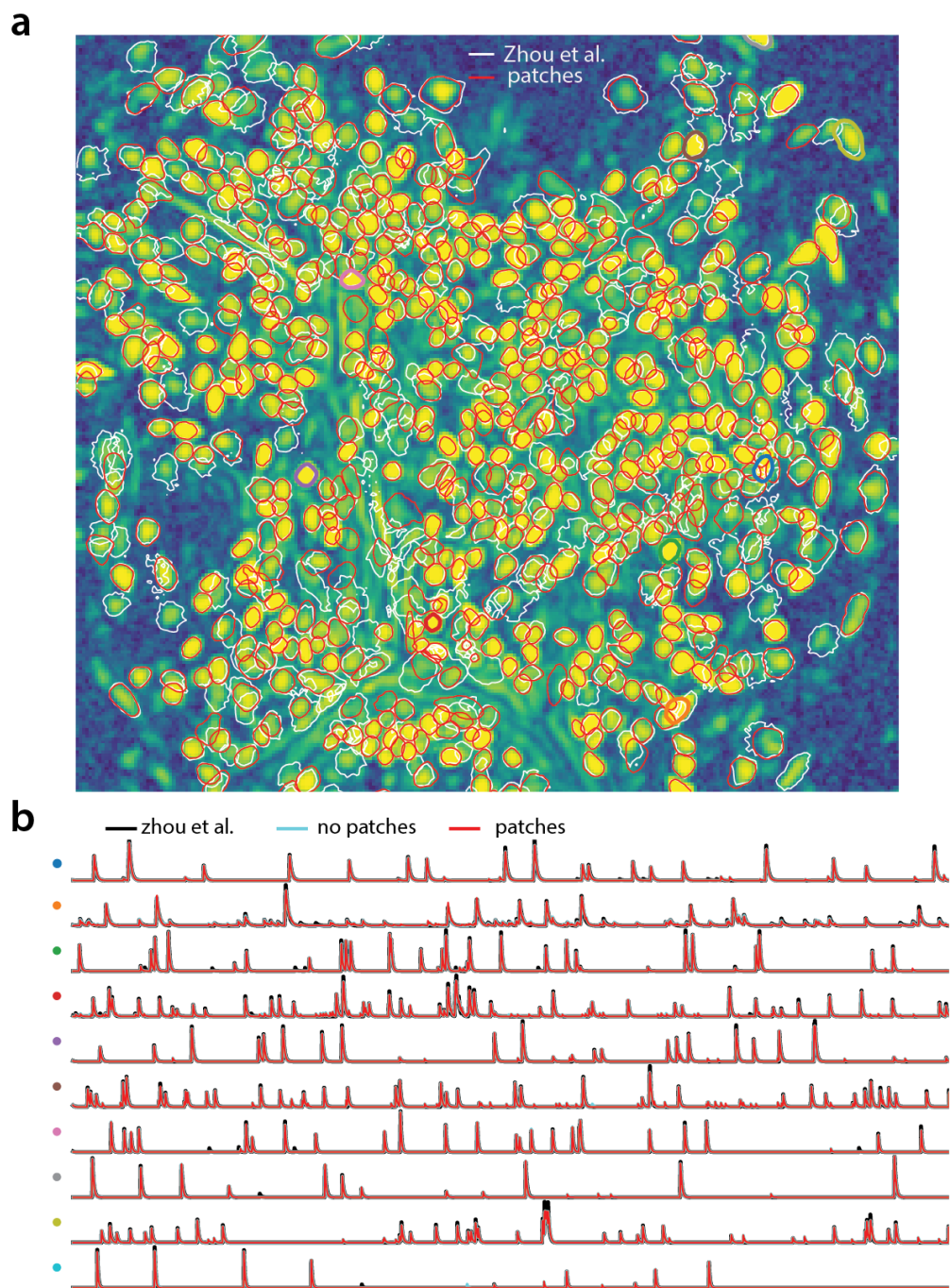


Figure 7. Analyzing microendoscopic 1p data with the CNMF-E algorithm using CAIMAN BATCH. (a) Contour plots of all neurons detected by the CNMF-E (white) implementation of *Zhou et al. (2018)* and CAIMAN BATCH (red) using patches. Colors match the example traces shown in (b), which illustrate the temporal components of 10 example neurons detected by both implementations. CAIMAN BATCH reproduces with reasonable fidelity the results of *Zhou et al. (2018)*.

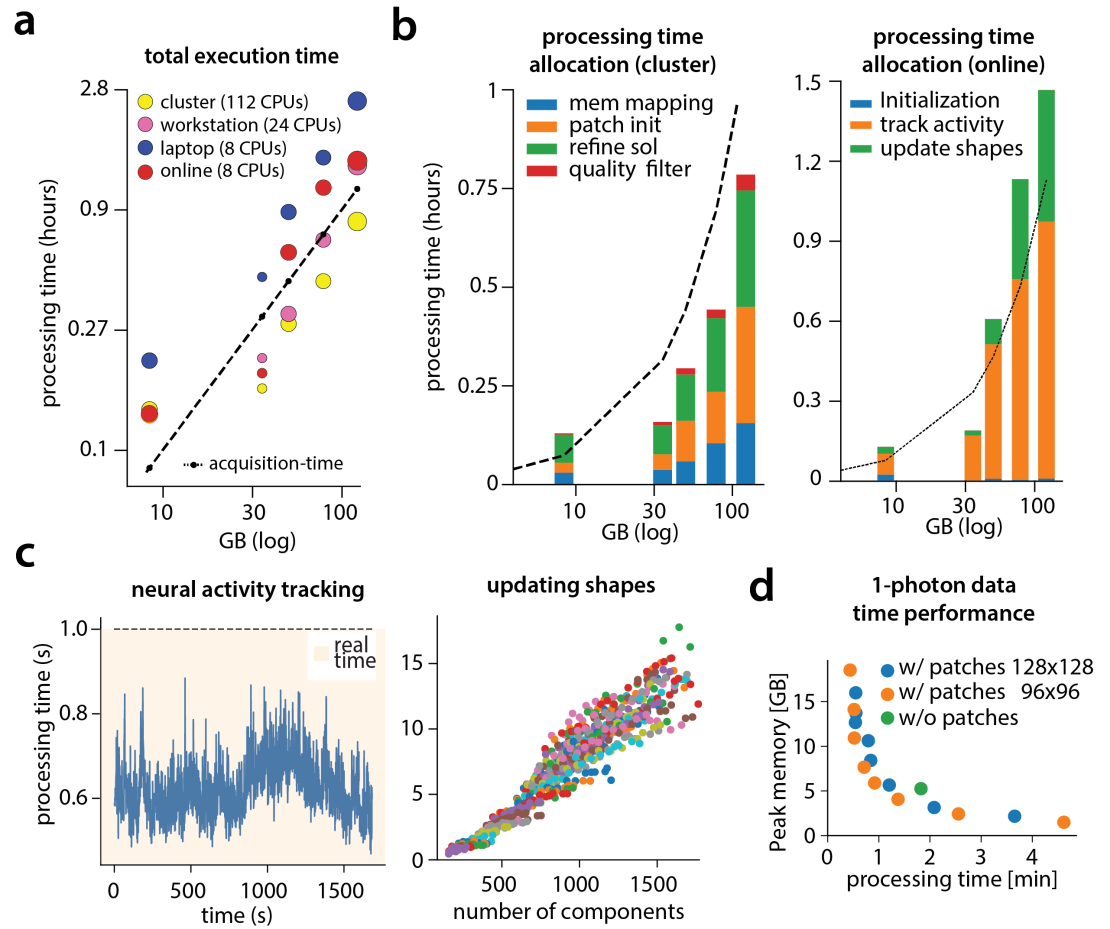


Figure 8. Time performance of CALMAN BATCH and CALMAN ONLINE for the analyzed datasets. (a) Log-log plot of total processing time as a function of data size for CALMAN BATCH for the 5 largest two-photon datasets using three different processing infrastructures: i) a laptop with 8 CPUs (blue), ii) a desktop workstation with 24 CPUs (magenta), and iii) a HPC where 112 CPUs are allocated (yellow). The results indicate a near linear scaling of the processing time with the size of dataset, with additional dependence on the number of found neurons (size of each point). Even very large datasets (> 100GB) can be processed efficiently with a single laptop, whereas access to a HPC enables processing with speed faster than the acquisition time (considered 30Hz for a 512x512 FOV here). The results of CALMAN ONLINE using the laptop are also plotted in red indicating near real-time processing speed. (b) Break down of processing time for CALMAN BATCH (left) and CALMAN ONLINE (right) (excluding motion correction). (Left) Processing with CNMF in patches and refinement takes most of the time for CALMAN BATCH. Right: Tracking neural activity and new neuron detection can be done in real-time for CALMAN ONLINE. (c) (Left) Cost of neural activity online tracking for the whole brain zebrafish dataset (maximum time over all planes per frame). Tracking can be done in real-time. (Right) The most expensive part during online processing occurs while updating the spatial footprints, a step that can be distributed or parallelized. Each color corresponds to the update cost for the various different planes. (d) Cost analysis of CNMF-E implementation for processing a 6000 frames long 1p dataset. Processing in patches in parallel induces a time/memory tradeoff and can lead to speed gains (patch size in legend).

506 lead to computational gains at the expense of increased memory usage. This is because the CNMF-E
507 introduces a background term that has the size of the dataset and needs to be loaded and updated
508 in memory in two copies. This leads to processing times that are slower compared to the standard
509 processing of 2p datasets, and higher memory requirements. However, as Fig. 8d demonstrates,
510 memory usage can be controlled enabling scalable inference at the expense of slower processing
511 speeds.

512 **CAIMAN successfully tracks neurons across multiple days**

513 Fig. 9 shows an example of tracking neurons across 6 different sessions corresponding to 6 different
514 days of mouse cortex *in vivo* data using our multi-day registration algorithm REGISTERMULTI (see
515 Methods, Alg. 6). 453, 393, 375, 378, 376, and 373 active components were found in the six sessions,
516 respectively. Our tracking method detected a total of 686 distinct active components. Of these, 172,
517 108, 70, 92, 82, and 162 appeared in exactly 1, 2, 3, 4, 5, and all 6 sessions respectively. Contour
518 plots of the 162 components that appeared in all sessions are shown in Fig. 9a, and parts of the
519 FOV are highlighted in Fig. 9d showing that components can be tracked in the presence of non-rigid
520 deformations of the FOV between the different sessions.

521 To test the stability of REGISTERMULTI for each subset of sessions, we repeated the same
522 procedure running backwards in time starting from day 6 and ending at day 1, a process that
523 also generated a total of 686 distinct active components. We identified the components present
524 in *at least* a given subset of sessions when using the forward pass, and separately when using
525 the backwards pass, and compared them against each other (Fig. 9b) for all possible subsets.
526 Results indicate a very high level of agreement between the two approaches with many of the
527 disagreements arising near the boundaries (data not shown). Disagreements near the boundaries
528 can arise because the forward pass aligns the union with the FOV of the last session, whereas the
529 backwards pass with the FOV of the first session, potentially leading to loss of information near the
530 boundaries.

531 A step by step demonstration of the tracking algorithm for the first three sessions is shown in
532 the appendix (Fig. 10). Our approach allows for the comparison of two non-consecutive sessions
533 through the union of components without the need of a direct pairwise registration (Fig. 10f), where
534 it is shown that registering sessions 1 and 3 directly and through the union leads to nearly identical
535 results. Fig. 9c compares the registrations for all pairs of sessions using the forward (red) or the
536 backward (blue) approach, with the direct pairwise registrations. Again, the results indicate a very
537 high level of agreement, indicating the stability and effectiveness of the proposed approach.

