A APPENDIX

Algorithm 1 realSEUDO Algorithm

Initialize: $\boldsymbol{X}_{temp} \leftarrow []$; $\boldsymbol{X}_{stab} \leftarrow []$
for each frame $\boldsymbol{y}_t$ **do**
    Denoise $\boldsymbol{y}_t$
    Identify the profiles in the current frame
    for all new profiles **do**
        if current profile overlaps any of $\boldsymbol{X}_{temp}$ **then**
            Merge new $\boldsymbol{X}_{temp}$ profiles into the current$\boldsymbol{X}_{temp}$
        else
            Add the current profile to $\boldsymbol{X}_{temp}$
        end if
    end for
    for all profiles in $\boldsymbol{X}_{temp}$ that have not been updated in the last few frames **do**
        Move them from $\boldsymbol{X}_{temp}$ to $\boldsymbol{X}_{stab}$
        Merge the moved profiles with existing $\boldsymbol{X}_{stab}$ profiles
    end for
    $\boldsymbol{\phi}_t, \boldsymbol{r}_t \leftarrow \text{SEUDO}(\boldsymbol{y}_t, \boldsymbol{X}_{stab})$
    Report $\boldsymbol{\phi}_t$ as detections
    $\boldsymbol{\phi}'_t \leftarrow \text{SEUDO}(\boldsymbol{r}_t, \boldsymbol{X}_{temp})$
    for each profile k in $\boldsymbol{X}_{temp}$ **do**
        if the $\phi'_{kt} > \gamma$ or this profile was previously active **then**
            Report this activation as early detection
        end if
    end for
end for

A.1 APPLICATION OF MODIFIED FISTA TO NEURAL NETWORK OPTIMIZATION

We further tested the modified FISTA momentum descent algorithm to problems outside of neuroscience—the training of neural networks—to evaluate the scope of applicability of our improvements. We used the problem of recognition of handwritten digits on a data set from AT&T Research available at https://hastie.su.domains/StatLearnSparsity_files/DATA/zipcode.html, reduced to 8x8 pixels, with a training set of 7291 images. The neural network (NN) model used the Leaky ReLU activation, with layer sizes 64, 64, 32, 10.

The common approach to training NNs uses stochastic gradient descent, including stochastic momentum methods. Thus to compare to a non-stochastic momentum method we established a non-stochastic baseline.

The dynamic estimation of the Lipshitz constant L from TFOCS cannot be applied to the neural network optimization because the optimization cost is highly non-linear. The multi-dimensional estimation of L is also computed only for the specific SEUDO function. The common practice is to use a fixed descent rate, which serves as an analog of $\frac{1}{2L}$. Estimating the highest descent rate is still not a fully solved problem. Algorithms for dynamic evaluation of the descent rate do exist (e.g., the ADAM algorithm in Kingma & Ba, 2015), however they rely on a constant to be picked for a particular problem.

The advantage of stochastic methods is that they can use a higher descent rate without diverging, as seen by observing the dependency of logarithm of mean square error from the number of training passes for various descent rates (Fig 6). In these results we ensure that we start from the same fixed randomized initial state since different initial states can produce wildly different results.

We observe that for the same descent rate (0.05 per pass), the stochastic and non-stochastic methods produce very similar error values, however the graph for the stochastic method is more smooth. The roughness of the graph represents the small divergences that manage to converge again over time, and shows that the descent rate is close to the maximum. However the stochastic method can accommodate a 100 times higher descent rate without diverging, and even a 1000 higher descent

