Recovering the building blocks of separable computations from neural population activity with Sparse Component Analysis

Andrew J. Zimnik[1,2#], Xinyue An[3,4#], K. Cora Ames[1,2,5,6], Andrew Ulmer[3,4] , Antonio H. Lara[1,2], Abigail A. Russo[1,2], Vladislav Susoy[7,8], Laura Driscoll[9,10,11], John P. Cunningham[2,5,6,12], Liam Paninski[2,5,6,12], Mark M. Churchland[1,2,5,13]*, Joshua I. Glaser[3,14,15]*

[1]Department of Neuroscience, Columbia University Medical Center, New York, NY, USA; [2]Zuckerman Institute, Columbia University, New York, NY, USA; [3]Department of Neurology, Northwestern University, Chicago, IL, USA; [4]Interdepartmental Neuroscience Program, Northwestern University, Chicago, IL, USA; [5]Grossman Center for the Statistics of Mind, Columbia University, New York, NY, USA; [6]Center for Theoretical Neuroscience, Columbia University, New York, NY, USA; [7]Department of Physics, Harvard University, Cambridge, MA, USA; [8]Center for Brain Science, Harvard University, Cambridge, MA, USA; [9]Department of Electrical Engineering, Stanford University, Stanford, CA, USA; [10]Allen Institute for Neural Dynamics, Allen Institute, Seattle, WA, USA; [11]Department of Neurobiology and Biophysics, University or Washington, Seattle, WA, USA, [12]Department of Statistics, Columbia University, New York, NY, USA; [13]Kavli Institute for Brain Science, Columbia University Medical Center, New York, NY, USA; [14]Department of Computer Science, Northwestern University, Evanston, IL, USA; [15]National Institute for Theory and Mathematics in Biology, Chicago, IL, USA

Co-first authors

* Co-senior authors.

Correspondence: mc3502@columbia.edu, j-glaser@northwestern.edu

Abstract

In many neural populations, the computationally relevant signals are posited to be a set of 'latent factors' – signals shared across many individual neurons. A given brain area may perform multiple computations, each associated with distinct factors, that can together compose an overall action. Methods for uncovering such structure typically require supervision, which can limit discovery of novel aspects of activity. Here, we introduce Sparse Component Analysis (SCA), an unsupervised approach. SCA facilitated surprisingly clear parcellations of neural activity across a range of behaviors, when seeking both linear and nonlinear embeddings. We applied SCA to motor cortex activity from reaching and cycling monkeys, single-trial imaging data from *C. elegans*, and activity from a multitask artificial network. SCA revealed both simple and unexpected instances where the overall population response was built compositionally from sets of factors with distinct computational roles.

Introduction

The study of computation by neural populations has experienced a conceptual shift. Traditionally, one first characterized single-neuron responses, then extrapolated to determine how the population would behave. This worked well in situations where population-level function (e.g. saccade specification) was a straightforward extension of single-neuron responses (e.g. tuning for target amplitude and direction)[1,2]. Yet with time, it became appreciated that this strategy must often be inverted. In many areas, single-neuron responses are complex, heterogeneous, and make little sense except in the context of population-level computation. This realization demanded tools for directly investigating computation at the population level.

In many biological and artificial networks, population-level computations involve 'latent factors'[3–10]: signals that are shared across the population and consistent across trials, despite variable single-neuron spiking[11]. Factor co-evolution can yield explanations regarding how a computation is accomplished (e.g.,[11–20]). However, factors cannot be observed directly. They must be estimated from population activity, which poses a hurdle: there are typically many statistically valid estimates of factors, yet not all provide equally interpretable views of the underlying computations. This is particularly true if a given neural population performs multiple computations. Behavioral flexibility is thought to result in part from the ability to switch amongst computations. This may even involve compositional reuse: the arrangement of subcomputations, in specific combinations or particular sequences, to perform an overall computation. There is thus a need for a method that can identify, in mixed single-neuron responses, factors that correspond to potentially distinct computations.

Motor cortical activity during delayed-reach tasks provides a canonical example (Fig. 1A). Most neurons are active during delay and movement, with no clear link between these activity patterns[21–24]. Yet one can identify two distinct sets of factors. 'Preparatory factors' (Fig. 1A, *left*) are active during the delay. 'Execution-related factors' (Fig. 1A, *right*) are active during the reach[25,26]. The ability to estimate such factors facilitated experiments addressing how movements are prepared[26], executed[15], corrected[27], and arranged in sequences[28]. Progress was possible because it was suspected that preparation and execution might be distinct yet linked processes: preparatory factors were hypothesized to seed execution dynamics that generate descending commands[15,21,29,30], and must therefore become active first. Critically, task design allowed researchers to infer when these factors were likely active. Reaching also provides an example of compositionality: reach sequences leverage repeated reuse of the same preparatory and execution-related factors as single reaches[28,31], with each reach preceded by its own preparatory stage. It is increasingly appreciated that more sophisticated forms of computational flexibility likely also involve compositional reuse of distinct computations, subserved by distinct factors[13,32,33].

Standard unsupervised methods (e.g. principal component analysis; PCA) are often ill-suited to discover factors with computationally distinct roles; PCA-estimated factors are frequently as 'mixed' as the individual neurons one started with. The field has thus developed multiple supervised approaches. A simple form of supervision is to divide data into temporal epochs based on prior knowledge or assumptions, before applying an otherwise-unsupervised method[17,19,25]. Alternatively, explicitly supervised methods such as demixed PCA (dPCA) use labeling based on within-trial time and experimental condition[34]. Related methods such as targeted dimensionality reduction (TDR) and its successor, model-based TDR,[35] use regression to identify factors that covary with task-related variables[12]. Methods such as jPCA[15] avoid labels, but seek hypothesized structure that is typically dataset- or experiment-specific (e.g., factors that obey rotational[15], or unconstrained[36] linear dynamics). While they can be powerful, these approaches cannot be applied unless one can infer the correct labels, temporal divisions, or potential forms of population-level structure.

Here, we present an unsupervised approach, Sparse Component Analysis (SCA), for identifying latent factors that reflect distinct computations. SCA leverages a basic premise of compositionality: when multiple distinct computations exist, they are not necessarily coactive, but may unfold with temporal flexibility relative to one another. SCA seeks factors that 1) are sparse in time, facilitating discovery of temporally flexible aspects of activity, and 2) evolve within orthogonal dimensions, facilitating discovery of activity related to potentially distinct computations. SCA's simple cost function reflects very general assumptions and can therefore be applied to a wide range of datasets. This approach can assume either linear or nonlinear factor embeddings.

We applied SCA to data collected from three experimental models (rhesus monkeys, *C. elegans*, and artificial networks) and multiple behavioral tasks. SCA identified factors previously documented using supervised methods, verifying that SCA can recover known features without need for supervision. SCA also identified previously undescribed factors, demonstrating its utility for scientific discovery. In monkey motor cortical activity during reaching, SCA identified distinct factors related to reach preparation, execution, and postural maintenance. During unimanual cycling, SCA identified surprisingly analogous factors to those observed during reaching, and also identified factors related to stopping an ongoing movement. These sets of factors were reused compositionally across cycling bouts of different durations. In a bimanual cycling task, SCA identified arm-selective factors that were used compositionally during bimanual movements. When applied to neural data collected from *C. elegans* during mating, SCA identified factors, and individual neurons, related to specific mating motifs. Finally, in a multi-task RNN, SCA recovered factors that subserved compositional network computations[13,37]. Thus, SCA can both recover known findings without supervision, and can uncover previously unknown factor categories that inform novel hypotheses.

Results

Discovering compositional, non-synchronous factors with sparsity and orthogonality

We designed SCA to respect the assumption that behavioral flexibility likely reflects compositionality: the presence of multiple computations (or subcomputations), combined in different ways. Such flexibility may take many forms[16,17,19,34,38–42], including computation order, duration, and when or whether they overlap temporally. By contrast, two computations with consistently identical timecourses cannot produce flexibility via compositionality – indeed one probably wouldn't consider them separable. SCA leverages the fact that, if computation-specific factors have non-identical timecourses (i.e., they are not completely co-active, or "non-synchronous"), they will be more sparsely active (at least modestly and perhaps dramatically) than a 'mixed' factor-basis would be.

Consider preparatory and execution-related computations during reaching (Fig. 1A). Preparation involves (roughly) fixed-point dynamics. Execution unfolds in orthogonal dimensions and involves rich dynamics with a strong rotational component. Preparation must precede execution, and they overlap modestly. They nevertheless exhibit temporal flexibility: preparation can be lengthy or brief[26] and can be updated[27] or canceled[43] prior to execution. In sequences, preparation for the next reach can occur during or after the current reach[28], depending on pacing. Figure 1B illustrates a simplified form of such flexibility: process A always begins before process B, but their relative timecourses are flexible. Bimanual movements provide a complementary example (Fig. 1C). Distinct factor sets (employing orthogonal dimensions) are active when the left or right arm moves[44,45], and become co-active when both move simultaneously[46]. Figure 1D illustrates a simplified version of such flexibility: process A and process B can occur either simultaneously or alone.

Our goal was to develop a method for discovering factors that display one or more such types of temporal flexibility, without requiring prior knowledge regarding the factor types that might be present or when they might be active. For reasons discussed above, encouraging sparsity and orthogonality is likely to be valuable in discovering such factors. Depending on circumstances, current methods can be agnostic to sparsity, can implicitly discourage it, or can implicitly encourage it. Methods also differ regarding whether orthogonality is enforced / encouraged. To yield a method that consistently and explicitly encourages sparsity and orthogonality, the SCA cost function directly embodies that goal:

$$\arg\min_{\mathbf{U},\,\mathbf{V}} \left(\|\mathbf{W}\left(\mathbf{X} - \mathbf{XUV}\right)\|_F^2 + \lambda_{\text{sparse}} \|\mathbf{XU}\|_1 + \lambda_{\text{orth}} \|\mathbf{VV}^\top - \mathbf{I}\|_F^2 \right)$$

The first term ensures the learned latent factors capture a large fraction of neural variance (thus minimizing error when reconstructing individual-neuron responses from the factors). Unlike PCA, SCA seeks to capture variance across all identified factors, rather than concentrating variance within initial components. The second term encourages factors to be sparse in time (either within-trial or across conditions). The identified factors may not actually be sparse, but should be more sparse than if they mixed non-synchronous subcomputations. The final term encourages orthogonal neural-latent mappings (loadings). This accords with the common assumption (as in the examples above) that subcomputations employ orthogonal dimensions to avoid interference (this assumption can be relaxed by appropriately choosing λ_{orth}).

Concretely, let $\mathbf{X}^{T\times N}$ be a data matrix. N indicates the number of recorded neurons. Each column of $\mathbf{X}$ may concatenate that neuron's response across conditions; T is the total number of timepoints across conditions. The factors, $\mathbf{Z}$, are estimated via a mapping, from neural activity space to K-dimensional factor space (Fig. 1E): $\mathbf{Z} = \mathbf{X}\mathbf{U}$. The matrix $\mathbf{V}^{K\times N}$, in which we constrain rows to be unit norm (see *Methods*), reconstructs neural activity from the factors, with the goal that $\mathbf{X} \approx \mathbf{Z}\mathbf{V}$. We include offset terms in both mappings (omitted for brevity in the equation above, see *Methods*). Neural activity in $\mathbf{X}$ is weighted, via the diagonal matrix $\mathbf{W}^{T\times T}$, such that reconstruction when firing rates are overall high (e.g. during execution) is not given inherent precedence over reconstruction when rates are overall lower (e.g., during preparation). SCA's cost function is agnostic regarding the experimental 'meaning' of different times; no labels are assigned nor are data divided into epochs. For example, SCA has no knowledge regarding whether activity corresponds to an imposed preparatory period versus an execution epoch, or belongs to one experimental condition versus another.

The fundamental intuition is that temporal flexibility of one computation, relative to another, implies that estimated factors will be more sparse if each pertains to only one computation. Consider two idealized 'ground truth' latent factors, active at separate times (Fig. 1F: times with black and red backgrounds). Consider factor-activity, at a given moment, in state space (Fig. 1G). Activity lies on the x-axis, at [1,0], when factor 1 is active. Activity lies on the y-axis, at [0,1], when factor 2 is active. If we rotate the estimated factors, $\mathbf{Z}$, anywhere along the unit circle (dashed vectors), we can equivalently explain neural activity by rotating $\mathbf{V}$ in the opposite direction. However, rotation mixes ground-truth factors: both rotated factors are active during both black and red epochs in Fig. 1F. Consequently, although ground-truth and rotated factors capture response variance equally, the ground-truth factors are sparser: each time point has $|\mathbf{Z}| = 1$, instead of $|\mathbf{Z}| = \sqrt{2}$. Thus, when ground-truth factors are active at non-overlapping times, the sparsest solution is the ground truth.

This continues to be true when ground-truth factors overlap partially, until overlap becomes greater than ~58%; (e.g. if two 100ms processes co-occur for >58 ms; Fig. 1I,J - see

Supplementary Text 1 for derivation and further details). This highlights a potential limitation of SCA. Penalizing sparsity is insufficient to demix highly overlapping latent factors; the sparsest solution will, interestingly, be a 45 degree rotation where one dimension captures the commonality between ground-truth factors and the other dimension captures their differences (Fig. 1H). In practice, this limitation is acceptable and possibly desirable: in situations where factors are mostly co-active, it may be unclear scientifically whether they should be considered part of the same process. Answering that question will depend not on analysis, but on further experiments that probe their capacity for temporal flexibility.

Unsurprisingly then, SCA can reveal temporal flexibility of one factor, relative to another, only if such flexibility is recruited by the experimental design. When the experimental design is sufficient (including the idealized scenarios in Figure 1B,D), SCA can successfully recover ground-truth factors without supervision (second column in Figure 1B,D; see Supplementary Text 1 for another example) even when individual-neuron responses (last column) are completely mixed (here rates were modeled as randomly weighted sums of factors). In contrast, PCA (third column) does not recover the ground truth. This is unsurprising; depending on the situation, PCA may be agnostic to sparsity, or may implicitly tend to mix sparsely active factors because doing so can concentrate variance in the top principal components. Several other existing unsupervised approaches, including Independent Component Analysis and Nonnegative Matrix Factorization, also fail to consistently recover the ground-truth factors (Fig. S1).

In empirical data, where we typically lack access to 'ground truth' factors, why would SCA factors be preferable to PCA factors given that both are statistically valid views of the data? SCA factors aren't necessarily more interpretable, but can be in situations where distinct subcomputations are used compositionally or are otherwise temporally flexible relative to one another. As we will show, if such factors are not present, SCA is unlikely to 'invent' them (when factors are consistently coactive, it is typically not possible to reduce the second term in the cost function without substantially increasing the first term). A failure to find such factors can thus argue for rejecting the motivating hypothesis. Conversely, if SCA finds such factors, their nature (when and how they are active) may suggest more specific hypotheses that can be tested via additional analyses or experiments. Indeed, a strength of SCA is that it can explore a general hypothesis without making specific assumptions regarding how factors behave, beyond them having some degree of temporal flexibility relative to one another. We illustrate these points below and show, across a variety of datasets, that SCA-identified factors can both confirm known results and inform novel hypotheses.

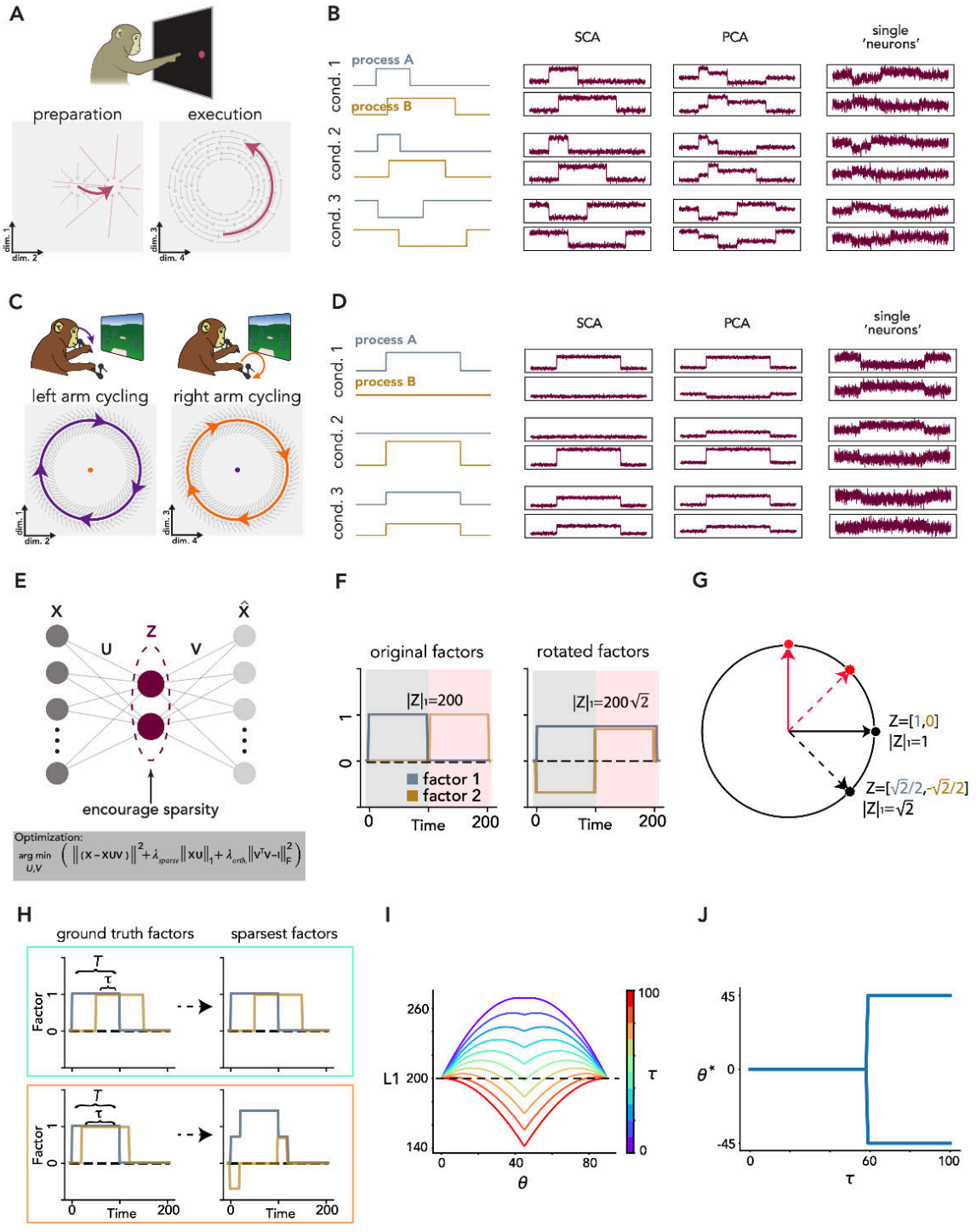


Figure 1. SCA seeks non-synchronous factors. A. Idealized view of reaching factors. Left: during preparation, condition-specific inputs alter the fixed point of population dynamics, resulting in reach-specific preparatory-factor activity. Right: execution dynamics, interacting with the state set during preparation, determine the execution-related factor trajectory. **B.** Timecourse of two idealized neural processes with flexibility resembling that of preparation and

execution. Process A always precedes process B, and they always overlap, but they are non-synchronous and have non-identical durations across conditions. A population of 50 'neurons' was simulated by multiplying the ground truth factors (assuming one factor per process) by an orthonormal matrix, plus added Gaussian noise (two examples shown at right). SCA, but not PCA, recovered the ground-truth latents. **C.** Idealized view of cycling factors. Motor cortical dynamics (*gray arrows*) maintain stable limit cycles. Those dynamics govern factors active when the left-arm (*purple)* and right-arm *(orange)* moves. Left-arm and right-arm factors evolve in orthogonal subspaces; the space that captures left-arm factors *(left*) accounts for little variance when the right-arm moves (*orange dot*, *left space*), and *vice versa*. **D.** Time course of two idealized neural processes with flexibility resembling that of left-arm and right-arm cycling. Neurons were simulated from these processes, and dimensionality reduction applied, as in panel C. **E.** SCA architecture and cost function. Bias terms not shown for brevity. **F.** *Left*: An example neural process with two underlying latent factors, active at non-overlapping times. *Right*: These same factors but rotated by -45 degrees. The L1 norm (the second term of the SCA cost function) is shown for the original and rotated factors. **G.** State-space view, for two timepoints, of the factors from panel F. Solid and dashed lines correspond to original and rotated factors, respectively. Black and red colors correspond to earlier and later timepoints. **H.** Exploration of how the sparsity penalty interacts with factor overlap. Two factors overlap for duration τ (total duration T=100). For moderate overlap (τ = 50; *top*), no rotation of ground-truth factors is sparser than the ground-truth factors themselves. For high overlap, (τ = 80; *bottom*), there exists a rotation that increases sparsity. **I.** L1 norm versus rotation of the latents in H, for τ = 0 (no overlap) to τ = 100 (complete overlap). Only for larger values of τ, where the trace crosses below the dashed line (the L1 value with no rotation), does the sparsest solution involve rotation of ground-truth latents. Otherwise the sparsest solution is the ground-truth latents themselves. **J.** Summary of results in panel I. The sparsity-maximizing rotation is zero (thus identifying ground-truth factors) for temporal overlap <58 ms (of 100 ms).

Center-out reaching - SCA factors reflect distinct neural computations

Because of the established presence of distinct computations[14,15,26,47,48] in motor cortex during center-out reaching (Fig. 2A,B), we thoroughly consider how SCA compares with other approaches in this context, then proceed more swiftly for additional datasets. Monkeys held their hand at a central touchpoint and were shown one of eight peripheral targets. Reaching was not allowed during a randomized delay period. Following a go cue, monkeys swiftly reached, holding their new posture until reward delivery. They then returned to the starting position to begin the next trial. Return reaches were self-initiated; there was no requirement to do so within a certain time.

It is established that preparatory factors become active shortly after target onset and typically remain relatively static throughout the delay[26,49]. After the go-cue, just before reach onset, preparatory-factor activity collapses as execution-related factors become active[25,26]. It has recently become appreciated that additional factors, related to postural-maintenance[50,51], are active while holding a target[31]. A straightforward hypothesis is thus that a full center-out-reach trial will be composed of multiple distinct computations: preparing then executing an outward reach, maintaining posture at that target, then preparing and executing a return reach. We

asked whether SCA and/or PCA could identify this hypothesized progression without supervision. To focus on factors where activity varies with direction, we employed the common preprocessing step of subtracting the time-varying cross-condition mean from each neuron's response[25,28].

The presence of distinct processes was apparent neither at the level of single neurons (Fig. 2C) nor in PCA-derived factors (Fig. 2D); for both, activity relating to preparation, execution, and posture were mixed. For example, the neuron shown in the top row of Fig. 2C was active during the delay period, the outward reach, the hold period (when holding the captured target), and the return reach. Many PCA factors were also continuously active throughout most of the trial (e.g., Fig. 2D, *factors 2 and 3*). Neural responses and PCA factors sometimes hinted at the presence of discrete processes: individual-neuron tuning typically changed between delay and execution epochs (e.g., Fig. 2C, *top row*) and some PCA factors were more active during a specific epoch (e.g., Fig. 2D, *top row*). Yet these hints are somewhat subtle.

In contrast, each SCA factor (Fig. 2E) was primarily active during only one task epoch: preparation, execution, or postural-maintenance. Shortly after target onset, activity emerged in dimensions 1 and 2 and was sustained throughout the delay, a pattern typical of preparatory activity[21,26,49]. Just before reach onset, activity in these dimensions collapsed, while activity grew in dimensions 4, 6, 7, and 8. Dimensions 6, 7, and 8 were occupied briefly: for approximately the same duration as the outward reach. In contrast, activity in dimensions 4 and 5 remained high as the target was held. These patterns are consistent with dimensions 1 and 2 capturing preparatory factors, dimensions 6, 7, and 8 capturing execution-related factors, and dimensions 4 and 5 capturing posture-related factors. Thus, despite lack of supervision (SCA had no knowledge regarding the number of epochs or their boundaries), factor activity agreed with established and emerging beliefs regarding the presence of multiple factor categories.

Examination of these naturally emerging factor-categories during return reaches provides a further test of current interpretations. During return reaches, posture returns to roughly where it started. In agreement, putative posture-factor activity receded before the self-initiated return reach, returning roughly to baseline. At the same time, activity reemerged in dimensions 1 and 3, likely reflecting preparation for the return reach. (That dimension 3 is occupied only prior to the return reach likely reflects kinematic differences between outward and return reaches, a point discussed later). Just before return-reach onset, the activity of the putatively preparatory factors decreased, while activity of putatively execution factors rose. Patterns of execution-related activity were 'flipped' between outward and return reaches, in agreement with the reversal of reach direction. Similar results were obtained for a second monkey (Fig. S2).

SCA thus confirms the known distinction between preparatory and execution factors, while also adding multiple nuances to our understanding of out-and-return sequences. First, it was

recently shown that distinct posture dimensions are engaged when tasks are performed from a given experimenter-set posture[51]. SCA reveals that distinct posture factors also actively track hand position across voluntary reaches (also see[31]). Factors with this property simply emerge when applying SCA, without need to specifically seek them; had SCA been a standard method a decade ago, it would have been almost impossible not to discover this basic phenomenon. Second, present results reveal that preparation is brief before self-paced return reaches, suggesting that the standard plateau seen during the delay is likely atypical, and reflects monkeys maintaining readiness until the unpredictable go cue. Lastly, although SCA revealed a novel class of posture-related factors, it did not reveal 'sequence related' factors, even though these certainly could have been present (e.g. [52]) and are present in other areas[53]. Instead of employing novel factor-level events, return reaches simply employed events mirroring those during outward reaches. These results mesh with recent conclusions regarding explicitly instructed externally paced sequences[28,31].

SCA-factor activity also speaks to the linkage between preparation and execution. Consider activity in a state-space (Fig. 2F,G) defined by one preparatory dimension (SCA dimension 1) and two execution-related dimensions (SCA dimensions 6 and 8). Just before the outward reach, preparatory activity rotates into the plane defined by execution-related dimensions (Fig. 2F, *gray plane*). Activity within this plane then rotates in a consistent direction across conditions[15]. This view confirms prior conclusions (e.g. compare with Fig. 7a of [25] and Fig 2a of [33]) but also extends them: during return reaches (Fig. 2G), the same process unfolds in the same three dimensions. Preparatory activity is weaker (likely because return reaches are slower) and condition-order reverses (reflecting reversal of the reaches themselves). Yet the same neural events occur, with the same rotation from preparatory dimension into execution dimensions, and the same rotation direction during execution.

Center-out reaching - Quantitative comparisons with PCA

The fact that each factor corresponds to a plausibly distinct process (preparation, execution, or posture) aids interpretation of SCA factors. In contrast, PCA factors (Fig. 2D) reflect mixtures of those processes (much as do single neurons). To explore further, we compared SCA factors with factors recovered by weighted PCA (wPCA, see *Methods*). wPCA uses the same weighting matrix ($\mathbf{W}$) as SCA; the only substantial difference between the methods is that wPCA does not seek sparsity.

Seeking of sparsity did not come at the cost of decreased reconstruction accuracy: SCA and wPCA factors accounted for virtually identical total neural variance (Fig. 2I), and performed equally well when generalizing to held-out neurons and conditions (Fig. 2I, *inset*). The major difference was simply the degree to which each factor reflected a single putative computation. To quantify this effect, we computed the cross-condition variance of activity in each dimension

(the dimension's 'occupancy', *Methods*) during preparatory, execution-related, and posture-related epochs (Fig. 2H, left). For all 8 SCA dimensions, one epoch accounted for >60% of that dimension's total occupancy. This concentration of occupancy was significantly higher for SCA-factors versus wPCA-factors (Fig. 2H, right, $p < 0.01$ *bootstrap test*).

This feature did not make SCA more 'data intensive' (less reliable for a given number of neurons) than PCA. When subsampling neurons, SCA-dimension consistency was comparable (or greater) than that of wPCA (Fig. S3). More broadly, SCA dimensions typically account for similar data variance as wPCA dimensions; they capture similar signals, just in a different basis, and are thus unlikely to 'over sparsify' responses. A similar point was made by a control analysis where we applied SCA solely to delay-period activity. Preparatory-factor activity is generally viewed as sustained throughout the delay: after its initial rise, there is no strong tendency to have 'early' or 'late' preparatory factors[25,26]. Consequently, if SCA-identified factors approximate ground-truth factors, they should not be particularly sparse. To test this, we divided the delay into three epochs and calculated occupancy concentration. Occupancy concentration did not differ significantly between SCA and wPCA factors ($p = 0.68$, *bootstrap test*), confirming that SCA did not find distinct factors in a situation where they are unlikely to exist.

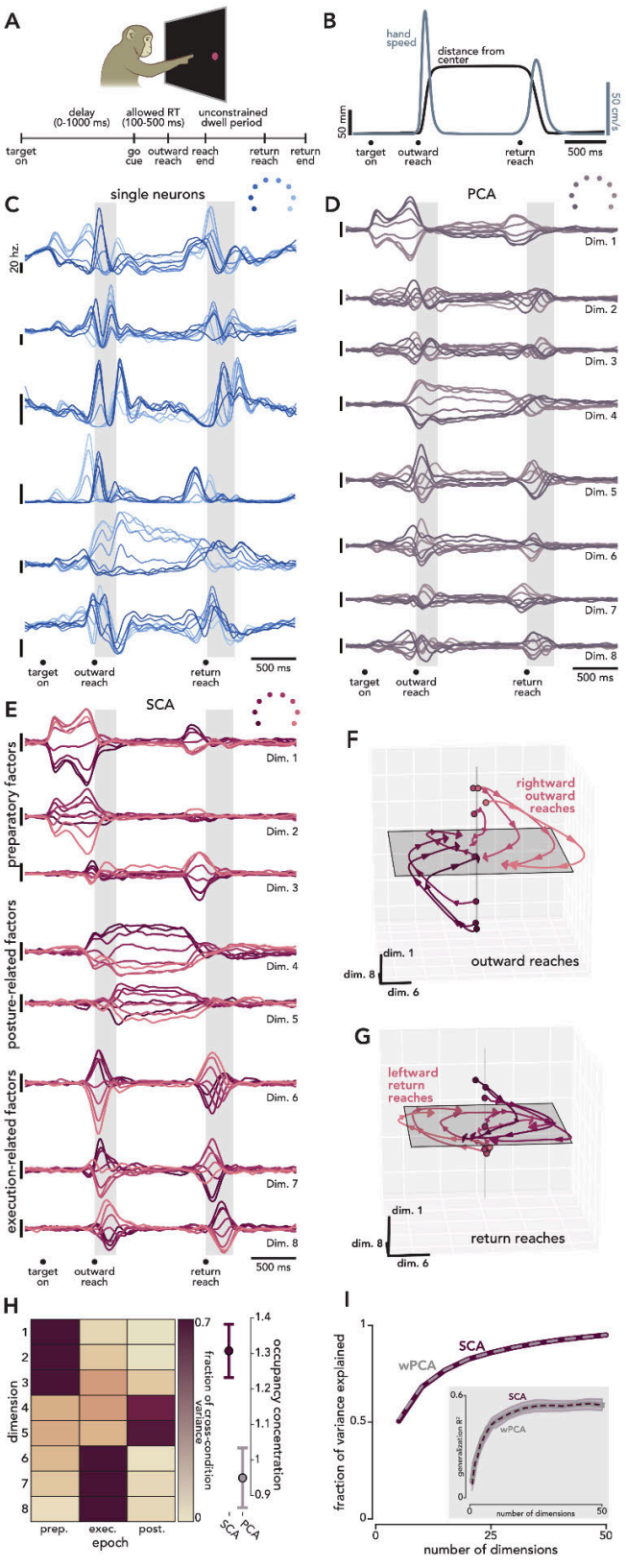

A
delay (0-1000 ms)
allowed RT (100-500 ms)
unconstrained dwell period
target on
go cue
outward reach
reach end
return reach
return end
B
hand speed
distance from center
50 mm
50 cm/s
target on
outward reach
return reach
500 ms
C
single neurons
20 hz.
target on
outward reach
return reach
500 ms
D
PCA
Dim. 1
Dim. 2
Dim. 3
Dim. 4
Dim. 5
Dim. 6
Dim. 7
Dim. 8
target on
outward reach
return reach
500 ms
E
SCA
preparatory factors
posture-related factors
execution-related factors
Dim. 1
Dim. 2
Dim. 3
Dim. 4
Dim. 5
Dim. 6
Dim. 7
Dim. 8
target on
outward reach
return reach
500 ms
F
rightward outward reaches
outward reaches
dim. 1
dim. 8
dim. 6
G
leftward return reaches
return reaches
dim. 1
dim. 8
dim. 6
H
dimension
prep.
exec.
post.
epoch
fraction of cross-condition variance
occupancy concentration
SCA
PCA
I
fraction of variance explained
SCA
wPCA
generalization R²
number of dimensions
number of dimensions

Figure 2. SCA identifies preparatory, execution-related, and posture-related factors in motor cortical reaching data. A. Trial timing. **B.** Example trial-averaged behavior for one condition. **C.** Firing rates of six example neurons (one trace per reach direction). *Gray rectangles:* median reach duration. **D.** Top eight PCA factors, ordered by the time in the trial when maximum occupancy (cross-condition variance) occurred. **E.** Eight SCA factors (when requesting eight) ordered as in panel D. Each factor's magnitude directly reflects its impact on neural activity, due the unit-norm constraint on the V matrix. **F.** Outward reach activity projected into three SCA dimensions. **G.** Projection of return reach activity in the same space. **H.** *Left*: For each SCA factor, the fraction of total across-condition variance accounted for by each epoch. *Right*: Mean and standard deviation of occupancy concentration (*Methods*) calculated across bootstrapped neural populations ($p < 0.01$, n=100 resampled populations). **I.** SCA and wPCA account for virtually identical fractions of neural variance. *Inset*. SCA and wPCA were trained on activity from a subsample of neurons during all but one condition. R^2 was calculated between predicted and actual activity of held-out neurons during the held-out condition (see *Methods*). *Solid* (wPCA) and *dashed* (SCA) traces plot mean R^2 across held-out conditions. Shading indicates standard error.

Center-out reaching - Comparisons with supervised approaches

A desired feature of an unsupervised method is that it should perform almost as well as a supervised method that makes accurate assumptions, and better than a supervised method that makes imperfect assumptions. To find factors in a supervised way, we used the method introduced by Elsayed *et al.*[25] to identify orthogonal subspaces, corresponding to preparation, execution, and posture (Fig. 3A-C). This approach leverages division of data into three types of epochs, based on task timing and prior results. Doing so thus leverages considerable prior knowledge, including when preparation occurs with and without a delay period. If SCA naturally discovers those same divisions, each SCA factor should be a linear combination of supervised-method factors that are restricted to be of only one type (e.g. only preparation-epoch factors or only execution-epoch factors). Indeed, across SCA factors, the mean best fit was 0.93±0.04 (R^2±std). In contrast, wPCA factors were more 'mixed': repeating the above analysis, mean fit was 0.79±0.04 (std). This is roughly as 'mixed' as one expects by chance: mean fit was 0.72±0.11 (std) when randomly rotating the wPCA basis.