538 **Discussion**

539 **Reproducible and scalable analysis for the 99%**

540 Significant advances in the reporting fidelity of fluorescent indicators, and the ability to simulta-
541 neously record and modulate neurons granted by progress in optical technology, have propelled
542 calcium imaging to being the main experimental method in systems neuroscience alongside elec-
543 trophysiology recordings. The resulting increased adoption rate has generated an unprecedented
544 wealth of imaging data which poses significant analysis challenges. The goal of CAIMAN is to provide
545 the experimentalist with a complete suite of tools for analyzing this data in a formal, scalable,
546 and reproducible way. The goal of this paper is to present the features of CAIMAN and examine
547 its performance in detail. CAIMAN embeds existing methods for preprocessing calcium imaging
548 data into a MapReduce framework and augments them with supervised learning algorithms and
549 validation metrics. It builds on the CNMF algorithm of *Pnevmatikakis et al. (2016)* for source
550 extraction and deconvolution, extending it along the lines of i) reproducibility and performance
551 improvement, by automating quality assessment through the use of unsupervised and supervised
552 learning algorithms for component detection and classification, and ii) scalability, by enabling fast
553 large scale processing with standard computing infrastructure (e.g., a commodity laptop or worksta-

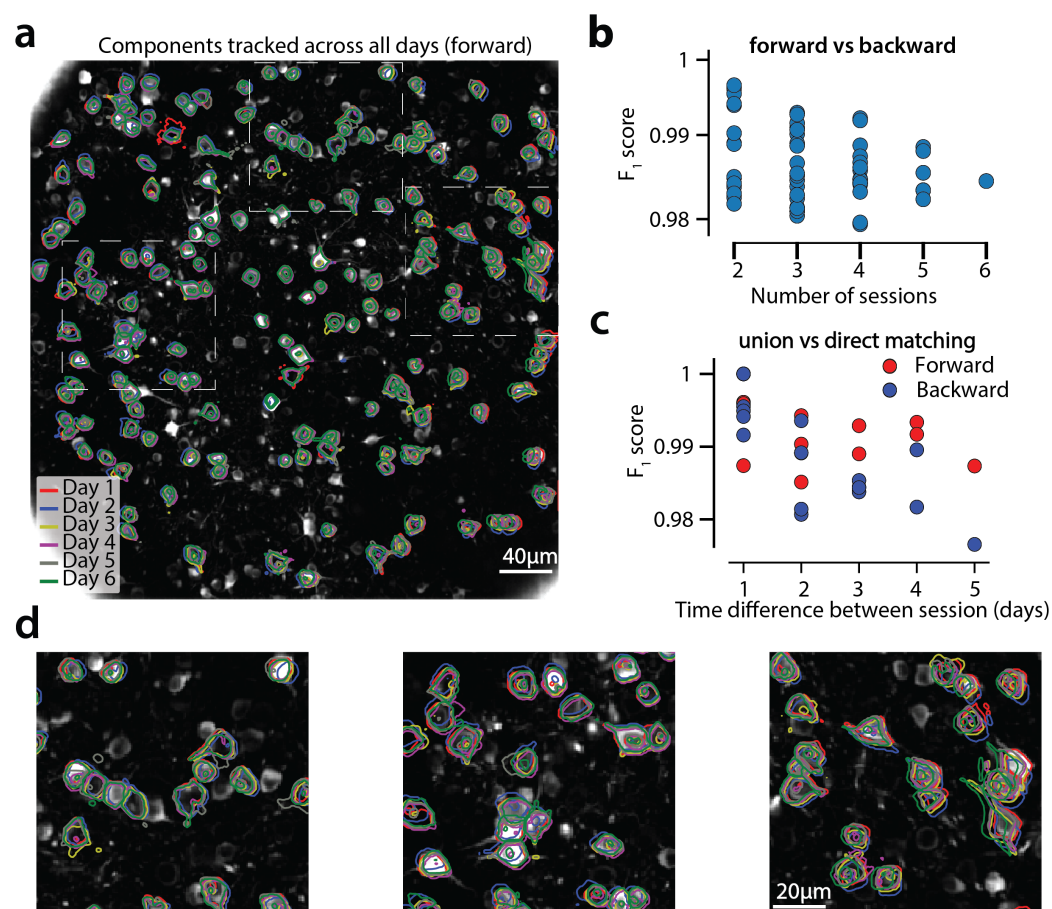


Figure 9. Components registered across six different sessions (days). (a) Contour plots of neurons that were detected to be active in all six imaging sessions overlaid on the correlation image of the sixth imaging session. Each color corresponds to a different session. (b) Stability of multiday registration method. Comparisons of forward and backward registrations in terms of F_1 scores for all possible subsets of sessions. The comparisons agree to a very high level indicating the stability of the proposed approach. (c) Comparison (in terms of F_1 score) of pair-wise alignments using readouts from the union vs direct alignment. The comparison is performed for both the forward and the backwards alignment. For all pairs of sessions the alignment using the proposed method gives very similar results compared to direct pairwise alignment. (d) Magnified version of the tracked neurons corresponding to the squares marked in panel (a). Neurons in different parts of the FOV exhibit different shift patterns over the course of multiple days, but can nevertheless be tracked accurately by the proposed multiday registration method.

554 tion). Scalability is achieved by either using a MapReduce batch approach, which employs parallel
555 processing of spatially overlapping, memory mapped, data patches; or by integrating the online
556 processing framework of *Giovannucci et al. (2017)* within our pipeline. Apart from computational
557 gains both approaches also result in improved performance. Towards our goal of providing a
558 single package for dealing with standard problems arising in analysis of imaging data, CALMAN
559 also includes an implementation of the CNMF-E algorithm of *Zhou et al. (2018)* for the analysis
560 of microendoscopic data, as well as with a novel method for registering analysis results across
561 multiple days.

562 **Towards surpassing human neuron detection performance**

563 To evaluate the performance of CALMAN BATCH and CALMAN ONLINE, we generated a corpus of
564 multiply annotated two-photon imaging datasets. The results indicated a surprising level of dis-
565 agreement between individual labelers, highlighting both the difficulty of the problem, and the
566 non-reproducibility of the laborious task of human annotation. CALMAN reached near-human
567 performance with respect to this ground truth, by using the *same* parameters for all the datasets
568 without dataset dependent parameter tweaking. Such tweaking could for example include setting
569 the SNR threshold based on the noise level of the recording, the complexity of the neuropil signal
570 based on the level of background activity, or specialized treatment around the boundaries of the
571 FOV to compensate for eventual imaging artifacts.

572 Apart from being used as a benchmarking tool, the set of manual annotations can also be used
573 as labeled data for supervised learning algorithms. CALMAN uses two CNN based classifiers trained
574 on (a subset of) this data, one for post processing component classification in CALMAN BATCH, and
575 the other for detecting new neurons in residual images in the CALMAN ONLINE. The deployment
576 of these classifiers resulted in significant gains in terms of performance, and we expect further
577 advances in the future. The annotations will be made freely available to the community upon
578 publication of the paper for benchmarking and training purposes.

579 **CALMAN BATCH VS CALMAN ONLINE**

580 Our results suggest similar performance between CALMAN BATCH and CALMAN ONLINE in terms of
581 processing speed and quality of results with CALMAN ONLINE outperforming CALMAN BATCH on
582 longer datasets in terms of neuron detection, possibly due to its inherent ability to adapt to non-
583 stationarities arising during the course of a large experiment, and underperforming on shorter
584 datasets potentially due to lack of enough information. By contrast, CALMAN BATCH extracts better
585 traces compared to CALMAN ONLINE with respect to “ground truth” traces. While multiple passes
586 over the data with CALMAN ONLINE can mitigate these shortcomings, this still depends on good
587 initialization with CALMAN BATCH, as the analysis of the whole brain zebrafish dataset indicates. In
588 offline setups, CALMAN ONLINE could also benefit from the post processing component evaluation
589 tools used in batch mode. e.g., using the batch classifier for detecting false positive components at
590 the end of the experiment.

591 What sets the two algorithms apart is the streaming processing mode of CALMAN ONLINE which,
592 besides lowering memory requirements, can be used to enable novel types of closed-loop all-
593 optical experiments (*Packer et al., 2015; Carrillo-Reid et al., 2017*). As discussed in *Giovannucci*
594 *et al. (2017)*, typical all-optical closed-loop experiments require the pre-determination of ROIs that
595 are monitored/modulated. Processing with CALMAN ONLINE can improve upon this by allowing
596 identification and modulation of new neurons on the fly, greatly expanding the space of possible
597 experiments. Even though our simulated online processing setup is not integrated with hardware to
598 an optical experimental setup, our results indicate that CALMAN ONLINE performed close to real-time
599 in most cases, without optimizing for speed. This suggest that large scale closed-loop experiments
600 with single cell resolution are feasible by combining existing all-optical technology and our proposed
601 analysis method.

602 **Future directions**

603 While CALMAN uses a highly scalable processing pipeline for two-photon datasets, processing of
604 one-photon microendoscopic imaging data is less scalable due to the more complex background
605 model that needs to be retained in memory during processing. Adapting CALMAN ONLINE to the one-
606 photon data processing algorithm of *Zhou et al. (2018)* is a promising way for scaling up efficient
607 processing in this case. The continuing development and quality improvement of neural activity
608 indicators has enabled direct imaging of neural processes (axons/dendrites), imaging of synaptic
609 activity (*Xie et al., 2016*), or direct imaging of voltage activity *in vivo* conditions (*Piatkevich et al.,*
610 *2018*). While the approach presented here is tuned for somatic imaging through the use of various
611 assumptions (space localized activity, CNN classifiers trained on images of somatic activity), the
612 technology of CALMAN is largely transferable to these domains as well. These extensions will be
613 pursued in future work.

614 **Methods and Materials**

615 **Memory mapping**

616 In order to efficiently access data in parallel, CALMAN BATCH relies on memory mapping. With
617 memory mapped (mmap) arrays, arithmetic operations can be performed on data residing on the
618 hard drive without explicitly loading it to RAM, and slices of data can be indexed and accessed
619 without loading the full file in memory, enabling out-of-core processing (*Toledo, 1999*). The order
620 in which data in a memory mapped file is stored on the hard drive can dramatically affect the
621 read-write performance of out-of-core operations on spinning disks, and to a lesser degree on solid
622 state drives. On modern computers tensors are stored in linear format, no matter the number of
623 the array dimensions. Therefore, one has to decide which elements of an array are contiguous in
624 memory: in *row-major order*, consecutive elements of a row (first-dimension) are next to each other,
625 whereas in *column-major order* consecutive elements of a column (last dimension) are contiguous.
626 Such decisions significantly affect the speed at which data is read or written: in *column-major order*
627 reading a full column is fast because memory is read in a single sequential block, whereas reading a
628 row is inefficient since only one element can be read at a time and all the data needs to be accessed.
629 Therefore, the original dataset must be saved in the right order to avoid performance problems.

630 In the context of calcium imaging datasets, CALMAN BATCH represents the datasets in a matrix
631 form Y , where each row corresponds to a different imaged pixel, and each column to a different
632 frame. As a result, a *column-major order* mmap file enables the fast access of individual frames at a
633 given time, whereas a *row-major order* files enables the fast access of an individual pixel at all times.
634 To facilitate processing in patches CALMAN BATCH stores the data in *row-major order*. In practice,
635 this is opposite to the order with which the data appears, one frame at a time. In order to reduce
636 memory usage and speed up computation CALMAN BATCH employs a MapReduce approach, where
637 either multiple files or multiple chunks of a big file composing the original datasets are processed
638 and saved in mmap format in parallel. This operation includes two phases, first the chunks/files are
639 saved in multiple row-major mmap format, and then chunks are simultaneously combined into a
640 single large row-major mmap file. In order to reduce preprocessing steps, if the file(s) need to be
641 corrected for motion artifacts, chunks of the registered data can be stored on-the-fly during motion
642 correction.

643 **Mathematical model of the CNMF framework**

644 The CNMF framework (Fig. 1d) for calcium imaging data representation can be expressed in mathe-
645 matical terms as (*Pnevmatikakis et al., 2016*)

$$Y = AC + B + E. \quad (1)$$

646 Here, $Y \in \mathbb{R}^{d \times T}$ denotes the observed data written in matrix form, where d is the total number
647 of observed pixels/voxels, and T is the total number of observed timesteps (frames). $A \in \mathbb{R}^{d \times N}$

648 denotes the matrix of the N spatial footprints, $A = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_N]$, with $\mathbf{a}_i \in \mathbb{R}^{d \times 1}$ being the spatial
 649 footprint of component i . $C \in \mathbb{R}^{N \times T}$ denotes the matrix of temporal components, $C = [\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_N]^T$,
 650 with $\mathbf{c}_i \in \mathbb{R}^{T \times 1}$ being the temporal trace of component i . B is the background/neuropil activity
 651 matrix. For two-photon data it is modeled as a low rank matrix $B = \mathbf{b}\mathbf{f}$, where $\mathbf{b} \in \mathbb{R}^{d \times n_b}$, $\mathbf{f} \in \mathbb{R}^{n_b \times T}$
 652 correspond to the matrices of spatial and temporal background components, and n_b is the number
 653 of background components. For the case of micro-endoscopic data the integration volume is much
 654 larger and the low rank model is inadequate. A solution comes from the CNMF-E algorithm of **Zhou**
 655 **et al. (2018)** where the background is modeled as

$$B = W(Y - AC), \quad (2)$$

656 where $W \in \mathbb{R}^{d \times d}$ is an appropriate weight matrix, where the (i, j) entry models the influence of the
 657 neuropil signal of pixel j to the neuropil signal at pixel i .