Algorithm 2 Modified FISTA algorithm

Initialize $t = 1$; $x[] = (initial\ values)$; diff[] $= [0]$; gradient_last[] $= [0]$
for $step = 1$ to $maxstep$ **do**
  $t_{next} = \frac{1+\sqrt{1+t^2*4}}{2}$
  $\eta = \frac{t_{-1}}{t_{next}}$
  if $step \neq 1$ **then**
    $t = t_{next}$
  end if
  $x[] = x[] + \eta * \text{diff}[]$
  for each i in dimensions of x **do**
    if $x[i] < 0$ **then**
      $x[i] = 0$; diff$[i] = 0$;
    end if
  end for
  gradient[] $=$ compute_gradient_f$(x[])$
  $x[] = x[] - \frac{gradient[]}{L}$
  for each i in dimensions of x **do**
    if $x[i] < 0$ **then**
      $x[i] = 0$; diff$[i] = 0$; gradient$[i] = 0$
    else
      if gradient$[i] *$ gradient_last$[i] < 0$ **then**
        diff$[i] = 0$
      else
        diff$[i] =$ diff$[i] - \frac{\text{gradient}[i]}{L}$
      end if
    end if
  end for
  gradient_last[] $=$ gradient[]
end for

Method	error	log(error)
stochastic	0.0956	-2.3476
baseline non-stochastic	0.0952	-2.3515
FISTA	0.1662	-1.7946
momentum+stop on gradient sign change	0.0699	-2.6607
momentum+stop on gradient sign change + $\eta = 1$	0.0701	-2.6578
momentum+stop on gradient sign change + $\eta = 1$ at 4x rate	0.0751	-2.5889
auto-adjusted rate for momentum+stop on gradient sign change	0.0631	-2.7630
auto-adjusted rate for momentum+stop on gradient sign change + $\eta = 1$	0.0654	-2.7272

Table 1: Performance of algorithms on handwritten dataset

rate becomes rough but still converges. The non-stochastic method is able to make the passes faster, because it performs the same accumulation of partial gradients, but saves the overhead of updating the weights after each training case (or batch). However even adjusted for time, the stochastic method performs faster. It is possible to compute the non-stochastic gradient in parallel by multiple threads but we have not implemented this. We used this example of the non-stochastic descent as a baseline for the FISTA-based momentum methods.

The summary of training errors in the momentum methods can be found in Figure 7A, and the mean square errors after 10,000 training passes are listed in the Table 1.

The unmodified FISTA algoithm with $\lambda = 0$ performed on this task out of its domain worse than the non-momentum baseline. Adding the momentum stop in the dimenstions with gradient sign change produced a substantial improvement over the baseline. Fixing the parameter $\eta = 1$ produced a close result to not fixing η, but with a less rough curve. We tested if the smoothness indicated that setting

$\eta = 1$ could accommodate a substantial increase in descent rate by re-running the algorithm at a 4x rate (0.2 instead of 0.05), and while this run did not diverge, we did observe a higher error rate.

Finally, we attempted to devise an algorithm that acts similar to the TFOCS dynamic evaluation of L but using the ratios of mean square values of gradient dimensions that change or not change sign as an indication of roughness. The rapid growth of gradient dimensions after sign change is seen as a beginning of a divergence, that causes the reduction of descent rate. This algorithm allowed training at a substantially higher rate in the first few thousands of passes but then flattened out. The automatically determined rate is close to the empirically found 0.05, and is higher in the initial passes where it reaches higher values, but then drops to the lower values (Fig. 7B). It is possible that the chosen criteria were not aggressive enough, and can be improved.

While event with the momentum descent the stochastic methods specialized for NN training can still achieve faster speeds (Fig. 7C-D), we have demonstrated that our more general optimization still represent a major improvement over both simple gradient descent and plain FISTA in a different domain.