Let us make the simplifying assumption that the supervised approach is indeed 'properly supervised' and estimates factors that correspond closely to ground-truth distinctions. Under this assumption, SCA not only comes very close to estimating the ground-truth factors, but performs better than a 'naively supervised' approach that makes reasonable but imperfect assumptions. To illustrate, we applied PCA after dividing the data into just two epoch types (Fig. 3A). This corresponds to a plausible approach if one appreciated the well-known division between preparation and execution factors, but not the recently discovered posture factors. This 'naively supervised' approach recovers factors that mix movement and posture-related computations (Fig. 3D), and thus performs less-well than SCA (Fig. 3B).

Another common supervised approach is demixed PCA (dPCA), which learns factors that explain variance linked to sets of task conditions (reach direction in this experiment). dPCA factors were strongly 'tuned' for reach direction (as expected given dPCA's goal) but mixed tuning related to preparation, execution, and posture (Fig. 3B,E). This is not a flaw of dPCA, but simply reflects the fact that the form of demixing it employs does not map onto the present goal.

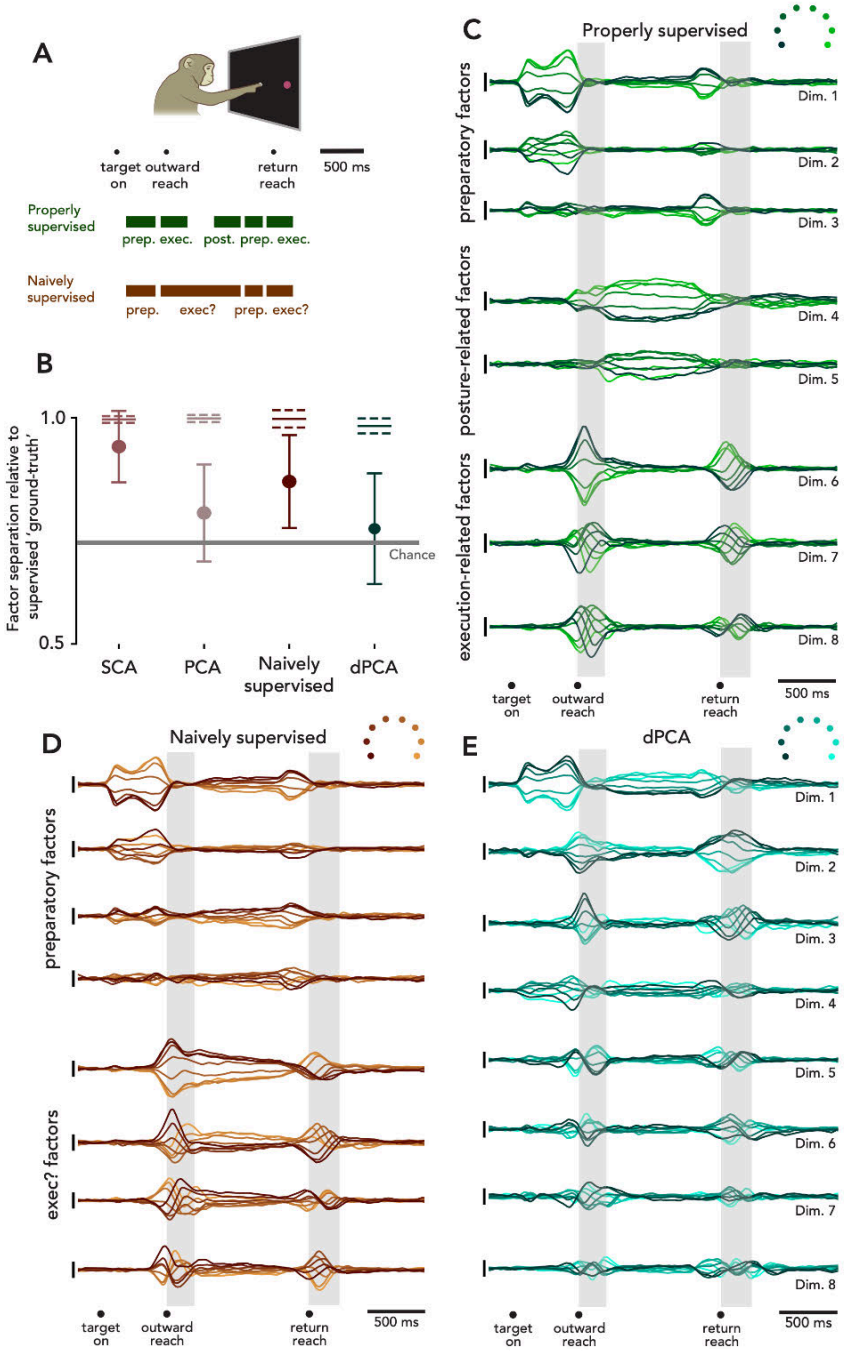


Figure 3. Quantitative comparisons of SCA with supervised approaches. A. Schematic illustrating how supervision[25] was used to identify orthogonal subspaces, one per task epoch. 'Proper' supervision uses known epochs when preparation, execution, and posture-related activity occur. 'Naive' supervision uses only two epochs, and ignores the fact that activity after movement onset may include both execution and posture-related activity. **B.** Factors discovered via 'proper' supervision formed three categories (preparatory, execution, posture). Each factor found by the other methods was reconstructed via regression against properly supervised factors, limited to only one category (whichever performed best). Mean and standard deviation of reconstruction performance (R^2) were computed across 100 resampled populations. The resulting 'factor separation' metric will be near unity if a method discovers the same factor categories as proper supervision. Factor separation was higher for SCA versus PCA ($p<0.01$ via bootstrap), dPCA ($p<0.01$), and naive supervision ($p=0.01$). Lines with flanking dashed traces (at *top*) show reconstruction performance when using all supervised factors. The difference in R^2 when using all properly-supervised factors, versus one category, was smaller for SCA versus PCA ($p < 0.01$), dPCA ($p = 0.01$), and naive supervision ($p = 0.01$).. Chance-level factor separation was estimated by reconstructing randomly rotated PCA factors. **C.** Eight 'Properly supervised' factors, ordered by variance explained within supervised categories. **D.** Same for eight 'Naively supervised' factors. **E.** Same for eight dPCA factors.

Center-out reaching - practical considerations

SCA has three hyperparameters: the requested number of factors; $\lambda_{\mathrm{sparse}}$ (which scales the penalty for non-sparse factors); and λ_{orth} (which scales the penalty for non-orthogonal dimensions when reconstructing neural responses from the factors). The key outcome above – distinct preparatory, execution, and posture-related factors – was robust to relatively large changes in all hyperparameters (Fig. S4).

The primary impact of hyperparameters regarded a subtler issue: the degree to which activity during outwards and return reaches occurred within the same dimensions. The original analysis (Fig 2E) requested eight dimensions, and the same execution-related dimensions (6-8) captured activity during outward and return reaches. In contrast, when requesting twenty-four dimensions, outward- and return-reaches shared some execution dimensions but not others (Fig. S5A monkey B; S6 monkey A). This is largely expected, as return reaches are considerably slower (Fig. 2B; 29% and 52% longer durations and 30% and 49% lower peak speeds, monkeys B and A, respectively) and are thus likely to use overlapping-but-not-identical dimensions (including preparatory dimensions, as noted above). Dimensions indeed differed when assessed independently of SCA. The alignment index[25] (which reaches unity when activity uses identical subspaces) between outward- and return-reach activity was 0.43. Alignment was thus far from complete, but also much larger than for truly distinct factor-sets (e.g. alignment was 0.1 for delay-period activity versus outward-reach-related activity, in agreement with prior results[25,26,28]). When more dimensions are requested, SCA is able to capture this partial overlap of execution factors. At the same time, for practical reasons, one might prefer to request fewer factors, which forces SCA to focus on the more-complete divisions: between preparatory,

execution-, and posture-related factors. Importantly (and as will be discussed further below) requesting too many factors does not typically yield 'spurious' sparsity. There should thus usually be a reasonable range that can be leveraged to request a more consolidated (fewer factors) versus more exhaustive (more factors) view of the data.

A larger practical concern relates to the orthogonality of the dimensions used to reconstruct the data from the factors. Values of λ_{orth} near the default promote approximately orthonormal dimensions. So long as not too many dimensions are requested, orthogonality is also modestly encouraged by the first term in the cost function; ten-dimensional neural data will be better reconstructed using ten nearly orthogonal dimensions versus ten nearly co-linear dimensions. Of course, in practice, one may have little idea of the true dimensionality of the data. One thus desires that SCA should not identify 'overly sparse' factors, even if requesting too many dimensions. Indeed, when using default values of λ_{orth}, we never found a case where SCA over-sparsified the data. In contrast, this did occur if λ_{orth}= 0 and too many factors were requested. For example, activity during outward and return reaches becomes split into largely separate factors (Fig. S5), in conflict with the fact (known from the alignment-index) that these subspaces overlap considerably. This same over-sparsification occurs when applying Independent Component Analysis (ICA), a well-known unsupervised approach without orthogonality constraints. ICA seeks to minimize mutual information between factors[54] and can yield factors that are sparse in time[55]. Requesting too many ICA dimensions results in variance being 'spread out' across all dimensions (Fig. S7). In contrast, for default values of λ_{orth}, SCA behaves as desired: 'extra' dimensions largely reflect noise.

A related concern is that, by seeking sparsity, SCA could overemphasize 'non-normal' dynamics: activity rotating from dimension-one into dimension-two, then into dimension-three without returning[56–58]. In practice, such overemphasis appears minimal. SCA recovered the true oscillatory / multiphasic latents when applied to activity from an RNN that used such dynamics (Fig. S8). In contrast, the empirical execution factors in this monkey (e.g. Fig. 2E, dimension 6) were only modestly multiphasic, consistent with rotations that possess a large non-normal component. In contrast, for a second monkey, SCA recovered strongly oscillatory latent factors (Fig. S8), much as it did for the RNN. This suggests that rotational dynamics can be closer to non-normal or oscillatory depending on the animal, and that SCA can help identify this heterogeneity.

Center-out reaching - Further comparisons with unsupervised approaches

There exist many approaches to dimensionality reduction. Some can encourage sparsity, or can be modified to do so, under at least some circumstances. As noted above (Fig. S1) such methods can sometimes perform nearly as well as SCA when estimating ground-truth factors, though

they are less consistent. We further explored how various methods perform in the context of center-out-reaching data (Figs. S9-S11). Unlike in the main analyses, we did not subtract the cross-condition mean, partly to provide a complementary test-case but also because one method (non-negative matrix factorization, NMF) requires non-negative data, and thus would be unable to be used in our previous example. The neural data thus contain condition-invariant signals in addition to the direction-dependent signals analyzed above.

As above, SCA identified distinct factors reflecting preparation, execution and posture. SCA identified additional factors that were largely condition-invariant, including the putative 'trigger signal' that changes abruptly before both outward and return reaches[28,59]. A variety of methods (PCA, factor analysis, and sparse PCA) did not demix; factors mixed preparation, posture, and/or execution. Additionally, factor analysis and sparse PCA largely failed to isolate condition-invariant signals.

Three methods (ICA, NMF, and PCA followed by a non-standard varimax rotation[60]) implicitly encourage sparsity and thus demixed factors, although not always quite as effectively as SCA. Concentration-by-epoch, computed as above, was 1.05, 1.30, and 1.12 (ICA, NMF, PCA+varimax), versus 1.30 for SCA. The similarity amongst these methods is reassuring: all three can encourage sparsity and thus should tend to perform similarly in many situations. Indeed, the non-standard variant of PCA+Varimax (recently employed in[60]), can be viewed as similar to a step-wise approximation of SCA. One would thus hope that they produce similar results, and indeed they often do. For this reason, our software toolbox provides code for PCA+varimax as a fast approximation of SCA.

That said, these different methods are less consistent, and can occasionally produce very different results from SCA (as in Fig. S1). In large part this is because none of these methods seek the two key features (sparsity and orthogonality of dimensions) as directly as SCA. Features peculiar to individual methods are also sometimes relevant (e.g. NMF can only accommodate non-negative factors and loadings, which cannot capture bidirectional modulation – e.g. left-right modulation of posture factors – and thus captures much less variance than the other methods; Fig. S11). As discussed above, ICA often performs similarly to SCA but is more sensitive to requesting the 'right' number of factors because it does not seek orthogonality. Both ICA and NMF tend to over-sparsify the data when many dimensions are requested (Fig. S12), a failure mode that SCA largely avoids.

Overall, no alternative method performed as well as SCA: either concentration was lower (factors were more mixed), condition-invariant signals were less-well isolated, variance captured was lower, and/or a lack of orthogonality could result in misleadingly sparse factors. At the same time, examination of these other methods highlights the importance of sparsity; implicit seeking of sparsity was the common feature of all alternative methods that demixed factors

fairly successfully. SCA has the advantage that its cost function explicitly encourages both sparsity and orthogonality.

Unimanual Cycling

Do more extended movements – those that span seconds – also involve distinct factors reflecting preparation, execution, and posture? As such movements terminate, do they involve preparation to stop? Are distinct stages reused compositionally? For example, does a 7-cycle movement involve the same preparation to start (and stop?) as a 4-cycle movement? Such questions remain open, in part because the nature and timing of these putative processes is unknown. This provides an opportunity to leverage the unsupervised nature of SCA.

We applied SCA to motor-cortex activity as monkeys cycled a hand-held pedal to traverse a virtual track (Fig. 4A)[53,61]. In alternating trial-blocks, instructed by landscape color, progress along the track required cycling forward versus backward. Monkeys began each trial by moving to a defined starting location. Shortly thereafter a target appeared, at a distance corresponding to one-half, one, two, four, or seven movement cycles. Movement was disallowed during a variable delay. Following a go cue, monkeys cycled swiftly (~2 Hz, Fig. 4B) to the target and stopped upon it, awaiting reward. The starting location was set such that the hand started either near the cycle's top or bottom. The hand ended near that same location, excepting reversals during half-cycle movements. Postural muscle activity (Fig. 4C) reflected both overall top-versus-bottom position and cycling direction (which influenced the exact stopping position).

SCA factors formed discrete sets reflecting preparation, steady-state execution, and posture. Figure 4D plots representative examples of each factor-type (second monkey in Fig. S15; all factors plotted in Fig S13 and S16). Preparatory factors were active after target onset and before movement onset. Consistent with a role in preparation, preparatory-factor activity varied with starting position and cycling direction, which together determined the pending change in muscle activity. Preparatory-factor activity was largely insensitive to cycling distance – it was similar for all distances greater than one cycle (Fig S13C and S16C) – supporting the recent conclusion that motor cortex lacks sequence / long-timescale information[28,53]. Steady-state execution factors were continuously active throughout cycling. Consistent with a role in steady-state execution, their activity was rhythmic and reflected cycling direction but not (ignoring a phase-shift) starting position, a feature shared with muscle activity. Posture factors were active between movements, regardless of target presence. Consistent with a role in maintaining posture, their activity reflected the task features that determined postural muscle activity: cycling direction and stopping position.

Additionally, SCA identified 'stopping' factors, active for roughly 500 ms before movement terminated. These presumably reflected preparation to stop, and/or stopping-specific execution

processes. Stopping-factor activity was tuned for ending position and cycling direction, the two features that most influenced the final pattern of muscle activity. For example, the stopping-factor in Figure 4D is active only when forward-cycling terminates at the bottom: during 7- and 4-cycle bottom-start movements and 0.5-cycle top-start movements. Thus, SCA factors formed categories putatively related to preparation, steady-state execution, stopping, and postural maintenance. Within each putative category, both factor timing and factor 'tuning' agreed with that category's hypothesized role.

Unlike during reaching, these factor-categories, and their likely interpretations, were not previously established. At the same time, three of the factor categories – preparatory, execution, and posture – map cleanly onto categories present during reaching, giving them a natural interpretation. Other aspects of interpretation are necessarily tentative. For example, do stopping factors principally reflect preparation to stop, stopping-specific execution computations, or both? Additional experimental manipulations (e.g. cueing stopping position prior to an unpredictable stop-cue) are needed to resolve such questions. The key point here is simply that SCA can identify factors that lend themselves to forming and testing concrete hypotheses, and can sometimes support firm conclusions immediately. For example, factors are indeed used compositionally: 7-cycle and 4-cycle movements involve identical factor-sets, active at corresponding moments, just with a briefer bout of steady-state execution. Even half-cycle movements (whose duration is similar to a long reach) mostly reuse these same factors, with heavily truncated execution. In broad strokes, this agrees with conclusions drawn in[53]. Yet the presence of distinct factor-groups, used compositionally, was invisible to that study because it focused on single-neuron properties and PCA-based trajectories.

Although SCA factors extend what was known, their properties also fit reassuringly with independent findings. This was true not only regarding the factor-categories SCA identified, but also factor-categories that could have been present but weren't. For example, SCA did not identify top-start-specific or bottom-start-specific steady-state-execution factors. In agreement, subspace alignment was high during top-start and bottom-start conditions (0.85±0.05; mean and standard deviation across pairs of top-start/bottom-start conditions). Similarly, steady-state execution factors were not cycle-specific: they repeated across all middle cycles, in agreement with conclusions based on single-neuron activity and PCA-based trajectories[53,61]. Thus, SCA does not 'invent' sparse factors when they do not exist (this is discouraged by the first term in the SCA cost function). Yet SCA does identify situation-specific factors in situations where independent analyses suggest they likely exist. Many SCA steady-state-execution factors were specific (or partially specific) to forward versus backward cycling (Fig. S13A, S16A). This agrees with prior PCA-based results[61,62] and with the partial subspace alignment between forward- and backward-cycling activity (0.40±0.03 for this dataset). Stopping-factor activity also agrees with (and indeed helps explain) prior[53] and current (Fig. S14) alignment-based results.

Notably, the sets of factors found by SCA are not pre-determined but emerge naturally; SCA does not use labels indicating what factor types should be sought. This explains why dPCA (a common and powerful method for demixing) is inappropriate in this context (Fig. S17). dPCA leverages labels regarding time and condition type. dPCA will typically not demix if those labels (e.g. cycling direction, distance, and starting position) don't correspond to computational processes (e.g. preparing, steady-state-execution, stopping, posture).

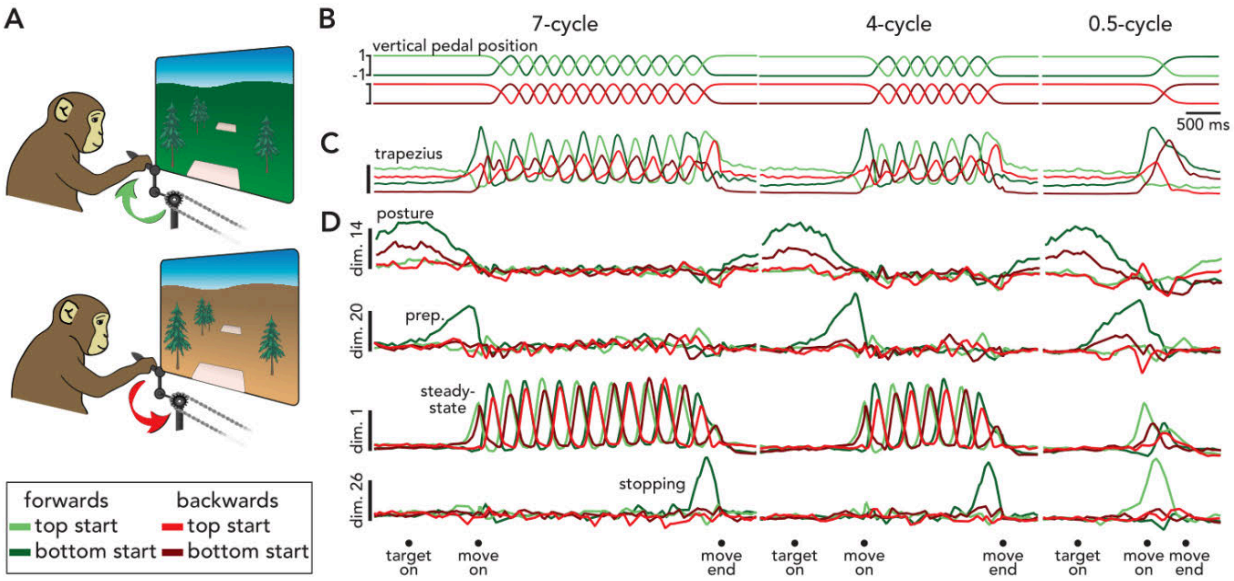


Figure 4. SCA identifies preparatory, execution-related, posture-related, and stopping-related factors in motor cortical unimanual cycling data. A. Task schematic. Monkeys pedaled forward (*top*) or backward (*middle*) from an initial target to a final target. Legend at bottom applies to all subsequent panels. **B.** Vertical pedal position for twelve total conditions: forward (*top, green*) and backward (*bottom, red*) cycling, starting at the top (*light traces*) or bottom (*dark traces*), across three distances. **C.** Trial-averaged EMG activity from the trapezius muscle for the same conditions shown in panel B. **D.** Four SCA-identified factors (corresponding to unordered dimensions 14, 20, 1 and 26) are shown for those same conditions. Factors were chosen to represent each category: posture-related, preparatory, steady-state execution, and stopping, respectively.

Bimanual cycling

We applied SCA to motor-cortex recordings, from both hemispheres, during a bimanual cycling task[44]. Data included previously analyzed conditions where only one arm moved, and not-yet-analyzed conditions where both arms move simultaneously. Monkeys cycled to a target (seven cycles distant), either forward or backward, from a starting location with the hand either

at the cycle's top or bottom. Monkeys cycled with the left arm, right arm, or bimanually (Fig. 5A). For bimanual movements, the arms were either in phase (bimanual-0) or out-of-phase (bimanual-180). Unlike a bicycle, moving one pedal did not cause the other to move. Thus, left-arm muscle activity (Fig. 5B, left) simply reflected what the left arm was doing, with little dependence on right-arm movement.

In contrast to the muscles, most M1 neurons were active during movements of either arm (Fig. 5B)[44,45,63,64], even though descending drive is primarily contralateral. A potential resolution to this seeming paradox is distinct left-arm and right-arm factors (with potentially distinct downstream effects) as found by two recent studies[44,45]. Can SCA reproduce this result without supervision? If so, can it address whether bimanual cycling leverages compositionality? A recent study, in which monkeys countered perturbations[46], found a compositional solution within-hemisphere, as did a study of finger movements[65]. Is bimanual cycling generated by superposition of left-arm and right-arm factors? Is this apparent in a population spanning both hemispheres? Are there additional 'bimanual specific' factors?

We applied SCA to steady-state activity (i.e., from cycles 2-6) from 586 neurons, during two cycling directions, two starting positions, and four handedness requirements. SCA reproduced, without supervision, the presence of distinct left-arm and right-arm factors. Left-arm factors were active when the left arm moved alone (Fig. 5C, the *purple trajectory is large*) but not when the right arm moved alone (Fig. 5C, the *orange trajectory* is very small). Right-arm factors (Fig. 5D) had the complementary property (the *orange trajectory* is large and the *purple trajectory* is small). During bimanual movements, left-arm and right-arm factors became simultaneously active (*light-green* and *dark-green trajectorie*s are sizable in both Fig. 5C and D). This was true not only of these example factors but in general (Fig. S18). If a factor was active when one arm moved alone, it was rarely active when the other arm moved alone, yet became active again during bimanual movements, consistent with bimanual movement reusing unimanual computations. In contrast, there was no evidence of reuse between unimanual conditions (almost no factors were shared between left-arm only and right-arm only cycling) even though there certainly could have been factors reflecting shared abstract parameters such as direction[63]. Results were similar for a second monkey (Fig S19, S20).

Properties of SCA factors were confirmed by subspace-alignment analyses (Fig. 5F). There was near-zero alignment between left-arm-only and right-arm-only conditions, in agreement with SCA finding distinct left-arm and right-arm factors. Alignment was high between top-start versus bottom-start conditions, consistent with SCA finding essentially no 'top-start specific' or 'bottom-start specific' factors. Alignment was high (but less so) between bimanual and unimanual conditions, consistent with bimanual conditions reusing unimanual factors (but also, as described below, with the presence of some bimanual-specific factors).

Total neural variance during bimanual conditions was lower than the sum of the variances during component unimanual conditions (Fig. 5F, *left*). This was true of most individual dimensions (Fig. 5E) and is reflected in the modestly smaller bimanual-condition orbits (Fig. 5C,D; *dark and light green trajectories*). This replicates an effect observed when countering perturbations[46] and when moving multiple fingers[65]. It is presently unclear whether this is a simple consequence of neurons having limited dynamic ranges, or whether it reflects a feature or limitation of compositionality. Also in violation of idealized superposition, a few SCA factors were active only during bimanual movement. The unsupervised nature of SCA is thus helpful in highlighting features that a supervised method might miss, unless the user were aware of such potential nuances.

Along similar lines, SCA identified, primarily for monkey F, a handful of 'ramping' factors (Fig. S20). These agree with a behavioral idiosyncrasy that was pronounced in monkey F: speeding up across the last few cycles when approaching the final target. A prediction, derived from prior findings[66], is that increasing speed will yield a helical motor-cortex population trajectory. This prediction was confirmed by plotting SCA-derived factors in state-space (Fig. S21).

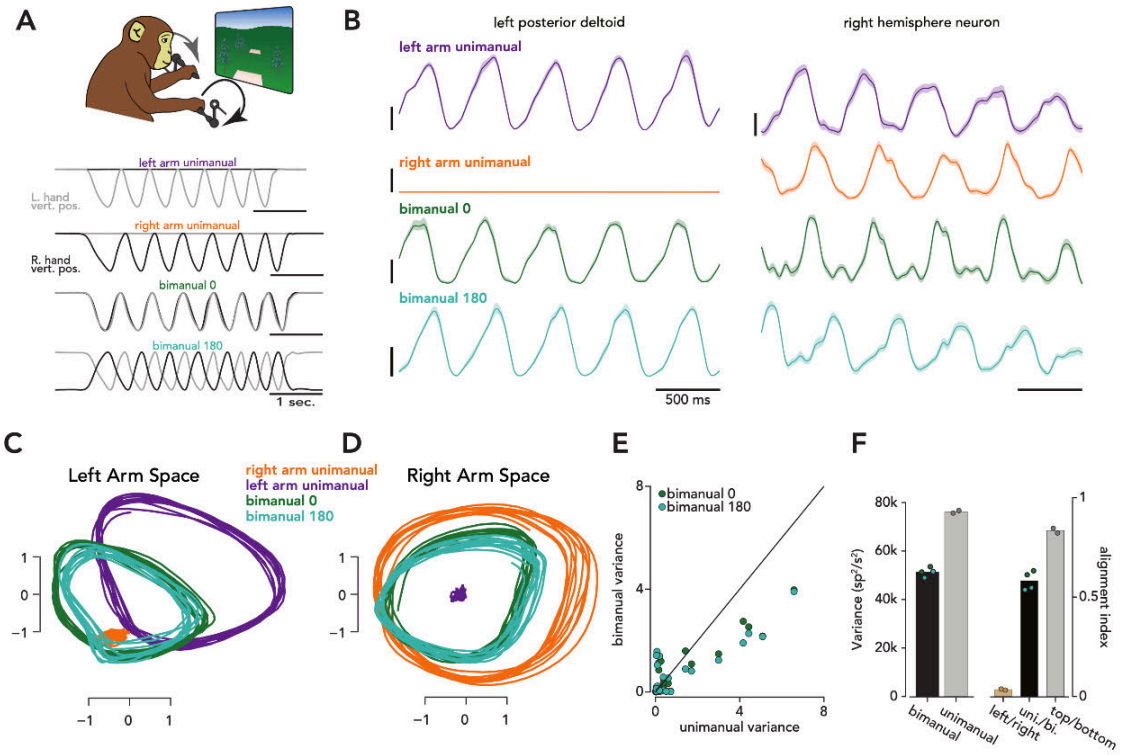


Figure 5. SCA identifies distinct left-arm and right-arm factors, used compositionally during unimanual and bimanual cycling. A. *Top*: Cycling through a virtual environment required using one or both pedals. *Bottom*: Vertical hand position is plotted for both hands for four trials (*rows*), illustrating the four handedness requirements: left-arm unimanual, right-arm

unimanual, bimanual 0, and bimanual 180.**B.** *Left.* Trial-averaged EMG (with standard error) from the posterior deltoid during four conditions. *Right.* Trial-averaged firing rate (with standard error) for an example motor-cortex neuron, recorded from the right hemisphere. **C.** Activity of two 'left-arm' factors (that captured the most left arm unimanual variance) identified by SCA, plotted during forward cycling. Traces correspond to steady state cycling across the four handedness requirements (different colors). Top-start and bottom-start conditions yield two highly overlapping traces per color. **D.** Same as panel C but for right-arm SCA factors. **E.** Bimanual versus unimanual variance for forward cycles (summed for left and right arms) for 50 SCA dimensions (one per symbol). **F.** *Left*: total neural variance during bimanual conditions is less than expected given the sum of variance from corresponding unimanual conditions. *Blue and green dots* correspond to variance during a single bimanual condition (four total conditions: bimanual 0 and 180, starting at the top and bottom), *Gray dots* correspond to top-start and bottom-start conditions, and give the sum of the variances for left-arm unimanual and right-arm unimanual conditions. Bar heights correspond to means across dots. *Right*: Alignment indices, using forward cycles, between left versus right unimanual conditions (*tan,* dots correspond to top-start and bottom-start conditions), unimanual versus bimanual conditions (*black,* four dots correspond to pairwise comparisons of the two 0/180 bimanual conditions, versus the two left/right unimanual conditions), and top-start versus bottom-start unimanual conditions (*gray,* dots correspond to left and right unimanual conditions).

SCA identifies factors related to specific mating behaviors in *C. elegans*

In mammalian cortex, neurons often exhibit heterogeneous responses with 'mixed selectivity'[67–73], consistent with single-neuron rates reflecting random projections of population factors[39,74–76]. To probe SCA's utility when neurons are more specialized, we considered single-trial data recorded during *C. elegans* mating[77], a complex behavior composed of discrete behavioral motifs (Fig. 6A). Prior work has mapped the individual neurons in the posterior brain of male *C. elegans* (the ganglia responsible for generating mating-related behaviors[78–80]) and found that the activity of many neurons is tightly linked to a few (or even a single) behavioral motif[77,81–83].

We analyzed calcium-imaging data from the posterior brain of eight male worms (49-54 simultaneously recorded and identified neurons, Fig. 6B, *bottom traces*). Behavior throughout the trial (i.e., a mating bout) was classified into one of twelve motifs (see *Methods*, Fig. 6B, *top*). We applied SCA to the neural activity of each individual worm separately. As in all analyses, SCA was unsupervised: i.e. was not provided with information such as when behavioral motifs occurred.

Activity in many individual SCA dimensions was tightly related to a single behavioral motif. Consider the top row of Fig. 6C, dimension 5 (for all SCA factors from all worms see Figures S22, S23). This factor's magnitude (*white trace*) is low except around vulva detection (*orange rectangles*). The three remaining factors (Fig. 6C, dimensions 1, 10, and 11) have similarly tight relationships with the onset of copulation (*red)*, turning (*blue*), and excretory pore detection (*black*), respectively. The loadings learned by SCA are consistent with previously described functional roles of many individual neurons (Fig. 6C, *right*). Neurons PCB, PCC, and HOB are sensory neurons known to contribute to vulva detection[77,82], and the factor that is active during vulva detection contributes heavily to these neurons

(Fig. 6C, *top trace and leftmost column*). Similarly, the factor related to turning contributes heavily to neuron PVV (Fig. 6C, *third trace and third column*), in agreement with prior work demonstrating that PVV drives turning behavior during mating[77].

We measured the correlation between each SCA factor and all behaviors (see *Methods*). A given SCA factor tended to be closely related to just one behavior (Fig. 6D, *top*). In contrast, most wPCA factors were weakly correlated with multiple behaviors (Fig. 6D, *bottom)*. Across all SCA factors, in all worms, the mean correlation between the factors and the most closely related behavior was 0.47 ±0.02 (standard error) (Fig. 6E). Performing the same analysis with PCA factors yielded significantly lower correlations (0.37 ±0.01, $p < 0.001$, rank sum test), indicating that the SCA factors bore a closer relationship to the timing of individual motifs than did wPCA factors.

We observed consistency across worms in many motif-related factors and their loadings (Fig. 6F). For example, we found SCA factors that were closely related to copulation in all worms that exhibited this behavioral motif (six of eight, Fig. 6F). To quantify the similarity of loadings, we measured the angle between the loading vectors of individual worms (Fig. S24). When compared to the angle between wPCA loadings, we found a modest but highly significant difference between SCA angles and wPCA angles (Fig. S24), indicating that the SCA loadings are indeed more replicable across worms than wPCA loadings. In fact, learned loadings could even generalize across worms - we could use SCA loadings from one worm to predict behavior in other worms (Fig S25). Qualitatively, SCA recovered similar relationships between individual neurons and particular motifs, across worms. For example, for copulation-related factors, SCA tended to learn large weights for neurons SPC, SPD, and HOA, neurons that are known to play an important role in copulation (Fig. 6G, *red box*)[77,81]. Similarly, for turning-related SCA factors (Fig. 6G, *blue box*), SCA recovered large weights for neuron PVV, a neuron known to be important for turning. Thus, SCA results are replicable and recapitulate prior knowledge about the roles of individual neurons during mating.