658 Combining results from different patches

659 To combine results from the different patches we first need to account for the overlap at the bound-
 660 aries. Neurons lying close to the boundary between neighboring patches can appear multiple times
 661 and must be merged. With this goal, we optimized the merging approach used in **Pnevmatikakis**
 662 **et al. (2016)**: Groups of components with spatially overlapping footprints whose temporal traces are
 663 correlated above a threshold are replaced with a single component, that tries to explain as much of
 664 the variance already explained by the "local" components (as opposed to the variance of the data
 665 as performed in (**Pnevmatikakis et al., 2016**)). If $A_{\text{old}}, C_{\text{old}}$ are the matrices of components to be
 666 merged, then the merged component $\mathbf{a}_m, \mathbf{c}_m$ are given by the solution of the rank-1 NMF problem:

$$\min_{\mathbf{a}_m \geq 0, \mathbf{c}_m \geq 0} \|A_{\text{old}} C_{\text{old}} - \mathbf{a}_m \mathbf{c}_m^T\|. \quad (3)$$

667 Prior to merging, the value of each component at each pixel is normalized by the number of patches
 668 that overlap in this pixel, to avoid counting the activity of each pixel multiple times.

669 We follow a similar procedure for the background/neuropil signals from the different patches.
 670 For the case of two-photon data, the spatial background/neuropil components for each patch can
 671 be updated by keeping their spatial extent intact to retain a local neuropil structure, or they can
 672 be merged when they are sufficiently correlated in time as described above to promote a more
 673 global structure. For the case of one-photon data, CNMF-E estimates the background using a
 674 local auto-regressive process (see Eq. (2)) (**Zhou et al., 2018**), a setup that cannot be immediately
 675 propagated when combining the different patches. To combine backgrounds from the different
 676 patches, we first approximate the backgrounds B^i from all the patches i with a low rank matrix
 677 using non-negative matrix factorization of rank g_b to obtain global spatial, and temporal background
 678 components.

$$[\mathbf{b}^i, \mathbf{f}^i] = \text{NNMF}(B^i, g_b). \quad (4)$$

679 The resulting components are embedded into a large matrix $B \in \mathbb{R}^{d \times T}$ that retains a low rank
 680 structure. After the components and backgrounds from all the patches have been combined,
 681 they are further refined by running CNMF iteration of updating spatial footprints, temporal traces,
 682 and neuropil activity. CAIMAN BATCH implements these steps in a highly parallel fashion (as also
 683 described in **Pnevmatikakis et al. (2016)**): Temporal traces whose corresponding spatial traces do
 684 not overlap can be updated in parallel. Similarly, the rows of the matrix of spatial footprints A can
 685 also be updated in parallel (2b). The process is summarized in algorithmic format in Alg. 1.

686 Initialization strategies

687 Source extraction using matrix factorization requires solving a bi-convex problem where initialization
 688 plays a critical role. The CNMF/CNMF-E algorithms use initialization methods that exploit the locality
 689 of the spatial footprints to efficiently identify the locations of candidate components (**Pnevmatikakis**
 690 **et al., 2016; Zhou et al., 2018**). CAIMAN incorporates these methods, extending them by using the

691 temporal locality of the calcium transient events. The available initialization methods for CALMAN
692 BATCH include:

693 **GREEDYROI:** This approach, introduced in *Pnevmatikakis et al. (2016)*, first spatially smooths the
694 data with a Gaussian kernel of size comparable to the average neuron radius, and then
695 initializes candidate components around locations where maximum variance (of the smoothed
696 data) is explained. This initialization strategy is fast but requires specification of the number
697 of components by the user.

698 **ROLLINGGREEDYROI:** The approach, introduced in this paper, operates like GREEDYROI by spatially
699 smoothing the data and looking for points of maximum variance. Instead of working across
700 all the data, ROLLINGGREEDYROI looks for points of maximum variance on a rolling window of
701 a fixed duration, e.g., 3 seconds, and initializes components by performing a rank one NMF on
702 a local spatial neighborhood. By focusing into smaller rolling windows, ROLLINGGREEDYROI
703 can better isolate single transient events, and as a result detect better neurons with sparse
704 activity. ROLLINGGREEDYROI is the default choice for processing of 2-photon data.

705 **GREEDYCORR:** This approach, introduced in *Zhou et al. (2018)*, initializes candidate components
706 around locations that correspond to the local maxima of an image formed by the pointwise
707 product between the correlation image and the peak signal-to-noise ratio image. By setting a
708 threshold for acceptance, this approach does not require the prior specification of number of
709 components. This comes at the expense of a higher computational cost. GREEDYCORR is the
710 default choice for processing of 1-photon data.

711 **SPARSENMF:** Sparse NMF approaches, when ran in small patches, can be effective for quickly uncov-
712 ering spatial structure in the imaging data, especially for neural processes (axons/dendrites)
713 whose shape cannot be easily parametrized and/or localized.

714 **Algorithm seeding with binary masks**

715 Often locations of components are known either from manual annotation or from labeled data
716 obtained in a different way, such as data from a static structural channel recorded concurrently
717 with the functional indicator. CALMAN can be seeded with binary (or real valued) masks for the
718 spatial footprints. Apart from A , these masks can be used to initialize all the other relevant matrices
719 C and B as well. This is performed by i) first estimating the temporal background components \mathbf{f}
720 using only data from parts of the FOV not covered by any masks and, ii) then estimating the spatial
721 background components \mathbf{b} , and then estimating A, C (with A restricted to be non-zero only at the
722 locations of the binary masks), using a simple NMF approach. Details are given in Alg. 2.

723 **Details of quality assessment tests**

724 Here we present the unsupervised and supervised quality assessment tests in more detail (Fig. 2).

725 **Matching spatial footprints to the raw data**

726 Let $\mathbf{a}_i, \mathbf{c}_i$ denote the spatial footprint and temporal trace of component i , and the let $A_{\setminus i}, C_{\setminus i}$ denote
727 the matrices A, C when the component i has been removed. Similarly, let $Y_i = Y - A_{\setminus i}C_{\setminus i} - B$ denote
728 the entire dataset when the background and the contribution of all components except i have been
729 removed. If component i is real then Y_i and $\mathbf{a}_i\mathbf{c}_i^T$ will look similar during the time intervals when
730 the component i is active. As a first test CALMAN finds the first N_p local peaks of c_i (e.g., $N_p = 5$),
731 constructs intervals around these peaks, (e.g., 50 ms in the past and 300ms in the future, to cover
732 the main part of a possible calcium transient around that point), and then averages Y_i across time
733 over the union of these intervals to obtain a spatial image $\langle Y_i \rangle$ (Fig. 2c). The Pearson's correlation
734 over space between $\langle Y_i \rangle$ and \mathbf{a}_i (both restricted on a small neighborhood around the centroid of
735 \mathbf{a}_i) is then computed, and component i is rejected if the correlation coefficient is below a threshold
736 value θ_{sp} , (e.g., $\theta_{sp} < 0.5$). Note that a similar test is used in the online approach of *Giovannucci et al.*
737 (2017) to accept for possible new components.

738 Detecting fluorescence traces with high SNR

739 For a candidate component to correspond to an active neuron its trace must exhibit dynamics
 740 reminiscent of the calcium indicator's transient. A criterion for this can be obtained by requiring
 741 the average SNR of trace \mathbf{c}_i over the course a transient to be above a certain threshold θ_{SNR} , e.g.,
 742 $\theta_{\text{SNR}} = 2$, (Fig. 2d). The average SNR is as a measure of how unlikely it is for the transients of \mathbf{c}_i (after
 743 some appropriate z-scoring) to have been a result of a white noise process.

744 To compute the SNR of a trace, let $R = Y - AC - B$ be the residual spatiotemporal signal. We
 745 can obtain the residual signal for each component i , \mathbf{r}_i , by projecting R into the spatial footprint \mathbf{a}_i :

$$\mathbf{r}_i = \frac{1}{\|\mathbf{a}_i\|^2} R^\top \mathbf{a}_i \quad (5)$$

746 Then the trace $\mathbf{c}_i + \mathbf{r}_i$ corresponds to the non-denoised trace of component i . To calculate its SNR
 747 we first compute a type of z-score:

$$\mathbf{z}_i = \frac{\mathbf{c}_i + \mathbf{r}_i - \text{BASELINE}(\mathbf{c}_i + \mathbf{r}_i)}{\text{NOISE}(\mathbf{c}_i + \mathbf{r}_i)} \quad (6)$$

748 The $\text{BASELINE}(\cdot)$ function determines the baseline of the trace, which can be varying in the case of
 749 long datasets exhibiting baseline trends, e.g., due to bleaching. The function $\text{NOISE}(\cdot)$ estimates
 750 the noise level of the trace. Since calcium transients around the baseline can only be positive, we
 751 estimate the noise level by restricting our attention only to the points t_n where $\mathbf{c}_i + \mathbf{r}_i$ is below the
 752 baseline value, i.e., $t_n = \{t : \mathbf{c}_i(t) + \mathbf{r}_i(t) \leq \text{BASELINE}(\mathbf{c}_i + \mathbf{r}_i)\}$, and compute the noise level as the scale
 753 parameter of a half-normal distribution (Fig. 2b):

$$\text{NOISE}(\mathbf{c}_i + \mathbf{r}_i) = \text{std}([\mathbf{c}_i + \mathbf{r}_i](t_n)) / \sqrt{1 - \frac{2}{\pi}} \quad (7)$$

754 We then determine how likely is that the positive excursions of \mathbf{z}_i can be attributed just to noise. We
 755 compute the probabilities $\mathbf{p}_i(t) = \Phi(-\mathbf{z}_i(t))$, where $\Phi(\cdot)$ denotes the cumulative distribution function
 756 of a standard normal distribution, and compute the most unlikely excursion over a window of N_s
 757 timesteps that corresponds to the length of a typical transient, e.g., $N_s = \lceil 0.4s \times F \rceil$, where 0.4s
 758 could correspond to the typical length of a GCaMP6f transient, and F is the imaging rate.

$$p_{\min}^i = \min_t \left(\prod_{j=0}^{N_s-1} \mathbf{p}_i(t+j) \right)^{1/N_s} \quad (8)$$

759 The (averaged peak) SNR of component i can then be defined as

$$\text{SNR}_i = \Phi^{-1}(1 - p_{\min}^i) = -\Phi^{-1}(p_{\min}^i), \quad (9)$$

760 where Φ^{-1} is the quantile function for the standard normal distribution (logit function) and a
 761 component is accepted if $\text{SNR}_i \geq \theta_{\text{SNR}}$. Note that for numerical stability we compute p_{\min}^i in the
 762 logarithmic domain and check the condition $p_{\min}^i \leq \Phi(-\theta_{\text{SNR}})$.

763 We can also use a similar test for the significance of the time traces in the spike domain after
 764 performing deconvolution. In this case, traces can be considered as spiking if the maximum height
 765 due to a spike transient is significantly larger than a threshold. If we assume that the shape of each
 766 calcium transient has been normalized to have maximum amplitude 1, then this corresponds to
 767 testing $\|\mathbf{s}_i\|_\infty \geq \theta_{\text{SNR}} \sigma_i$, where \mathbf{s}_i represents the deconvolved activity trace for component i , and θ_{SNR}
 768 is again an appropriate SNR threshold, e.g., $\theta_{\text{SNR}} = 2$, and σ_i is the noise level for trace i .

769 Classification through convolutional neural networks (CNNs)

770 The tests described above are unsupervised but require fine-tuning of two threshold parameters
 771 ($\theta_{sp}, \theta_{\text{SNR}}$) that might be dataset dependent and might be sensitive to strong non-stationarities. As a
 772 third test we trained a 4-layer CNN to classify the spatial footprints into true or false components,
 773 where a true component here corresponds to a spatial footprint that resembles the soma of a
 774 neuron (See Fig. 2e and section *Classification through convolutional networks* for details). A simple
 775 threshold θ_{CNN} can be used to tune the classifier (e.g., $\theta_{\text{CNN}} = 0.5$).

Table 2. Properties of manually annotated datasets. For each dataset the duration, imaging rate and calcium indicator are given, as well as the number of active neurons selected after consensus of the manual annotations.