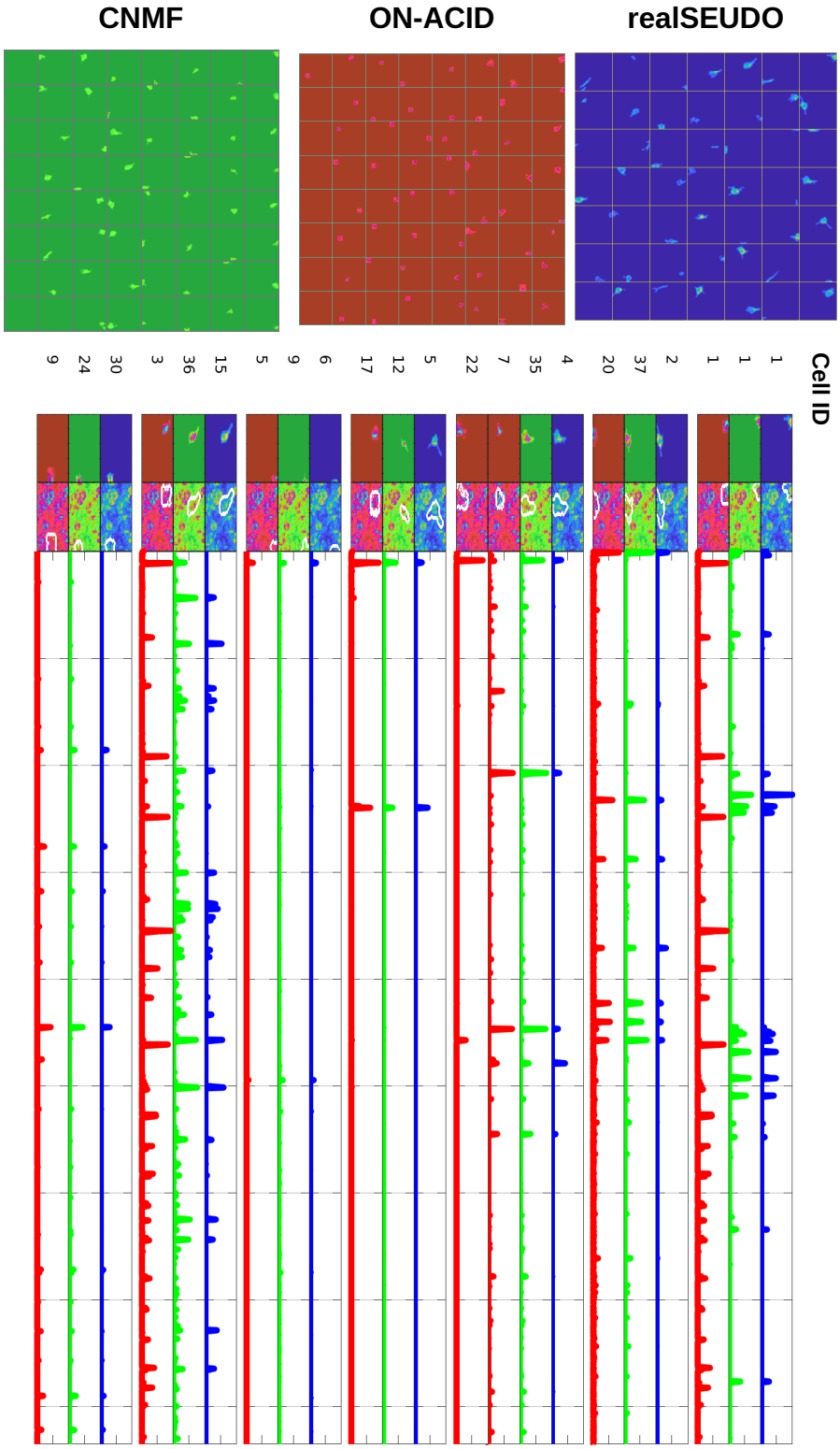

Figure 4: The whole set of cells detected by realSEUDO (blue), OnACID (red), and CNMF (green) in one movie, with selected time traces of matching cells.