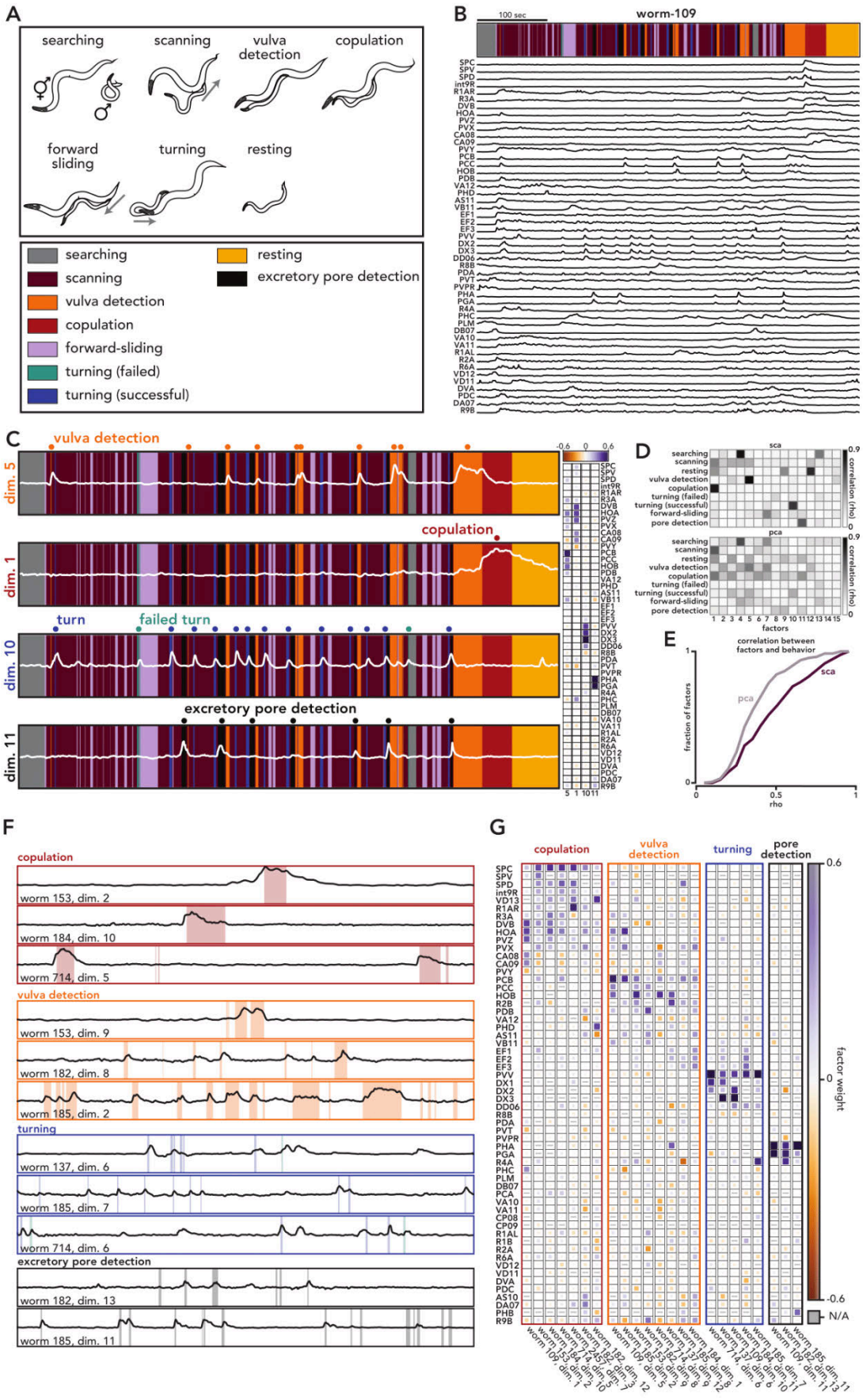

A
searching
scanning
vulva detection
copulation
forward sliding
turning
resting
searching
scanning
vulva detection
copulation
forward-sliding
turning (failed)
turning (successful)
resting
excretory pore detection
B
100 sec
worm-109
C
vulva detection
dim. 5
copulation
dim. 1
turn
failed turn
dim. 10
excretory pore detection
dim. 11
D
sca
pca
searching
scanning
resting
vulva detection
copulation
turning (failed)
turning (successful)
forward-sliding
pore detection
correlation (rho)
factors
E
correlation between factors and behavior
fraction of factors
rho
sca
pca
F
copulation
worm 153, dim. 2
worm 184, dim. 10
worm 714, dim. 5
vulva detection
worm 153, dim. 9
worm 182, dim. 8
worm 185, dim. 2
turning
worm 137, dim. 6
worm 185, dim. 7
worm 714, dim. 6
excretory pore detection
worm 182, dim. 13
worm 185, dim. 11
G
copulation
vulva detection
turning
pore detection
factor weight
N/A

Figure 6. SCA identifies factors related to specific mating motifs in *C. elegans*. A. Behavioral motifs exhibited by worm-109. Continuous behavior was automatically classified; not all worms exhibited all behavioral motifs. **B.** *Top*: ethogram for a single mating bout (worm-109). *Bottom*: Activity of forty-nine neurons during that bout. **C.** Left: Four SCA factors during that same bout - listed dimension numbers are intrinsically unordered. Factor-activity most closely corresponds to vulva detection, copulation, turning, and excretory pore detection (top to bottom, respectively). *Right*: Loadings describe the impact of these four SCA factors on single-neuron activity. **D.** Correlations between all worm-109 factors and all behaviors, for both SCA and PCA factors. **E.** For each SCA (or PCA) factor, we found the behavior with which it was most strongly correlated and took that correlation. Cumulative distributions plot those correlations across all factors and worms. Individual SCA factors had overall tighter relationships with individual behaviors, resulting in a rightward shift of the distribution ($p < 0.001$, rank-sum test, *Methods*). **F.** Example SCA factors related to specific behavioral motifs. **G.** Loadings for SCA dimensions related to the same behavioral motif. There are different numbers of columns for each motif because not all worms exhibited the same motifs, and SCA did not always identify a dimension that clearly corresponded to a particular motif. The latter is partially due to different combinations of neurons being recorded across worms. *Gray dashes* indicate neurons that were not recorded.

Identifying compositionality in a multitask RNN

Driscoll *et al.*[13], extending work by Yang *et al.*[37], found that single networks, trained to perform fifteen cognitive tasks, learned a small number of component computations. These computational building blocks were used across multiple tasks and allowed networks to quickly learn new tasks. Confirmation of compositional reuse required analyses of linearized dynamics. In some cases, the presence of distinct building blocks was apparent directly from activity. This was particularly true when network units used non-negative activation functions (softplus or rectified tanh) and individual units primarily participated in a small number of component computations. For other activation functions (e.g. tanh), units tended to be active across many component computations (Fig. 7B), perhaps similar to the mixed selectivity prevalent in the frontal lobe. In such situations, might the presence of distinct computations still be discerned, but at the level of population factors rather than individual neurons?

We applied SCA to population activity from a tanh network. For all tasks, the network received directional inputs (the cosine and sine of different angles) and generated a 2-dimensional directional output (Fig. 7A; see *Methods* and Fig. S26 for documentation of each task). SCA factors reflected the compositionality present in the network, and pointed towards a computational strategy used to solve the tasks. To illustrate this strategy, we highlight the activity of three SCA factors during nine tasks (Fig. 7C, Fig. S27 shows all recovered factors during all tasks). The network's putative strategy can be summarized as follows: use one set of factors to consolidate incoming stimuli, another to reflect the 'internal' decision, and additional condition-invariant 'trigger' factors to determine when activity flows into output-potent dimensions. We describe each factor-type in turn.

Directional inputs employed two 'channels' (Fig. 7A, *stimulus 1 and 2*). Some tasks used only one channel per trial (e.g., Delay-pro, Compare-stim1 and -stim2). In others (e.g., ContextCompare-stim1), both channels were simultaneously active and one had to be ignored. SCA identified 'stimulus factors' that reflected the relevant stimulus, regardless of input channel. For example, activity in dimension 1 (Fig. 7C) tracks stimulus timecourse for either or both channels, as appropriate depending on the task. SCA also identified additional channel-specific dimensions (Fig. S26, dimensions 3 and 4).

A second set of factors (e.g., Fig. 7C, dimension 6) reflected the internal decision – i.e. the signal that eventually determined network output. To appreciate how these factors behave, consider activity during Memory-pro and Memory-anti tasks, both of which involve a brief directional stimulus. In the Memory-pro task, the network must report this same direction after the go cue. In the Memory-anti task, the network must report the direction opposite the stimulus. After the stimulus is removed, activity grows in dimension 6, but with opposite ordering for pro and anti tasks, reflecting what will become the ultimate 'decision.' During Compare- and ContextCompare tasks, activity becomes strong only after the second epoch, because the decision requires comparing information across both.

A third set of factors reflects output timing. For example, dimensions 17 changes suddenly, in a similar way for all conditions and tasks, following the go cue. This is reminiscent of the condition-invariant 'trigger signal' observed during reaching in non-human primates[59], decision making in rodents[40], and speech in humans[76]. The hypothesized role of the trigger signal is to translate activity to a region of state-space where local dynamics cause output-null activity (here decision-factor activity) to impact output-potent dimensions. Such a clear parcellation of the network's underlying computations cannot be gleaned from single-unit activity (Fig. 7B) nor from PCA factors (Fig. S28).

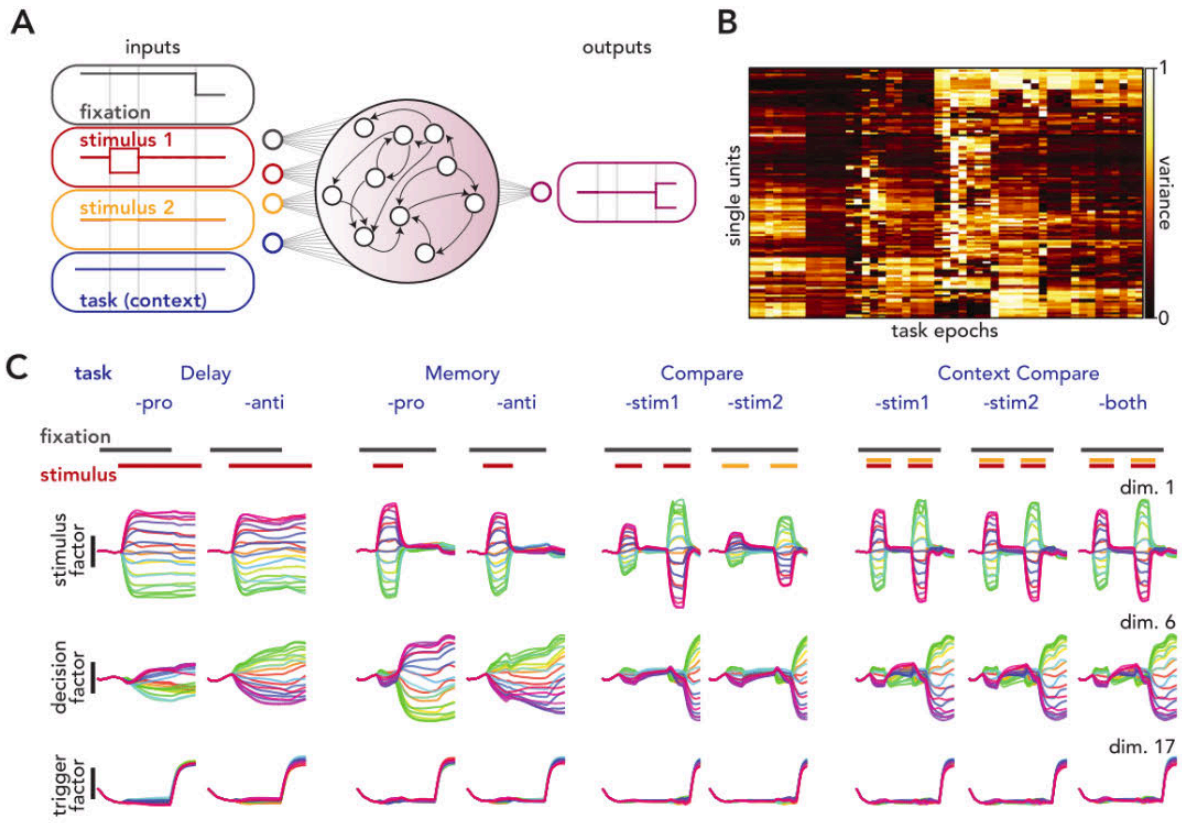


Figure 7. SCA identifies compositionality in a multitask network. A. Network architecture. Stimuli arrived in two 'channels' (analogous to two stimulus modalities), each consisting of two dimensions. **B.** Normalized firing-rate variance (across time and conditions) of individual units, for different task epochs. Epochs were delineated by the timing of the network inputs; the number of epochs differed across tasks. Individual units were typically active across many task epochs, preventing the underlying computations (and their compositionality) from being apparent at the individual-unit level. **C.** Activity in three SCA dimensions during nine tasks, with traces colored by stimulus angles. The first factor (*top row*) reflected the relevant stimulus, the second (*middle row*) reflected the network's ultimate decision, and the third (*bottom row*) reflected the timing of network output. During the 'Delay-pro' and 'Delay-anti' tasks, the network received a directional stimulus and generated an output in the same, or opposite, direction after receiving a 'go cue'. 'Memory-pro' and '-anti' tasks were the same as the 'Delay-' tasks, except the network did not receive a continual directional stimulus. The remaining five tasks required the network to compare the amplitude of two successive stimuli and respond in the direction of the larger of the two. For 'Compare-stim 1' and 'Compare-stim 2', the network compared the amplitude of two stimuli, both delivered through the same input channel (i.e. delivered either via *stimulus 1* dimensions or via *stimulus 2* dimensions). For the 'ContextCompare-' tasks, the network received stimuli from both pairs of stimulus channels (both *stimulus 1* dimensions and *stimulus 2* dimensions) and needed to either ignore one channel ('ContextCompareStim 1' and 'ContextCompareStim 2') or compare across both pairs of inputs ('ContextCompareStimuli').

Nonlinear SCA

Depending on the application, one may wish to allow nonlinear relationships between latent factors and neural responses, while maintaining interpretability. We thus created a nonlinear extension of SCA, which allows learning flexible mappings from latents to neural activity through a neural network within its architecture (Fig. 8A). This approach can be viewed as a sparse autoencoder with added orthonormality penalties. The cost function is almost identical to that of linear SCA, although we here encourage orthonormality in the weights in all layers of the neural network mapping latents to neural activity, which continues to facilitate discovery of distinct computations (Fig. 8A, see Methods).

When applied to unimanual cycling data (Fig. 8C, Fig. S29), non-linear SCA identified sets of latents – corresponding to preparation, steady-state movement, posture, and stopping – similar to linear-SCA factors. Thus, the presence of these distinct processes is supported by both linear and nonlinear methods. However, non-linear SCA needed fewer latents of each type, and could thus achieve better reconstruction accuracy (on held-out data) for a given number of dimensions (Fig. 8B). Nonlinear SCA required only 10 dimensions, as opposed to 40 for linear SCA, to achieve an R^2>0.7. One straightforward reason for this savings relates to the fact that many neurons had steady-state execution firing rates that were essentially nonlinearly distorted versions of a sinusoid (e.g. sinusoidal but non-negative).

Both sparsity and orthonormality played an important role in discovering factors reflecting distinct computations. A standard nonlinear autoencoder, with an equivalent architecture but no sparsity or orthonormality constraints, produced latents that mixed underlying task computations (Fig. 8D, Fig. S30). Moreover, encouraging orthonormality prevented 'oversparsification' of splitting computations into multiple latent factors (Fig. S32,S33).

Quantitatively, nonlinear SCA's predictive performance was actually superior to that of a standard autoencoder, especially for larger numbers of dimensions. This was likely due to greater overfitting within the standard autoencoder. Thus, beyond improving interpretability of the latent factors, the SCA sparsity penalty may be serving as an effective regularizer. This suggests that SCA's encouragement of sparsity does more than produce a basis that aids interpretation; if the true factors tend to be non-synchronous, seeking this feature may improve the accuracy with which latents are estimated.

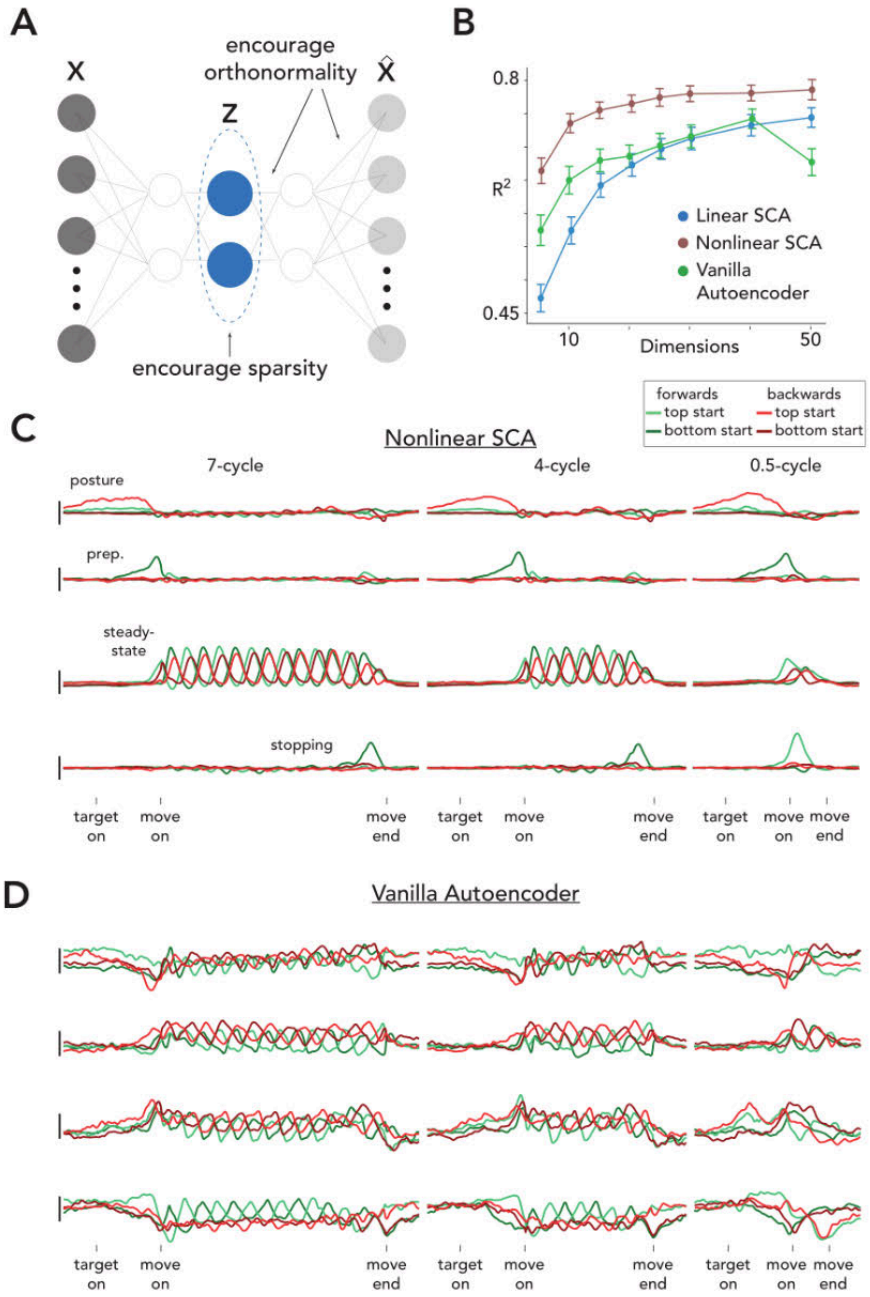


Figure 8. Nonlinear SCA discovers demixed factors and improves reconstruction accuracy in unimanual cycling data. A. Architecture schematic of nonlinear SCA. **B**. Reconstruction accuracy on held-out data, as a function of the number of latent dimensions, using recordings from the unimanual cycling task. We compare nonlinear SCA, a vanilla autoencoder (which has no sparsity or orthonormality penalties), and linear SCA. **C.** Four example latent factors found with nonlinear SCA, and their putative role, as in Fig. 4D. **D.** Four example latent factors from a vanilla autoencoder.

Discussion

It is increasingly appreciated that a given neural population often performs multiple computations, deployed flexibly and/or compositionally. Based on first principles, sparsity and orthonormality should be helpful for demixing neural computations. SCA's cost function is a straightforward implementation of these two goals. Empirically, SCA performed well across a range of datasets, confirming prior findings and revealing new ones. Without supervision, SCA reproduced the well-established distinction between preparation and execution. SCA additionally revealed posture factors that track hand position across reaches. Across outbound and return reaches, SCA factors reveal a straightforward instance of compositionality: lengthy preparation, execution, postural maintenance, brief preparation, then execution that returns the hand to its original posture. While easy to appreciate in retrospect, this series of events is challenging to discern at the single-neuron level. There was also no guarantee that prior findings, which depended upon assumptions regarding when preparation and execution occur, would be so closely replicated by a method that makes no such assumptions.

During cycling, SCA revealed that extended and abbreviated movements reuse the same factor types, similar to those used during reaches but with the addition of stopping-specific factors. SCA confirmed that compositional reuse of left-arm and right-arm factors, described when countering perturbations, extends to continuous cycling. SCA also revealed additional bimanual-specific aspects of activity. During worm mating, SCA recovered factors that correspond to established single-neuron properties, confirming that the concept of latent factors can apply to populations with both mixed and (relatively) unmixed selectivity. In a multi-task network, SCA parcellated activity into largely distinct computations, something that had previously been challenging when networks display mixed selectivity.

Every dataset we examined contained examples of distinct sets of factors, often used flexibly and/or compositionality. The prevalence of such structure, across species, brain areas and tasks, argues that SCA is likely to be broadly useful. Further supporting its utility, distinctions discovered by SCA, once known, were readily confirmed with other methods (e.g. by computing subspace alignment). This highlights that SCA can be used either as a primary method or as a means of discovering factor-categories that can then be isolated via supervision. In this context, it is noteworthy that we never found a situation where SCA discovered 'spurious' factors that disagreed with subsequent analyses. For example, when examining situations where SCA did, versus didn't, find distinct factors, subspace alignment (computed using PCA) was appropriately low versus high. This agrees with expectations given SCA's cost function: when ground-truth factors are not sparse, it is unlikely that sparsity can be significantly increased without increasing one or both of the other two terms.

Nonlinear embeddings

Latent factors discovered by nonlinear SCA were qualitatively similar to the dominant factors discovered by linear SCA, during both unimanual cycling and reaching. In fact, using latents found by linear SCA, followed by nonlinear mappings, led to prediction performance nearly as good as that of nonlinear SCA (Fig. S31). This aligns well with recent modeling work [11], where latent factors are defined as linear combinations of neurons' spiking activity (as in PCA or linear SCA) and single-neuron firing rates are nonlinear functions of the latents. In networks where this conceptual framework applies, nonlinear SCA may thus be preferable. At the same time, the broad similarity of findings, between linear and nonlinear SCA, argues that the common use of linear techniques is reasonable in the context of the motor system. However, this is unlikely to be the case in all systems; e.g. linear mappings may be very suboptimal when analyzing hippocampal populations.

Factor interpretation

Intentionally (given its unsupervised nature), SCA does not apply interpretation to the factors it discovers. Categories may naturally emerge (e.g. preparatory versus execution factors) but these are distinctions made by the user, not the method. Whether categories are valid, or are interpreted correctly, requires scientific rather than statistical inference. 'Posture factors' during reaching provide an example. The postural interpretation is suggested by the fact that two-dimensional factor activity roughly maps to two-dimensional hand position. The return of factor activity to baseline, following return reaches, supports that interpretation. Yet even after tentatively accepting the postural interpretation, additional questions remain: does postural activity reflect proprioceptive feedback, posture-maintaining muscle commands, or an internal representation of location? Are the factors that track posture across reaches identical to those found across experimenter-determined baseline postures[51]? Answering such questions requires additional experiments. Related points regard the 'rules' governing when and how factor-categories are used compositionally: specifically crafted experiments are typically needed to answer questions such as whether execution requires[26] versus merely benefits[84] from preparation, or whether rapid sequences involve dedicated computations[52] versus reuse those used during isolated reaches[28].

Interpretation of SCA factors will also necessarily depend upon the appropriateness of SCA's assumptions: that distinct computations use distinct factors, occupying largely orthogonal dimensions. Such properties are not guaranteed to be universal, but are likely common. In situations where expectations (or analysis goals) are different (e.g. aiming to find factors that explicitly map to known experimental variables) one would use different methods[34,85–88]. When SCA's assumptions don't apply, sparse factors may be absent. For example, SCA did not find cycle-specific factors during steady-state cycling, in agreement with prior results in primary motor cortex and in contrast to cycle-specific response aspects in the supplementary motor area[53]. Thus, SCA can reject hypotheses, in addition to generating new ones.

Orthogonality

In combination with penalizing reconstruction error, seeking orthogonality tends, in practice, to result in SCA performing very similarly to PCA from a variance-captured perspective. Knowing that SCA is capturing near-maximal variance reduces concerns that the sparse factors it finds are unrepresentative. In contrast, without orthogonality, and when requesting increasing numbers of dimensions, initially distinct factors will split (and continue splitting) into dimensions with increasingly collinear loadings. We observed this effect empirically both for SCA (when the orthogonality term was eliminated) and for ICA. In such cases, interpreting factors as distinct would be questionable without additional theoretical or empirical justification.

To date, many of the clearest examples of distinct factor categories involve dimensions fairly close to orthogonal[16,17,19,34,38–40]. If learning tends to place different computations in different subspaces (with no attempt to align them), they will naturally be near-orthogonal due to the geometry of high-dimensional spaces. Alternatively (or additionally) orthogonality may reduce interference between computations, allowing them to interact only when dynamics link their dimensions. For these reasons, encouraging orthogonality is generally thought to aid interpretation. Yet it does not eliminate the need to test hypotheses via further analyses / experiments. Beliefs regarding when orthogonal sets of dimensions do or don't map onto distinct biological processes will essentially always depend upon further examination. In situations where expectations of orthogonality don't apply, one might relax this term of SCA's cost function.

Sparsity

The idea of sparsity, in the context of sparse single-neuron responses, has a long history in neuroscience[89]. Existing data analyses leverage this concept by encouraging sparse loadings, as in sparse PCA (sPCA)[90,91], which seeks factors that only involve a subset of neurons and is therefore particularly useful for clustering[92]. Both sPCA and SCA fall under the broad umbrella of dictionary learning[93,94]: methods that decompose a matrix into a product of two matrices, one of which is encouraged to be sparse; prior work has even referred to dictionary learning as sparse component analysis[95,96]. Within neuroscience, dictionary learning is not typically used for estimating latent factors but rather as a preprocessing tool (e.g., for extracting neural activity from calcium imaging data[97–99] or for spike sorting[100]).

Given present goals, we were interested not in sparse loadings, but in latents that are sparsely active despite single-neuron responses that might be non-sparse due to mixed selectivity. For example, it is typical for a neuron that is active during preparation to also be active during execution. Yet preparation and execution are distinct processes, active at largely non-overlapping times, at the population level. By encouraging sparsity, the estimated factors naturally honor these two categories. This is true even though the factors are not truly sparse – simply more sparse than if they were

mixed. That said, we anticipate that highly sparse factor-activity sparsity will sometimes occur: when a given area performs many computations and different experimental conditions engage different computations. Factor sparsity may thus become more prevalent (making SCA particularly useful) as experimental tasks become richer. There have been only a few prior attempts to maximize sparsity of estimated latents within population activity. While ICA does not explicitly seek sparsity, minimizing mutual information between factors[54] can lead to sparse dimensions in practice[55]. Dekleva et al[60] used PCA followed by a varimax rotation of the latents, which also leads to sparse factors, and can be viewed as a stepwise approximation to SCA. Additionally, Tolooshams et al[101] recently developed an approach based on a convolutional dictionary learning generative model to find sparse events ('codes'), rather than continuously occurring latents, that reconstruct neural activity. Interestingly, within the Artificial Intelligence community, there has also been success in using sparsity to find interpretable latents within Large Language Models[102–104].

The goal of maximizing latent sparsity is also notably different from traditional theoretical ideas of sparse coding at the neuron level. Sparse coding typically arises if one encourages an autoencoder to use a sparse latent representation with *more* dimensions than input features[89]. SCA, on the other hand, finds a lower-dimensional representation. We would thus not expect SCA to be particularly insightful when applied to neural activity for which sparse coding is the predominant framework.

Factors and neurons

The degree to which computationally distinct factors relate to physically distinct neurons will vary by brain area and species. In *C. elegans*, where individual-neuron connectivity is preserved across animals, many SCA factors primarily reflect the activity of a small number of neurons. In this system, distinct processes are often reliant on the activity of single neurons (e.g., 'excretory pore detection' requires PHA activity[77]) and can be understood, at the circuit level, in terms of interactions between individual neurons. In contrast, in monkey motor cortex, individual neurons have mixed selectivity, arguing that each neuron participates in many computations and reflects many factors. For example, there exist distinct left-arm and right-arm factors, but relatively few 'right-arm-only' neurons, even in primary motor cortex[44]. This appears to be a common property in much of cortex[69,74,75,105].

Traditionally, one often speaks of a given cortical area as performing 'a computation', or a small set of fixed computations. Yet the remarkable behavioral and cognitive flexibility displayed by the brain presumably involves the ability of areas to switch amongst a multitude of learned computations[106]. Animals' ability to perform well in new situations may depend on compositionality: selecting, from a repertoire of computations, which to use in the present moment. It is precisely in such situations where factor sparsity may be most pronounced. This could be largely invisible at the level of single-neuron responses, which appear 'mixed' and confusing rather than sparse. The results above

reveal that SCA can uncover distinct factor types, presumably with distinct computational roles, even when this is hidden at the single-neuron level. If the above ideas regarding compositionality are correct, the need to recover the underlying computational building blocks will only grow.

Author Contributions

A.J.Z. and J.I.G. developed SCA; A.J.Z. and A.H.L. collected the reaching data; A.A.R. collected the unimanual cycling data; K.C.A. collected the bimanual cycling data; L.D., trained the multitask network; V.S., collected the *C. elegans* data; A.J.Z., X.A., A.W.U., V.S., L.D., M.M.C, J.I.G. performed analyses, A.J.Z., X.A., M.M.C., and J.I.G. wrote the manuscript. All authors contributed to manuscript editing.

Acknowledgements

We thank Y. Pavlova for expert animal care. This work was supported by the Grossman Center for the Statistics of Mind (M.M.C., J.P.C., and L.P.), the Simons Foundation (M.M.C., L.D., J.P.C., and L.P.), the McKnight Foundation (M.M.C. and J.P.C.), NSF Neuronex 520 Award DBI-1707398 (L.P.), the Gatsby Charitable Foundation (L.P. and J.P.C.), NIH DP2 NS083037 (M.M.C.), NIH CRCNS R01NS100066 (M.M.C.), NINDS 1U19NS104649 (M.M.C.), NINDS R01NS135240 (M.M.C.), the National Science Foundation (A.J.Z.), the Kavli Foundation (M.M.C.), NIH R01NS113119 (V.S.), NIH T32 MH067564 (A.W.U), NIH ROO NS119787 (X.A, J.I.G.). This research was also supported in part by grants from the NSF (DMS-2235451) and Simons Foundation (MPS-NITMB-00005320) to the NSF-Simons National Institute for Theory and Mathematics in Biology (NITMB).

Declaration of Interests

The authors declare no competing interests.

References

1. Munoz, D.P., and Wurtz, R.H. (1995). Saccade-related activity in monkey superior colliculus. I. Characteristics of burst and buildup cells. J. Neurophysiol. *73*, 2313–2333.

2. Munoz, D.P., and Wurtz, R.H. (1995). Saccade-related activity in monkey superior colliculus. II. Spread of activity during saccades. J. Neurophysiol. *73*, 2334–2348.

3. Vyas, S., Golub, M.D., Sussillo, D., and Shenoy, K. (2020). Computation Through Neural Population Dynamics. Annu. Rev. Neurosci.

4. Duncker, L., and Sahani, M. (2021). Dynamics on the manifold: Identifying computational dynamical activity from neural population recordings. Curr. Opin. Neurobiol. *70*, 163–170.

5. Shenoy, K.V., Sahani, M., and Churchland, M.M. (2013). Cortical control of arm movements: a dynamical systems perspective. Annu. Rev. Neurosci. *36*, 337–359.

6. Cunningham, J.P., and Yu, B.M. (2014). Dimensionality reduction for large-scale neural recordings. Nat. Neurosci. *17*, 1500–1509.

7. Ruff, D.A., Ni, A.M., and Cohen, M.R. (2018). Cognition as a window into neuronal population space. Annu. Rev. Neurosci. *41*, 77–97.

8. Khona, M., and Fiete, I.R. (2022). Attractor and integrator networks in the brain. Nat. Rev. Neurosci. *23*, 744–766.

9. Williamson, R.C., Doiron, B., Smith, M.A., and Byron, M.Y. (2019). Bridging large-scale neuronal recordings and large-scale network models using dimensionality reduction. Curr. Opin. Neurobiol. *55*, 40–47.

10. Sussillo, D., and Barak, O. (2013). Opening the black box: low-dimensional dynamics in high-dimensional recurrent neural networks. Neural Comput. *25*, 626–649.

11. DePasquale, B., Sussillo, D., Abbott, L.F., and Churchland, M.M. (2023). The centrality of population-level factors to network computation is demonstrated by a versatile approach for training spiking networks. Neuron *111*, 631–649.e10.

12. Mante, V., Sussillo, D., Shenoy, K.V., and Newsome, W.T. (2013). Context-dependent computation by recurrent dynamics in prefrontal cortex. Nature *503*, 78–84.

13. Driscoll, L., Shenoy, K., and Sussillo, D. (2022). Flexible multitask computation in recurrent networks utilizes shared dynamical motifs. bioRxiv, 2022.08.15.503870. https://doi.org/10.1101/2022.08.15.503870.

14. Sussillo, D., Churchland, M.M., Kaufman, M.T., and Shenoy, K.V. (2015). A neural network

that finds a naturalistic solution for the production of muscle activity. Nat. Neurosci. *18*, 1025–1033.

15. Churchland, M.M., Cunningham, J.P., Kaufman, M.T., Foster, J.D., Nuyujukian, P., Ryu, S.I., and Shenoy, K.V. (2012). Neural population dynamics during reaching. Nature *487*, 51–56.

16. Remington, E.D., Narain, D., Hosseini, E.A., and Jazayeri, M. (2018). Flexible Sensorimotor Computations through Rapid Reconfiguration of Cortical Dynamics. Neuron *98*, 1005–1019.e5.

17. Wang, J., Narain, D., Hosseini, E.A., and Jazayeri, M. (2018). Flexible timing by temporal scaling of cortical responses. Nat. Neurosci. *21*, 102–110.

18. Koay, S.A., Charles, A.S., Thiberge, S.Y., Brody, C.D., and Tank, D.W. (2022). Sequential and efficient neural-population coding of complex task information. Neuron *110*, 328–349. e11.

19. Panichello, M.F., and Buschman, T.J. (2021). Shared mechanisms underlie the control of working memory and attention. Nature *592*, 601–605.

20. Sohn, H., Narain, D., Meirhaeghe, N., and Jazayeri, M. (2019). Bayesian computation through cortical latent dynamics. Neuron *103*, 934–947. e5.

21. Churchland, M.M., Cunningham, J.P., Kaufman, M.T., Ryu, S.I., and Shenoy, K.V. (2010). Cortical preparatory activity: representation of movement or first cog in a dynamical machine? Neuron *68*, 387–400.

22. Crammond, D.J., and Kalaska, J.F. (2000). Prior information in motor and premotor cortex: activity during the delay period and effect on pre-movement activity. J. Neurophysiol. *84*, 986–1005.

23. Wise, S.P., Weinrich, M., and Mauritz, K.H. (1986). Movement-related activity in the premotor cortex of rhesus macaques. Prog. Brain Res. *64*, 117–131.

24. Kaufman, M.T., Churchland, M.M., Santhanam, G., Yu, B.M., Afshar, A., Ryu, S.I., and Shenoy, K.V. (2010). Roles of monkey premotor neuron classes in movement preparation and execution. J. Neurophysiol. *104*, 799–810.

25. Elsayed, G.F., Lara, A.H., Kaufman, M.T., Churchland, M.M., and Cunningham, J.P. (2016). Reorganization between preparatory and movement population responses in motor cortex. Nat. Commun. *7*, 13239.

26. Lara, A.H., Elsayed, G.F., Zimnik, A.J., Cunningham, J., and Churchland, M.M. (2018). Conservation of preparatory neural events in monkey motor cortex regardless of how movement is initiated. Elife *7*. https://doi.org/10.7554/eLife.31826.

27. Ames, K.C., Ryu, S.I., and Shenoy, K.V. (2019). Simultaneous motor preparation and

execution in a last-moment reach correction task. Nat. Commun. *10*, 2718.

28. Zimnik, A.J., and Churchland, M.M. (2021). Independent generation of sequence elements by motor cortex. Nat. Neurosci. *24*, 412–424.

29. Churchland, M.M., Yu, B.M., Ryu, S.I., Santhanam, G., and Shenoy, K.V. (2006). Neural variability in premotor cortex provides a signature of motor preparation. J. Neurosci. *26*, 3697–3712.