Name	Area brain	Lab	Rate (Hz)	Size (TxXxY)	Indicator	# labelers	# neurons GT
NF.03.00.t	Hippocampus	Losonczy	7	2250x498x467	GCaMP6f	3	178
NF.04.00.t	Cortex	Harvey	7	3000x512x512	GCaMP6s	3	257
NF.02.00	Cortex	Svoboda	30	8000x512x512	GCaMP6s	4	394
NF.00.00	Cortex	Svoboda	7	2936x512x512	GCaMP6s	3	425
NF.01.01	Visual Cortex	Hausser	7	1825x512x512	GCaMP6s	4	333
YST	Visual Cortex	Yuste	10	3000x200x256	GCaMP3	4	405
K53	Parietal Cortex	Tank	30	116043x512x512	GCaMP6f	4	920
J115	Hippocampus	Tank	30	90000x463x472	GCaMP5	3	891
J123	Hippocampus	Tank	30	41000x458x477	GCaMP5	4	183

776 **Collection of manual annotations and ground truth**

777 We collected manual annotations from four independent labelers who were instructed to find
778 round or donut shaped neurons of similar size using the ImageJ Cell Magic Wand tool *Walker (2014)*.
779 We focused on manually annotating only cells that were active within each dataset and for that
780 reason the labelers were provided with two summary statistics: i) A movie obtained by removing a
781 running 20th percentile (as a crude background approximation) and downsampling in time by a
782 factor of 10, and ii) the max-correlation image. The correlation image (CI) at every pixel is equal
783 to the average temporal correlation coefficient between that pixel and its neighbors *Smith and*
784 *Häusser (2010)* (8 neighbors were used for our analysis). The max-correlation image is obtained
785 by computing the CI for each batch of 33 seconds (1000 frames for a 30Hz acquisition rate), and
786 then taking the maximum over all these images. Neurons that are inactive during the course of the
787 dataset will be suppressed both from the baseline removed video (since their activity will always be
788 around their baseline), and from the max-correlation image since the variation around this baseline
789 will mostly be due to noise leading to practically uncorrelated neighboring pixels. 9 different mouse
790 in vivo datasets were used from various brain areas and labs. A description is given in Table 2. To
791 create the consensus ground truth, the labelers were asked to jointly resolve the inconsistencies
792 with each others annotations.

793 The annotation procedure provides a binary mask per selected component. On the other
794 hand, the output of CALMAN for each component is a non-negatively valued vector over the FOV
795 (a real-valued mask). The two sets of masks differ not only in their variable type but also in their
796 general shape: Manual annotation through the Cell Magic Wand tool tends to produce circular
797 shapes, whereas the output of CALMAN will try to accurately estimate the shape of each active
798 component. To construct ground truth that can be directly used for comparison, the binary masks
799 from the manual annotations were used to seed the CNMF algorithm (Alg. 2). This produced a set
800 of ground truth real valued components with spatial footprints restricted to the areas provided by
801 the annotations, and a corresponding set of temporal components that can be used to evaluate
802 the performance of CALMAN (Fig. 4). Registration was performed using the REGISTERPAIR algorithm
803 (Alg. 5) and match was counted as a true positive when the (modified) Jaccard distance (Eq. 11) was
804 below 0.7. Details of the registration procedure are given below (see *Component registration*).

805 **Classification through convolutional neural networks (CNNs)**

806 CALMAN uses two CNN classifiers; one for post processing component screening in CALMAN BATCH,
807 and a different one for screening candidate components in CALMAN ONLINE. In both cases a 4 layer
808 CNN was used, with architecture as described in Fig. 2e.

809 CALMAN BATCH classifier for post processing classification

810 The purpose of the batch classifier is to classify the components detected by CALMAN BATCH into
811 neuron somas or other shapes, by examining their spatial footprints. Only three annotated datasets
812 (.03.00.t, NF.04.00.t, NF.02.00) were used to train the batch classifier. The set of estimated
813 footprints from running CALMAN BATCH initialized with the consensus ground truth was matched
814 to the set of ground truth footprints. Footprints matched to ground truth components were
815 considered positive examples, whereas the remaining components were labeled as negatives. The
816 two sets were enriched using data augmentation (rotations, reflections, contrast manipulation etc.)
817 through the Keras library (keras.io) and the CNN was trained on 60% of the data, leaving 20% for
818 validation and 20% for testing. The CNN classifier reached an accuracy of 97% on test data; that
819 also generalized to the rest of the datasets (Fig. 2e) without any parameter change.

820 Online classifier for new component detection

821 The purpose of the CALMAN ONLINE classifier is to detect new components based on their spatial
822 footprints by looking at the mean across time of the residual buffer. To construct the ground truth
823 data for the online classifier, CALMAN BATCH was run on the first five annotated datasets seeded
824 with the masks obtained through the manual annotations. Subsequently the activity of random
825 subsets of found components and the background was removed from contiguous frames of the
826 raw datasets to construct residual buffers, which were averaged across time. From the resulting
827 images patches were extracted corresponding to positive examples (patches around a neuron that
828 was active during the buffer) and negative examples (patches around other positions within the
829 FOV). A neuron was considered active if its trace attained an average peak-SNR value of 4 or higher
830 during the buffer interval. Similarly to the batch classifier, the two sets were augmented and split
831 into training, validation and testing sets. The resulting classifier reached a 98% accuracy on the
832 testing set, and also generalized well when applied to different datasets.

833 Differences between the two classifiers

834 Although both classifiers examine the spatial footprints of candidate components, their required
835 performance characteristics are different which led us to train them separately. The batch classifier
836 examines each component as a post-processing step to determine whether its shape corresponds
837 to a neural cell body. As such, false positive and false negative examples are treated equally
838 and possible mis-classifications do not directly affect the traces of the other components. By
839 contrast, the online classifier operates as part of the online processing pipeline. In this case, a new
840 component that is not detected in a residual buffer is likely to be detected later should it become
841 more active. On the other hand, a component that is falsely detected and incorporated in the online
842 processing pipeline will continue to affect the future buffer residuals and the detection of future
843 components. As such the online algorithm is more sensitive to false positives than false negatives.
844 To ensure a small number of false positive examples under testing conditions, only components
845 with average peak-SNR value at least 4 were considered as positive examples during training of the
846 online classifier.

847 Component registration

848 Fluorescence microscopy methods enable imaging the same part of the brain across different
849 sessions that can span multiple days or weeks. While the microscope can visit the same location
850 in the brain with reasonably high precision, the FOV might not precisely match due to
851 misalignments or deformations in the brain medium. CALMAN provides routines for FOV alignment
852 and component registration across multiple sessions/days. Let $\mathbf{a}_1^1, \mathbf{a}_2^1, \dots, \mathbf{a}_{N_1}^1$ and $\mathbf{a}_1^2, \mathbf{a}_2^2, \dots, \mathbf{a}_{N_2}^2$ the
853 sets of spatial components from sessions 1 and 2 respectively, where N_1 and N_2 denote the total
854 number of components from each session. We first compute the FOV displacement by aligning
855 some summary images from the two sessions (e.g., mean or correlation image), using some non-
856 rigid registration method, e.g., NoRMCorre (*Pnevmatikakis and Giovannucci, 2017*). We apply the

857 estimated displacement field to the components of A_1 to align them with the FOV of session 2. To
 858 perform the registration, we construct a pairwise distance matrix $D \in \mathbb{R}^{N_1 \times N_2}$ with $D(i, j) = d(\mathbf{a}_i^1, \mathbf{a}_j^2)$,
 859 where $d(\cdot, \cdot)$ denotes a distance metric between two components. The chosen distance corresponds
 860 to the Jaccard distance between the binarized versions of the components. A real valued component
 861 \mathbf{a} is converted into its binary version $m(\mathbf{x})$ by setting to 1 only the values of \mathbf{a} that are above the
 862 maximum value of \mathbf{a} times a threshold θ_b , e.g., $\theta_b = 0.2$:

$$m(\mathbf{a})(x) = \begin{cases} 1, & \mathbf{a}(x) \geq \theta_b \|\mathbf{a}\|_\infty \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

863 To compute the distance between two binary masks m_1, m_2 , we use the Jaccard index (intersection
 864 over union) which is defined as

$$J(m_1, m_2) = \frac{|m_1 \cap m_2|}{|m_1 \cup m_2|}, \quad (11)$$

865 and use it to define the distance metric as

$$d(\mathbf{a}_i^1, \mathbf{a}_j^2) = \begin{cases} 1 - J(m(\mathbf{a}_i^1), m(\mathbf{a}_j^2)), & 1 - J(m(\mathbf{a}_i^1), m(\mathbf{a}_j^2)) \leq \theta_d \\ 0, & (m(\mathbf{a}_i^1) \subseteq m(\mathbf{a}_j^2)) \text{ OR } (m(\mathbf{a}_j^2) \subseteq m(\mathbf{a}_i^1)) \\ \infty, & \text{otherwise.} \end{cases} \quad (12)$$

866 where θ_d is a distance threshold, e.g., 0.5 above which two components are considered non-
 867 matching and their distance is set to infinity to prevent false assignments.

868 After the distance matrix D has been completed, an optimal matching between the components
 869 of the two sessions is computed using the Hungarian algorithm to solve the linear assignment
 870 problem. As infinite distances are allowed, it is possible to have components from both sessions
 871 that are not matched with any other component. This process of registering components across
 872 two sessions (REGISTERPAIR) is summarized in Alg. 5.

873 To register components across multiple sessions, we first order the sessions chronologically
 874 and register session 1 against session 2. From this registration we construct the union of the
 875 distinct components between the two sessions by keeping the matched components from session
 876 2 as well as the non-matched components from both sessions aligned to the FOV of session
 877 2. We then register this union of components to the components of session 3 and repeat the
 878 procedure until all sessions are have been registered. This process of multi session registration
 879 (REGISTERMULTI) is summarized in Alg. 6. At the end of the process the algorithm produces a list of
 880 matches between the components of each session and the union of all active distinct components,
 881 allowing for efficient tracking of components across multiple days (Fig. 9), and the comparison
 882 of non-consecutive sessions through the union without the need of direct pairwise registration
 883 (Fig. 10)). An alternative approach to the problem of multiple session registration (CELLREG) was
 884 presented recently by *Sheintuch et al. (2017)* where the authors register neurons across multiple
 885 days by first constructing a similar union set of all the components which is then refined using a
 886 clustering procedure. REGISTERMULTI differs from the CELLREG method of *Sheintuch et al. (2017)* in
 887 a few key ways, that highlight its simplicity and robustness:

- 888 • REGISTERMULTI uses a very simple intersection over union metric to estimate the distance
 889 between two neighboring neurons after the FOV alignment. Cells that have a distance above
 890 a given threshold are considered different by default and are not tested for matching. This
 891 parameter is intuitive to set a priori for each dataset. In contrast CELLREG uses a probabilistic
 892 framework based on the joint probability distribution between the distance of two cells
 893 and the correlation of their shapes that makes specific parametric assumptions about the
 894 distributions of centroid distances between the same and different cells, as well as their shape
 895 correlations. This model needs to be re-evaluated for every different set of sessions to be
 896 registered and potentially requires a lot of data to learn the appropriate distance metric.

897 • REGISTERMULTI uses the Hungarian algorithm to register two different set of components, a
898 practice that solves the linear assignment problem optimally under the assumed distance
899 function. In contrast CELLREG uses a greedy method for initializing the assignment of cells to
900 the union superset relying on the following clustering step to refine these estimates, and thus
901 adding extra computational burden to the registration procedure.

902 **Implementation details for CALMAN BATCH**

903 Each dataset was processed using the same set of parameters, excepting the expected size of
904 neurons (estimated by inspecting the correlation image), the size of patches and expected number
905 of neurons per patch (estimated by inspecting the correlation image). For the dataset N.01.01,
906 where optical modulation was induced, the threshold for merging neurons was slightly higher (the
907 stimulation caused clustered synchronous activity). For shorter datasets, rigid motion correction
908 was sufficient; for longer datasets K53, J115 we applied non-rigid motion correction. The parameters
909 for the automatic selection of components were optimized using only the first three datasets and
910 fixed for all the remaining files. For all datasets the background neuropil activity was modeled as a
911 rank 2 matrix, and calcium dynamics were modeled as a first order autoregressive process. The
912 remaining parameters were optimized so that all the datasets could be run on a machine with less
913 than 128GB RAM.