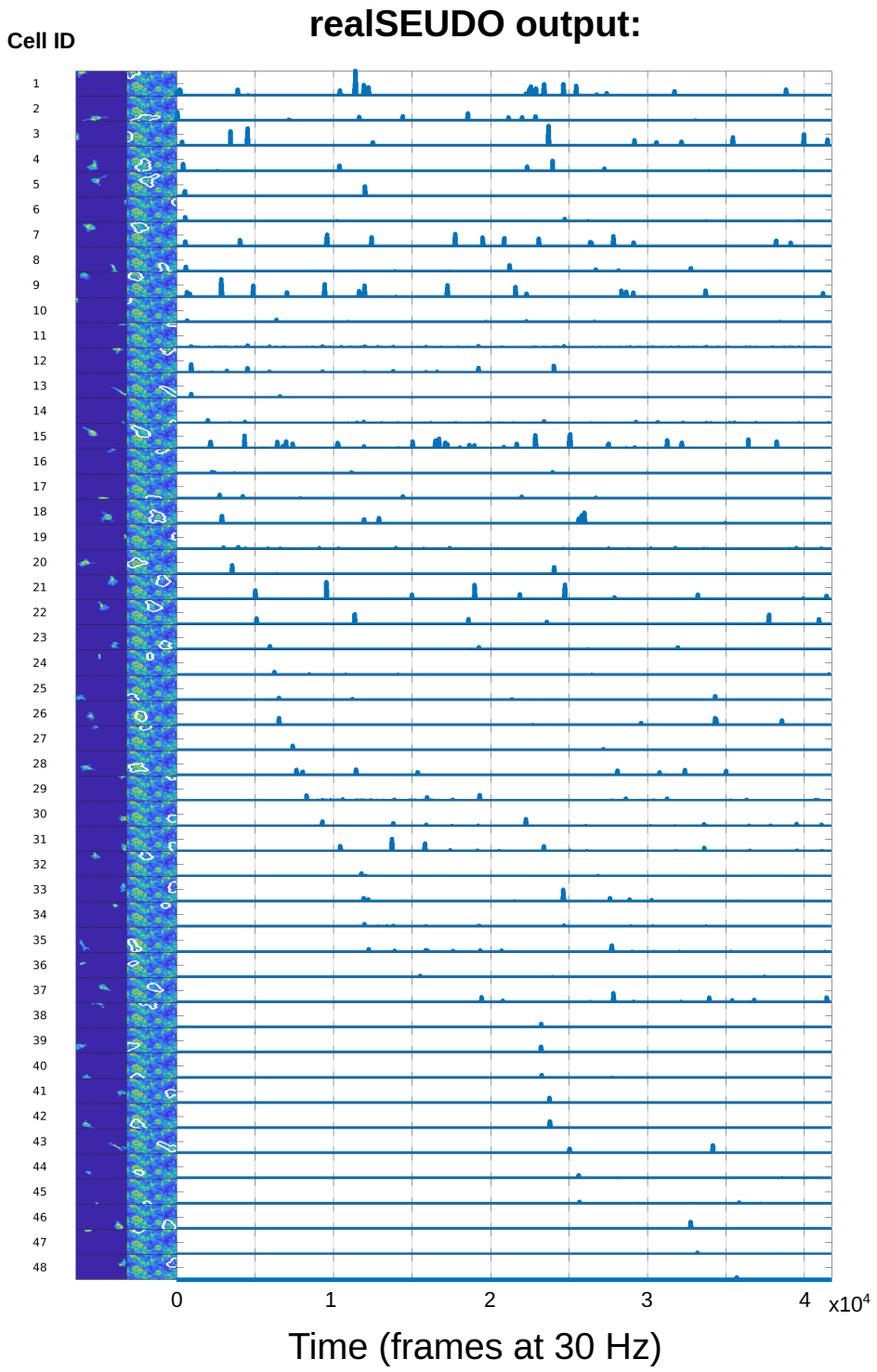

Figure 5: Example realSEUDO cells and traces from a single patch, ordered by the discovery time.