30. Churchland, M.M., Santhanam, G., and Shenoy, K.V. (2006). Preparatory activity in premotor and motor cortex reflects the speed of the upcoming reach. J. Neurophysiol. *96*, 3130–3146.

31. Kashefi, M., Michaels, J.A., Kersten, R., Lau, J.C., Diedrichsen, J., and Pruszynski, J.A. (2025). Compositional neural dynamics during reaching. bioRxiv. https://doi.org/10.1101/2025.09.04.674069.

32. Riveland, R., and Pouget, A. (2024). Natural language instructions induce compositional generalization in networks of neurons. Nat Neurosci *27*, 988–999.

33. Churchland, M.M., and Shenoy, K.V. (2024). Preparatory activity and the expansive null-space. Nat. Rev. Neurosci. *25*, 213–236.

34. Kobak, D., Brendel, W., Constantinidis, C., Feierstein, C.E., Kepecs, A., Mainen, Z.F., Qi, X.L., Romo, R., Uchida, N., and Machens, C.K. (2016). Demixed principal component analysis of neural population data. Elife *5*. https://doi.org/10.7554/eLife.10989.

35. Aoi, M.C., Mante, V., and Pillow, J.W. (2020). Prefrontal cortex exhibits multidimensional dynamic encoding during decision-making. Nat. Neurosci. *23*, 1410–1420.

36. Lara, A.H., Cunningham, J.P., and Churchland, M.M. (2018). Different population dynamics in the supplementary motor area and motor cortex during reaching. Nat. Commun. *9*, 2754.

37. Yang, G.R., Joglekar, M.R., Song, H.F., Newsome, W.T., and Wang, X.-J. (2019). Task representations in neural networks trained to perform many cognitive tasks. Nat. Neurosci. *22*, 297–306.

38. Kirk, E.A., Hope, K.T., Sober, S.J., and Sauerbrei, B.A. (2023). An output-null signature of inertial load in motor cortex. bioRxiv, 2023.11. 06.565869.

39. Warriner, C.L., Fageiry, S., Saxena, S., Costa, R.M., and Miri, A. (2022). Motor cortical influence relies on task-specific activity covariation. Cell Rep. *40*.

40. Inagaki, H.K., Chen, S., Ridder, M.C., Sah, P., Li, N., Yang, Z., Hasanbegovic, H., Gao, Z., Gerfen, C.R., and Svoboda, K. (2022). A midbrain-thalamus-cortex circuit reorganizes

cortical dynamics to initiate movement. Cell *185*, 1065–1081.e23.

41. Xing, D., Truccolo, W., and Borton, D.A. (2022). Emergence of distinct neural subspaces in motor cortical dynamics during volitional adjustments of ongoing locomotion. bioRxiv. https://doi.org/10.1101/2022.04.03.486001.

42. Tafazoli, S., Bouchacourt, F.M., Ardalan, A., Markov, N.T., Uchimura, M., Mattar, M.G., Daw, N.D., and Buschman, T.J. (2025). Building compositional tasks with shared neural subspaces. Nature. https://doi.org/10.1038/s41586-025-09805-2.

43. Pani, P., Giamundo, M., Giarrocco, F., Mione, V., Fontana, R., Brunamonti, E., Mattia, M., and Ferraina, S. (2022). Neuronal population dynamics during motor plan cancellation in nonhuman primates. Proc. Natl. Acad. Sci. U. S. A. *119*, e2122395119.

44. Ames, K.C., and Churchland, M.M. (2019). Motor cortex signals for each arm are mixed across hemispheres and neurons yet partitioned within the population response. Elife *8*. https://doi.org/10.7554/eLife.46159.

45. Heming, E.A., Cross, K.P., Takei, T., Cook, D.J., and Scott, S.H. (2019). Independent representations of ipsilateral and contralateral limbs in primary motor cortex. Elife *8*. https://doi.org/10.7554/eLife.48190.

46. Cross, K.P., Heming, E.A., Cook, D.J., and Scott, S.H. (2020). Maintained Representations of the Ipsilateral and Contralateral Limbs during Bimanual Control in Primary Motor Cortex. J. Neurosci. *40*, 6732–6747.

47. Michaels, J.A., Dann, B., and Scherberger, H. (2016). Neural Population Dynamics during Reaching Are Better Explained by a Dynamical System than Representational Tuning. PLoS Comput. Biol. *12*, e1005175.

48. Kao, T.-C., Sadabadi, M.S., and Hennequin, G. (2021). Optimal anticipatory control as a theory of motor preparation: A thalamo-cortical circuit model. Neuron *109*, 1567–1581.e12.

49. Even-Chen, N., Sheffer, B., Vyas, S., Ryu, S.I., and Shenoy, K.V. (2019). Structure and variability of delay activity in premotor cortex. PLoS Comput. Biol. *15*, e1006808.

50. Kurtzer, I., Herter, T.M., and Scott, S.H. (2005). Random change in cortical load representation suggests distinct control of posture and movement. Nat. Neurosci. *8*, 498–504.

51. Marino, P.J., Bahureksa, L., Fisac, C.F., Oby, E.R., Smoulder, A.L., Motiwala, A., Degenhart, A.D., Grigsby, E.M., Joiner, W.M., Chase, S.M., et al. (2025). A posture subspace in the primary motor cortex. Neuron *113*, 3647–3660.e10.

52. Lu, X., and Ashe, J. (2005). Anticipatory activity in primary motor cortex codes memorized

movement sequences. Neuron *45*, 967–973.

53. Russo, A.A., Khajeh, R., Bittner, S.R., Perkins, S.M., Cunningham, J.P., Abbott, L.F., and Churchland, M.M. (2020). Neural trajectories in the supplementary motor area and motor cortex exhibit distinct geometries, compatible with different classes of computation. Neuron *107*, 745–758.e6.

54. Hyvärinen, A., and Oja, E. (2000). Independent component analysis: algorithms and applications. Neural Netw. *13*, 411–430.

55. Laubach, M., Shuler, M., and Nicolelis, M.A. (1999). Independent component analyses for quantifying neuronal ensemble interactions. J. Neurosci. Methods *94*, 141–154.

56. Hennequin, G., Vogels, T.P., and Gerstner, W. (2014). Optimal control of transient dynamics in balanced networks supports generation of complex movements. Neuron *82*, 1394–1406.

57. Goldman, M.S. (2009). Memory without feedback in a neural network. Neuron *61*, 621–634.

58. Murphy, B.K., and Miller, K.D. (2009). Balanced amplification: a new mechanism of selective amplification of neural activity patterns. Neuron *61*, 635–648.

59. Kaufman, M.T., Seely, J.S., Sussillo, D., Ryu, S.I., Shenoy, K.V., and Churchland, M.M. (2016). The Largest Response Component in the Motor Cortex Reflects Movement Timing but Not Movement Type. eNeuro *3*. https://doi.org/10.1523/ENEURO.0085-16.2016.

60. Dekleva, B.M., Chowdhury, R.H., Batista, A.P., Chase, S.M., Yu, B.M., Boninger, M.L., and Collinger, J.L. (2023). Motor cortex retains and reorients neural dynamics during motor imagery. bioRxiv. https://doi.org/10.1101/2023.01.17.524394.

61. Russo, A.A., Bittner, S.R., Perkins, S.M., Seely, J.S., London, B.M., Lara, A.H., Miri, A., Marshall, N.J., Kohn, A., Jessell, T.M., et al. (2018). Motor Cortex Embeds Muscle-like Commands in an Untangled Population Response. Neuron. https://doi.org/10.1016/j.neuron.2018.01.004.

62. Schroeder, K.E., Perkins, S.M., Wang, Q., and Churchland, M.M. (2022). Cortical Control of Virtual Self-Motion Using Task-Specific Subspaces. J. Neurosci. *42*, 220–239.

63. Cisek, P., Crammond, D.J., and Kalaska, J.F. (2003). Neural activity in primary motor and dorsal premotor cortex in reaching tasks with the contralateral versus ipsilateral arm. J. Neurophysiol. *89*, 922–942.

64. Steinberg, O., Donchin, O., Gribova, A., Cardosa de Oliveira, S., Bergman, H., and Vaadia, E. (2002). Neuronal populations in primary motor cortex encode bimanual arm movements. Eur. J. Neurosci. *15*, 1371–1380.

65. Shah, N.P., Avansino, D., Kamdar, F., Nicolas, C., Kapitonava, A., Vargas-Irwin, C., Hochberg, L.R., Pandarinath, C., Shenoy, K.V., Willett, F.R., et al. (2025). Pseudo-linear summation explains neural geometry of multi-finger movements in human premotor cortex. Nat. Commun. *16*, 5008.

66. Saxena, S., Russo, A.A., Cunningham, J., and Churchland, M.M. (2022). Motor cortex activity across movement speeds is predicted by network-level strategies for generating muscle activity. Elife *11*. https://doi.org/10.7554/eLife.67620.

67. Rigotti, M., Barak, O., Warden, M.R., Wang, X.-J., Daw, N.D., Miller, E.K., and Fusi, S. (2013). The importance of mixed selectivity in complex cognitive tasks. Nature *497*, 585–590.

68. Kira, S., Safaai, H., Morcos, A.S., Panzeri, S., and Harvey, C.D. (2023). A distributed and efficient population code of mixed selectivity neurons for flexible navigation decisions. Nat. Commun. *14*, 2121.

69. Machens, C.K., Romo, R., and Brody, C.D. (2010). Functional, But Not Anatomical, Separation of “What” and “When” in Prefrontal Cortex. Journal of Neuroscience *30*, 350–360.

70. Jun, J.K., Miller, P., Hernández, A., Zainos, A., Lemus, L., Brody, C.D., and Romo, R. (2010). Heterogenous population coding of a short-term memory and decision task. J. Neurosci. *30*, 916–929.

71. Scott, S.H. (2008). Inconvenient truths about neural processing in primary motor cortex. J. Physiol. *586*, 1217–1224.

72. Kalaska, J.F. (2009). From intention to action: motor cortex and the control of reaching movements. Adv. Exp. Med. Biol. *629*, 139–178.

73. Willett, F.R., Deo, D.R., Avansino, D.T., Rezaii, P., Hochberg, L.R., Henderson, J.M., and Shenoy, K.V. (2020). Hand knob area of premotor cortex represents the whole body in a compositional way. Cell *181*, 396–409. e26.

74. Raposo, D., Kaufman, M.T., and Churchland, A.K. (2014). A category-free neural population supports evolving demands during decision-making. Nat. Neurosci. *17*, 1784–1792.

75. Churchland, M.M., and Shenoy, K.V. (2007). Temporal complexity and heterogeneity of single-neuron activity in premotor and motor cortex. J. Neurophysiol. *97*, 4235–4257.

76. Stavisky, S.D., Willett, F.R., Wilson, G.H., Murphy, B.A., Rezaii, P., Avansino, D.T., Memberg, W.D., Miller, J.P., Kirsch, R.F., Hochberg, L.R., et al. (2019). Neural ensemble dynamics in dorsal motor cortex during speech in people with paralysis. Elife *8*. https://doi.org/10.7554/eLife.46015.

77. Susoy, V., Hung, W., Witvliet, D., Whitener, J.E., Wu, M., Park, C.F., Graham, B.J., Zhen, M.,

Venkatachalam, V., and Samuel, A.D.T. (2021). Natural sensory context drives diverse brain-wide activity during C. elegans mating. Cell *184*, 5122–5137.e17.

78. Sulston, J.E., Albertson, D.G., and Thomson, J.N. (1980). The Caenorhabditis elegans male: postembryonic development of nongonadal structures. Dev. Biol. *78*, 542–576.

79. Jarrell, T.A., Wang, Y., Bloniarz, A.E., Brittin, C.A., Xu, M., Thomson, J.N., Albertson, D.G., Hall, D.H., and Emmons, S.W. (2012). The connectome of a decision-making neural network. Science *337*, 437–444.

80. Cook, S.J., Jarrell, T.A., Brittin, C.A., Wang, Y., Bloniarz, A.E., Yakovlev, M.A., Nguyen, K.C.Q., Tang, L.T.-H., Bayer, E.A., Duerr, J.S., et al. (2019). Whole-animal connectomes of both Caenorhabditis elegans sexes. Nature *571*, 63–71.

81. LeBoeuf, B., and Garcia, L.R. (2017). Caenorhabditis elegans Male Copulation Circuitry Incorporates Sex-Shared Defecation Components To Promote Intromission and Sperm Transfer. G3 *7*, 647–662.

82. Liu, K.S., and Sternberg, P.W. (1995). Sensory regulation of male mating behavior in Caenorhabditis elegans. Neuron *14*, 79–89.

83. Koo, P.K., Bian, X., Sherlekar, A.L., Bunkers, M.R., and Lints, R. (2011). The robustness of Caenorhabditis elegans male mating behavior depends on the distributed properties of ray sensory neurons and their output through core and male-specific targets. J. Neurosci. *31*, 7497–7510.

84. Ames, K.C., Ryu, S.I., and Shenoy, K.V. (2014). Neural dynamics of reaching following incorrect or absent motor preparation. Neuron *81*, 438–451.

85. Schneider, S., Lee, J.H., and Mathis, M.W. (2023). Learnable latent embeddings for joint behavioural and neural analysis. Nature *617*, 360–368.

86. Hurwitz, C., Srivastava, A., Xu, K., Jude, J., Perich, M., Miller, L., and Hennig, M. (2021). Targeted neural dynamical modeling. Adv. Neural Inf. Process. Syst. *34*, 29379–29392.

87. Sani, O.G., Abbaspourazad, H., Wong, Y.T., Pesaran, B., and Shanechi, M.M. (2021). Modeling behaviorally relevant neural dynamics enabled by preferential subspace identification. Nat. Neurosci. *24*, 140–149.

88. Zhou, D., and Wei, X.-X. (2020). Learning identifiable and interpretable latent models of high-dimensional neural activity using pi-VAE. In Advances in Neural Information Processing Systems, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, eds. (Curran Associates, Inc.), pp. 7234–7247.

89. Olshausen, B.A., and Field, D.J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature *381*, 607–609.

90. Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. J. Comput. Graph. Stat. *15*, 265–286.

91. Jolliffe, I.T., Trendafilov, N.T., and Uddin, M. (2003). A modified principal component technique based on the LASSO. J. Comput. Graph. Stat. *12*, 531–547.

92. Gouwens, N.W., Sorensen, S.A., Berg, J., Lee, C., Jarsky, T., Ting, J., Sunkin, S.M., Feng, D., Anastassiou, C.A., and Barkan, E. (2019). Classification of electrophysiological and morphological neuron types in the mouse visual cortex. Nat. Neurosci. *22*, 1182–1195.

93. Tošić, I., and Frossard, P. (2011). Dictionary learning. IEEE Signal Process. Mag. *28*, 27–38.

94. Kreutz-Delgado, K., Murray, J.F., Rao, B.D., Engan, K., Lee, T.-W., and Sejnowski, T.J. (2003). Dictionary learning algorithms for sparse representation. Neural Comput. *15*, 349–396.

95. Georgiev, P., Theis, F., and Cichocki, A. (2005). Sparse component analysis and blind source separation of underdetermined mixtures. IEEE Trans. Neural Netw. *16*, 992–996.

96. Lin, X.-X., Nieder, A., and Jacob, S.N. (2023). The neuronal implementation of representational geometry in primate prefrontal cortex. Sci. Adv. *9*, eadh8685.

97. Giovannucci, A., Friedrich, J., Kaufman, M., Churchland, A., Chklovskii, D., Paninski, L., and Pnevmatikakis, E.A. (2017). Onacid: Online analysis of calcium imaging data in real time. Adv. Neural Inf. Process. Syst. *30*.

98. F. Diego, S. Reichinnek, M. Both, and F. A. Hamprecht (2013). Automated identification of neuronal activity from calcium imaging by sparse dictionary learning.

99. Charles, A.S., Cermak, N., Affan, R.O., Scott, B.B., Schiller, J., and Mishne, G. (2022). GraFT: Graph Filtered Temporal Dictionary Learning for Functional Neural Imaging. IEEE Trans. Image Process. *31*, 3509–3524.

100. Song, A.H., Flores, F., and Ba, D. (2018). Spike Sorting by Convolutional Dictionary Learning. arXiv [stat.ME].

101. Tolooshams, B., Matias, S., Wu, H., Temereanca, S., Uchida, N., Murthy, V.N., Masset, P., and Ba, D. (2025). Interpretable deep learning for deconvolutional analysis of neural signals. Neuron *113*, 1151–1168.e13.

102. Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., et al. (2023). Towards Monosemanticity: Decomposing Language Models With Dictionary Learning. Transformer Circuits Thread.

103. Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. (2023). Sparse autoencoders find highly interpretable features in language models. arXiv preprint arXiv:2309. 08600.

104. Rajamanoharan, S., Conmy, A., Smith, L., Lieberum, T., Varma, V., Kramár, J., Shah, R., and Nanda, N. (2024). Improving dictionary learning with gated sparse autoencoders. arXiv preprint arXiv:2404. 16014.

105. Kaufman, M.T., Churchland, M.M., Ryu, S.I., and Shenoy, K.V. (2014). Cortical activity in the null space: permitting preparation without movement. Nat. Neurosci. *17*, 440–448.

106. Amematsro, E.A., Trautmann, E.M., Marshall, N.J., Abbott, L.F., Shadlen, M.N., Wolpert, D.M., and Churchland, M.M. (2025). Motor cortex flexibly deploys a high-dimensional repertoire of subskills. bioRxiv. https://doi.org/10.1101/2025.09.07.674717.

107. Lezcano-Casado, M. (2022). Geometric optimisation on manifolds with applications to deep learning. arXiv preprint arXiv:2203.04794.

108. Kingma, D.P., and Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

109. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res. *12*, 2825–2830.

110. Zitnik, M., and Zupan, B. (2012). Nimfa: A Python Library for Nonnegative Matrix Factorization. Journal of Machine Learning Research *13*, 849–853.

111. Lee, D., and Seung, H.S. (2000). Algorithms for non-negative matrix factorization. Advances in neural information processing systems *13*.

Methods

Sparse Component Analysis

Parameters and Cost Function: Let $\mathbf{X}^{T\times N}$ be a matrix containing the activity of *N* neurons over *T* time bins. The goal of SCA is to find a low dimensional latent representation, with dimensionality *K,* that 1) accurately reconstructs the original data $\mathbf{X}$ via an affine mapping (linear mapping $\mathbf{V}^{K\times N}$ and offset $\boldsymbol{b_v}^{1\times N}$), 2) has temporally sparse latents, $\mathbf{Z}^{T\times K}$, and 3) encourages the mapping from the latent space to neural space to be orthonormal. We additionally include optional sample weighting, via the diagonal matrix $\mathbf{W}^{T\times T}$, which allows providing different importance to different time points within the cost function.

This cost function can be written as follows, with the summed terms corresponding to the three above goals, respectively:

$$\underset{\boldsymbol{Z},\boldsymbol{V},\boldsymbol{b_v}}{\operatorname{argmin}}\|\boldsymbol{W}\big(\boldsymbol{X}-(\boldsymbol{Z}\boldsymbol{V}+\boldsymbol{b_v})\big)\|_F^2+\lambda_{\text{sparse}}\|\boldsymbol{Z}\|_1+\lambda_{\text{orth}}\|\boldsymbol{V}\boldsymbol{V}^\top-\boldsymbol{I}\|_F^2$$

In practice, rather than directly optimizing the latents, we learn the latents via a mapping from the neural activity space (which can be viewed as a linear autoencoder, Fig. 1E).

This is accomplished with the following cost function:

$$\underset{\boldsymbol{U},\boldsymbol{V},\boldsymbol{b_v},\boldsymbol{b_u}}{\operatorname{argmin}}\|\boldsymbol{W}\Big(\boldsymbol{X}-\big((\boldsymbol{X}\boldsymbol{U}+\boldsymbol{b_u})\boldsymbol{V}+\boldsymbol{b_v}\big)\Big)\|_F^2+\lambda_{\text{sparse}}\|\boldsymbol{X}\boldsymbol{U}+\boldsymbol{b_u}\|_1+\lambda_{\text{orth}}\|\boldsymbol{V}\boldsymbol{V}^\top-\boldsymbol{I}\|_F^2,$$

where $\mathbf{U}^{N\times K}$ and $\boldsymbol{b_u}^{1\times K}$ are the linear matrix and offset in the "encoder" part of the autoencoder, that map the neural activity to the low-dimensional space.

In addition to the orthonormality penalty, we also strictly constrain each row of $\mathbf{V}$ to have unit norm. Thus, even if λ_{orth} is set to 0, optimization will not lead to a solution in which the magnitude of $\mathbf{Z}$ endlessly becomes smaller while $\mathbf{V}$ endlessly becomes larger. Additionally, this unit norm allows for easier interpretation of SCA factor magnitudes, as they can be interpreted as the magnitude of its effect on neural population activity..

We note that in our accompanying code package, we also include an option to enforce strict orthogonality in the recovered dimensions by performing manifold optimization[107]. However, we have found that using the orthogonality penalty (as opposed to an orthogonality constraint) is advantageous for two primary reasons. First, while the two options generally produced very similar factors, we found that, on occasion, constraining SCA to recover orthogonal dimensions could produce factors that were more 'mixed'. For example, the separation between posture-related factors and preparatory factors was less complete for monkey C (Fig. S13) if we required SCA to recover

orthogonal dimensions. The second benefit was purely practical; fitting SCA with manifold optimization was notably slower. For example, fitting SCA on the reaching dataset from monkey B was slowed by a factor of 1.8 if we used a strict orthogonality constraint.

Sample weighting: When we use sample weighting, we default the sample weight at a given time point to be inversely proportional to the sum squared activity of the neural population at that time point. This has the purpose of encouraging the model to still use latent dimensions to reconstruct lower-activity time points (e.g. to care as much about motor preparation as motor execution, when the former has substantially lower activity).

Optimization: All optimization is done in PyTorch with the Adam optimizer[108] using default hyperparameters. We use learning rate scheduling, where the initial learning rate is set at 1e-3, and the learning rate is halved every time the loss does not decrease for 100 iterations, until a minimum learning rate of 5e-4.

Initialization of parameters: We initialize the model parameters using sample-weighted PCA on the data. That is, if $\mathbf{Q}^{N\times K}$ are the sample-weighted PCA loadings, we initialize with $\mathbf{U}=\mathbf{Q}$, $\mathbf{V}=\mathbf{Q}^T$, $\boldsymbol{b}_v=0$, $\boldsymbol{b}_u=0$. In practice, we find nearly equivalent solutions using a random initialization of parameters, but convergence generally takes significantly longer. Both sample-weighted PCA and random initializations are available within our code package.

Hyperparameters: In our code package, we have the following default settings for model hyperparameters. λ_{orth} is defaulted so that, if the off-diagonal entries of $\boldsymbol{V}\boldsymbol{V}^\top$ were all equal to 0.1, the orthonormality penalty would be 10% of the initial reconstruction cost (with sample-weighted PCA initialization). λ_{sparse} is defaulted so that the initial cost associated with the sparsity penalty is 10% of the initial reconstruction cost. This serves to make sparsity and orthogonality important, but not at great expense of reconstructing the neural data. These defaults allow for easy out-of-the-box model-fitting without hyperparameter tuning. Still, we have found that it is often slightly beneficial to increase λ_{sparse} beyond this default value - i.e., further sparsity can be achieved without harming the reconstruction accuracy. Thus, for the demonstrations in this paper, we sometimes set λ_{sparse} above the default value - using λ_{sparse} values at the elbow of the plot of reconstruction accuracy versus λ_{sparse}. We also provide a code notebook so that users can also find λ_{sparse} in this way. See below for a table of the hyperparameters used for each dataset.

Similarly, we have also seen a general robustness to the dimensionality chosen, with the primary change that occurs over differing dimensionalities detailed in *Results: Center-out reaching - practical considerations*. Importantly, we have seen that requesting too many dimensions generally yields highly similar 'signal' dimensions, with the extra dimensions being primarily capturing noise. For instance, in the reaching data, when overspecifying dimensionality, dimensions beyond 20 appear to

be primarily noise (Fig. S5), which matches the dimensionality at which the performance on held-out data plateaus (Fig. 2I; we include such bi-cross-validation code within our code package). We do note that this robustness does depend on keeping nearly orthogonal loadings (Fig. S7).

Advantages of neural-network based optimization: We chose to optimize SCA as a linear autoencoder using standard neural network machinery for two primary reasons. 1) This framework scales well with increased dataset duration. The number of encoding and decoding weights depends only on the number of neurons and number of requested factors, whereas directly optimizing latents yields a parameter count that scales with the number of timepoints. Additionally, neural network machinery can easily allow for GPU use, providing improved speed for larger datasets. 2) Our approach makes the method readily extendable, as we have done to create a nonlinear version. As another possible example, if a dataset contains neural activity from two brain regions, the model can be easily altered to identify interpretable factors within one brain region that predict activity in the second region. Additionally, one can easily change the model's cost function (e.g. including Poisson statistics). This flexibility will encourage further development of interpretable dimensionality reduction tools for neural population activity.

Nonlinear SCA

Parameters and Cost Function: As in our linear SCA description, we start by describing the mapping from latents, $\mathbf{Z}$ to neural activity, $\mathbf{X}$ (the 'decoder' part of the autoencoder). Let g be the decoder function mapping the latent space to neural activity predictions, and θ_g represent the parameters (weights and biases) of the neural networks g. Let $\boldsymbol{V}_i$ be the weights of layer i in the decoder (so all $\boldsymbol{V}_i \in \theta_g$). We aim to find:

$$\arg\min_{\boldsymbol{Z},\,\theta_g}\left(\left|\boldsymbol{W}\left(\boldsymbol{X}-\boldsymbol{g}(\boldsymbol{Z})\right)\right|_F^2+\lambda_{\text{sparse}}\left|\boldsymbol{Z}\right|_1+\lambda_{\text{orth}}\sum_i\left|\boldsymbol{V}_i\boldsymbol{V}_i^{\top}-\boldsymbol{I}\right|_F^2\right)$$

As in linear SCA, we don't directly optimize $\mathbf{Z}$, but rather learn the mapping from $\mathbf{X}$ to $\mathbf{Z}$. Here, this is a nonlinear neural network f. Let θ_f represent the parameters (weights and biases) of the neural network f. We thus optimize:

$$\arg\min_{\theta_f,\,\theta_g}\left(\left|\boldsymbol{W}\left(\boldsymbol{X}-\boldsymbol{g}(\boldsymbol{f}(\boldsymbol{X}))\right)\right|_F^2+\lambda_{\text{sparse}}\left|\boldsymbol{f}(\boldsymbol{X})\right|_1+\lambda_{\text{orth}}\sum_i\left|\boldsymbol{V}_i\boldsymbol{V}_i^{\top}-\boldsymbol{I}\right|_F^2\right)$$

The three terms of the cost function, in order, relate to 1) the reconstruction cost, 2) sparsity of the latents, and 3) orthonormality. Here, we encourage the weights of each layer of the decoder neural network to be orthonormal. It's important to note that the concept of orthonormality is defined for linear matrix transformations, rather than nonlinear functions. Here, our nonlinear mapping contains a nonlinearity between two matrix multiplications. In a linear network, the multiplication of two

orthonormal matrices remains orthonormal. While the nonlinear activation functions between the matrices in the neural network will alter this mapping, our approach nonetheless promotes separate latents to have more distinct mappings to neural activity, and empirically to separate different computations involving different patterns of neural firing into different latent factors. We also note that while this orthonormality penalty was important for interpretation (Fig. S32,S33), orthonormality had minimal impact on reconstruction accuracy (Fig. S31).

Similar to linear SCA, In addition to the orthonormality penalty, we also strictly constrain each row of $\boldsymbol{V}_i$ to have unit norm, for all layers i. Thus, even if λ_{orth} is set to 0, optimization will not lead to a solution in which the magnitude of $\mathbf{Z}$ endlessly becomes smaller while all $\boldsymbol{V}_i$ endlessly become larger.

In all our demonstrations, f and g were neural networks with a single hidden layer with a tanh nonlinearity. For K (the number of SCA dimensions) $< N/2$ (where N is the number of neurons), the hidden layer contained $N/2$ units. For $K \geq N/2$, we let the hidden layer contain K units.

Optimization: Optimization was done as above in linear SCA.

Initialization of parameters and hyperparameters: Neural network parameters were randomly initialized. As in linear SCA, we have default hyperparameter values that allow the technique to work out-of-the-box. λ_{orth} was defaulted so that, if the off-diagonal entries of $\boldsymbol{V}_i\boldsymbol{V}_i^\top$, for both layers of the neural network, were all equal to 0.1, the total summed orthonormality penalty would be 100% of the sample-weighted PCA reconstruction cost. λ_{sparse} was defaulted identically to SCA - the value that would lead to the sparsity term of the cost function leading to 10% of the sample-weighted PCA reconstruction loss.

Ordering of latent factors

There is not a meaningful intrinsic ordering to SCA dimensions. However, after model fitting, re-ordering dimensions can facilitate discovering a meaningful interpretation of the data. For instance, in the reaching data, we typically ordered dimensions by the time points at which the dimensions explained the most cross-condition variance, which allowed us to clearly observe dimensions that activated at distinct time epochs - i.e., preparation, then movement, then posture. Such a temporal ordering also helped us to discover 'stopping' dimensions in the unimanual cycling datasets, although we ultimately manually grouped these dimensions into their putative roles, after further examination, for the clearest presentation.

Dimensions can also be automatically sorted by the amount of neural activity explained, comparable to variance explained in PCA, which can also provide insight. For instance, the non-interpretable dimensions that appear like noise often have relatively low variance, so sorting in this manner can

help to filter out such 'noise' dimensions. Still, it is important to note there are often interpretable signals that explain a low amount of variance overall, but a high amount of variance at a small number of time points (e.g. when stopping a movement or preparing a movement in the unimanual cycling task). For instance, the preparatory dimensions each explain about 1-2% of the total explained variance in the unimanual cycling data, despite being interpretable and computationally meaningful dimensions. Nonetheless, this activity-explained-based sorting can help to provide intuition for the extent to which each factor is present in the overall population activity. We provide code in our package for sorting factors in both these temporal-ordering and variance-ordering manners.

Responsible use of SCA

We want to give some advice regarding how to responsibly use SCA to decrease the risk of arriving at incorrect scientific conclusions. One set of possible concerns relates to sampling error (due to too few neurons or too few trials) leading to false positives or false negatives.

False positives: By false positive, we mean the discovery of seemingly sparse and seemingly well-behaved factors (possibly suggesting some hypothesis) where none actually exist. For example, suppose that preparatory and execution activity actually involved the same set of factors (i.e. they involve different patterns of activity within the same subspace) but SCA returned distinct preparatory and execution factors. This would constitute a false positive. In practice, we have generally found SCA to be robust to false positives. As we highlight in *Discussion,* we never found a situation in which SCA found spurious factors; SCA factors always agreed with a known ground truth, or were validated by subsequent non-SCA-based analyses (e.g. subspace alignment). Furthermore, when applying SCA to poorly (minimally) filtered single-trial data, we did not find spurious factors, but noise. To be clear, this does not mean that the factor divisions found by SCA are necessarily the right ones for understanding computation – the degree to which this is true will depend on the degree to which the goals of SCA align with one's scientific goals. Nor does it mean that initially-appealing interpretations will be correct – further analyses or experiments will almost always be needed, as with any method. What it does mean is that SCA is resistant to hallucinating false positives.

To see why, consider that each SCA latent is simply a weighted sum of the empirical neural activity. Those latents in turn form a basis set for the activity of all the single neurons. To create a false positive, SCA must take activity where there is no natural basis of sparsely active factors, yet choose weights so as to create such factors. To do so, optimization could follow the gradient in weight space that maximally reduces the sparsity term in SCA's cost function. Yet this will come at the cost of increased reconstruction error (precisely because there is no natural sparse basis). Put differently, if there is no natural sparse basis, a change in weights that increases factor sparsity will typically decrease the sparsity term much less than it will increase reconstruction error. Optimization will thus avoid such changes. It is for this reason that the variance captured by SCA is typically almost as large

as that captured by PCA – indeed the SCA basis set is frequently close to a rotated version of the PCA basis set (which is why PCA with a varimax rotation can be a good approximation to SCA).

These observations are reassuring, but also highlight that care must be taken when the orthogonality hyperparameter is substantially decreased (e.g see Figs. S5, S33). The same is true when the sparsity hyperparameter is turned up into the regime where reconstruction performance degrades (which we do not recommend, as described above in *Sparse Component Analysis: Hyperparameters*).

If one remains concerned about false positives, particularly due to noisy or limited data, we recommend confirming the found factors on held-out data. Held-out data can take many forms, including held-out neurons, trials, or conditions. One can also replicate findings on another dataset from the same task, either from another session of the same animal or from a different animal (as we often did in this study). One can also replicate results in a variation of the original situation, e.g. a variation of the task.

False negatives: False negatives are potentially a larger concern, in particular when data-limited. Suppose two factors are truly orthogonal in the full population, but one measures the activity of too few neurons. The loadings may then be accidentally correlated, leading to non-orthogonality simply due to lack of data. For instance, in the center-out reaching dataset, preparatory activity and movement-related activity occur in orthogonal dimensions when using all neurons. However, when using only 20 neurons, this scientific conclusion becomes less obvious, as some noisy latents are active both during preparatory and movement epochs (see 2nd-from-top and 2nd-from-bottom latents in Fig. S3). In general, the number of needed neurons will scale with the total number of factors that one seeks. In situations where one is concerned that neuron-counts are marginal, we would suggest applying SCA to simulated data to estimate the stability of results with neuron count.

Conceptual Interpretation: In general, we think the larger concerns typically relate to the conceptual interpretation of the latent factors. SCA often produces factors that nicely lend themselves to interpretation, which is helpful, but only if one is careful. It can be a little too tempting, upon finding factors that are active before stopping to cycle, to then conclude 'those must be related to preparing to stop'. Similarly, upon observing factors that mirror posture, it may be tempting to conclude 'those must be related to posture'. Those are indeed reasonable interpretations, but that doesn't reduce the need for follow-up analyses or experiments to test them. This is of course nothing new, or particular to SCA or even to dimensionality reduction. For example, motor-cortex preparator' activity (at both single-neuron and population levels) was called 'preparatory' for a long time before enough experiments were performed to truly nail down that that was indeed its role. As we describe in *Discussion: Factor Interpretation,* initial interpretations should not be fully accepted until follow-up experiments are used to test them.

SCA Runtime

To provide an approximate intuition of the runtime for fitting SCA, we provide examples below. These models were fit with CPU, on a 2020 MacBook Pro, 13" with M1 Chip and 8GB memory. Runtimes are provided until model convergence.

Reaching data (size 124 neurons x 2048 timepoints)*:*
Linear SCA: 2 seconds (3000 epochs)
Nonlinear SCA: 29 seconds (15000 epochs)

Unimanual Cycling data (size 116 neurons x 7023 timepoints):
Linear SCA: 19 seconds (5000 epochs)
Nonlinear SCA: 1 minute 54 seconds (15000 epochs)

Sample-weighted Principal Components Analysis

Sample-weighted PCA (wPCA) is a variant of principal component analysis that includes providing a weight for each sample (time point). Formulated as a cost function, it aims to solve:

$$\underset{\mathbf{U}}{\operatorname{argmin}} ||\mathbf{W}\left(\mathbf{X} - (\mathbf{X}\mathbf{U}\mathbf{U}^T\right)||_F^2 \quad \text{s.t. } \mathbf{U}^T\mathbf{U} = \mathbf{I}$$

$\mathbf{U}$ is found by running singular value decomposition (SVD) on the matrix $\mathbf{W}\mathbf{X}$, in the same way that standard PCA runs SVD on the matrix $\mathbf{X}$.