914 **Implementation details for CALMAN ONLINE**

915 Datasets were processed for two epochs with the exception of the longer datasets K53, J115, J123
916 where only one pass of the data was performed to limit computational cost. For each dataset
917 the online CNN classifier was used to detect new neurons, and five candidate components were
918 considered for each frame. The online CNN classifier had the same threshold 0.5 for all datasets,
919 with the exception of the longest datasets J115, J123 where the threshold was set to 0.75. Setting
920 the threshold to 0.5 for these datasets led to slightly poorer performance. Large datasets were
921 spatially decimated by a factor of 2 to enhance processing speed, a step that did not lead to changes
922 in detection performance. For all datasets the background neuropil activity was modeled as a rank
923 2 matrix, and calcium dynamics were modeled as a first order autoregressive process. For each
924 dataset, CALMAN ONLINE was initialized on the first 200 frames, using the BAREINITIALIZATION on
925 the entire FOV with only 2 neurons, so in practice all the neurons were detected during the online
926 mode. To highlight the *truly* online processing mode, no post-processing of the results was used, a
927 step that can further enhance the performance of the algorithm. Similarly to batch processing, the
928 expected size of neurons was chosen separately for each dataset after inspecting the correlation
929 image.

930 For the analysis of the whole brain zebrafish dataset, CALMAN ONLINE was run for 1 epoch with
931 the same parameters as above, with only differences appearing in the number of neurons during
932 initialization (600 vs 2), and the value of the threshold for the online CNN classifier (0.75 vs 0.5).
933 The former decision was motivated by the goal of retrieving with a single pass neurons from a
934 preparation with a denser level of activity over a larger FOV in this short dataset (1885 frames).
935 To this end, the number of candidate neurons at each timestep was set to 10 (per plane). The
936 threshold choice was motivated by the fact that the classifier was trained on mouse data only, and
937 thus a higher threshold choice would help diminish potential false positive components. Rigid
938 motion correction was applied online to each plane.

939 **Performance quantification as a function of SNR**

940 To quantify performance as a function of SNR we approximate the ground truth traces by running
941 CALMAN BATCH on the datasets seeded with the "consensus" binary masks obtained from the manual
942 annotators. After that the average peak-SNR of a trace \mathbf{c} with corresponding residual signal \mathbf{r} (5) is
943 obtained as

$$\text{SNR}(\mathbf{z}) = -\Phi^{-1}(p_{\min}), \quad (13)$$

944 where $\Phi^{-1}(\cdot)$ denotes the probit function (quantile function for the standard Gaussian distribution),
 945 \mathbf{z} is the z-scored version of $\mathbf{c} + \mathbf{r}$ (6) and p_{\min} is given by (8).

946 Let $c_1^{\text{gt}}, c_2^{\text{gt}}, \dots, c_N^{\text{gt}}$ be the ground truth traces and $c_1^{\text{cm}}, c_2^{\text{cm}}, \dots, c_N^{\text{cm}}$ be their corresponding CALMAN
 947 inferred traces. Here we assume that false positive and false negative components are matched with
 948 trivial components that have 0 SNR. Let also $\text{SNR}_i^{\text{gt}} = \text{SNR}(c_i^{\text{gt}})$ and $\text{SNR}_i^{\text{cm}} = \text{SNR}(c_i^{\text{cm}})$, respectively.
 949 After we compute the SNR for both ground truth and inferred traces the performance algorithm
 950 can be quantified in multiple ways as a function of a SNR thresholds θ_{SNR} :

Precision: Precision at level θ_{SNR} , can be computed as the fraction of detected components with
 $\text{SNR}^{\text{cm}} > \theta_{\text{SNR}}$ that are matched with ground truth components. It quantifies the certainty that
 a component detected with a given SNR or above corresponds to a true component.

$$\text{PREC}(\theta_{\text{SNR}}) = \frac{|\{i : (\text{SNR}_i^{\text{cm}} > \theta_{\text{SNR}}) \ \& \ (\text{SNR}_i^{\text{gt}} > 0)\}|}{|\{i : (\text{SNR}_i^{\text{cm}} > \theta_{\text{SNR}})\}|}$$

Recall: Recall at level θ_{SNR} , can be computed as the fraction of ground truth components with
 $\text{SNR}^{\text{gt}} > \theta_{\text{SNR}}$ that are detected by the algorithm. It quantifies the certainty that a ground truth
 component with a given SNR or above is detected.

$$\text{RECALL}(\theta_{\text{SNR}}) = \frac{|\{i : (\text{SNR}_i^{\text{gt}} > \theta_{\text{SNR}}) \ \& \ (\text{SNR}_i^{\text{cm}} > 0)\}|}{|\{i : (\text{SNR}_i^{\text{gt}} > \theta_{\text{SNR}})\}|}$$

F_1 score: An overall F_1 score at level θ_{SNR} , can be obtained by computing the harmonic mean
 between precision and recall

$$F_1(\theta_{\text{SNR}}) = 2 \frac{\text{PREC}(\theta_{\text{SNR}}) \times \text{RECALL}(\theta_{\text{SNR}})}{\text{PREC}(\theta_{\text{SNR}}) + \text{RECALL}(\theta_{\text{SNR}})}$$

951 The cautious reader will observe that the precision and recall quantities described above are
 952 not computed in the same set of components. This can be remedied by recomputing the quantities
 953 in two different ways:

AND framework: Here we consider a match only if *both* traces have SNR above the given threshold:

$$\text{PREC}_{\text{AND}}(\theta_{\text{SNR}}) = \frac{|\{i : (\text{SNR}_i^{\text{cm}} > \theta_{\text{SNR}}) \ \& \ (\text{SNR}_i^{\text{gt}} > \theta_{\text{SNR}})\}|}{|\{i : (\text{SNR}_i^{\text{cm}} > \theta_{\text{SNR}})\}|}$$

$$\text{RECALL}_{\text{AND}}(\theta_{\text{SNR}}) = \frac{|\{i : (\text{SNR}_i^{\text{gt}} > \theta_{\text{SNR}}) \ \& \ (\text{SNR}_i^{\text{cm}} > \theta_{\text{SNR}})\}|}{|\{i : (\text{SNR}_i^{\text{gt}} > \theta_{\text{SNR}})\}|}$$

OR framework: Here we consider a match if *either* trace has SNR above the given threshold and
 its match has SNR above 0.

$$\text{RECALL}_{\text{OR}}(\theta_{\text{SNR}}) = \frac{|\{i : (\max(\text{SNR}_i^{\text{gt}}, \text{SNR}_i^{\text{cm}}) > \theta_{\text{SNR}}) \ \& \ (\min(\text{SNR}_i^{\text{gt}}, \text{SNR}_i^{\text{cm}}) > 0)\}|}{|\{i : (\text{SNR}_i^{\text{cm}} > 0)\}|}$$

$$\text{PREC}_{\text{OR}}(\theta_{\text{SNR}}) = \frac{|\{i : (\max(\text{SNR}_i^{\text{gt}}, \text{SNR}_i^{\text{cm}}) > \theta_{\text{SNR}}) \ \& \ (\min(\text{SNR}_i^{\text{gt}}, \text{SNR}_i^{\text{cm}}) > 0)\}|}{|\{i : (\text{SNR}_i^{\text{gt}} > 0)\}|}$$

It is easy to show that

$$\begin{aligned} \text{PREC}_{\text{AND}}(\theta_{\text{SNR}}) &\leq \text{PREC}(\theta_{\text{SNR}}) \leq \text{PREC}_{\text{OR}}(\theta_{\text{SNR}}) \\ \text{RECALL}_{\text{AND}}(\theta_{\text{SNR}}) &\leq \text{RECALL}(\theta_{\text{SNR}}) \leq \text{RECALL}_{\text{OR}}(\theta_{\text{SNR}}) \\ F_{1\text{AND}}(\theta_{\text{SNR}}) &\leq F_1(\theta_{\text{SNR}}) \leq F_{1\text{OR}}(\theta_{\text{SNR}}), \end{aligned}$$

954 with equality holding for $\theta_{\text{SNR}} = 0$. As demonstrated in Fig. 4d, these bounds are tight.

955 **Additional features of CALMAN**

956 CALMAN contains a number of additional features that are not presented in the results section for
 957 reasons of brevity. These include:

958 Volumetric data processing

959 Apart from planar 2D data, CALMAN BATCH is also applicable to 3D volumetric data arising either
 960 from dense raster scanning methods, or from direct volume imaging methods such as light field
 961 microscopy (*Prevedel et al., 2014; Grosenick et al., 2017*).

962 Segmentation of structural indicator data

963 Structural indicators expressed in the nucleus and functional indicators expressed in the cytoplasm
 964 can facilitate source extraction and help identify silent or specific subpopulations of neurons
 965 (e.g., inhibitory). CALMAN provides a simple adaptive thresholding filtering method for segmenting
 966 summary images of the structural channel (e.g., mean image). The obtained results can be used
 967 for seeding source extraction from the functional channel in CALMAN BATCH or CALMAN ONLINE as
 968 already discussed.

969 Duplicate Detection

970 The ground truth obtained through the consensus process was screened for possible duplicate
 971 selections. To detect for duplicate components we define the degree of spatial overlap matrix O as

$$O_{ij} = \begin{cases} 0, & i = j \\ \frac{|m(\mathbf{a}^i) \cap m(\mathbf{a}^j)|}{|m(\mathbf{a}^j)|}, & i \neq j \end{cases}, \quad (14)$$

972 that defines the fraction of component i that overlap with component j , where $m(\cdot)$ is the thresh-
 973 olding function defined in (10). Any entry of O that is above a threshold θ_o (e.g., $\theta_o = 0.7$ used
 974 here) indicates a pair of duplicate components. To decide which of the two components should be
 975 removed, we use predictions of the CALMAN BATCH CNN classifier, removing the component with the
 976 lowest score.

977 Extraction of $\Delta F/F$

978 The fluorescence trace \mathbf{f}_i of component i can be written as

$$\mathbf{f}_i = \|\mathbf{a}_i\|^2(\mathbf{c}_i + \mathbf{r}_i). \quad (15)$$

979 The fluorescence due to the component's transients overlaps with a background fluorescence due
 980 to baseline fluorescence of the component and neuropil activity, that can be expressed as

$$\mathbf{f}_{0,i} = \text{BASELINE}(\mathbf{f}_i + \mathbf{B}^T \mathbf{a}_i), \quad (16)$$

981 where $\text{BASELINE} : \mathbb{R}^T \mapsto \mathbb{R}^T$ is a baseline extraction function, and \mathbf{B} is the estimated background
 982 signal. Examples of the baseline extraction function are a percentile function (e.g., 10th percentile),
 983 or a for longer traces, a running percentile function, e.g., 10th percentile over a window of a hundred
 984 seconds⁶. To determine the optimal percentile level an empirical histogram of the trace (or parts of
 985 it in case of long traces) is computed using a diffusion kernel density estimator (*Botev et al., 2010*),
 986 and the mode of this density is used to define the baseline and its corresponding percentile level.
 987 The $\Delta F/F$ activity of component i can then be written as

$$\mathbf{c}_i^{\Delta F/F} = \frac{\mathbf{f}_i - \text{BASELINE}(\mathbf{f}_i)}{\mathbf{f}_{0,i}} \quad (17)$$

988 The approach we propose here is conceptually similar to practical approaches where the $\Delta F/F$ is
 989 computed by averaging over the spatial extent of an ROI (*Jia et al., 2011*) with some differences:
 990 i) instead of averaging with a binary mask we use the a weighed average with the shape of each
 991 component, ii) signal due to overlapping components is removed from the calculation of the

⁶Computing the exact running percentile function can be computationally intensive. To reduce the complexity we compute the running percentile with a stride of W , where W is equal or smaller to the length of the window, and then linearly interpolate the values.

992 background fluorescence, and iii) the traces have been extracted through the CNMF process prior
993 to the $\Delta F/F$ extraction. Note that the same approach can also be performed to the trace $\|a_i\|_2^2 c_i$,
994 that does not include the residual traces for each component. In practice it can be beneficial to
995 extract $\Delta F/F$ traces prior to deconvolution, since the $\Delta F/F$ transformation can alleviate the effects
996 of drifting baselines, e.g., due to bleaching. For the non-deconvolved traces f_i some temporal
997 smoothing can also be applied to obtain more smooth $\Delta F/F$ traces.