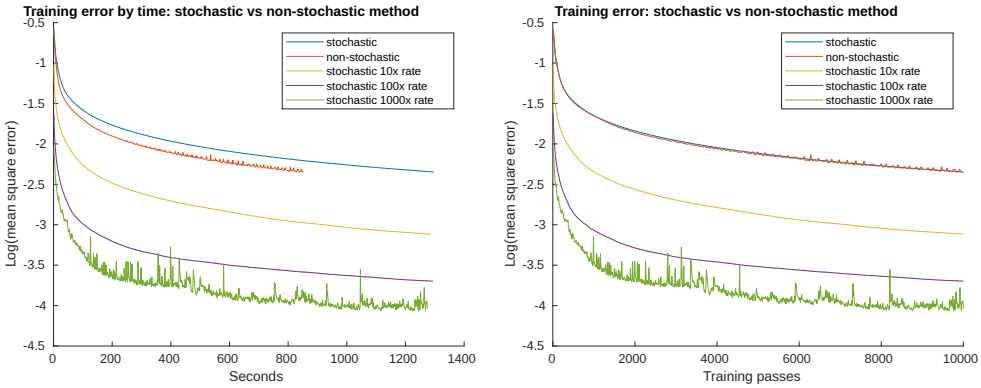

Figure 6: Comparison of different algorithm's learning curves on handwritten datasets. Left: mean-squared error (MSE) as a function of optimization time. Right: MSE as a function of training passes.

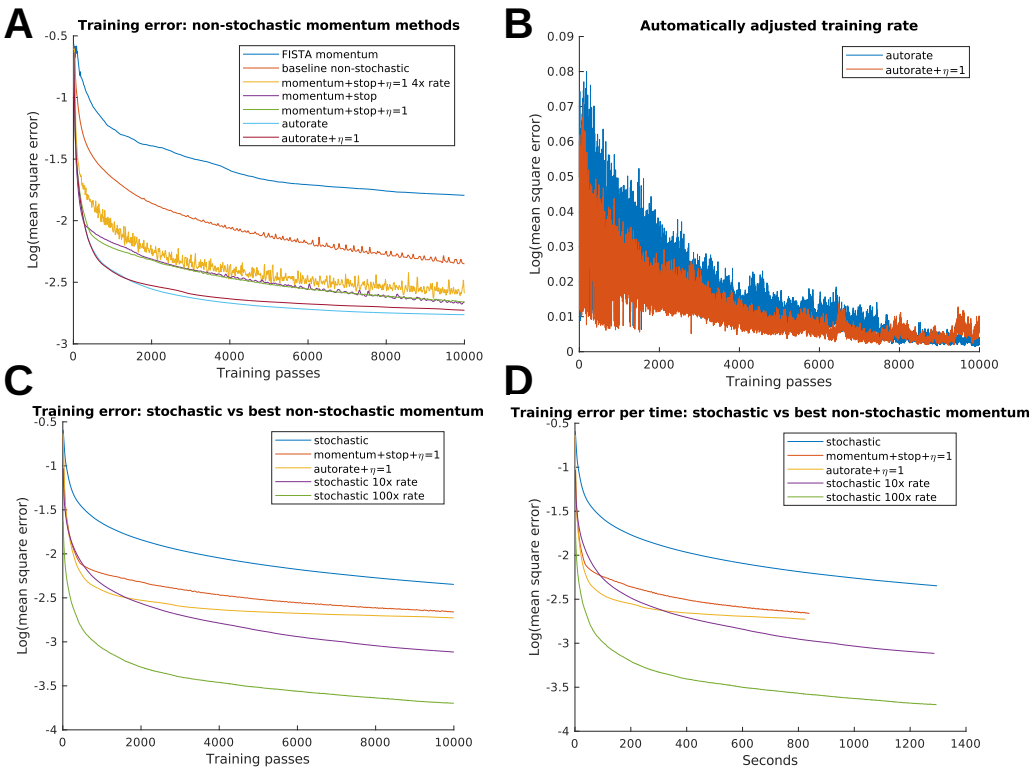

Figure 7: Comaprison of training curves for different algorithms. A: Training MSE as a function of training passes for different variants of the improved FISTA algorithm. B: Training MSE improvement when setting $\eta = 1$. C: Training MSE as a function of training passes for the best tested non-stochastic methods vs. momentum-improved FISTA D: Training MSE as a function of optimization time for the best tested non-stochastic methods vs. momentum-improved FISTA