Additional Comparison Methods

Beyond PCA, we compared SCA with several additional unsupervised dimensionality reduction approaches:

Factor analysis (FA): FA is a commonly used linear probabilistic generative model that enables fitting separate noise levels on each neuron. We used the scikit-learn[109] implementation with default hyperparameters.

Sparse PCA (sPCA): This is a variant of PCA in which sparsity is encouraged in the loadings ($\mathbf{U}$ in the description of PCA in the above subsection). We used the scikit-learn implementation with default hyperparameters.

Non-negative matrix factorization (NMF): NMF factorizes a non-negative matrix into the multiplication of two matrices with non-negative entries. We used the NIMFA package[110]. For both simulations (Fig. S1) and reaching data,, we used a NNDSVD initialization and increased the maximum number of iterations, to improve performance, particularly as convergence was not reached with the standard number of iterations. NMF can implicitly promote sparsity and orthogonality[111].

I*ndependent component analysis (ICA):* ICA aims to find dimensions that minimize mutual information between those dimensions. We used the scikit-learn implementation of FastICA. Default hyperparameters were used for simulations (Fig. S1). For reaching data, we increased the maximum number of iterations to improve performance.

PCA followed by a varimax rotation of the latents: This can be viewed as similar to a step-wise approximation of SCA. PCA first does dimensionality reduction, and ensures that the loadings are strictly orthonormal. Doing a varimax rotation on the factors, which finds a rotation of the latent factor space that maximizes the variance across factors at each time, leads to sparse latents. This is because having, for example, half the latents with zero activity and half the latents with high activity will have more variance than having all factors with moderate activity. PCA followed by a varimax rotation of the latents is part of our SCA code package.

Datasets

All analyzed datasets (center-out reaching[26], unimanual cycling[61], bimanual cycling[44], center-out reaching network[14], multi-task network[13], and *C. elegans* mating[77]) have previously been analyzed. Detailed descriptions of each can be found in the respective citations. Here, we briefly describe each dataset.

dataset	**number of units**
Multi-task RNN	128
Monkeys B and A, center-out reaching	124 and 130
Monkey N, maze task	118
Maze task RNN	231
Monkeys C and D, unimanual cycling	116 and 117
Monkeys F and E, bimanual cycling	738 and 586
Worms 109, 137, 153, 182, 184, 185, 714, and 2457, *C. elegans* mating	49, 53, 54, 50, 49, 51, 53, and 51

Center-out reaching

The animal protocols relating to the experiments involving rhesus macaques were approved by the Columbia University Institutional Animal Care and Use Committee. Single neuron and well-isolated multi-unit activity was collected from motor cortex (dorsal premotor cortex and primary motor cortex) while monkeys A and B performed center-out reaches. All recordings were performed in the hemisphere contralateral to the performing arm. Animals reached in eight directions during two contexts: cue-initiated and quasi-automatic. In the cue-initiated context, the animals performed standard center-out reaches with a random, unpredictable delay between 0-1000 ms. In the quasi-automatic context, the delay period was again 0-1000 ms long, and the peripheral target began to move outward from the center of the screen once the go-cue was delivered. In both conditions, the peripheral target needed to be held for 600 ms, after which reward was delivered. Following reward delivery, the monkeys reached back to the central touch-point to begin the next trial.

Due to extensive training, the monkeys' behavior during cue-initiated and quasi-automatic conditions is extremely similar. Additionally, activity in motor cortex is also extremely similar[26]. We therefore combined cue-initiated and quasi-automatic trials when creating trial-averaged PSTHs. When creating PSTHs, we excluded failed trials and trials where the monkeys' behavior deviated from normal. This included trials where the outward reaction time was greater than 500 ms, the monkey's kinematics were substantially different from the median kinematics, or the monkey's hand moved during the dwell period between outward and return reaches. In practice, we primarily excluded trials because of behavior during the return reach, where behavior was less constrained by task requirements. All analyses only included data from conditions with a long (i.e., 300 ms) delay.

To construct trial-averaged PSTHs, we spliced neural activity from around target onset, outward reach onset, and return reach onset together to form a single trace that was continuous in time. For each unit, we first aligned spikes from a single trial, within a window of time, to the relevant task events. We used windows of –200 to 350 ms, -225 to 1000 ms, and -225 to 1000 ms for peri-target, peri-reach, and peri-return activity, respectively. We then concatenated each peri-event spike train, convolved each with a 25 ms Gaussian filter, and averaged across trials. Prior to performing SCA or PCA, we performed two additional preprocessing steps. First, we 'soft-normalized' the activity of each neuron; each neuron's mean rate (across times and conditions) was normalized by the range of the rate (across times and conditions) + 5. We standardly use soft-normalization (e.g.,[15,28,61]) to balance the desire for SCA or PCA to explain the responses of all neurons with the desire that weak responses not contribute on an equal footing with robust responses. As a second preprocessing step, the responses for each neuron were mean-centered at each time. We calculated the mean activity across all conditions of each neuron at each time point, and subtracted this mean activity from each condition's response. This step ensures that dimensionality reduction focuses on dimensions where responses are selective across conditions, rather than dimensions where activity varies in a similar fashion across all conditions[59].

Unimanual cycling

Monkeys C and D performed unimanual cycling movements to traverse a virtual track[61]. All recordings were performed in motor cortex, contralateral to the performing arm. Here, only well-isolated single neurons were included in the analyses.

Each trial began with the monkey acquiring an initial target, whose location determined the starting location. A final target was then shown in the distance, cueing the monkey as to the number of required cycles. After a variable delay period (500-1000 ms), a go-cue was delivered. After acquiring the final target, the monkey needed to remain stationary for 1500 ms, after which they received a reward. Animals performed 20 unique conditions, each requiring 0.5, 1, 2, 4, or 7 cycles, forwards or backwards movements of the arm, starting at either the top or the bottom of a cycle.

To create trial-averaged PSTHs, we used a previously described time-warping procedure[61]. While behavior during the reaching task was brief, movements during the cycling task could unfold over multiple seconds, necessitating a more detailed alignment procedure. Briefly, the time-base for each individual trial was scaled such that the virtual position for that trial closely matched the mean position across all trials of a given condition. This procedure aligned neural activity (and behavior) across trials of a given condition. We then took a subsequent step (not used in[61]) to better align data across conditions; we applied a second scaling to the time-base calculated for each condition, such that the duration of each individual cycle was 500 ms (0.5 cycle conditions were not adjusted). Because both animals tended to cycle at approximately 2 Hz, this second adjustment was fairly minor. Prior to performing SCA, responses were soft-normalized, as described above.

Bimanual cycling

Monkeys E and F performed a cycling task similar to that described above; the animals performed cycling movements to traverse a virtual track[44]. While monkeys C and D performed only unimanual cycling movements, monkeys E and F performed both unimanual and bimanual conditions. As above, each condition required the monkey to move between two virtual targets, with a given condition requiring either forwards or backwards arm movements, beginning at either the top or bottom of a cycle. Unlike the task performed by monkeys C and D, each condition was seven cycles long and required movements of either the left, right, or both arms.

For unimanual conditions, only one arm (the 'performing arm') was allowed to move; a trial was aborted if the 'non-performing arm' moved outside of a small window centered around the bottom of the cycle (±0.05 and ±0.07 cycles for monkey E and F, respectively). During bimanual conditions, the animal was required to move both arms, maintaining either a 0-degree (bimanual0 conditions) or 180-degree (bimanual180 conditions) phase offset between the two hands. Virtual position was determined by the average position of the two arms. Critically, the pedals were not 'locked' together; the monkeys needed to actively maintain a particular phase offset. If the angle between the two

hands exceeded 180 degrees from the desired offset (i.e., if one arm became more than one half-cycle ahead of the other) the trial was aborted. While the position of the pedals were not yoked, we did provide a weak restorative force to help the animals maintain the instructed phase offset.

Recordings were made in the primary and premotor cortices of both hemispheres and included both single neurons and well-isolated multi-units. To generate trial-averaged PSTHs, we aligned single-trial spikes to movement onset and convolved each spike-train with a 25 ms Gaussian. We then used the same alignment procedure described previously[44]. This procedure results in an adjusted time-base in which each individual cycle, for each condition, is 500 ms in duration. As was true for monkeys C and D, monkeys E and F cycled at approximately 2 Hz for all conditions, so the warping procedure provided only a small adjustment. All analyses of the bimanual cycling dataset only included data from the middle cycles (cycles 2-6). Prior to performing SCA, responses were soft-normalized, as described above.

***C. elegans* mating**

Here, we analyzed calcium imaging data recorded from the posterior brain of eight male worms during mating[77]. All worms co-expressed GCaMP6s and mNeptune in all neuronal nuclei. Individual neurons were identified via their location, morphology, and expression of specific fluorescent markers, and the occurrence of specific behavioral motifs was detected automatically using various kinematic measures (e.g., swimming velocity, tail velocity) as well as the orientation and position of the male worms relative to the hermaphrodites. All analyses involved single-trial fluorescence signals; for details of the preprocessing steps see[77]. Each neuron's response was fully normalized by its dynamic range prior to performing SCA or PCA.

Multitask RNN

The multitask network[13] consisted of 128 recurrently connected units and received three inputs: fixation (1-dimensional), stimulus (4-dimensional), and context (15-dimensional). The fixation input served both as a 'hold' signal and a 'go cue', depending on the context. The set of stimuli contained two separate two-dimensional vectors composed of $A\sin(\theta)$ and $A\cos(\theta)$, where each vector encoded a different one-dimensional circular variable (θ_1, θ_2) scaled by an amplitude A. The context input indicates the current task on any given trial. Context input was encoded in a one-hot vector where the index associated with the current task was one and all other indices were zero. The network generated two outputs: a fixation output (1-dimensional) and a response angle (2-dimensional, the sine and cosine of the response angle). During training, the duration of the stimulus epochs (and memory epochs, when present) was determined by a random draw from a uniform distribution to prevent the network from predicting task period transitions.

The RNN was defined by standard functions:

$$\tau \frac{d\mathbf{h}}{dt} = -\mathbf{h}(t) + F(\mathbf{W}_{rec}\mathbf{h}(t) + \mathbf{W}_{in}\mathbf{u}(t) + \mathbf{b}_{in})$$

$$\mathbf{z}(t) = \mathbf{W}_{out}\mathbf{h}(t) + \mathbf{b}_{out}$$

where F is a tanh activation function, $\mathbf{h}$ is a vector of network activations, $\mathbf{u}$ are the inputs, $\mathbf{b}$ are offset terms, and $\mathbf{z}$ is the output vector. $\mathbf{W}_{rec}$, $\mathbf{W}_{in}$, and $\mathbf{W}_{out}$ are the recurrent, input, and output weight matrices, respectively. The network was trained with standard L2 penalties on the weights and rates. Additionally, private neuron noise and input noise were added during training. The weights were trained via backpropagation through time using the Adam optimizer[108]. During training, the network performed all tasks in an interleaved order.

Below, we summarize the 15 tasks performed by the network.

Delayed Response: Delay-pro: Move in the same direction as stimulus ($\phi_{response} = \theta_{stimulus}$) after a delay. Delayed-anti: Move in opposite direction as stimulus ($\phi_{response} = \theta_{stimulus} + \pi$) after a delay. Stimulus remains on throughout stimulus and response periods.

Memory Response: Same as above, except the stimulus disappears during the memory and response period.

Reaction Timed: Same as Delayed Response, but a response is required as soon as the stimulus is delivered.

Compare: Move in the direction of the stimulus with the largest amplitude. Compare-stim1: Compare stimuli delivered via stimulus modality 1. Compare-stim2: Compare stimuli delivered via stimulus modality 2.

Context Compare: Same as above, except stimuli are delivered using both modalities. ContextCompare-stim1: compare stimuli delivered via modality 1, ignore modality 2. ContextCompare-stim2: compare stimuli delivered via modality 2, ignore modality 1. ContextCompare-both: attend both modalities.

Category Match: Network is delivered two sequential stimuli (using the same stimulus modality). Category-pro: immediately respond in the direction of the second stimulus if the two stimuli belong to the same category ($\theta_{stim1}, \theta_{stim_2} < \pi$ or $\theta_{stim1}, \theta_{stim_2} > \pi$). Category-anti: immediately respond in the direction of the second stimuli if the two stimuli belong to opposite categories.

MatchToSample: Network is delivered two sequential stimuli (using the same stimulus modality). MatchToSample-match: immediately respond in the direction of the second stimulus if the two

stimuli are the same ($\theta_{stim1} = \theta_{stim2}$). MatchToSample-nonmatch: immediately respond in the direction of the second stimulus if the two stimuli are 180° apart ($\theta_{stim1} = \theta_{stim2} + \pi$)

SCA hyperparameters

As demonstrated in Figure S4, SCA produced highly similar results across a wide range of hyperparameter choices. Below we list the specific hyperparameters used for each dataset. 'Default', and rationale for non-default values is described above.

dataset	λ_{sparse}	λ_{orth}	**number of factors**
Multi-task RNN	default	default	25
center-out reaching	default	default	8, 16, or 24
maze task	default	default	12
Maze task RNN	default	default	12
unimanual cycling	0.03 (~10x default)	default	40
bimanual cycling	0.01 (~10x default)	default	50
C. elegans mating	0.05 (~10x default)	default	15

Reaching analyses

Occupancy and occupancy concentration: For the reaching datasets, we computed the occupancy (variance across conditions) of each dimension, k, at each timepoint, t, where t began with target onset and ended 300 ms after the onset of the return reach:

$$\mathrm{Occ}(t,k) = \frac{1}{C-1}\sum_{i=1}^{C}(\mathbf{u}_i^\intercal \mathbf{r}_{t,i})^2$$

where C is the number of conditions, $\mathbf{u}$ is a vector of PCA or SCA loadings, and $\mathbf{r}$ is a vector of firing rates. For each dimension, we calculated the fraction of the total occupancy accounted for by the preparatory, execution, or postural epochs (e.g., Fig. 2H, *left*). For example, the fractional occupancy accounted for by the preparatory epoch is defined as:

$$\mathrm{Occ\_fraction}_{\mathrm{prep}}(k) = \frac{\mathrm{Occ}_{\mathrm{prep}}(k)}{\mathrm{Occ}_{\mathrm{prep}}(k) + \mathrm{Occ}_{\mathrm{exec}}(k) + \mathrm{Occ}_{\mathrm{post}}(k)}$$

Where $\mathrm{Occ}_{\mathrm{prep}}$, $\mathrm{Occ}_{\mathrm{exec}}$, and $\mathrm{Occ}_{\mathrm{post}}$ are the sum of the occupancies during the preparatory, execution, and posture epochs. These three epochs are defined below.

epoch	time window(s)
preparatory	$T_{target} : T_{reach}$, $T_{return} - 300 : T_{return}$
execution	$T_{reach} : T_{reach} + 300$, $T_{return} : T_{return} + 300$
posture	$T_{reach} + 300 : T_{return} - 300$

T_{target}, T_{reach}, and T_{return} are the times of target onset, outward reach onset, and return reach onset, and the time ranges are defined in milliseconds.

To summarize the distribution of occupancy within a single dimension, we calculated 'occupancy concentration': the sum of the absolute difference between all fractional occupancy pairs:

$$\mathrm{Occ\_concentration}(k) = |\mathrm{Occ\_fraction}_{\mathrm{prep}}(k) - \mathrm{Occ\_fraction}_{\mathrm{exec}}(k)| + |\mathrm{Occ\_fraction}_{\mathrm{prep}}(k) - \mathrm{Occ\_fraction}_{\mathrm{post}}(k)| + |\mathrm{Occ\_fraction}_{\mathrm{exec}}(k) - \mathrm{Occ\_fraction}_{\mathrm{post}}(k)|$$

The maximum possible occupancy concentration for a single dimension would be 2 (e.g., fractional occupancies of 1,0, and 0 for the preparatory, execution, and posture epochs, respectively). To summarize occupancy concentration across all dimensions, we simply calculated the median concentration across all dimensions (Fig. 2H, *right*).

Supervised dimensionality reduction: To compare SCA and PCA latents to those recovered by a supervised method, we used the dimensionality reduction method documented in[25](Fig. 3). This method identifies orthogonal sets of dimensions that capture a large fraction of variance during user-defined epochs:

$$\hat{Q}_{\mathrm{prep}}, \hat{Q}_{\mathrm{exec}}, \hat{Q}_{\mathrm{post}} = \arg \max_{Q_{\mathrm{prep}}, Q_{\mathrm{exec}}, Q_{\mathrm{post}}} \frac{1}{2} \left(\frac{\mathrm{Tr}(Q_{\mathrm{prep}}^T C_{\mathrm{prep}} Q_{\mathrm{prep}})}{\sum_{i=1}^{d_{\mathrm{prep}}} \sigma_{\mathrm{prep}}(i)} + \frac{\mathrm{Tr}(Q_{\mathrm{exec}}^T C_{\mathrm{exec}} + Q_{\mathrm{exec}})}{\sum_{i=1}^{d_{\mathrm{exec}}} \sigma_{\mathrm{exec}}(i)} + \frac{\mathrm{Tr}(Q_{\mathrm{post}}^T C_{\mathrm{post}} Q_{\mathrm{post}})}{\sum_{i=1}^{d_{\mathrm{post}}} \sigma_{\mathrm{post}}(i)} \right)$$

$$\text{subject to } Q_{\text{prep}}^T Q_{\text{exec}} = 0, Q_{\text{prep}}^T Q_{\text{post}} = 0, Q_{\text{exec}}^T Q_{\text{post}} = 0$$

$$Q_{\text{prep}}^T Q_{\text{prep}} = I, Q_{\text{exec}}^T Q_{\text{exec}} = I, Q_{\text{post}}^T Q_{\text{post}} = I$$

Where Q are the loadings for the preparatory, execution, and posture-related spaces, C are the covariance matrices of neural activity during preparatory, execution, and posture-related epochs, and $\sigma(i)$ is the i^{th} singular value of C. For each epoch, we requested 12 dimensions, which were used as ground-truth computation-specific factors when evaluating other methods (Fig. 3).

In Figure 3, for properly supervised dimensionality reduction, the model was provided with preparatory, execution, and posture-related epochs. For naively supervised dimensionality reduction, the posture-related epoch was not provided; instead, execution and posture-related epochs along with the intervening time between them were grouped into a single "execution?" epoch.

Demixed PCA (dPCA): We applied demixed PCA using reaching directions and time as task parameters. The model was fit by requesting eight components in each marginalization, and we restricted our analysis to the factors within direction-time interaction marginalization - that is, the eight condition-dependent factors.

Bi-cross-validation: As a stringent test of whether SCA recovers structure that exists across conditions during reaching, we performed a bi-cross-validation analysis, which required generalization to both held-out conditions and held-out neurons. We first fit SCA using data from all conditions and a set of training neurons (90% of all neurons). We then used linear regression to fit the loadings between test-neurons and factors, using activity from 7 of 8 conditions. Finally, we use factor activity from the held-out condition and the loadings recovered via regression to predict the activity of the held-out neurons during the held-out condition. We repeated this process 80 times (10-fold validation for the neurons, 8-fold validation for the conditions).

Bimanual cycling analyses

Validating existence of a single speed dimension during bimanual cycling: Training SCA on bimanual cycling activity from monkey E yielded a single dimension that was highly correlated with the monkey's average cycling speed over a cycle (Figure S21E). To validate this result, we used linear regression to identify a single speed dimension from the neural activity from one condition. We then asked whether activity in this dimension was correlated with cycling speed during a second condition. More specifically, we used linear regression to solve for $\mathbf{b}$:

$$\mathbf{y} = \mathbf{X}\mathbf{b}$$

Where $\mathbf{y}$ is a vector corresponding to the angular cycling speed for a single condition after low-pass filtering (third-order Butterworth, cutoff frequency: 0.8 Hz) and $\mathbf{X}$ is a TxN matrix of trial-averaged

firing rates for one bottom-start condition. To prevent overfitting, we used ridge-regression with $\lambda = 0.01$. After calculating $\mathbf{b}$, we projected neural activity from all top-start conditions into the same dimension and calculated the R^2 between the projection and low-pass filtered angular speed for each condition.

***C. elegans* analyses**

Correlating factors with behavior: The magnitude of individual SCA factors often correlated with the onset of specific behavioral motifs (e.g., Fig. 6C). To quantify this relationship (Fig. 6D), we generated boolean vectors that corresponded to when each individual behavioral motif occurred for each worm. We convolved each vector with a gamma distribution (shape parameter: 2, scale parameter: 5) to produce a continuous behavioral trace, then calculated the correlation between these behavioral traces and each SCA (or PCA) factor.

Generating pseudo-factors: SCA often recovered qualitatively similar loading vectors for the same behavioral motif in different worms (e.g., Fig. 6G). We quantified this relationship in two ways. First, we measured the angle between loading vectors (Figure S24). Second, we generated 'pseudo-factors' from the loadings of one worm and the neural activity of a second. If SCA is recovering truly similar loading vectors for different worms, then the magnitude of the pseudo-factor should correlate with the behavior of the worm that provided the neural activity.

For Figure S25, we generated pseudo-factors using the loading vectors related to factors with high correlations to behavior. Specifically, we identified the loading vectors associated with the highest 25% of these correlations. We then generated pseudo-factors for each second worm (i.e., the worm other than the one that provided the loading vector) that exhibited the relevant behavioral motif. The critical result – that pseudo-factors generated from SCA loadings were more correlated than those generated from PCA loadings – held if we constructed pseudo-factors from all of the original loading vectors (mean correlation(standard error): 0.34(0.009) vs. 0.29(0.008), $p < 0.001$, SCA vs. PCA, rank sum test).

Nonlinear SCA - unimanual cycling quantitative performance

As a rigorous comparison between linear autoencoders, SCA, and non-linear SCA, we performed a bi-cross-validation analysis across varying numbers of latent dimensions. For each model type, using *k* latent dimensions (x-axis), we fit the model to 80% of the 116 neurons within M1 across all 20 trial conditions in the cycling dataset. We then trained a linear regression model to map from the latent factors on 80% of the trials to the held-out 20% of neurons. Finally, we evaluated the fit linear regression model by predicting activity of the same 20% of held-out neurons using latent factors on the 20% of trials withheld from the linear regression model. This entire procedure was repeated five

times with independent splits of neurons and trial conditions, yielding a 5-fold cross-validation over neurons and a 5-fold cross-validation over trial conditions.

Software Availability

A code package for SCA is available at https://github.com/glaserlab/sca.

Supplemental Figures

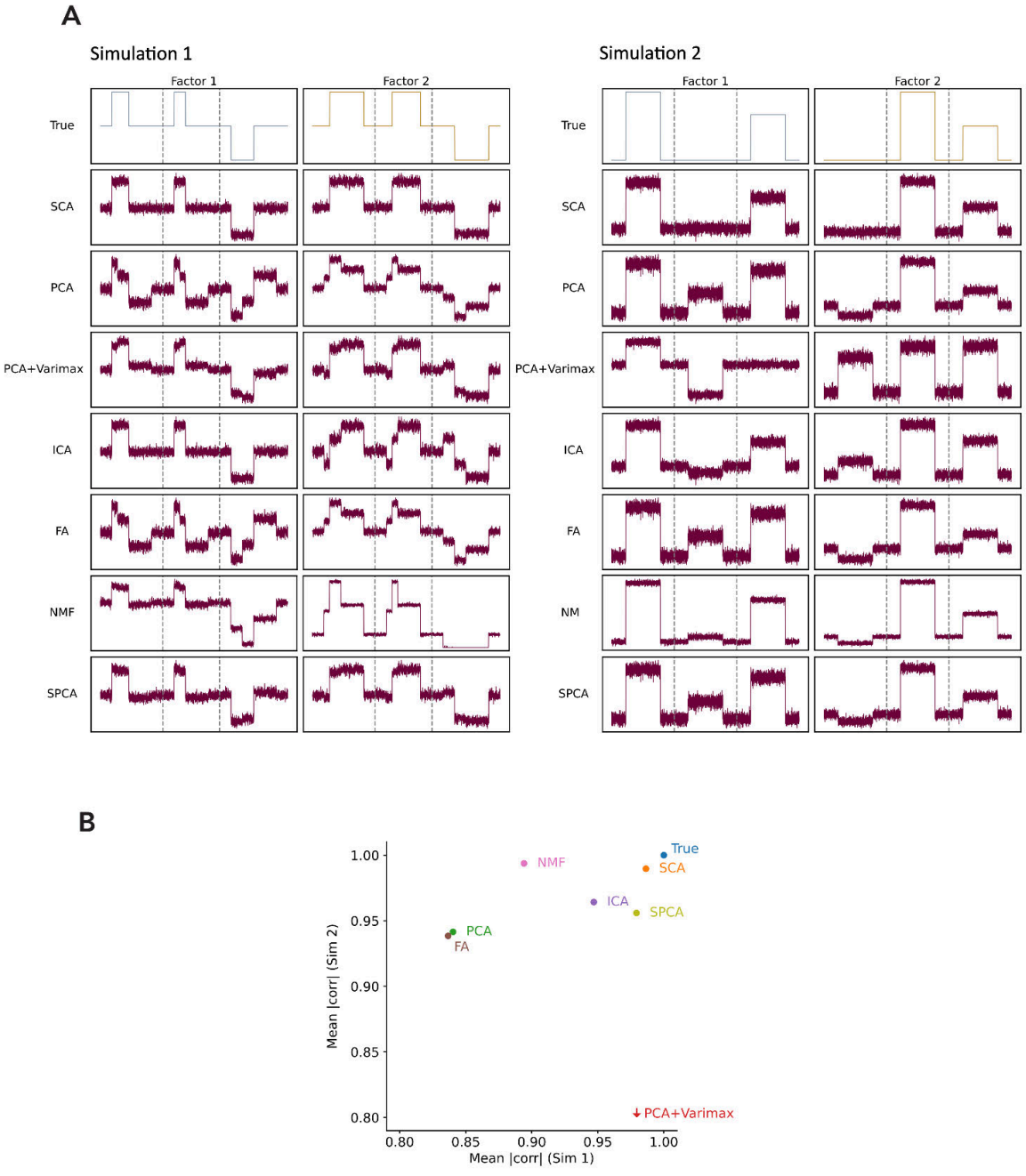


Figure S1. Comparison of methods on simulated datasets. (A) On top, ground truth latents, as in Fig. 1B (here, left column) and 1D (here, right column). In Fig. 1, conditions are shown in separate rows - here, they are shown concatenated together in time, with dashed vertical lines between conditions. In rows 2-8, the reconstructions are shown for SCA, PCA, and additional alternative unsupervised methods - PCA followed by a varimax rotation of the latents (PCA+Varimax),

Independent Component Analysis (ICA), Factor Analysis (FA), Nonnegative Matrix Factorization (NMF), and Sparse PCA, which encourages sparse loadings instead of latents (SPCA). We provide further descriptions of these methods in the Methods section. For all methods besides NMF, simulations were identical to Fig. 1 - creating neural activity modulation by multiplying the latents by orthonormal matrices. Because this leads to negative values of activity modulation, which cannot be analyzed with NMF, to give NMF the best chance, we multiplied latents by random positive loadings in those simulations. An alternative approach, using the original simulations and adding a constant to all activity values to create positive values, led to worse NMF reconstructions. Qualitatively, we can see that SCA most accurately decomposes neural activity into its constituent non-synchronous latent factors. **(B)**. We aimed to quantify how well each method decomposed neural activity into the ground-truth non-synchronous latents in the above example simulations. To do so, for each of the above two simulations, we calculated the mean (across the two latents) of the absolute value of the correlation of the recovered latents versus the ground truth latents. This was calculated after pairing each recovered latent with the most similar ground truth latent to maximize this correlation value. We plot these values against each other for each simulation (x and y axes). SCA most accurately decomposes neural activity into its constituent non-synchronous latent factors - i.e., it is closest to [1,1] in this plot.

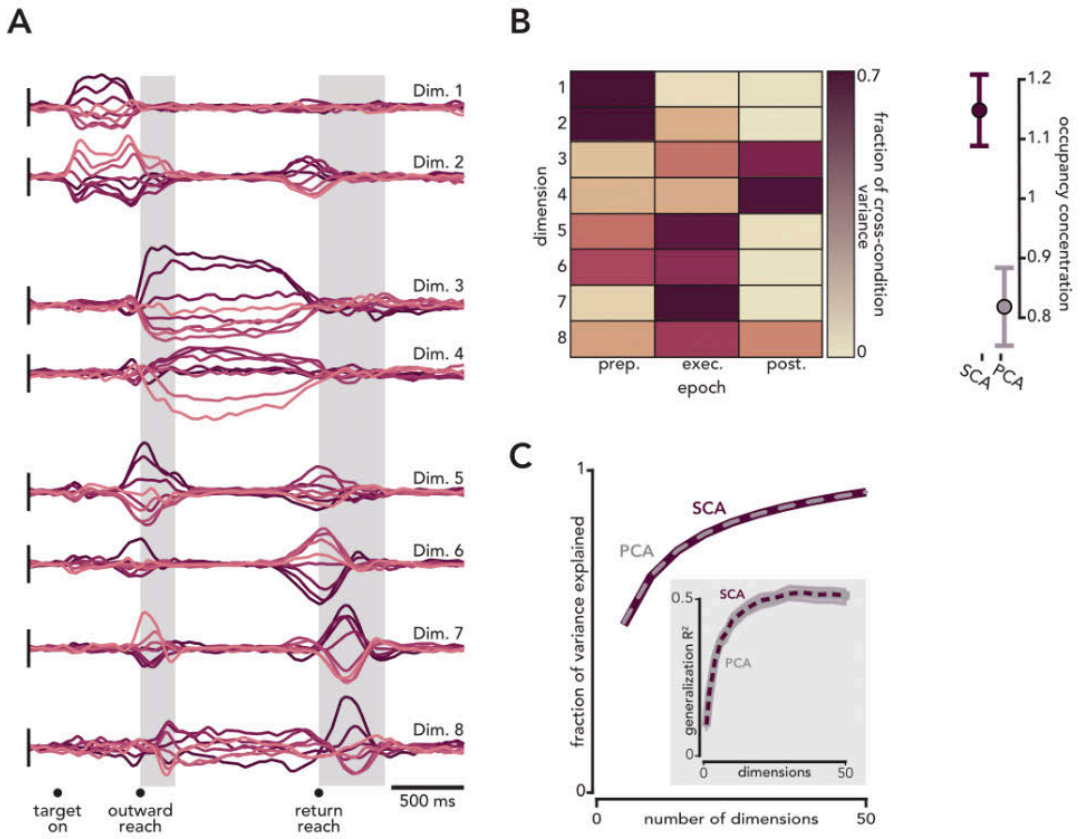


Figure S2. SCA applied to motor cortical reaching data from a second monkey. Same format as Figure 2. A. Eight SCA dimensions, ordered by the time point of the trial with maximum cross-condition variance. **B.** Left: fraction of occupancy in each SCA dimension across three task epochs. Right: Occupancy concentration (mean and standard deviation, calculated across redrawn populations, $p < 0.01$, n=100 populations). **C.** Cumulative fraction of variance explained. *Inset*. Reconstructing activity of held-out neurons during held-out conditions from SCA or PCA factors.

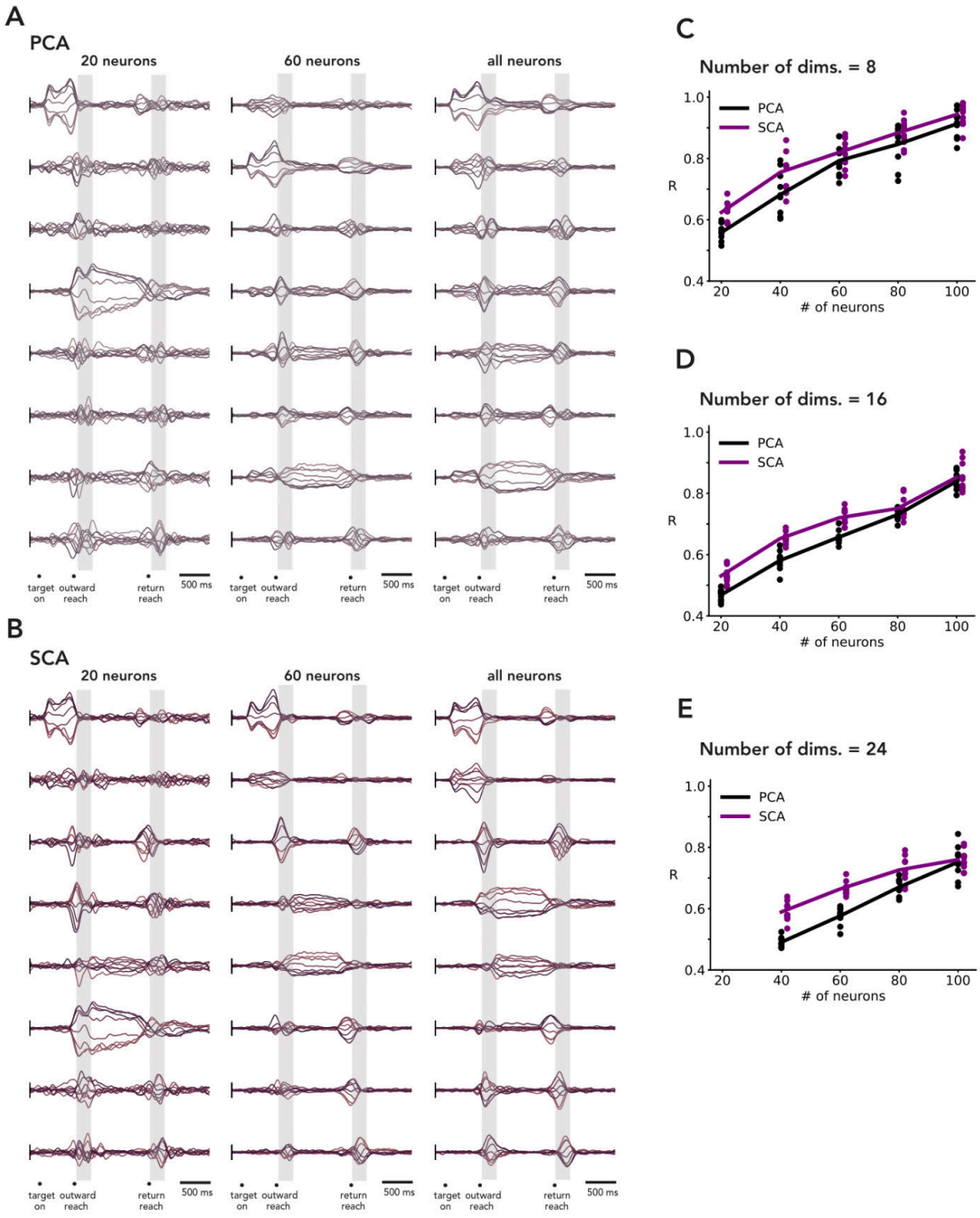


Figure S3. Similar consistency of SCA and PCA factors in sub-sampled neural populations. A. Eight PCA dimensions, using subsampled neural populations (Left: "20 neurons", Middle: "60 neurons") or the full population (Right: "all neurons", 124 neurons). Dimensions are ordered by the time point of the trial with maximum cross-condition variance. **B.** Same as A, for SCA. **C.** SCA factors are similarly, or slightly more, consistent across subsampled populations than PCA factors. SCA (or PCA) was performed on the subsampled populations to generate eight factors. The maximum correlation between each subsampling-generated factor and all original (i.e., full population) factors was calculated. The average maximum

correlation was then calculated for PCA and SCA. This process was repeated ten times for each number of subsampled neurons. **D, E.** Same as panel C, but for 16 and 24 SCA (or PCA) factors. In situations where there exist concerns regarding whether the number of neurons is sufficient to perform conclusions, we suggest performing a version of this analysis. If consistency is high, even when downselecting, that provides reassurance. One would also likely choose to inspect individual components (as in panel B). For example, a consistency of 0.6 may actually be tolerable, if the factor categories are consistent, and only the within-category bases differ.