998 Acknowledgments

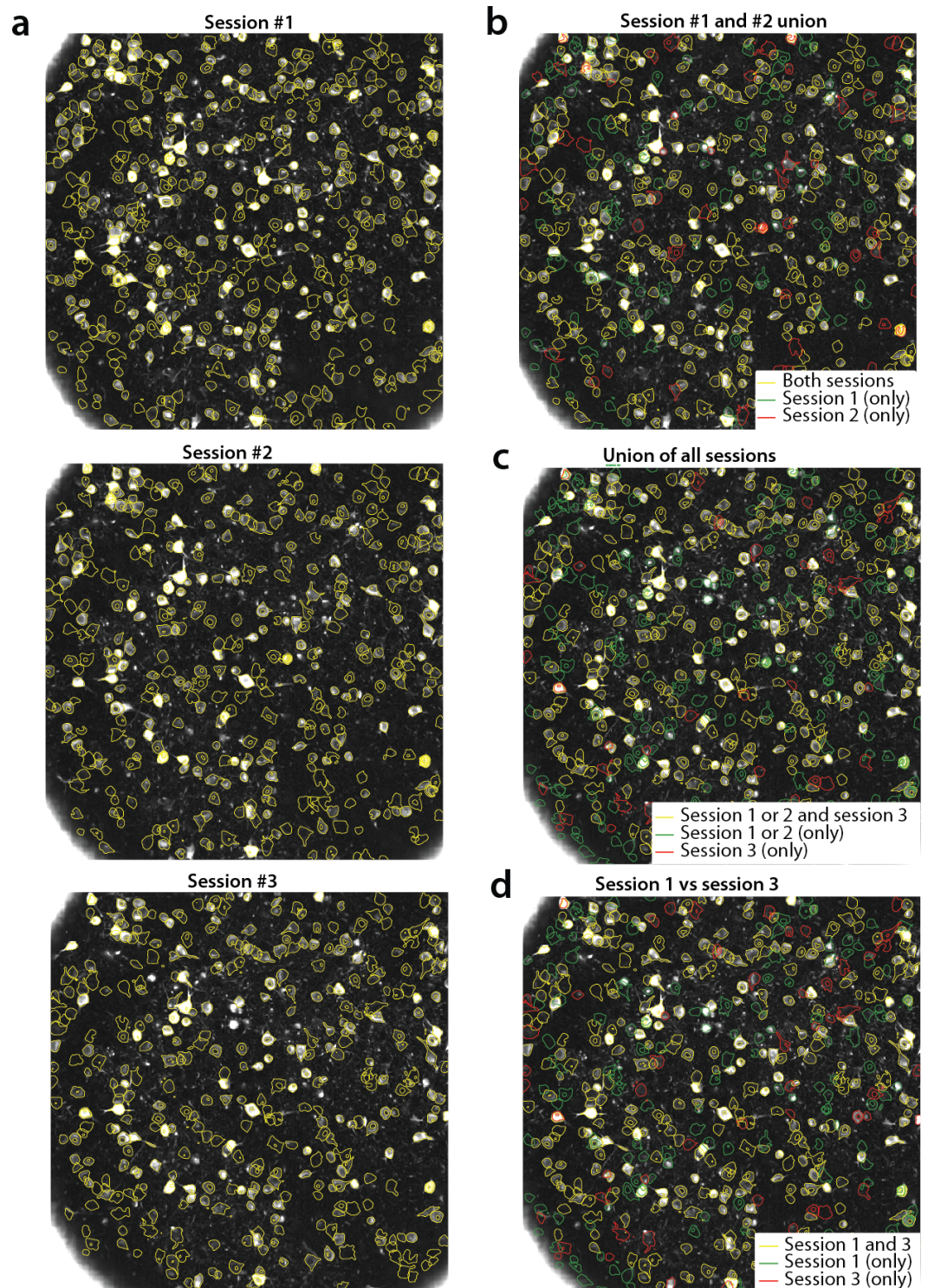
999 We thank B. Cohen, L. Myers, N. Roumelioti, and S. Villani for providing us with manual annotations.
1000 We thank V. Staneva and B. Deverett for contributing to the early stages of CALMAN, and L. Paninski
1001 for numerous useful discussions. We thank N. Carriero, I. Fisk, and D. Simon from the Flatiron
1002 Institute (Simons Foundation) for useful discussions and suggestions to optimize High Performance
1003 Computing code. We thank T. Kawashima and M. Ahrens for sharing the whole brain zebrafish
1004 dataset. Last but not least, we thank the active community of CALMAN users for their great help in
1005 terms of code/method contributions, bug reporting, code testing and suggestions that have led
1006 to the growth of CALMAN into a widely used open source package. A partial list of contributors
1007 (in the form of GitHub usernames) can be found in <https://github.com/flatironinstitute/CalmAn/graphs/contributors> (Python) and <https://github.com/flatironinstitute/CalmAn-MATLAB/graphs/contributors> (MATLAB®). The authors acknowledge support from following funding sources: A.G.,
1010 E.A.P., J.F., P.G. (Simons Foundation), J.G., S.A.K., D.W.T. (NIH NRSA F32NS077840-01, 5U01NS090541,
1011 1U19NS104648 and Simons Foundation SCGB), P.Z. (NIH NIBIB R01EB022913, NSF NeuroNex DBI-
1012 1707398, Gatsby Foundation), J.T. (NIH R01-MH101198), F.N. (MURI, Simons Collaboration on the
1013 Global Brain and Pew Foundation).

1014 References

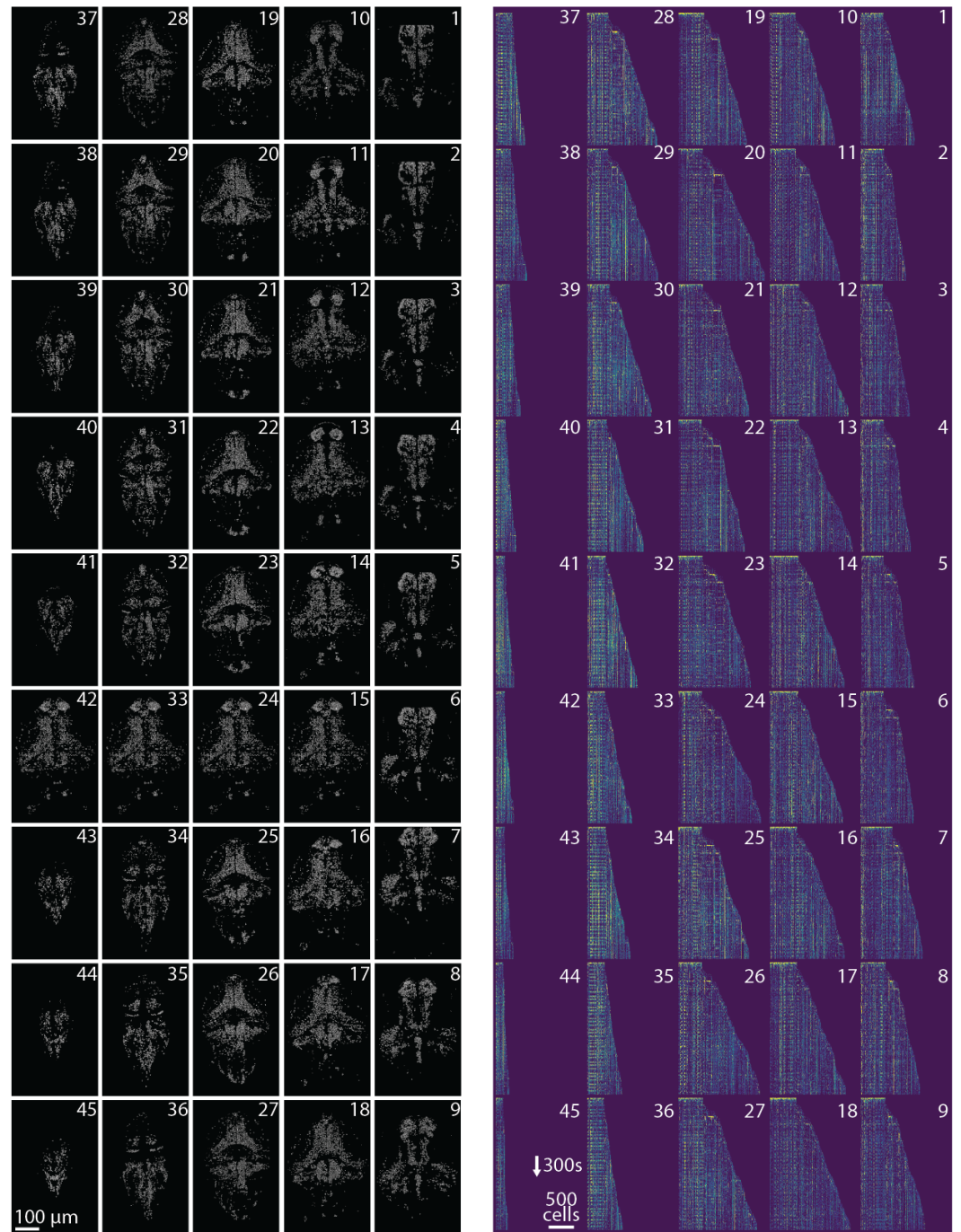
- 1015 **Ahrens MB**, Orger MB, Robson DN, Li JM, Keller PJ. Whole-brain functional imaging at cellular resolution using
1016 light-sheet microscopy. *Nature methods*. 2013; 10(5):413–420.
- 1017 **Apthorpe N**, Riordan A, Aguilar R, Homann J, Gu Y, Tank D, Seung HS. Automatic Neuron Detection in Calcium
1018 Imaging Data Using Convolutional Networks. In: *Advances in Neural Information Processing Systems*; 2016. p.
1019 3270–3278.
- 1020 **Berens P**, Freeman J, Deneux T, Chenkov N, McColgan T, Speiser A, Macke JH, Turaga S, Mineault P, Rupprecht
1021 P, Gerhard S, Friedrich RW, Friedrich J, Paninski L, Pachitariu M, Harris KD, Bolte B, Machado TA, Ringach
1022 D, Reimer J, et al. Community-based benchmarking improves spike inference from two-photon calcium
1023 imaging data. *bioRxiv*. 2017 Aug; p. 177956. <https://www.biorxiv.org/content/early/2017/08/18/177956>, doi:
1024 10.1101/177956.
- 1025 **Botev ZI**, Grotowski JF, Kroese DP. Kernel density estimation via diffusion. *The Annals of Statistics*. 2010;
1026 38(5):2916–2957.
- 1027 **Bouchard MB**, Voleti V, Mendes CS, Lacefield C, Grueber WB, Mann RS, Bruno RM, Hillman EM. Swept confocally-
1028 aligned planar excitation (SCAPE) microscopy for high-speed volumetric imaging of behaving organisms.
1029 *Nature photonics*. 2015; 9(2):113–119.
- 1030 **Bradski G**. The OpenCV Library. *Dr Dobb's Journal: Software Tools for the Professional Programmer*. 2000;
1031 25(11):120–123.
- 1032 **Cai DJ**, Aharoni D, Shuman T, Shobe J, Biane J, Song W, Wei B, Veshkini M, La-Vu M, Lou J, et al. A shared neural
1033 ensemble links distinct contextual memories encoded close in time. *Nature*. 2016; 534(7605):115.
- 1034 **Carrillo-Reid L**, Yang W, Kang Miller Je, Peterka DS, Yuste R. Imaging and Optically Manipulating Neuronal
1035 Ensembles. *Annual review of biophysics*. 2017; 46:271–293.
- 1036 **Chen TW**, Wardill TJ, Sun Y, Pulver SR, Renninger SL, Baohan A, Schreiter ER, Kerr RA, Orger MB, Jayaraman V,
1037 Looger LL, Svoboda K, Kim DS. Ultrasensitive fluorescent proteins for imaging neuronal activity. *Nature*. 2013
1038 Jul; 499(7458). <https://www.nature.com/articles/nature12354>, doi: 10.1038/nature12354.