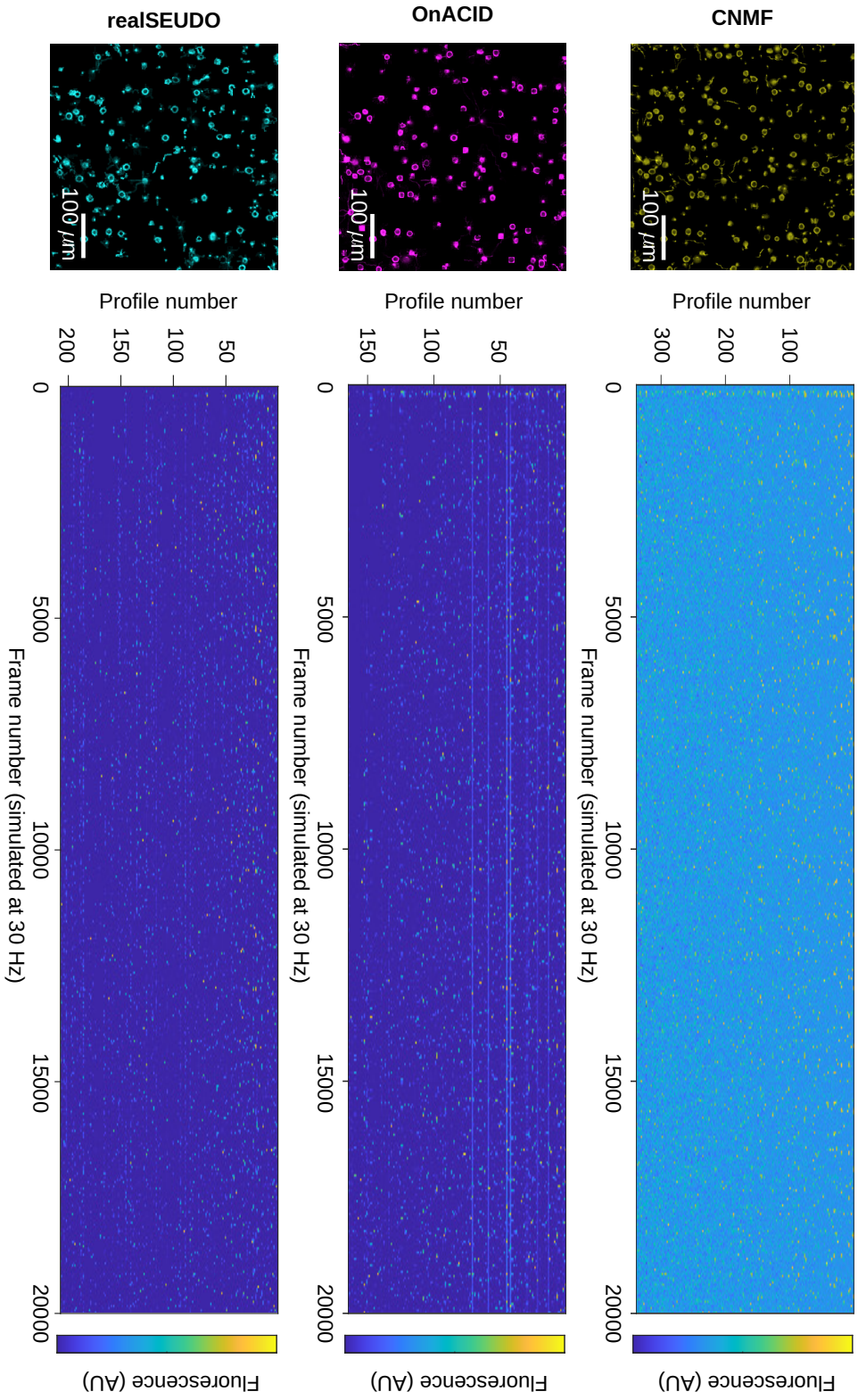

Figure 8: Full sets of strongly-paired traces from CNMF, OnACID and realSEUDO.