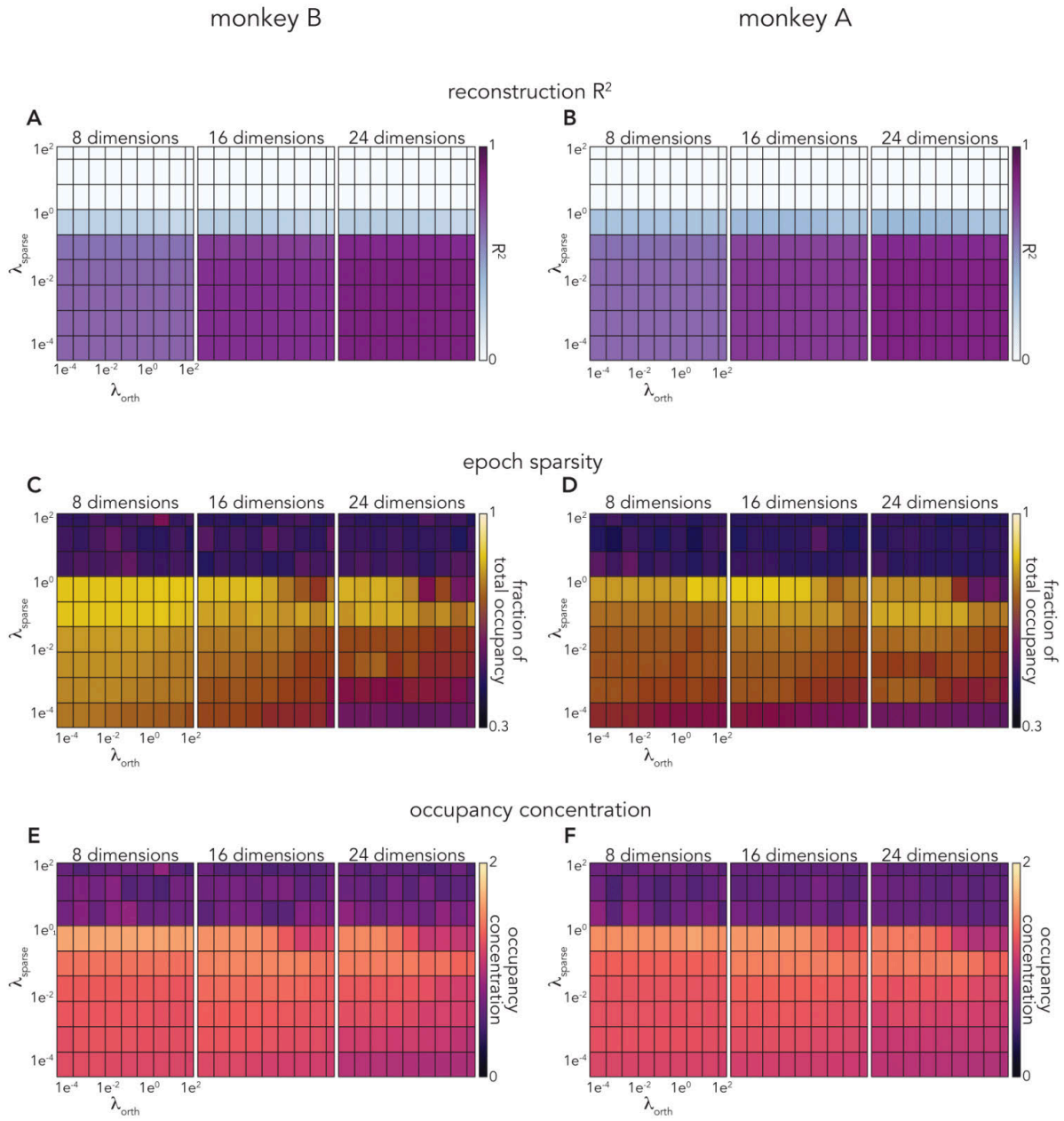


Figure S4. SCA performs similarly across a wide range of hyperparameters. To quantify the effects of changing λ_{orth} and $\lambda_{\mathrm{sparse}}$, we swept these parameters across six orders of magnitude. We found a relatively minor quantitative differences across the entire range of λ_{orth} values. Increasing $\lambda_{\mathrm{sparse}}$ had similarly small quantitative effects across roughly four orders of magnitude. However, above some threshold value ($\lambda_{\mathrm{sparse}}$ ≈ 0.1, *white rectangles*, in **A** and **B**), SCA prioritizes sparsity over minimizing reconstruction error, and the magnitude of the recovered factors tends towards 0. **A,B.** R^2 between trial-averaged neural activity and SCA reconstruction, monkeys B and A, respectively. **C,D.** For each SCA dimension, we first calculated the maximum fraction of the total occupancy accounted for within a single epoch (preparatory, execution, or posture). We then took the median (across all SCA dimensions) of the maximum fractional

occupancy values. **C** and **D** correspond to monkey B and A, respectively. **E,F.** Occupancy concentration (calculated as outlined in *Methods*), monkeys B and A, respectively.

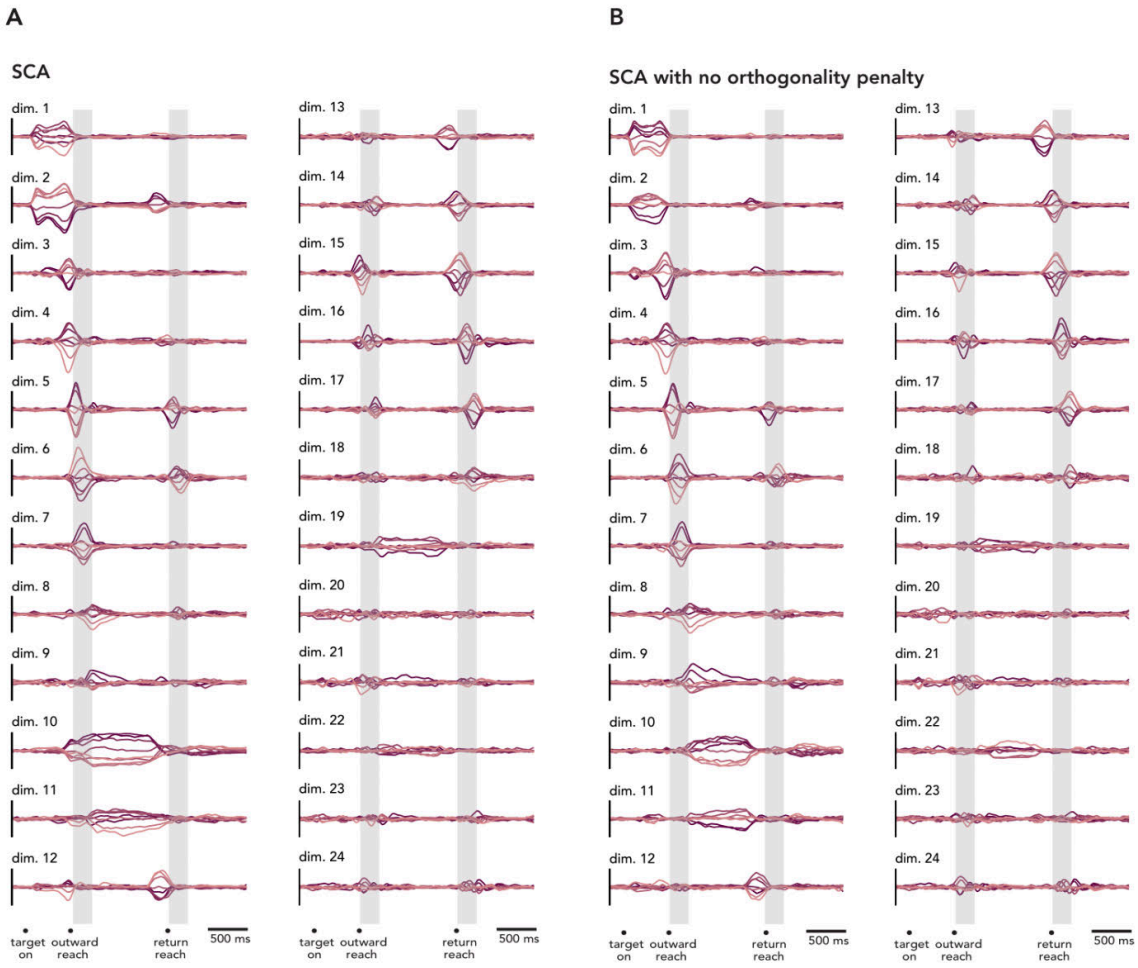


Figure S5. Using orthogonality makes SCA less likely to generate 'overly-sparse' factors. A. Projections in twenty-four SCA dimensions, using the standard orthogonality penalty, ordered by the time point of the trial with maximum cross-condition variance. These SCA dimensions share many of the same features as those plotted in Fig. 2, namely dimensions that are primarily occupied during preparatory, execution, or posture epochs. In the set of 24 SCA dimensions however, there are more dimensions that are only occupied prior to or during outward or return reaches (rather than occupied prior to/during both outward and return reaches). **B.** Projections in twenty-four SCA dimensions, without using any orthogonality penalty. A greater number of dimensions have substantial activity and are only occupied prior to/during outward or return reaches (rather than occupied prior to/during both outward and return reaches).

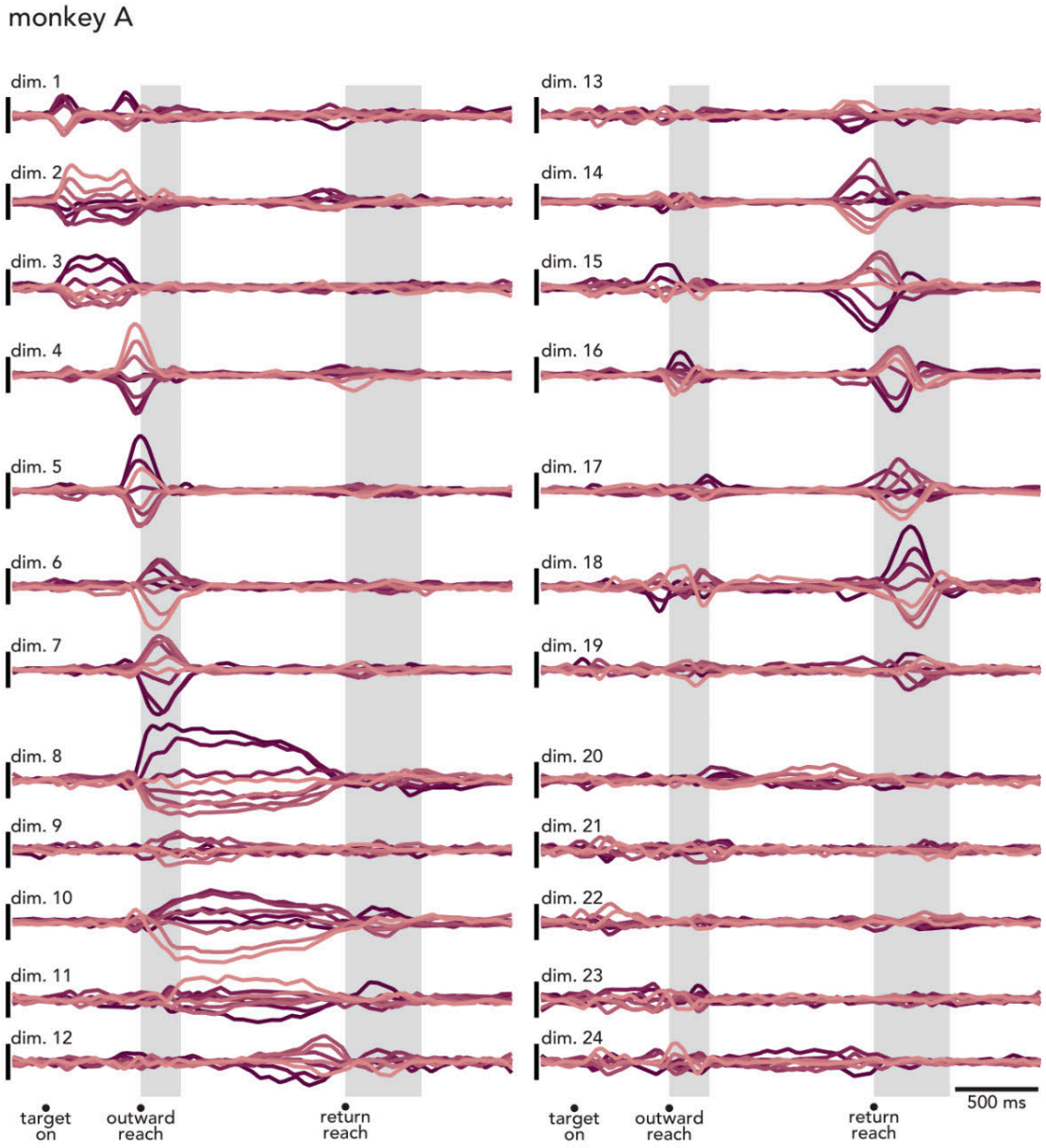


Figure S6. Twenty-four SCA dimensions, motor cortical reaching data, monkey A. Unlike monkey B, the SCA dimensions found for monkey A tended to be occupied prior to or during outward or return reaches, but not both. This difference is due to the greater dissimilarity between outward and return reaches for monkey A. For monkey A, return reaches were roughly half as fast as outward reaches (the mean peak speed for return reaches was 49% that of outward reaches). Relatedly, the alignment index for outward and return reaches was 0.29. For monkey B, the alignment index for outward and return reaches was 0.43.

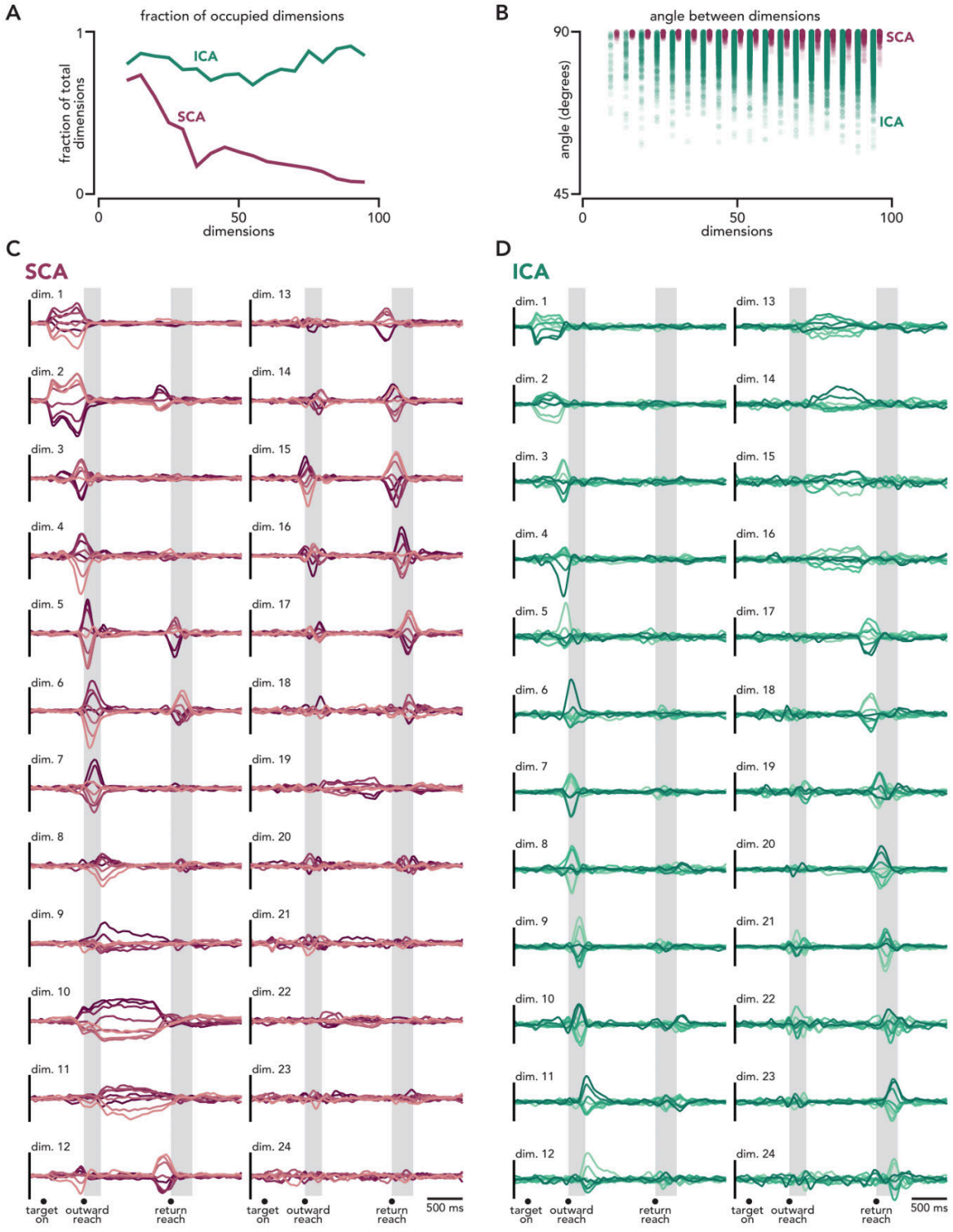


Figure S7. SCA is less likely to generate 'overly-sparse' factors than ICA. A. Fraction of dimensions with appreciable occupancy as function of total dimensions. A dimension was considered 'appreciably occupied' if the sum of the

cross-condition variance within that dimension was greater than 30% than that of the dimension with the largest cross-condition variance. **B.** Angle between all individual dimensions. As a larger number of dimensions are requested, both SCA (*purple*) and ICA (*green*) tend to find dimensions that are more aligned (have a smaller angle). However, this effect is much more dramatic for ICA. Note that even for a moderate number of dimensions (e.g., less than 20), ICA dimensions are more aligned than SCA dimensions. **C.** Projections in twenty-four SCA dimensions. These SCA dimensions share many of the same features as those plotted in Fig. 2, namely dimensions that are primarily occupied during preparatory, execution, or posture epochs. In the set of 24 SCA dimensions however, there are more dimensions that are only occupied prior to (or during) outward or return reaches. Note the number of dimensions that are largely unoccupied (e.g., dimensions 21-24). **D.** Projections in twenty-four ICA dimensions. Nearly all dimensions have substantial activity and are only occupied prior to/during outward or return reaches. Dimensions are ordered by the time point of the trial with maximum cross-condition variance in C and D.

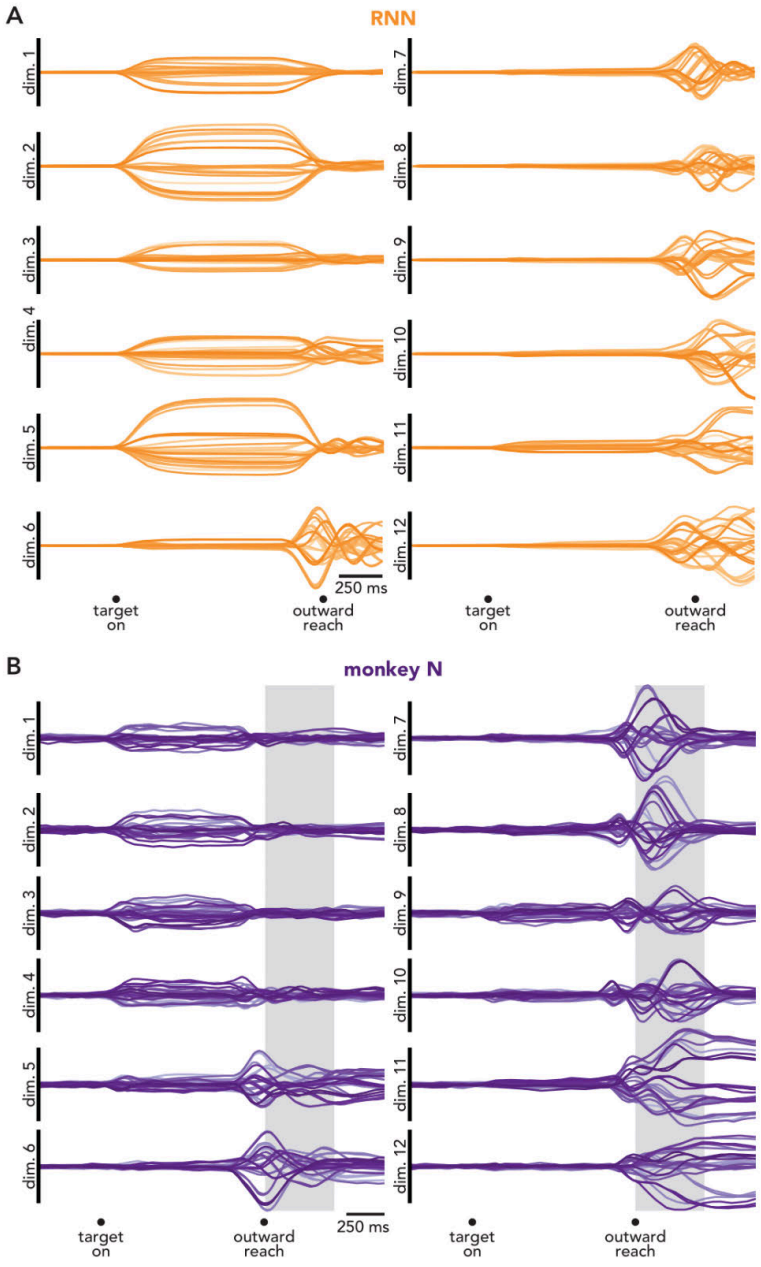


Figure S8. SCA can capture factors reflecting quasi-oscillatory dynamics. An obvious concern regarding SCA is that it might 'over-sparsify' data, parcellating activity into more dimensions than necessary. This concern interacts with present uncertainty regarding the nature of rotational dynamics in motor cortex during reaching. Neural network models of reach generation reproduce the empirical rotations using nonlinear dynamics whose linear approximation can be either quasi-oscillatory[14] or non-normal[56]. With oscillatory dynamics, neural trajectories exit dimension one, rotate into dimension two, then subsequently rotate back into dimension one (typically decaying as they do so). With non-normal dynamics, neural trajectories exit dimension one, rotate into dimension two, and then rotate into a third dimension without returning to dimension one[57,58]. Networks can blend both strategies[28] and the distinction can become blurry for nonlinear dynamics (e.g., time-varying eigenvectors can cause oscillatory dynamics to exhibit non-normal-like features).

Yet the solutions are different enough that it is worth distinguishing between them when possible. The data in Figure 2E, analyzed via SCA, are overall most consistent with non-normal dynamics. There are some biphasic aspects to responses, suggesting a modest oscillatory component, but overall activity tends to be monophasic and to rotate from some dimensions (e.g., dimension 6) into others (e.g., dimension 8) without returning. A key question is whether this feature – largely monophasic factors – simply appears because SCA optimizes for sparsity and monophasic factors are more sparse than multiphasic factors. Here we address that question in two ways. First, we analyzed population activity from a recurrent neural network (RNN) that is known to use a quasi-oscillatory solution. Second, we re-analyzed population activity that had previously been identified as having a sizable quasi-oscillatory component. If SCA works as desired – i.e., if it does not over-sparsify data – then it should return multiphasic factors in both cases. This was indeed the case. **A.** Twelve SCA dimensions, identified from the activity of an RNN[14] trained to generate empirical patterns of muscle activity, recorded from a monkey performing a task that required both curved and straight reaches (i.e, the 'maze' task[15]). Each *orange trace* represents one condition. Previous work[14] has thoroughly characterized the dynamics of this network. During reach execution, across all conditions, the network exhibits a single fixed point. The linearized dynamics, local to this fixed point, involve a small number of oscillatory modes. In agreement with this ground truth, SCA identifies multiphasic factors. Also in agreement with a quasi-oscillatory (rather than non-normal) solution, activity tends not to leave a dimension once it enters it. Most dimensions that are active just before movement onset (dimensions 6-12) remain active >250 ms after movement onset. This argues that the mostly monophasic factors in Figure 2E are not a distortion; if the empirical dynamics had been strongly quasi-oscillatory, SCA would have found more strongly multiphasic factors as it does for the network model shown here. For the data in Figure 2E, a better approximation is likely to be non-normal dynamics. Alternatively, oscillatory dynamics may involve eigenvectors that change swiftly as the overall location in state-space changes[59]. **B.** Twelve SCA dimensions identified in motor cortex activity from a monkey performing the maze task. Each *purple trace* represents a condition. Data are the same as those analyzed in Figure 3 and Supplementary Movie 3 of[15]. For this monkey, the factors identified by SCA resemble those when analyzing the network's population response. Many individual factors are multiphasic and are active across fairly long durations. This is in contrast to the data in Figure 2E (from monkey B) where execution-active factors are mostly monophasic and short-lived. The fact that SCA identified multiphasic factors for monkey N demonstrates that SCA recover this structure when it exists in the data. These results also suggest the two monkeys (N and B) use modestly different internal solutions despite performing similar tasks. The similarities and differences of solutions can be compared by considering what occurs in plots of the sort shown in Figure 2F. In a quasi-oscillatory solution (e.g., monkey N), the set of preparatory states (arranged along the vertical axis) rotates into execution dimensions (gray plane) and then continues to rotate, producing multiphasic factors. In a non-normal solution (e.g., monkey B), the set of preparatory states rotates into execution dimensions (as before). However, as rotations continue, they continuously enter new dimensions and leave old ones, resulting in monophasic factors. Mixed solutions are also possible. Indeed, neither monkey appears to use an entirely pure solution. There is some biphasic activity in the factors of monkey B, and monkey N shows some factors that are primarily active early in execution and others that are primarily active late. Presumably these two solutions lie on a spectrum of 'good' solutions that can be used by nonlinear recurrent networks.

A potentially related phenomenon was found in the case of the cycling task (Figure S13, S16). Some monkeys used largely distinct factors for forward and backward cycling (e.g., monkey D, Figure S16) and others used a few SCA factors during both conditions (e.g., monkey C, Figure S13). Whether such differences have meaningful computational or behavioral consequences, or whether they simply reflect redundancy in the available solutions, is an interesting future question.

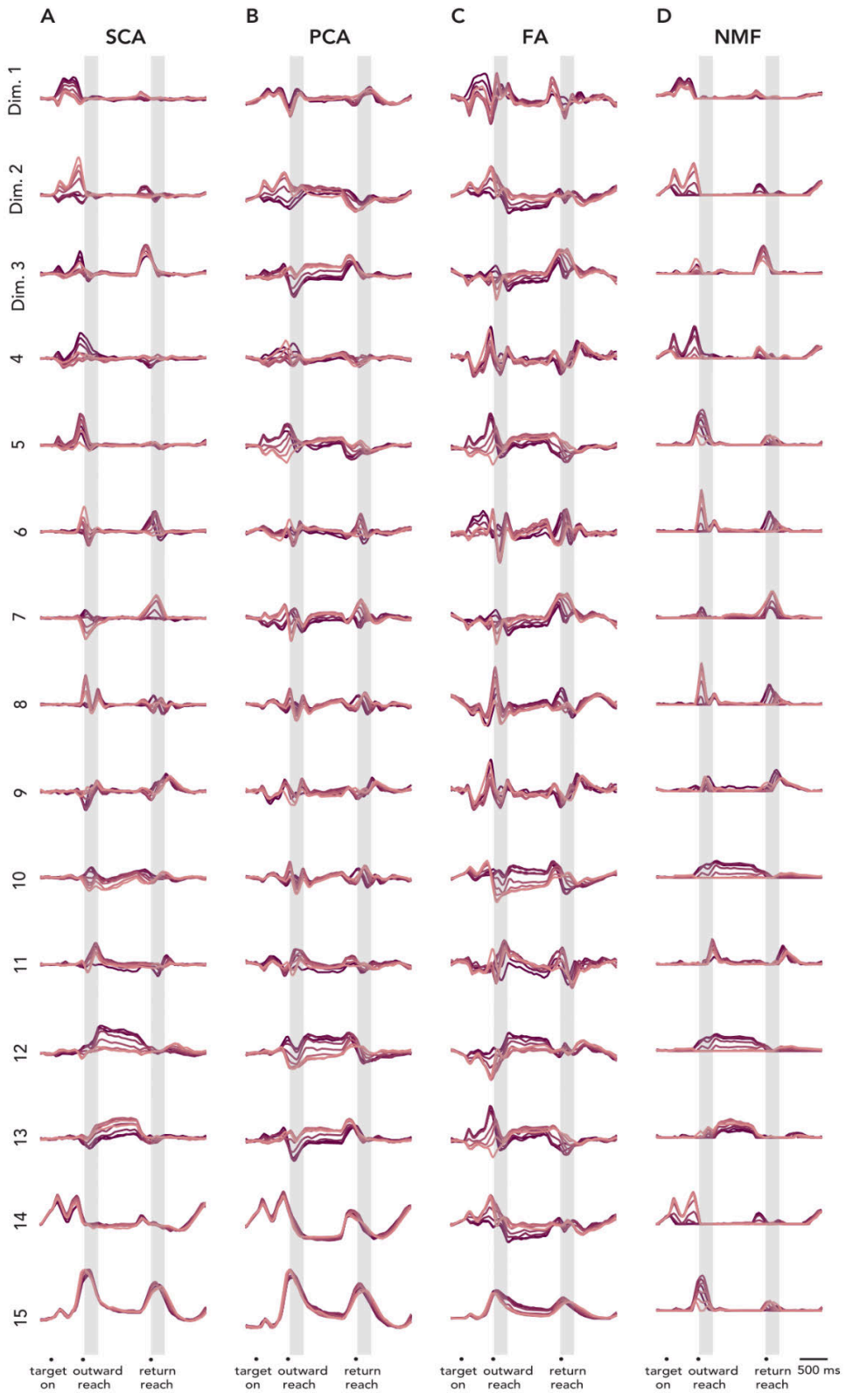

A
SCA
B
PCA
C
FA
D
NMF
Dim. 1
Dim. 2
Dim. 3
4
5
6
7
8
9
10
11
12
13
14
15
target on
outward reach
return reach
500 ms

Figure S9. Comparisons of unsupervised dimensionality reduction approaches on center-out reaching data. Data is from monkey B, and does not have the cross-condition mean removed. 15 dimensions are shown from **A.** SCA (without sample weighting, to enable clearer comparisons with all other methods), **B.** PCA, **C.** Factor Analysis (FA), **D.** Nonnegative matrix factorization (NMF). Data is colored by reach direction. SCA dimensions were manually ordered according to their putative computational role. For easier visual comparisons, factors from other methods were ordered such that they have maximum correlations with the corresponding SCA factor.

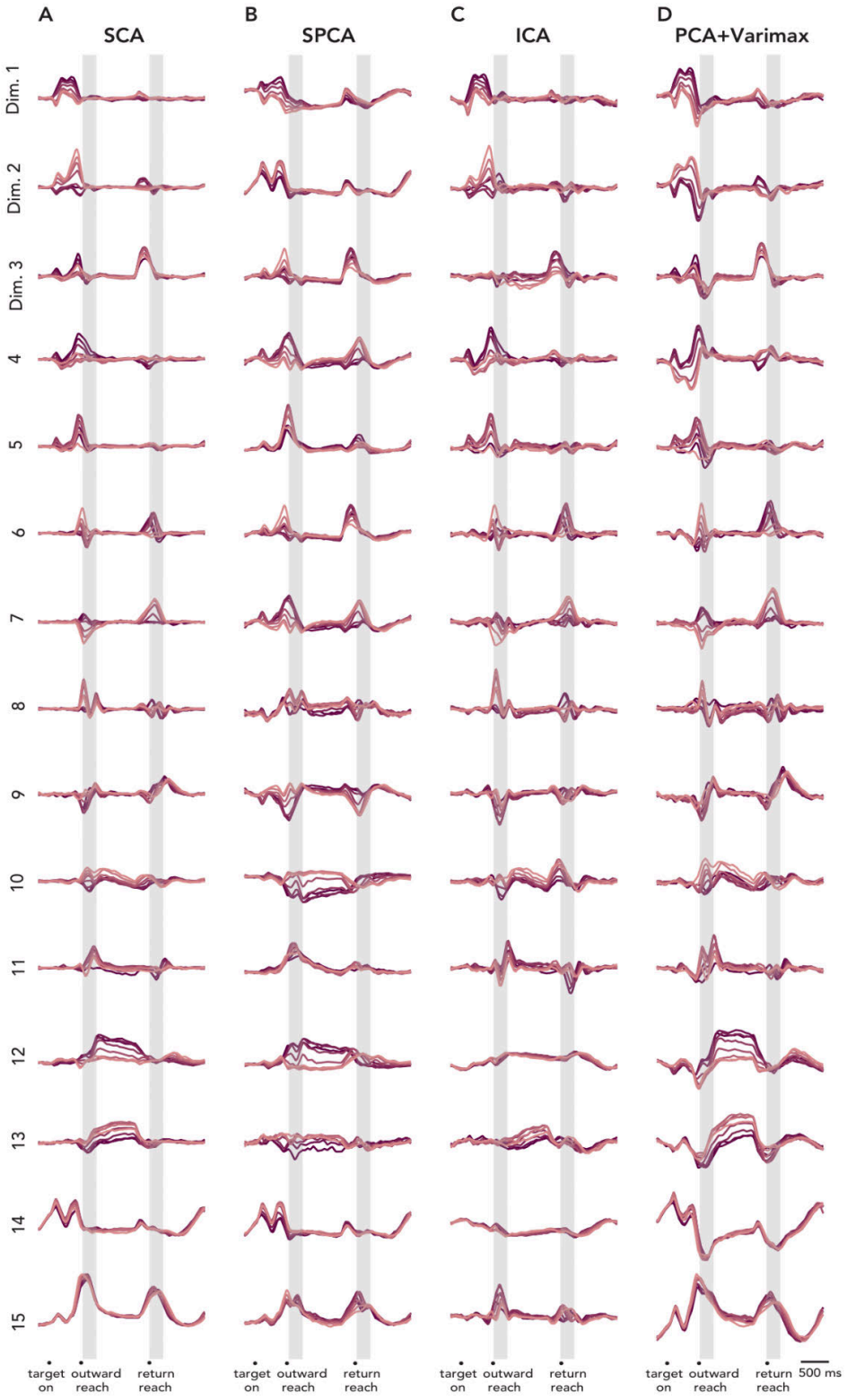

A
SCA
B
SPCA
C
ICA
D
PCA+Varimax
Dim. 1
Dim. 2
Dim. 3
4
5
6
7
8
9
10
11
12
13
14
15
target on
outward reach
return reach
500 ms

Figure S10. Comparisons of unsupervised dimensionality reduction approaches on center-out reaching data (continued). Data is from monkey B, and does not have the cross-condition mean removed. 15 dimensions are shown from **A.** SCA (without sample weighting, to enable clearer comparisons with all other methods), **B.** Sparse PCA (sPCA), which encourages sparse loadings rather than sparse factors, **C.** Independent Component Analysis (ICA), **D.** PCA followed by a varimax rotation of the factors. Data is colored by reach direction. SCA dimensions were manually ordered according to their putative computational role. For easier visual comparisons, factors from other methods were ordered such that they have maximum correlations with the corresponding SCA factor.