- 1039 **Cichocki A**, Zdunek R, Amari Si. Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization.
1040 In: *International Conference on Independent Component Analysis and Signal Separation* Springer; 2007. p.
1041 169–176.
- 1042 **Dean J**, Ghemawat S. MapReduce: simplified data processing on large clusters. *Communications of the ACM*.
1043 2008; 51(1):107–113.
- 1044 **Deneux T**, Kaszas A, Szalay G, Katona G, Lakner T, Grinvald A, Rózsa B, Vanzetta I. Accurate spike estimation
1045 from noisy calcium signals for ultrafast three-dimensional imaging of large neuronal populations in vivo.
1046 *Nature communications*. 2016; 7:12190.
- 1047 **Flusberg BA**, Nimmerjahn A, Cocker ED, Mukamel EA, Barretto RP, Ko TH, Burns LD, Jung JC, Schnitzer MJ.
1048 High-speed, miniaturized fluorescence microscopy in freely moving mice. *Nature methods*. 2008; 5(11):935.
- 1049 **Freeman J**, Vladimirov N, Kawashima T, Mu Y, Sofroniew NJ, Bennett DV, Rosen J, Yang CT, Looger LL, Ahrens MB.
1050 Mapping brain activity at scale with cluster computing. *Nature methods*. 2014; 11(9):941–950.
- 1051 **Friedrich J**, Yang W, Soudry D, Mu Y, Ahrens MB, Yuste R, Peterka DS, Paninski L. Multi-scale approaches for high-
1052 speed imaging and analysis of large neural populations. *PLoS computational biology*. 2017; 13(8):e1005685.
- 1053 **Friedrich J**, Zhou P, Paninski L. Fast online deconvolution of calcium imaging data. *PLOS Computational Biology*.
1054 2017; 13(3):e1005423.
- 1055 **Giovannucci A**, Friedrich J, Kaufman J, Churchland A, Chklovskii D, Paninski L, Pnevmatikakis EA. OnACID: Online
1056 analysis of calcium imaging data in real time. In: *Neural Information Processing Systems (NIPS)*; 2017. .
- 1057 **Grosenick LM**, Broxton M, Kim CK, Liston C, Poole B, Yang S, Andalman AS, Scharff E, Cohen N, Yizhar O,
1058 Ramakrishnan C, Ganguli S, Suppes P, Levoy M, Deisseroth K. Identification Of Cellular-Activity Dynamics
1059 Across Large Tissue Volumes In The Mammalian Brain. *bioRxiv*. 2017 May; p. 132688. [https://www.biorxiv.](https://www.biorxiv.org/content/early/2017/05/01/132688)
1060 [org/content/early/2017/05/01/132688](https://www.biorxiv.org/content/early/2017/05/01/132688), doi: 10.1101/132688.
- 1061 **Jia H**, Rochefort NL, Chen X, Konnerth A. In vivo two-photon imaging of sensory-evoked dendritic calcium signals
1062 in cortical neurons. *Nature protocols*. 2011; 6(1):28.
- 1063 **Kaifosh P**, Zaremba JD, Danielson NB, Losonczy A. SIMA: Python software for analysis of dynamic fluorescence
1064 imaging data. *Frontiers in neuroinformatics*. 2014; 8:80.
- 1065 **Kawashima T**, Zwart MF, Yang CT, Mensh BD, Ahrens MB. The serotonergic system tracks the outcomes of
1066 actions to mediate short-term motor learning. *Cell*. 2016; 167(4):933–946.
- 1067 **Klibisz A**, Rose D, Eicholtz M, Blundon J, Zakharenko S. Fast, Simple Calcium Imaging Segmentation with Fully
1068 Convolutional Networks. In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical*
1069 *Decision Support* Springer; 2017.p. 285–293.
- 1070 **Mairal J**, Bach F, Ponce J, Sapiro G. Online learning for matrix factorization and sparse coding. *Journal of*
1071 *Machine Learning Research*. 2010; 11(Jan):19–60.
- 1072 **Mukamel EA**, Nimmerjahn A, Schnitzer MJ. Automated analysis of cellular signals from large-scale calcium
1073 imaging data. *Neuron*. 2009; 63(6):747–760.
- 1074 **Pachitariu M**, Packer AM, Pettit N, Dalgleish H, Häusser M, Sahani M. Extracting regions of interest from
1075 biological images with convolutional sparse block coding. In: *Advances in Neural Information Processing*
1076 *Systems*; 2013. p. 1745–1753.
- 1077 **Pachitariu M**, Stringer C, Dipoppa M, Schröder S, Rossi LF, Dalgleish H, Carandini M, Harris KD. Suite2p: beyond
1078 10,000 neurons with standard two-photon microscopy. *BioRxiv*. 2017; p. 061507.
- 1079 **Packer AM**, Russell LE, Dalgleish HW, Häusser M. Simultaneous all-optical manipulation and recording of neural
1080 circuit activity with cellular resolution in vivo. *Nature Methods*. 2015; 12(2):140–146.
- 1081 **Pedregosa F**, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg
1082 V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay É. Scikit-learn: Machine Learning in
1083 Python. *Journal of Machine Learning Research*. 2011; 12(Oct):2825–2830. [http://www.jmlr.org/papers/v12/](http://www.jmlr.org/papers/v12/pedregosa11a.html)
1084 [pedregosa11a.html](http://www.jmlr.org/papers/v12/pedregosa11a.html).
- 1085 **Petersen A**, Simon N, Witten D. SCALPEL: Extracting Neurons from Calcium Imaging Data. *arXiv preprint*
1086 *arXiv:170306946*. 2017; .

- 1087 **Piatkevich KD**, Jung EE, Straub C, Linghu C, Park D, Suk HJ, Hochbaum DR, Goodwin D, Pnevmatikakis E, Pak N,
1088 Kawashima T, Yang CT, Rhoades JL, Shemesh O, Asano S, Yoon YG, Freifeld L, Saulnier JL, Riegler C, Engert
1089 F, et al. A robotic multidimensional directed evolution approach applied to fluorescent voltage reporters.
1090 *Nature Chemical Biology*. 2018 Apr; 14(4):352–360. <https://www.nature.com/articles/s41589-018-0004-9>, doi:
1091 10.1038/s41589-018-0004-9.
- 1092 **Pnevmatikakis EA**, Giovannucci A. NoRMCorre: An online algorithm for piecewise rigid motion correction of
1093 calcium imaging data. *Journal of Neuroscience Methods*. 2017; 291:83–94.
- 1094 **Pnevmatikakis EA**, Merel J, Pakman A, Paninski L. Bayesian spike inference from calcium imaging data. In:
1095 *Signals, Systems and Computers, 2013 Asilomar Conference on IEEE*; 2013. p. 349–353.
- 1096 **Pnevmatikakis EA**, Soudry D, Gao Y, Machado TA, Merel J, Pfau D, Reardon T, Mu Y, Lacefield C, Yang W,
1097 Ahrens M, Bruno R, Jessell TM, Peterka DS, Yuste R, Paninski L. Simultaneous Denoising, Deconvolution, and
1098 Demixing of Calcium Imaging Data. *Neuron*. 2016 Jan; 89(2):285–299. [https://www.cell.com/neuron/abstract/](https://www.cell.com/neuron/abstract/S0896-6273(15)01084-3)
1099 [S0896-6273\(15\)01084-3](https://www.cell.com/neuron/abstract/S0896-6273(15)01084-3), doi: 10.1016/j.neuron.2015.11.037.
- 1100 **Prevedel R**, Yoon YG, Hoffmann M, Pak N, Wetzstein G, Kato S, Schrödel T, Raskar R, Zimmer M, Boyden ES, Vaziri
1101 A. Simultaneous whole-animal 3D-imaging of neuronal activity using light field microscopy. *Nat Methods*.
1102 2014; 11:727–730.
- 1103 **Reynolds S**, Abrahamsson T, Schuck R, Sjöström PJ, Schultz SR, Dragotti PL. ABLE: An Activity-Based Level Set
1104 Segmentation Algorithm for Two-Photon Calcium Imaging Data. *eNeuro*. 2017; 4(5):ENEURO–0012.
- 1105 **Sheintuch L**, Rubin A, Brande-Eilat N, Geva N, Sadeh N, Pinchasof O, Ziv Y. Tracking the Same Neurons across
1106 Multiple Days in Ca²⁺ Imaging Data. *Cell reports*. 2017; 21(4):1102–1115.
- 1107 **Smith SL**, Häusser M. Parallel processing of visual space by neighboring neurons in mouse visual cortex. *Nature*
1108 *neuroscience*. 2010; 13(9):1144–1149.
- 1109 **Sofroniew NJ**, Flickinger D, King J, Svoboda K. A large field of view two-photon mesoscope with subcellular
1110 resolution for in vivo imaging. *Elife*. 2016; 5.
- 1111 **Spaen Q**, Hochbaum DS, Asín-Achá R. HNCcorr: A Novel Combinatorial Approach for Cell Identification in
1112 Calcium-Imaging Movies. arXiv preprint arXiv:170301999. 2017; .
- 1113 **Speiser A**, Yan J, Archer EW, Buesing L, Turaga SC, Macke JH. Fast amortized inference of neural activity from
1114 calcium imaging data with variational autoencoders. In: *Advances in Neural Information Processing Systems*;
1115 2017. p. 4027–4037.
- 1116 **Theis L**, Berens P, Froudarakis E, Reimer J, Rosón MR, Baden T, Euler T, Tolias AS, Bethge M. Benchmarking spike
1117 rate inference in population calcium imaging. *Neuron*. 2016; 90(3):471–482.
- 1118 **Toledo S**. A survey of out-of-core algorithms in numerical linear algebra. *External Memory Algorithms and*
1119 *Visualization*. 1999; 50:161–179.
- 1120 **Vogelstein JT**, Packer AM, Machado TA, Sippy T, Babadi B, Yuste R, Paninski L. Fast nonnegative deconvolution for
1121 spike train inference from population calcium imaging. *Journal of neurophysiology*. 2010; 104(6):3691–3704.
- 1122 **Walker T**, Cell magic wand tool; 2014. [https://www.maxplanckflorida.org/fitzpatricklab/software/](https://www.maxplanckflorida.org/fitzpatricklab/software/cellMagicWand/)
1123 [cellMagicWand/](https://www.maxplanckflorida.org/fitzpatricklab/software/cellMagicWand/).
- 1124 **Van der Walt S**, Schönberger JL, Nunez-Iglesias J, Boulogne F, Warner JD, Yager N, Gouillart E, Yu T. scikit-image:
1125 image processing in Python. *PeerJ*. 2014; 2:e453.
- 1126 **Xie Y**, Chan AW, McGirr A, Xue S, Xiao D, Zeng H, Murphy TH. Resolution of high-frequency mesoscale intracortical
1127 maps using the genetically encoded glutamate sensor iGluSnFR. *Journal of Neuroscience*. 2016; 36(4):1261–
1128 1272.
- 1129 **Yoo AB**, Jette MA, Grondona M. Slurm: Simple linux utility for resource management. In: *Workshop on Job*
1130 *Scheduling Strategies for Parallel Processing* Springer; 2003. p. 44–60.
- 1131 **Zhou P**, Resendez SL, Rodriguez-Romaguera J, Jimenez JC, Neufeld SQ, Giovannucci A, Friedrich J, Pnevmatikakis
1132 EA, Stuber GD, Hen R, Kheirbek MA, Sabatini BL, Kass RE, Paninski L. Efficient and accurate extraction of in
1133 vivo calcium signals from microendoscopic video data. *eLife*. 2018 Feb; 7:e28728. [https://elifesciences.org/](https://elifesciences.org/articles/28728)
1134 [articles/28728](https://elifesciences.org/articles/28728), doi: 10.7554/eLife.28728.



Appendix 0 Figure 10. Tracking neurons across days, step-by-step description of multi session registration (Fig. 9). (a) Correlation image overlaid to contour plots of the neurons identified by CALMAN BATCH in day 1 (top, 453 neurons), 2 (middle, 393 neurons) and 3 (bottom, 375 neurons). (b) Result of the pairwise registration between session 1 and 2. The union of distinct active components consists of the components that were active in i) both sessions (yellow - where only the components of session 2 are displayed), ii) only in session 2 (green), and iii) only in session 1, aligned to the FOV of session 2 (red). (c) At the next step the union of sessions 1 and 2 is registered with the results of session 3 to produce the union of all distinct components aligned to the FOV of session 3. (d) Comparison of non-consecutive sessions without pairwise registration. Keeping track of which session each component was active in, enables efficient and stable comparisons.



Appendix 0 Figure 11. Profile of spatial (left) and temporal (right) components found in each plane of the whole brain zebrafish recording. (Left) Components are extracted with CAIMAN ONLINE and then max-thresholded. (Right) See Results section for a complete discussion.

1135 Supplemental Data

1136 Description of Supplemental Movies

1137 **Movie 1:** Depiction of CALMAN ONLINE on a small patch of *in vivo* cortex data. Top left: Raw
1138 data. Bottom left: Footprints of identified components. Top right: Mean residual buffer and
1139 proposed regions for new components (in white squares). Enclosings of accepted regions are
1140 shown in magenta. Several regions are proposed multiple times before getting accepted. This is
1141 due to the strict behavior of the classifier to ensure a low number of false positives. Bottom right:
1142 Reconstructed activity.