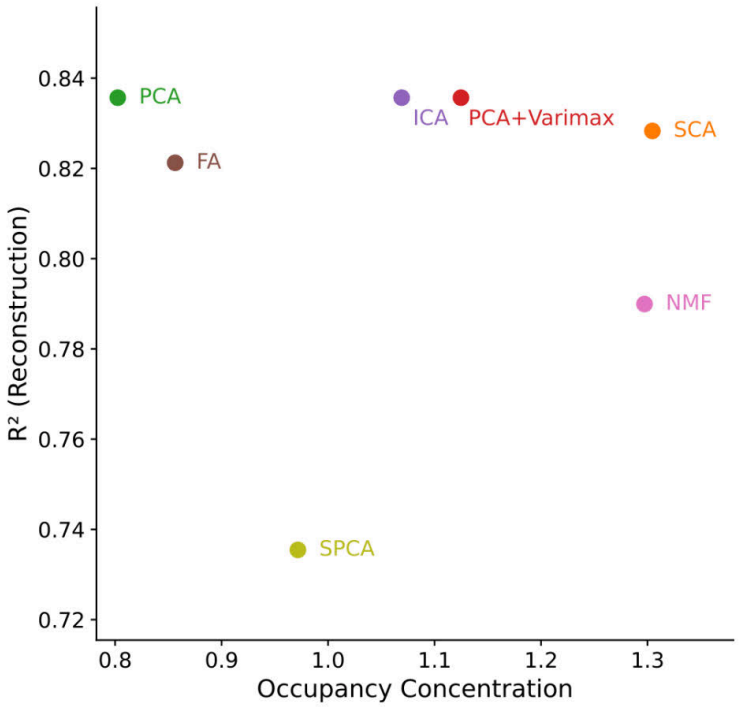


Figure S11. Summary of comparisons of unsupervised dimensionality reduction approaches on center-out reaching data. We quantified how well each of the methods from Figs. S9 and S10 were able to 1) reconstruct neural activity, as quantified by reconstruction R^2 (plotted on the y-axis) and 2) find distinct factors related to preparation, posture, and execution, as quantified by their occupancy concentration (as in Fig. 2H; plotted on the x-axis). Unlike other methods, SCA simultaneously achieved both goals, corresponding to being most top-right in the scatterplot.

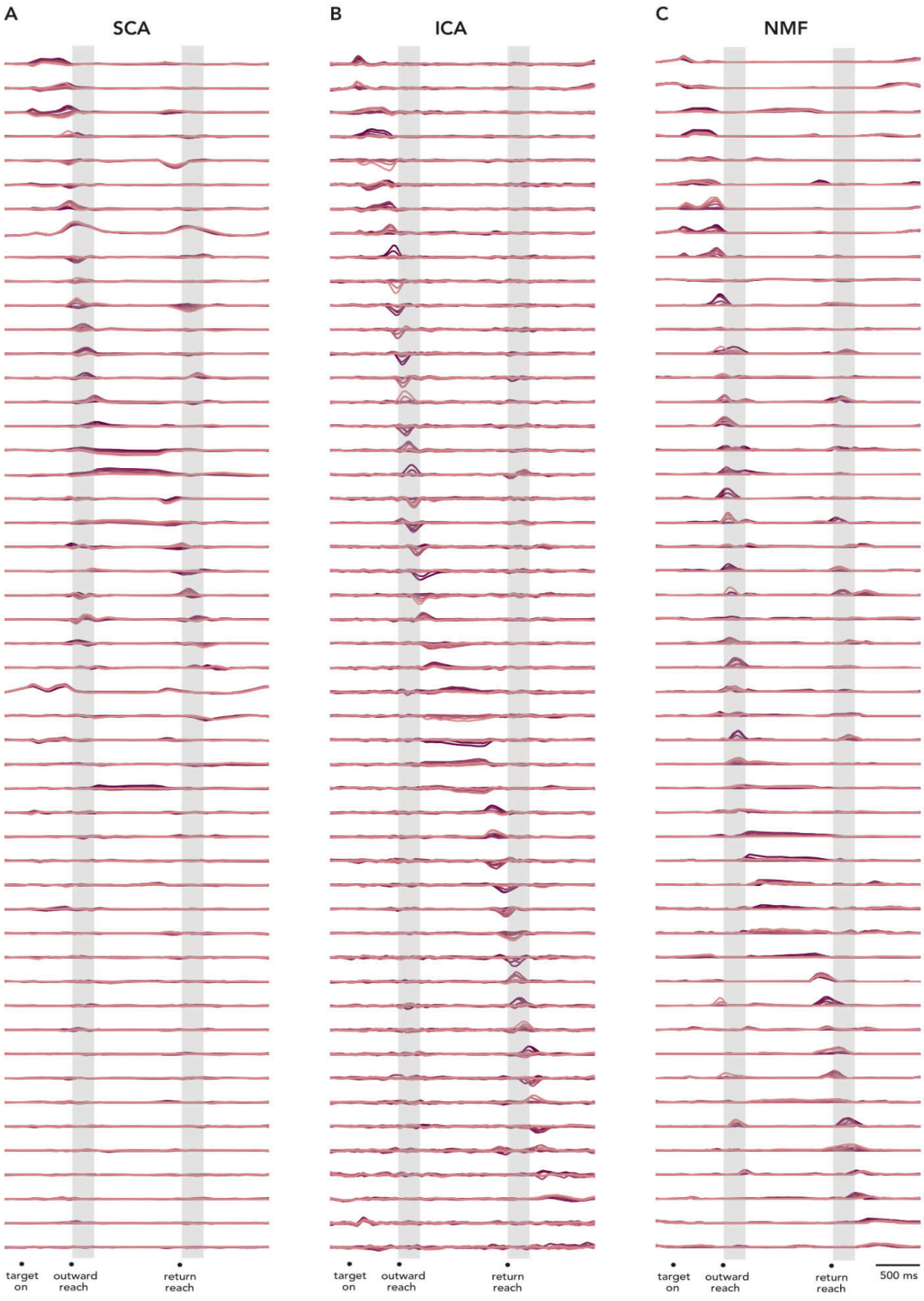

A
SCA
B
ICA
C
NMF
target on
outward reach
return reach
target on
outward reach
return reach
target on
outward reach
return reach
500 ms

Figure S12. SCA is less likely to generate 'overly-sparse' factors than ICA and NMF. Data is from monkey B, and does not have the cross-condition mean removed. 50 fit dimensions are shown from **A.** SCA, **B.** ICA, **C.** NMF. Data is colored by reach direction, and ordered by the time point of the trial with maximum variance. ICA and NMF tend to 'oversparsify' the latent factors and spread out the variance across more dimensions. We quantified the extent to which each method oversparsified the latents in terms of the extent which outwards and return reaches shared the same factors or were separated into separate factors. To do so, for each method, we identified execution-related factors as those that explained most variance during the execution epoch (outward + return) relative to the preparatory or posture epochs. We then computed the shared variance between outward and return reaches by dividing the smaller of the two variances by the larger. A factor was considered "shared" when this ratio exceeded X%. For X=25%, outward and return reaches share 78% (SCA), 26% (ICA), and 56% (NMF) factors. For X=50%, outward and return reaches share 52% (SCA), 4% (ICA), and 30% (NMF) factors. For X=75%, outward and return reaches share 35% (SCA), 4% (ICA), and 7% (NMF) factors. Note that the loadings of SCA are typically closer to orthogonal than those of NMF, which are closer to orthogonal than those of ICA, which likely explains these trends.

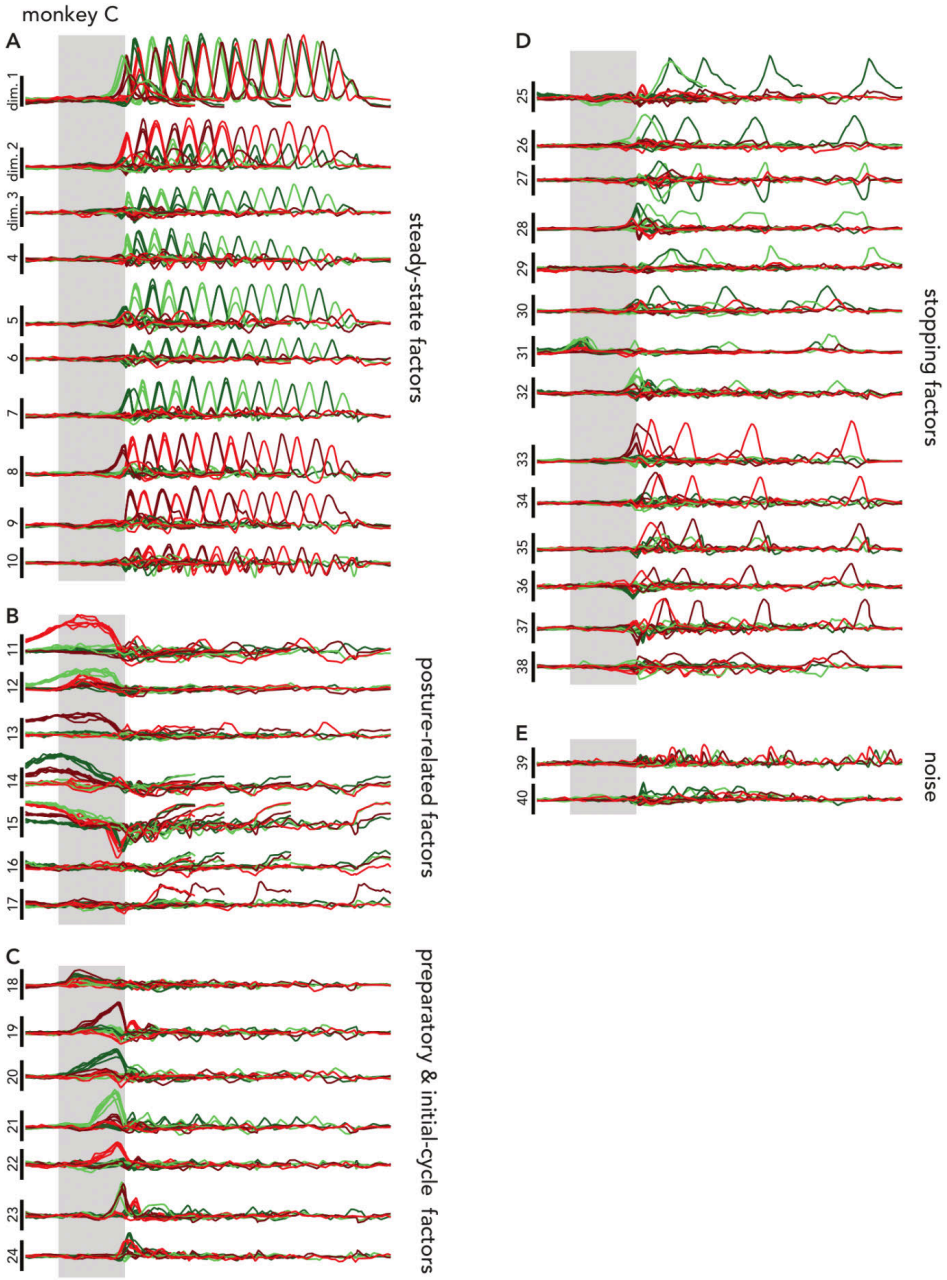


Figure S13. All SCA dimensions identified for monkey C. Dimensions are manually grouped by putative function. **A.** Steady-state factors. **B.** Posture-related factors. **C.** Preparatory factors**. D.** Stopping-related factors **E.** Other. Note that SCA's unsupervised approach does not itself group factors by putative function - that grouping depends on a user's scientific interpretation goals.

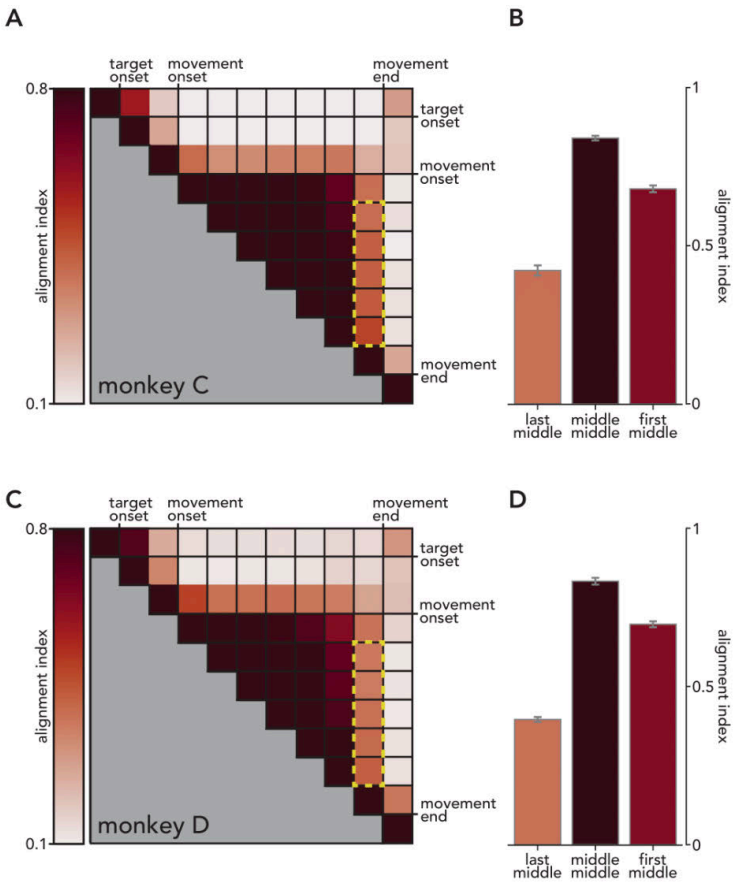


Figure S14. Alignment Indices for motor cortical activity during unimanual cycling. A. Pairwise alignment indices for all times during seven-cycle forwards movements. Each square corresponds to 500 ms (the duration of one cycle). The *dashed yellow box* corresponds to the alignment indices between the middle cycles and the final cycle. **B.** Mean alignment index (and standard error) for all pairs of individual cycles within four or seven cycle conditions (half, one, and two cycle conditions did not have middle cycles). **C,D.** Same as **A,B** but for monkey D.

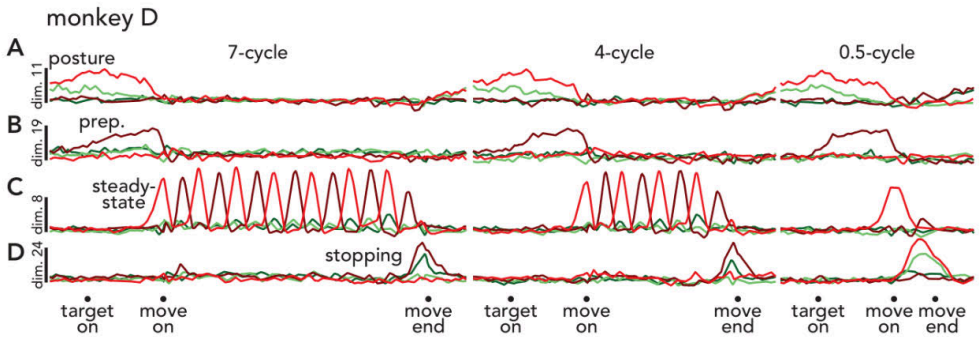


Figure S15. Example SCA dimensions from a second cycling monkey. Same format as Figure 4. **A.** Projections from all seven cycle (left), four cycle (middle), and half cycle (right) conditions in a single posture-related SCA dimension. During seven and four cycle conditions, activity for top-start conditions (*light traces*) is high at the start of the trial and begins to increase after movement ends. However, during half cycle conditions, activity related to bottom start conditions (*dark traces*) begins to increase after movement onset. This is expected, as the monkey's arm ends at the top of a cycle during bottom start, half cycle conditions. **B.** Projections from all seven, four, and half-cycle conditions in a single preparatory dimension. **C.** Projections in a steady-state dimension. **D.** Projections in a stopping-related dimension.

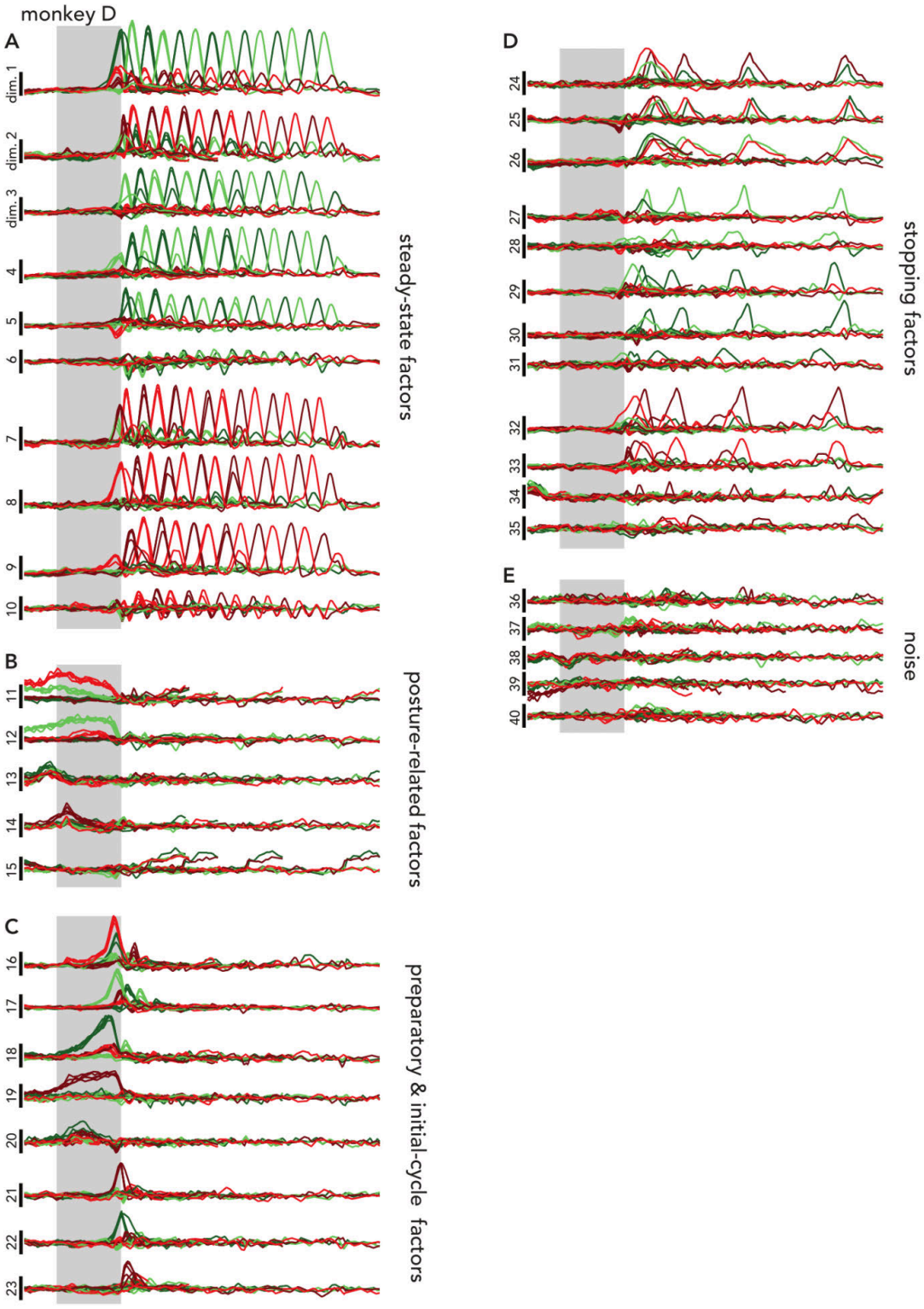

monkey D
A
dim. 1
dim. 2
dim. 3
4
5
6
7
8
9
10
steady-state factors
B
11
12
13
14
15
posture-related factors
C
16
17
18
19
20
21
22
23
preparatory & initial-cycle factors
D
24
25
26
27
28
29
30
31
stopping factors
E
32
33
34
35
36
37
38
39
40
noise

Figure S16. All SCA dimensions identified for monkey D. Dimensions are manually grouped by putative function. **A.** Steady-state factors. **B.** Posture-related factors. **C.** Preparatory factors**. D.** Stopping-related factors **E.** Other.

When comparing the SCA factors for the two unimanual cycling monkeys (monkey C and monkey D), the most notable difference was a higher degree of overlap between forward and backward steady-state activity in monkey C than in monkey D (compare *steady-state* factors here versus in Figure S13). In agreement with this difference in steady-state factors, the alignment between forward and backward cycling differed in the two animals. As stated in the main text, forwards and backwards cycling activity occupied partially overlapping spaces for monkey C (alignment index: 0.40±0.03 (mean and standard deviation, calculated across conditions), while the alignment index for these conditions was half as large for monkey D (alignment index: 0.20±0.01). While determining the functional significance of this difference (if any) is beyond the scope of this work, the fact that SCA preserves this structure demonstrates its utility as a discovery tool.

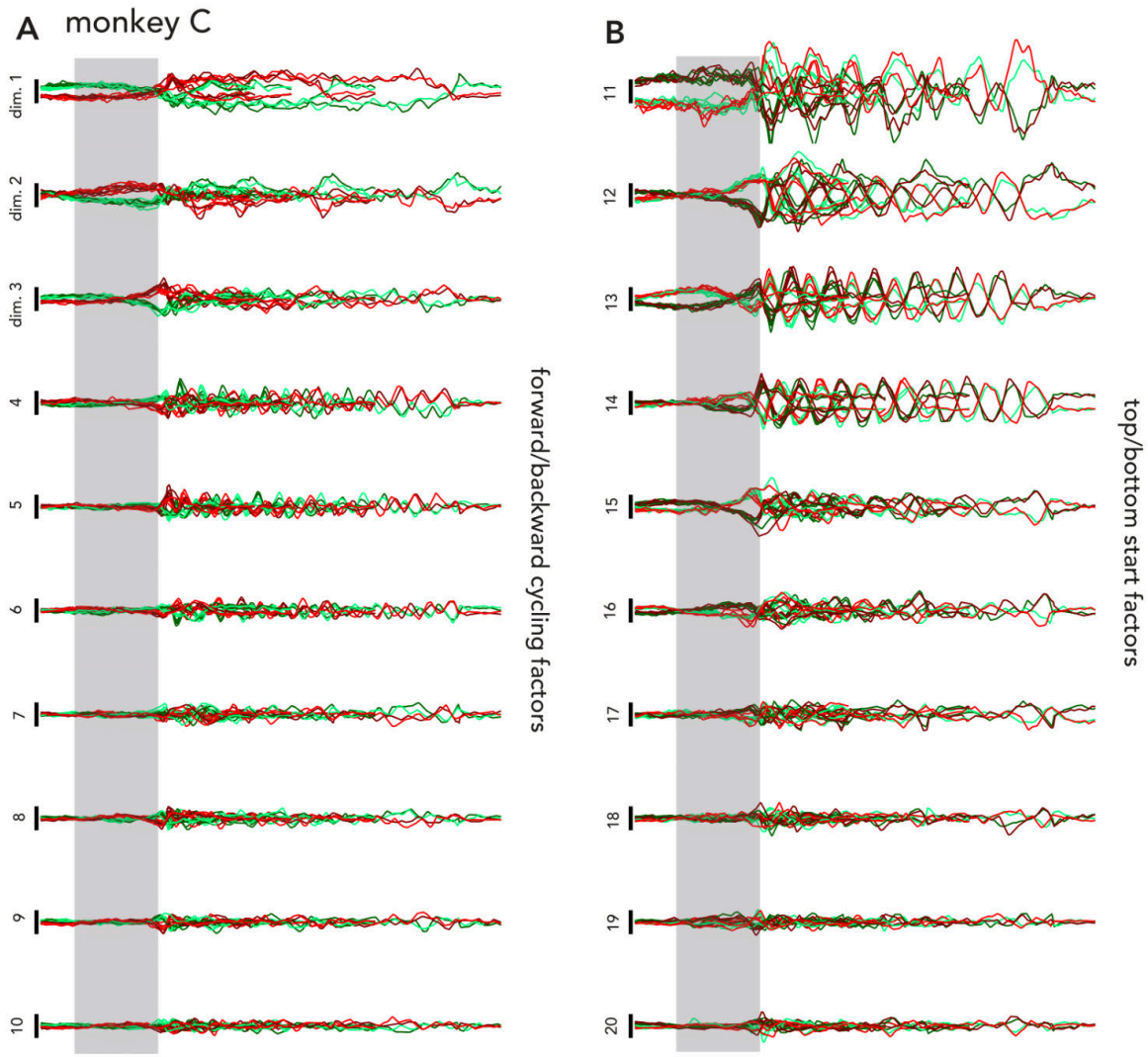


Figure S17. Demixed PCA factors on unimanual cycling data. We applied demixed PCA using forward/backward cycling, top/bottom start posture, and time as task parameters. The model was fit by requesting 20 factors in each marginalization, and we plotted the factors found within the two condition-time interaction marginalizations (forward/backward temporal factors and top/bottom temporal factors). 10 factors each are shown for the experimental conditions of forward vs backward cycling (column 1), and top versus bottom start posture (column 2), ordered by variance within each supervised factor category. Coloring is according to condition as done in Fig. 4. Demixed PCA learns factors related to these experimental conditions, rather than factors related to computations which are shared across task conditions (e.g. preparation and stopping).

Figure S18. All SCA dimensions for monkey F. Factors were grouped manually (and imperfectly) by the conditions during which they were most active. A factor was placed in a 'unimanual' group if the factor was strongly active during a unimanual condition. A factor was placed in a 'bimanual' group if it was strongly active during a bimanual condition and weakly active during all unimanual conditions.

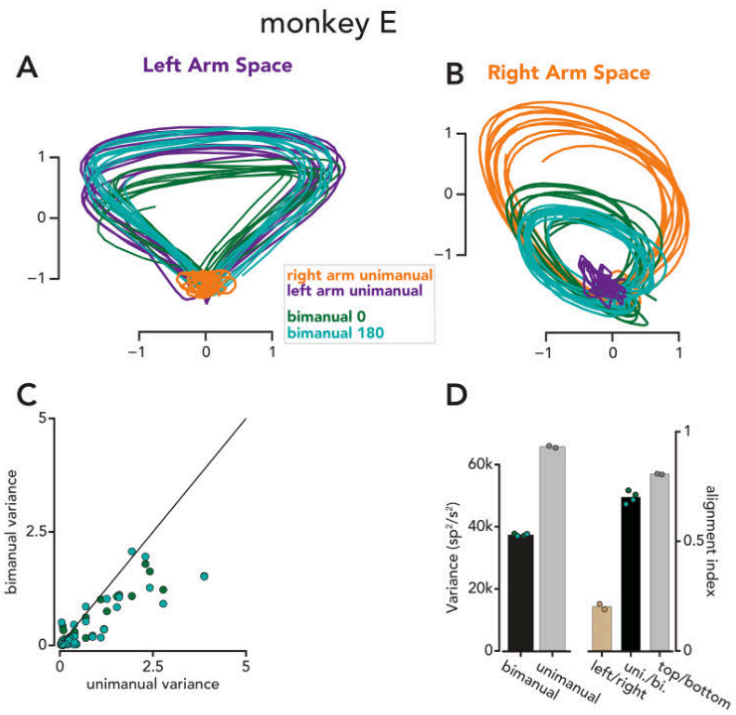


Figure S19. Summary of SCA results from a second bimanual cycling monkey. Same format as Figure 5. **A.** All forward conditions plotted in a space that is spanned by the two SCA dimensions that accounted for the largest fraction of left arm unimanual variance. **B.** Same as **A**, but for SCA dimensions that captured the largest fraction of right arm unimanual variance. **C.** Bimanual and unimanual variance in 50 SCA dimensions. **D.** Left: total neural variance during bimanual and unimanual conditions. Right: Alignment indices between left and right unimanual conditions.


monkey E
right arm unimanual, forwards
right arm unimanual, backwards
left arm unimanual, forwards
left arm unimanual, backwards
bimanual
early/late cycles
ramping
noise
500 ms
forwards
right arm
left arm
bimanual 0
bimanual 180
backwards
right arm
left arm
bimanual 0
bimanual 180

Figure S20. All SCA dimensions for monkey E. Same format as Figure S18. Across the four cycling monkeys, the degree of overlap between the neural dimensions occupied during forward and backward cycling differed fairly substantially; the alignment index between forwards and backwards cycling varied from 0.16 to 0.47 across monkeys. This difference in alignment was mirrored by the degree to which individual SCA factors were active for forward or backward cycling (compare Figures. S13, S16, S18, and S20). Whether these monkey-to-monkey differences reflect meaningful computational differences, with different, perhaps subtle, behavioral consequences, remains to be seen. Alternatively, the differences in overlap between forward and backward cycling could be largely arbitrary; networks may be able to fulfill computational requirements (e.g., generate neural trajectories with low tangling[61]) using either partially-aligned or largely-unaligned subspaces.

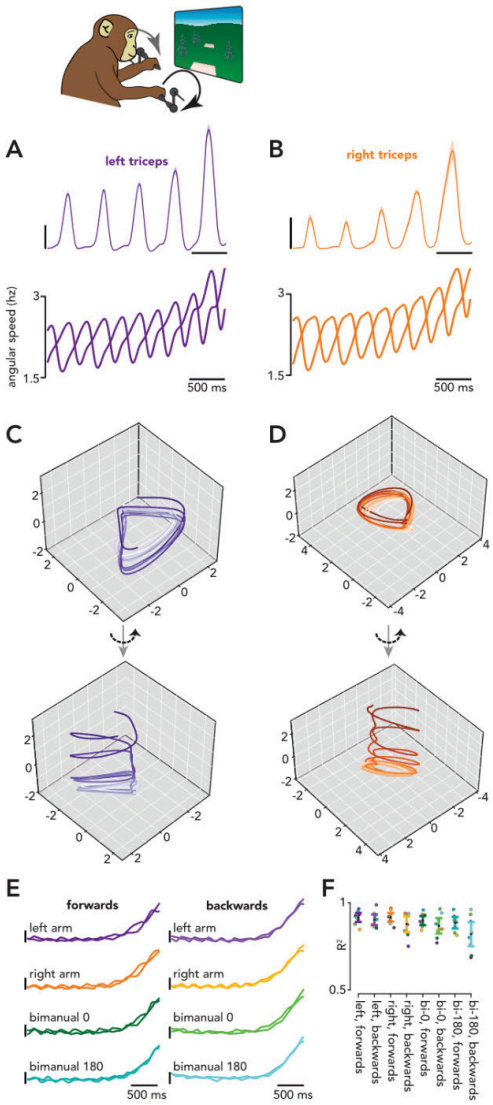


Figure S21. SCA identifies a shared 'speed' dimension in motor cortical bimanual cycling data. Prior work has explored the mechanisms underlying cycling at different speeds in both motor cortex and artificial networks[66]. Networks trained to empirical muscle activity recorded during cycling exhibit large, oscillatory signals that are correlated with the phase of the cycle; faster cycling speeds are associated with faster rotations in network activity. However, if network activity were to traverse the same region of state space at different speeds, 'tangling' (see[61]) would be high, and the network would be susceptible to noise[61]. To maintain low tangling, the networks translate the oscillatory signals associated with cycling at different speeds and/or 'tilt' these signals into different dimensions; such structure was also observed in motor cortex activity[66]. These prior results make the straightforward prediction that if a monkey were to steadily increase its cycling speed over time (rather than cycle at different speeds on different trials, as was done in Saxena *et al.*), activity in motor cortex should be helical.

During the bimanual cycling task, one monkey (monkey E) exhibited the idiosyncratic behavior of substantially increasing his cycling speed towards the end of a trial. **A.** Top: trial-averaged EMG from the lateral head of the left triceps during a

left arm unimanual condition. Bottom: Trial-averaged angular speed of the left arm during two left arm unimanual cycling conditions. **B.** Same as **A,** but for the right arm during right arm unimanual conditions. We performed SCA on all 16 conditions (2 cycling directions x 4 arm conditions x 2 starting pedal positions) and recovered the helical structure hypothesized by Saxena *et al.* SCA recovered a single dimension in which activity correlated with average cycling speed (rho = 0.92±0.03, mean and standard deviation, across conditions). To illustrate the geometry of these signals, we have plotted activity in two three-dimensional state spaces (**C,D**); two of the dimensions capture the large, oscillatory signals that are present throughout the trial, and the third dimension is the 'speed dimension'. The projections can be oriented such that the trajectories trace out a repeating, rotational pattern (**C,D,** top). However, rotating the projection reveals that the trajectories actually form a helix, with the first three cycles evolving along the base of the helix and the final two cycles ascending the helix (**C,D,** bottom).

Monkey E increased his cycling speed toward the end of all conditions. **E.** Projections from all sixteen conditions in a single SCA dimension that correlates with intra-cycle speed. The two traces in each dimension correspond to top start and bottom start conditions. Given this result, two possibilities exist. Either activity truly does primarily evolve along a single speed dimension, or there are multiple speed dimensions, and SCA has found a rotation of these dimensions in which activity evolves along a stereotyped trajectory. To disambiguate these possibilities, we performed a regression analysis. For each combination of moving arm and cycling direction, we used activity from the top-start condition to learn a set of neural weights that predicted low-pass filtered cycling speed for that condition. We then predicted the (low-pass filtered) cycling speed for all other bottom-start conditions. **F.** Validation of a single 'speed axis' across conditions. For each bottom-start condition, regression was used to identify a single neural dimension that correlated with the angular speed during that condition (see *Methods*). Neural activity from all top start conditions was then projected into this dimension, and the R^2 between the projection and angular speed of that top start condition was calculated (*colored dots*). Error bars correspond to mean and standard deviation across conditions. While the decoder often performed the best for the matched bottom-start condition (compare the colors of the *error bars* and the highest *circle* in each column of **F**), this was a minor improvement over the non-matched conditions, indicating that the neural activity largely translated along a single dimension as the monkey cycled faster, regardless of condition.