1143 **Movie 2:** Depiction of CALMAN ONLINE on a single plane of mesoscope data courtesy of E. Froudarakis,
1144 J. Reimers and A. Tolias (Baylor College of Medicine). Top left: Raw data. Top right: Inferred activity
1145 (without neuropil). Bottom left: Mean residual buffer and accepted regions for new components
1146 (magenta squares). Bottom right: Reconstructed activity.

1147 **Movie 3:** Results of CALMAN ONLINE initialized by CALMAN BATCH on a whole brain zebrafish dataset.
1148 Each panel shows the active neurons in a given plane (top-to-bottom) without any background
1149 activity. See the text for more details.

1150 Algorithmic Details

1151 In the following we present in pseudocode form several of routines introduced and used by CALMAN.
1152 Note that the pseudocode descriptions do not aim to present a complete picture and may refer to
1153 other work for some of the steps.

Algorithm 1 PROCESSINPATCHES

Require: Input data matrix Y , patch size p , overlap size o_s , initialization method, rest of parameters.

```
1:  $Y^{(1)}, \dots, Y^{(N_p)} = \text{CONSTRUCTPATCHES}(Y, p_s, o_s)$            ▷ Break data into memory mapped patches.
2: for  $i = 1, \dots, N_p$  do                                         ▷ Process each patch
3:    $[A^{(i)}, C^{(i)}, \mathbf{b}^{(i)}, \mathbf{f}^{(i)}] = \text{CNMF}(Y^{(i)}, \text{options})$    ▷ Run CNMF on each patch
4: end for
5:  $[A, C] = \text{MERGECOMPONENTS}[\{A^{(i)}, C^{(i)}\}_{i=1, \dots, N_p}]$        ▷ Merge Components
6:  $[\mathbf{b}, \mathbf{f}] = \text{MERGEBACKGROUNDS}[\{\mathbf{b}^{(i)}, \mathbf{f}^{(i)}\}_{i=1, \dots, N_p}]$    ▷ Merge background components
7:  $M \leftarrow (A > 0)$ .                                           ▷ Find masks of spatial footprints.
8: repeat                                                           ▷ Optionally keep updating  $A, C, \mathbf{b}, \mathbf{f}$  using HALS (Cichocki et al., 2007).
9:    $[\mathbf{b}, \mathbf{f}] \leftarrow \text{NNMF}(Y - AC, n_b)$ 
10:   $C \leftarrow \arg \min_{C \geq 0} \|Y - \mathbf{b}\mathbf{f} - AC\|$ 
11:   $A \leftarrow \arg \min_{A \geq 0, A(\sim M)=0} \|Y - \mathbf{b}\mathbf{f} - AC\|$ 
12: until Convergence
13: return  $A, C, \mathbf{b}, \mathbf{f}$ 
```

Algorithm 2 SEEDEDINITIALIZATION

Require: Input data matrix Y , matrix of binary masks M , number of background components n_b .

- 1: $\mathbf{p} = \text{find}(A\mathbf{1} == 0)$ ▷ Find the pixels not covered by any component.
- 2: $[\tilde{\sim}, \mathbf{f}] \leftarrow \text{NNMF}(Y[\mathbf{p}, :], n_b)$ ▷ Run NMF on these pixels just to get temporal backgrounds \mathbf{f}
- 3: $\mathbf{b} \leftarrow \arg \min_{\mathbf{b} \geq 0} \|Y - \mathbf{b}\mathbf{f}\|$ ▷ Obtain spatial background \mathbf{b} .
- 4: $C \leftarrow \max((M^T M)^{-1} M^T (Y - \mathbf{b}\mathbf{f}), 0)$ ▷ Initialize temporal traces.
- 5: $A \leftarrow \arg \min_{A \geq 0, A(\sim M) = 0} \|Y - \mathbf{b}\mathbf{f} - AC\|.$ ▷ Initialize spatial footprints constrained within the masks.
- 6: **repeat** ▷ Optionally keep updating $A, C, \mathbf{b}, \mathbf{f}$ using HALS.
- 7: $[\mathbf{b}, \mathbf{f}] \leftarrow \text{NNMF}(Y - AC, n_b)$
- 8: $C \leftarrow \arg \min_{C \geq 0} \|Y - \mathbf{b}\mathbf{f} - AC\|$
- 9: $A \leftarrow \arg \min_{A \geq 0, A(\sim M) = 0} \|Y - \mathbf{b}\mathbf{f} - AC\|$
- 10: **until** Convergence
- 11: **return** $A, C, \mathbf{b}, \mathbf{f}$

Algorithm 3 CALMAN ONLINE (See *Giovannucci et al. (2017)* for explanation of routines)

Require: Data matrix Y , initial estimates $A, \mathbf{b}, C, \mathbf{f}, S$, current number of components K , current timestep t' , rest of parameters.

- 1: $W = Y[:, 1 : t'] C^T / t'$
- 2: $M = CC^T / t'$ ▷ Initialize sufficient statistics (*Giovannucci et al., 2017*)
- 3: $\mathcal{G} = \text{DETERMINEGROUPS}([A, \mathbf{b}], K)$ ▷ *Giovannucci et al. (2017)*, Alg. S1-S2
- 4: $R_{\text{buf}} = [Y - [A, \mathbf{b}][C; \mathbf{f}]][:, t' - l_b + 1 : t']$ ▷ Initialize residual buffer
- 5: $t = t'$
- 6: **for** $i = 1, \dots, N_{\text{epochs}}$ **do**
- 7: **while** there is more data **do**
- 8: $t \leftarrow t + 1$
- 9: $\mathbf{y}_t \leftarrow \text{MOTIONCORRECT}(\mathbf{y}_t, \mathbf{b}\mathbf{f}_{t-1})$ ▷ (*Pnevmatikakis and Giovannucci, 2017*)
- 10: $[\mathbf{c}_t; \mathbf{f}_t] \leftarrow \text{UPDATETRACES}([A, \mathbf{b}], [\mathbf{c}_{t-1}; \mathbf{f}_{t-1}], \mathbf{y}_t, \mathcal{G})$ ▷ *Giovannucci et al. (2017)*, Alg. S3
- 11: $C, S \leftarrow \text{OASIS}(C, \gamma, s_{\text{min}}, \lambda)$ ▷ *Friedrich et al. (2017b)*
- 12: $A_{\text{new}}, C_{\text{new}} \leftarrow \text{FINDNEWCOMPONENTS}(R_{\text{buf}}, N_{\text{comp}})$ ▷ Alg. 4
- 13: $[A, \mathbf{b}], [C, \mathbf{f}], K, \mathcal{G}, R_{\text{buf}}, W, M \leftarrow \text{INTEGRATENEWCOMPONENTS}([A, \mathbf{b}], [C, \mathbf{f}], K, \mathcal{G}, A_{\text{new}}, C_{\text{new}}, R_{\text{buf}}, \mathbf{y}_t, W, M)$ ▷ *Giovannucci et al. (2017)*, Alg. S4
- 14: $R_{\text{buf}} \leftarrow [R_{\text{buf}}[:, 2 : l_b], \mathbf{y}_t - A\mathbf{c}_t - \mathbf{b}\mathbf{f}_t]$ ▷ Update residual buffer
- 15: $W, M \leftarrow \text{UPDATESUFFSTATISTICS}(W, M, \mathbf{y}_t, [\mathbf{c}_t; \mathbf{f}_t])$
- 16: **if** $\text{mod}(t - t', l_b) = 0$ **then** ▷ Update $W, M, [A, \mathbf{b}]$ every l_b timesteps
- 17: $[A, \mathbf{b}] \leftarrow \text{UPDATESHAPES}[W, M, [A, \mathbf{b}]]$ ▷ *Giovannucci et al. (2017)*, Alg. S5
- 18: **end if**
- 19: **end while**
- 20: $t \leftarrow 0$
- 21: **end for**
- 22: **return** $A, \mathbf{b}, C, \mathbf{f}, S$

Algorithm 4 FINDNEWCOMPONENTS

Require: Residual buffer R_{buf} , number of new candidate components N_{comp} , neuron radius r .

- 1: $E \leftarrow \sum_i \max(R_{\text{buf}(i)}, 0)^2$
- 2: $E \leftarrow \text{HIGHPASSFILTER}(E)$ ▷ Spatial high pass filtering for contrast enhancement.
- 3: $P = \text{FINDLOCALPEAKS}(E, N_{\text{comp}}, r)$ ▷ Find local maxima at least $2r$ apart.
- 4: $A_{\text{test}} \leftarrow \emptyset$
- 5: **for** $p \in P$ **do**
- 6: $N_p = \{(x, y) : |x - p_x| \leq r, |y - p_y| \leq r\}$ ▷ Define a neighborhood around p
- 7: $A_{\text{test}} \leftarrow A_{\text{test}} \cup \text{MEAN}(R_{\text{buf}})$
- 8: **end for**
- 9: $I_{\text{accept}} \leftarrow \text{ONLINECNNCLASSIFIER}(A_{\text{test}})$ ▷ Find indices of accepted components
- 10: $A_{\text{new}} \leftarrow \emptyset, C_{\text{new}} \leftarrow \emptyset$
- 11: **for** $i \in I_{\text{accept}}$ **do**
- 12: $[a, c] \leftarrow \text{NNMF}(R_{\text{buf}}[N_{p^i}, :], 1)$
- 13: $A_{\text{new}} \leftarrow A_{\text{new}} \cup a$
- 14: $C_{\text{new}} \leftarrow C_{\text{new}} \cup c$
- 15: **end for**
- 16: **return** $A_{\text{new}}, C_{\text{new}}$

Algorithm 5 REGISTERPAIR

Require: Spatial footprint matrices A_1, A_2 , field of view templates I_1, I_2 , thresholds for binarization θ_b and matching θ_m .

- 1: $S = \text{COMPUTEMOTIONFIELD}(I_1, I_2)$ ▷ Compute motion field between the templates.
- 2: $A_1 \leftarrow \text{APPLYMOTIONFIELD}(A_1, S)$ ▷ Align A_1 to the template I_2
- 3: $[M_1, M_2] = \text{BINARIZE}([A_1, A_2], \theta_b)$ ▷ Turn components into binary masks.
- 4: $D = \text{COMPUTEDISTANCEMATRIX}(M_1, M_2, \theta_D)$ ▷ Compute distance matrix.
- 5: $P_1, P_2, L_1, L_2 = \text{HUNGARIAN}(D)$ ▷ Match using the Hungarian algorithm.
- 6: **return** Matched components P_1, P_2 , non-matched components L_1, L_2 and aligned components from first session A_1 .

Algorithm 6 REGISTERMULTI

Require: List of spatial footprint matrices A_1, A_2, \dots, A_N , list of FOV templates I_1, I_2, \dots, I_N , thresholds for binarization θ_b and matching θ_m .

- 1: **for** $i = 1, \dots, N$ **do**
- 2: $K_i = \text{SIZE}(A_i, 2)$ ▷ Number of components in each session.
- 3: **end for**
- 4: $A_u \leftarrow A_1$ ▷ Initialize A_u matrix
- 5: $m[1] = [1, 2, \dots, K_1]$ ▷ Initialize matchings list
- 6: $K_{\text{tot}} \leftarrow K_1$ ▷ Total # of distinct components so far.
- 7: **for** $i = 2, \dots, N$ **do**
- 8: $P_u, P_i, L_u, L_i, A_u = \text{REGISTERPAIR}(A_u, A_i, I_{i-1}, I_i, \theta_b, \theta_m)$ ▷ Register A_u to session i .
- 9: $A_u[:, P_u] \leftarrow A_i[:, P_i]$ ▷ Keep the matched components from session i .
- 10: $A_u \leftarrow [A_u, A_i[:, L_i]]$ ▷ Include the non-matched components from session i .
- 11: $m[i][P_i] = P_u$ ▷ $m[i][j] = k$ if component j from session i is mapped to component k in A_u .
- 12: $m[i][L_i] = [K_{\text{tot}} + 1, K_{\text{tot}} + 2, \dots, K_{\text{tot}} + |L_i|]$ ▷ Include newly added components.
- 13: $K_{\text{tot}} \leftarrow K_{\text{tot}} + |L_i|$ ▷ Update total number of distinct components.
- 14: **end for**
- 15: **return** Union of all distinct components A_u , and list of matchings m .