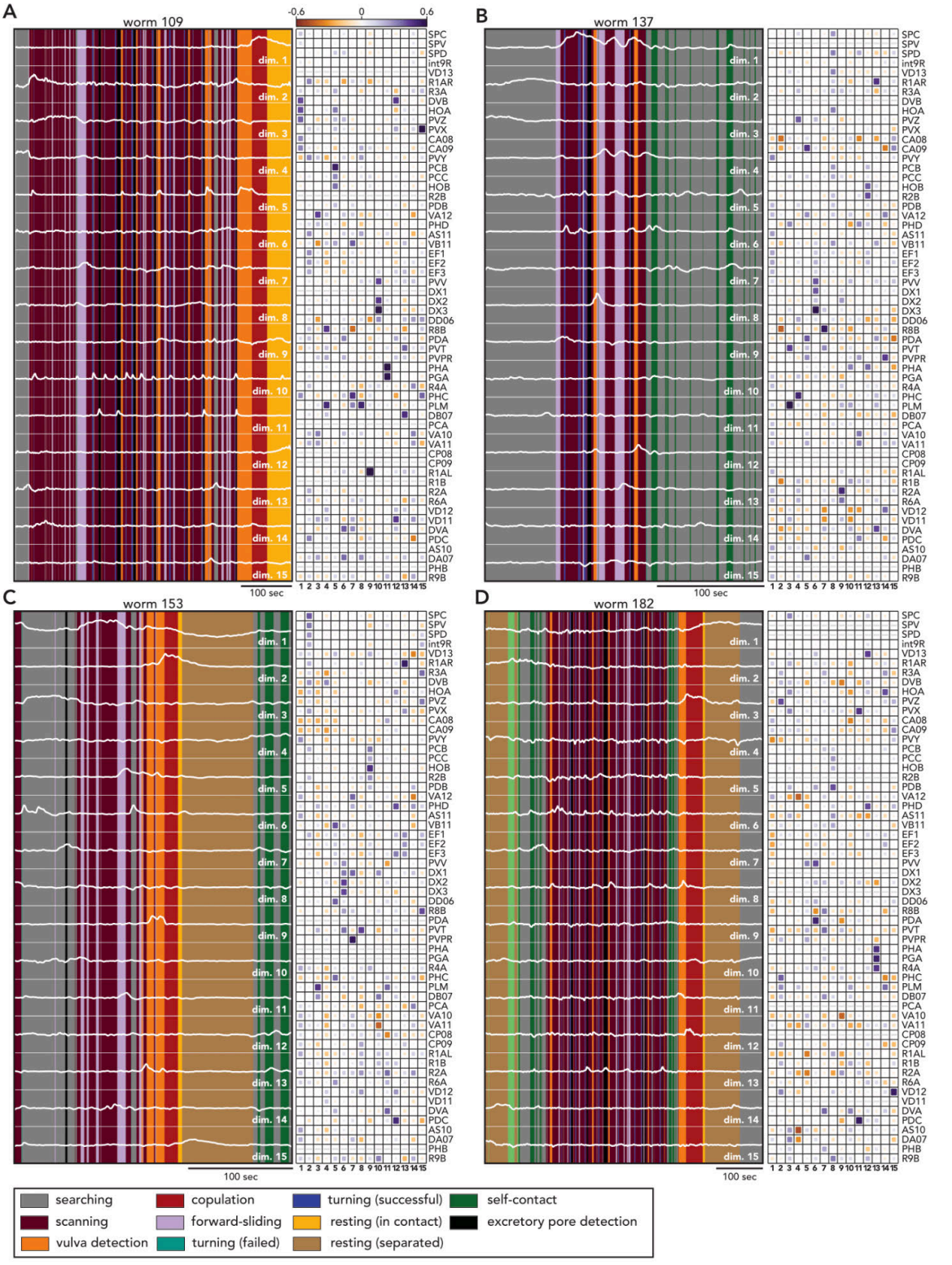

A
worm 109
B
worm 137
C
worm 153
D
worm 182
-0.6
0
0.6
dim. 1
dim. 2
dim. 3
dim. 4
dim. 5
dim. 6
dim. 7
dim. 8
dim. 9
dim. 10
dim. 11
dim. 12
dim. 13
dim. 14
dim. 15
100 sec
SPC SPV SPD int9R VD13 R1AR R3A DVB HOA PVZ PVX CA08 CA09 PVY PCB PCC HOB R2B PDB VA12 PHD AS11 VB11 EF1 EF2 EF3 PVV DX1 DX2 DX3 DD06 R8B PDA PVT PVPR PHA PGA R4A PHC PLM DB07 PCA VA10 VA11 CP08 CP09 R1AL R1B R2A R6A VD12 VD11 DVA PDC AS10 DA07 PHB R9B
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
searching
scanning
vulva detection
copulation
forward-sliding
turning (failed)
turning (successful)
resting (in contact)
resting (separated)
self-contact
excretory pore detection

Figure S22. All SCA dimensions for four worms. A. Left: Projections in all SCA dimensions (*white traces*), plotted on top of the worm's ethogram. The dimensions were not ordered. Right: SCA loadings. Neurons that were not recorded are indicated with a *gray dash*. **B-D.** Same as **A**, for three additional worms.

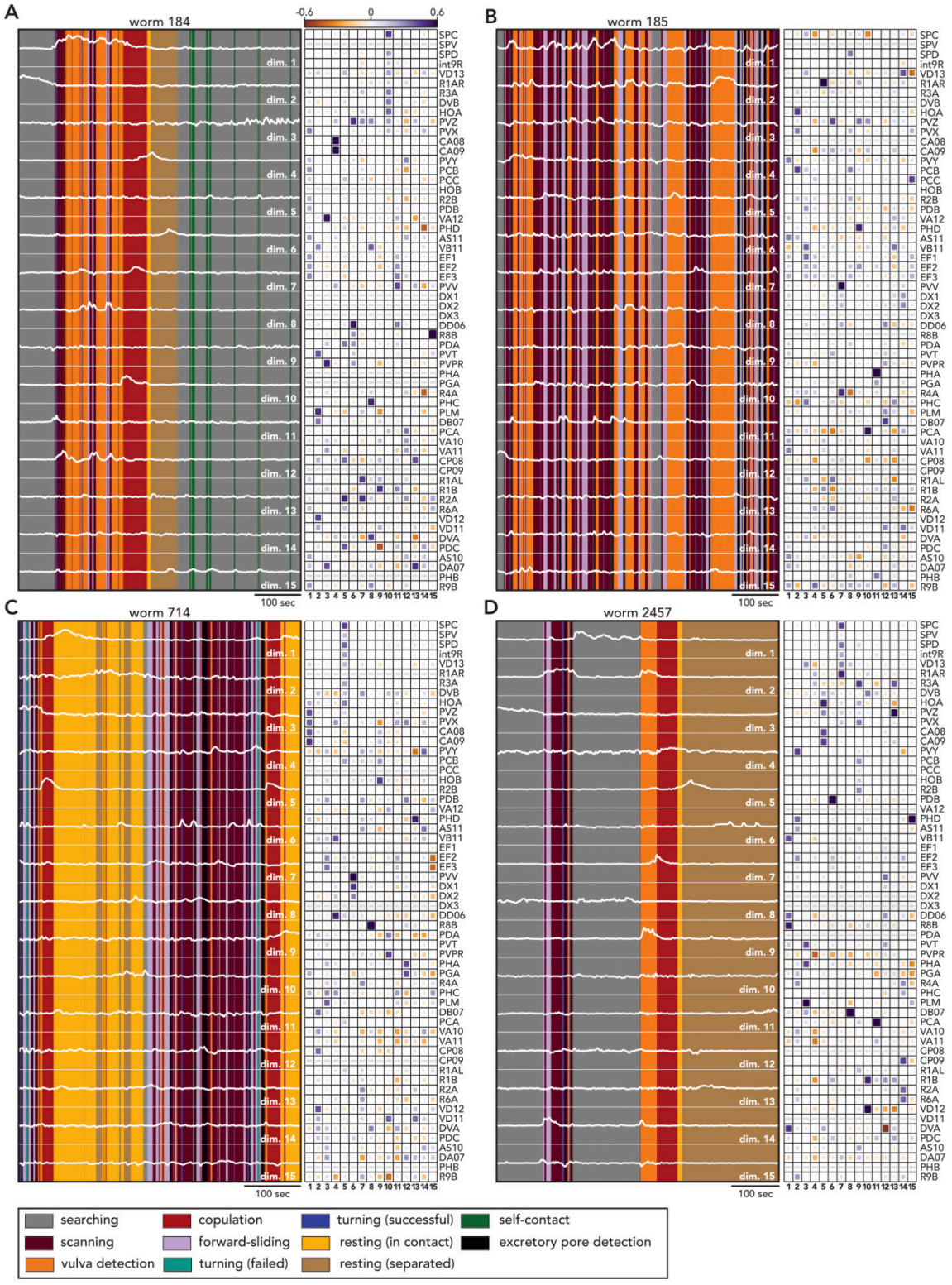


Figure S23. All SCA dimensions for remaining worms. Same format as Figure S22.

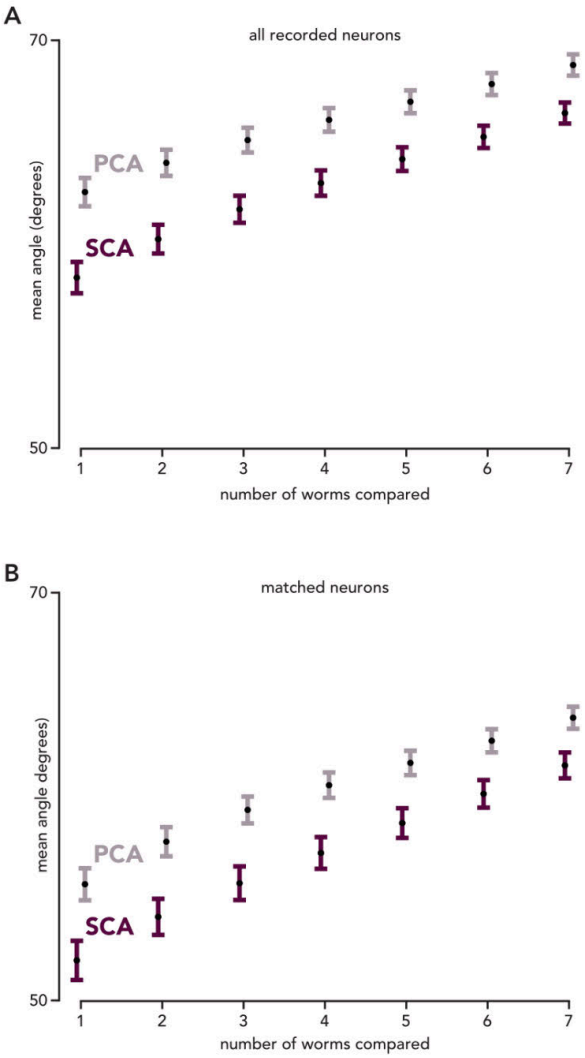


Figure S24. SCA loading vectors are more similar between worms than PCA loading vectors. A. For each SCA (or PCA) loading vector for each worm, we identified the seven most similar loading vectors from all other worms (each worm could only contribute a single vector). We then calculated the mean angle between the most similar vectors, the two most similar vectors, etc. Here, the loading values for missing neurons were set to 0. *Error bars* correspond to standard error, $p < 0.001$ for all SCA-PCA comparisons, rank-sum test. **B.** Same as above, except the analysis is limited to the 33 neurons that were shared between all 8 worms; $p < 0.01$ for all comparisons, rank-sum test.

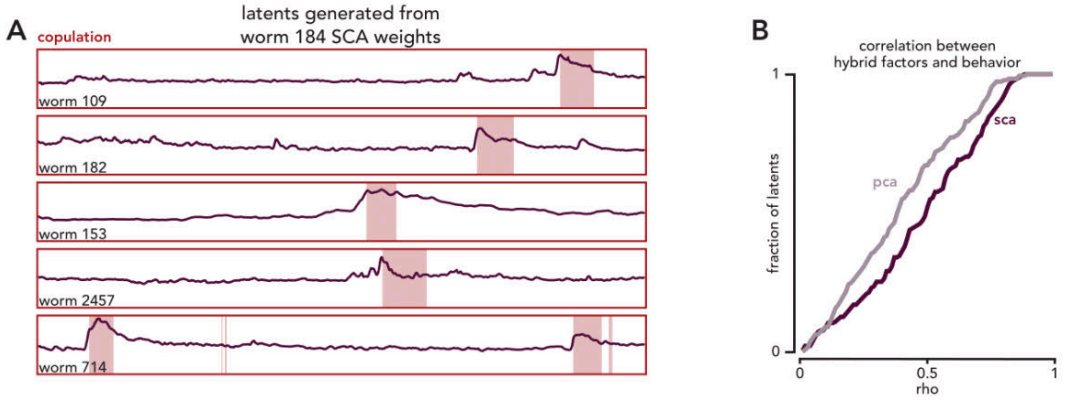


Figure S25. SCA identifies similar dimensions across individual worms. To test generalizability of SCA loadings, we used the SCA loadings from one worm to predict a second worm's behavior. Given an SCA factor that was related to a particular motif in one worm (worm-A) and a second worm that exhibited the same motif (worm-B), we generated a 'hybrid factor' using SCA loadings from worm-A and neural activity from worm-B. We asked how well the hybrid factor correlated with worm-B's behavior. **A.** 'Hybrid factors' generated from SCA weights from worm 184 and the neural activity of five other worms that copulated. These hybrid factors were well correlated with copulation timing (rho = 0.59±0.13, mean and standard deviation, across worms). **B.** Correlations between hybrid factors generated from SCA or PCA weights and behavior (mean rho = 0.48±0.02 for SCA versus 0.39±0.02 for PCA, $p < 0.001$, rank sum test). The rightwards shift of the *purple trace* (relative to the *gray trace*) indicates that the correlations between SCA factors and behavior were higher than those between PCA factors and behavior.

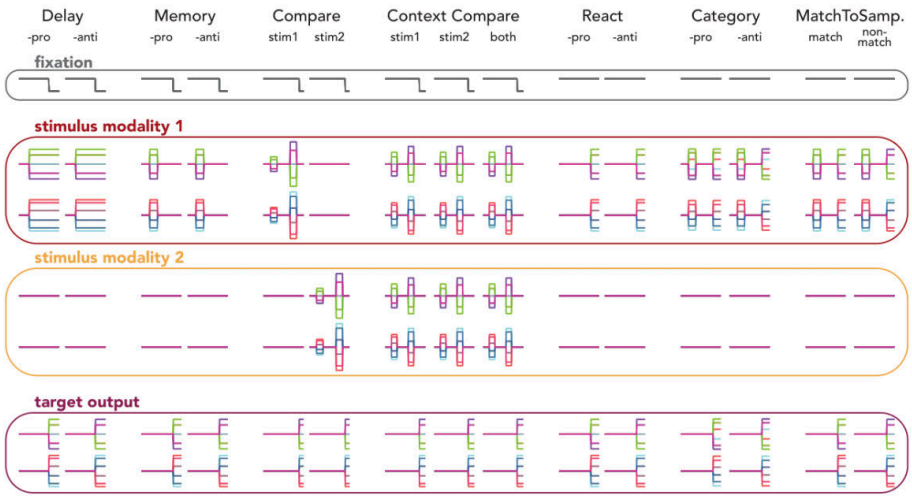


Figure S26. Inputs and target outputs for the multitask network. Across all tasks, the network received three inputs: a 1-dimensional fixation signal (*gray box*), a 4-dimensional stimulus (corresponding to two, 2-dimensional circular stimuli, both scaled by amplitude A, *red* and *yellow boxes*), and a 15-dimensional context cue throughout each trial (not shown). On each trial, the network generated a 2-dimensional output (*maroon box*), corresponding to a single circular variable (the sine and cosine of an angle). Traces are colored by stimulus angles

The removal of the fixation signal acted as a go cue during Delay-, Memory-, Compare-, and ContextCompare- tasks. During the remaining tasks, output was triggered by the delivery of directional stimuli.

The context cue was provided throughout each trial and determined how the network should respond to the directional stimuli. Depending on the task, the network needed to attend to stimulus modality 1 (*red box*), stimulus modality 2 (*yellow box*), or both. For a complete description of each task, see *Methods*.

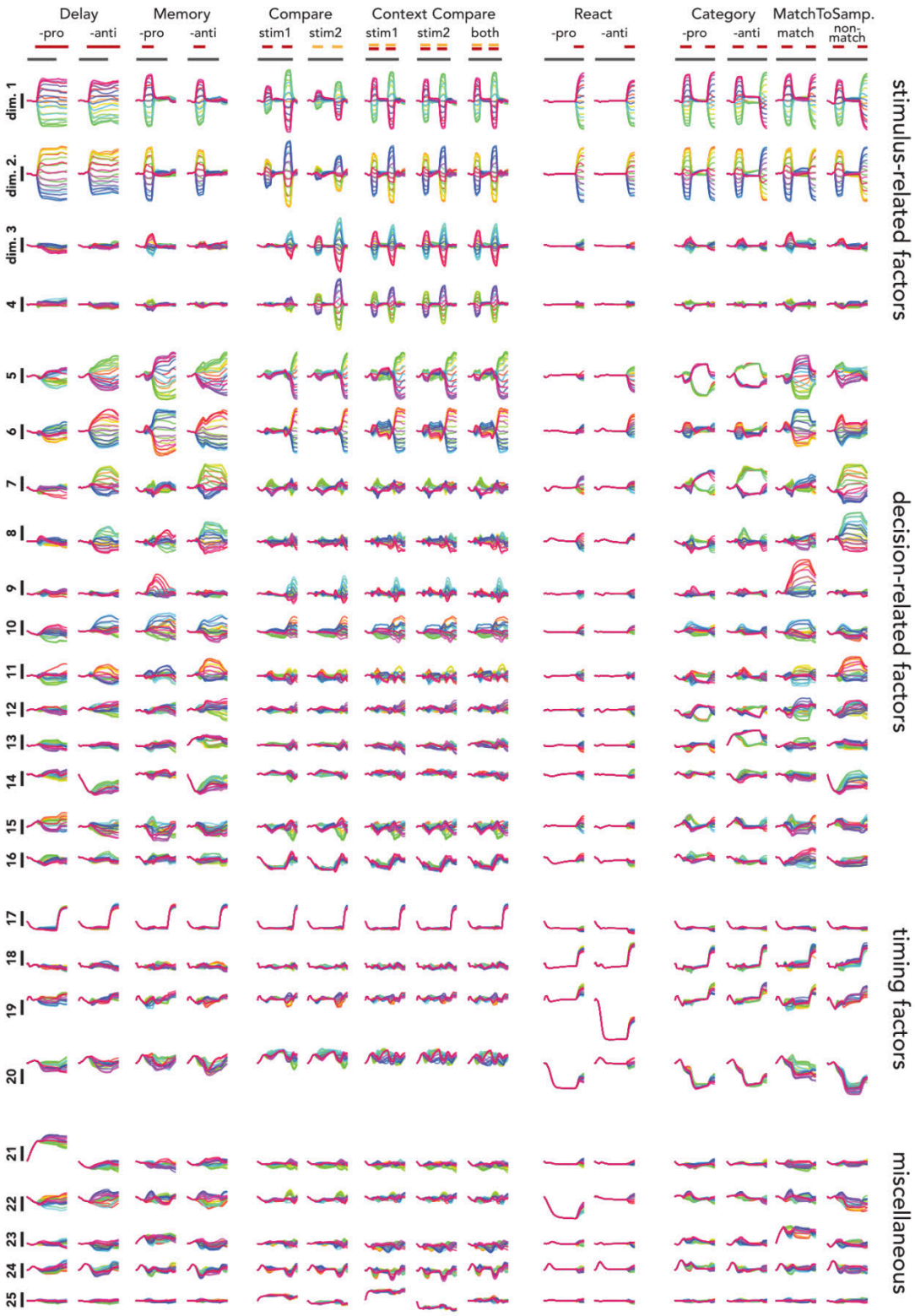

Delay
-pro
-anti
Memory
-pro
-anti
Compare
stim1
stim2
Context Compare
stim1
stim2
both
React
-pro
-anti
Category
-pro
-anti
MatchToSamp.
match
non-match
dim. 1
dim. 2.
dim. 3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
stimulus-related factors
decision-related factors
timing factors
miscellaneous

Figure S27. SCA factors identified from a multitask network. SCA was trained on the activity from all fifteen tasks (see *Methods* for a description of each task), and dimensions were manually sorted according to putative role. Traces were colored by stimulus angles. Dimensions 1-4 reflected the directional stimuli. Dimensions 1 and 2 were more closely related to input from stimuli 1 (e.g., CompareStim 1), while dimensions 3 and 4 more strongly reflected input from stimuli 2 (e.g., CompareStim 2). In contrast to dimensions 1-4, dimensions 5-16 reflected the network's ultimate output (i.e., decision). Interestingly, there was only partial overlap between the dimensions that reflected the network's decision during tasks that involved an external go cue (Delay-, Memory-, Compare-, and Context Compare-) and the dimensions that carried the network's decision during tasks without an explicit go cue (React-, Category-, and MatchToSample). Consider dimensions 5 and 6. As outlined in *Results*, these dimensions strongly reflect the network's decision during the first nine tasks (i.e., the across-condition variance in these dimensions after the go cue is delivered is substantial). During the Category- tasks, however, across-condition variance in these dimensions decreases after the go cue. However, during the Category- tasks, variance increases after the go cue in dimensions 7 and 8. The partial overlap of the 'decision dimensions' for different tasks likely relates to the fact that the network seemed to use different sets of factors to trigger output generation, depending on which task was being performed. During the first nine tasks (Delay-, Memory-, Compare-, and ContextCompare-) the fixation cue was removed at the end of each trial, acting as a go cue. During the final six tasks (React-, Category-, and MatchToSamp.-) the fixation cue was never removed; here, the timing of the network's output was determined by the directional stimuli. SCA recovered factors that reflected two different triggering mechanisms. As discussed in *Results*, the condition-invariant activity in dimension 17 resembles the 'trigger signal' observed in neural data[59]. Note that during the final six tasks, factor 17 was not active. However, a different factor (factor 18) was only active during tasks without a fixation cue. During these tasks, factor 18 also exhibited condition-invariant activity that was time-locked to the production of an output, suggesting that it was acting as a 'trigger signal' during tasks without a dedicated external go cue.

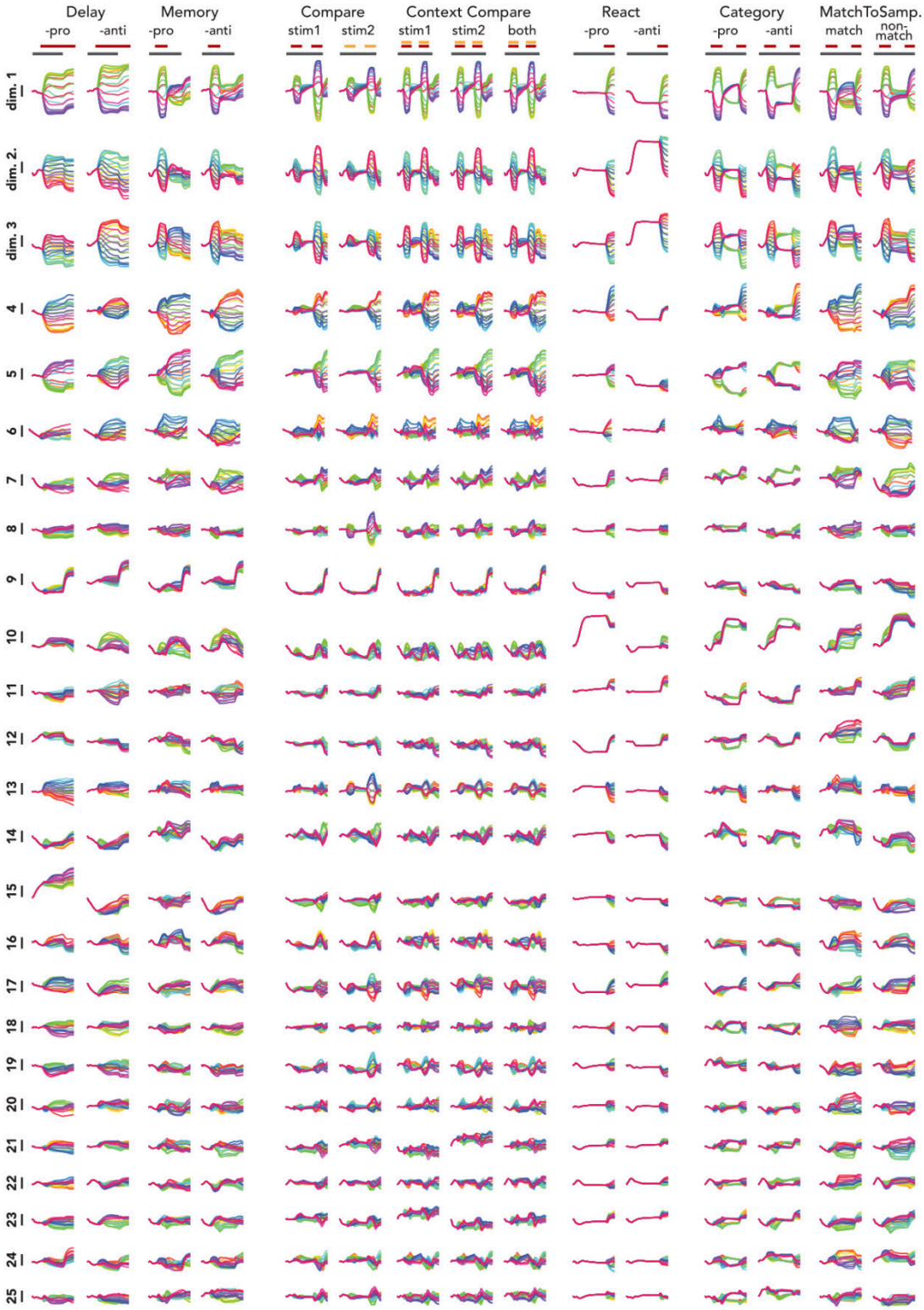

Delay
-pro
-anti
Memory
-pro
-anti
Compare
stim1
stim2
Context Compare
stim1
stim2
both
React
-pro
-anti
Category
-pro
-anti
MatchToSamp.
match
non-match
dim. 1
dim. 2.
dim. 3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

Figure S28. PCA factors identified from a multitask network. Factors identified via PCA (ordered by variance explained) do not offer the same clear view of the underlying computations. While some individual dimensions are similar to single SCA dimensions (e.g., dimensions 9 appears to be the 'trigger signal' observed in SCA dimension 17, Figure S27), PCA factors more often appear to be 'mixed' versions of SCA factors. For example, dimensions 1 and 2 appear to be a mix of stimulus and decision related activity.

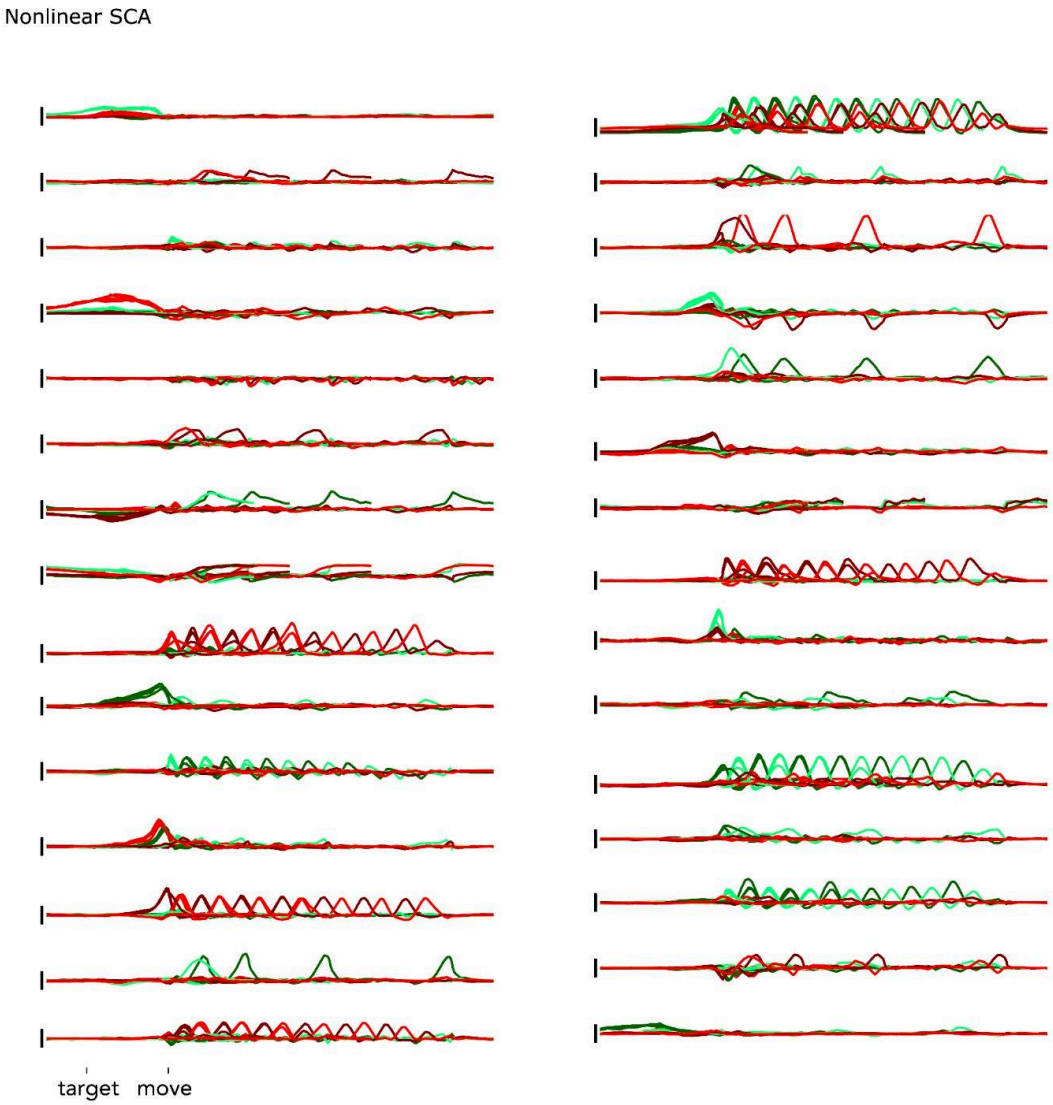


Figure S29. Nonlinear SCA recovers demixed factors on unimanual cycling data. All 40 latent factors are shown from the nonlinear SCA model fit that generated the examples shown in Fig. 8C. Dimensions are not ordered. Coloring of conditions is the same as in Fig. 8C. As in the linear SCA model fit, factors generally have separate putative roles related to steady-state movement, posture, preparation, and stopping.

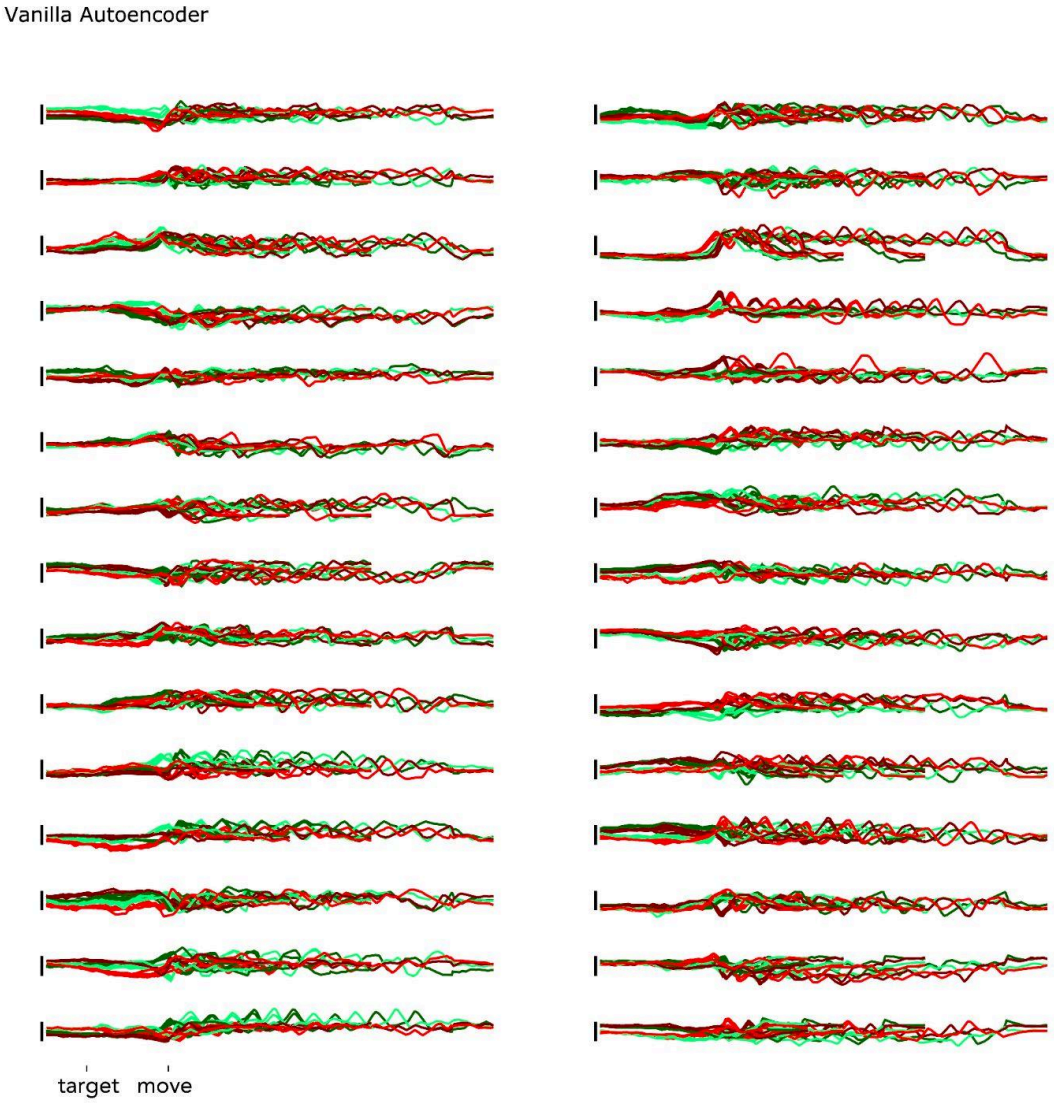


Figure S30. Vanilla autoencoders recover mixed factors on unimanual cycling data. All 40 latent factors are shown from the vanilla autoencoder fit that generated the examples shown in Fig. 8D. Dimensions are not ordered. Coloring of conditions is the same as in Fig. 8. Factors mix individual neural computations.

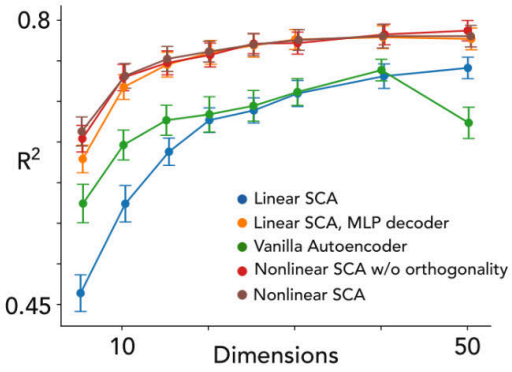


Figure S31. Quantitative comparison of linear and nonlinear approaches. Reconstruction accuracy on held-out data, as a function of the number of latent dimensions, using recordings from the unimanual cycling task. Results for nonlinear SCA, a vanilla autoencoder (which has no sparsity or orthonormality penalties), and linear SCA, are identical to Fig. 8B. Accuracy is also shown for fitting a linear SCA model, followed by fitting a nonlinear multi-layer perceptron from the learned factors to the neurons (orange; "Linear SCA, MLP decoder"). Interestingly, many of the nonlinear accuracy benefits are retained. Additionally, accuracy is shown for nonlinear SCA without any orthogonality penalty, which is close to a standard sparse autoencoder (red). Orthogonality has minimal impact on quantitative performance.

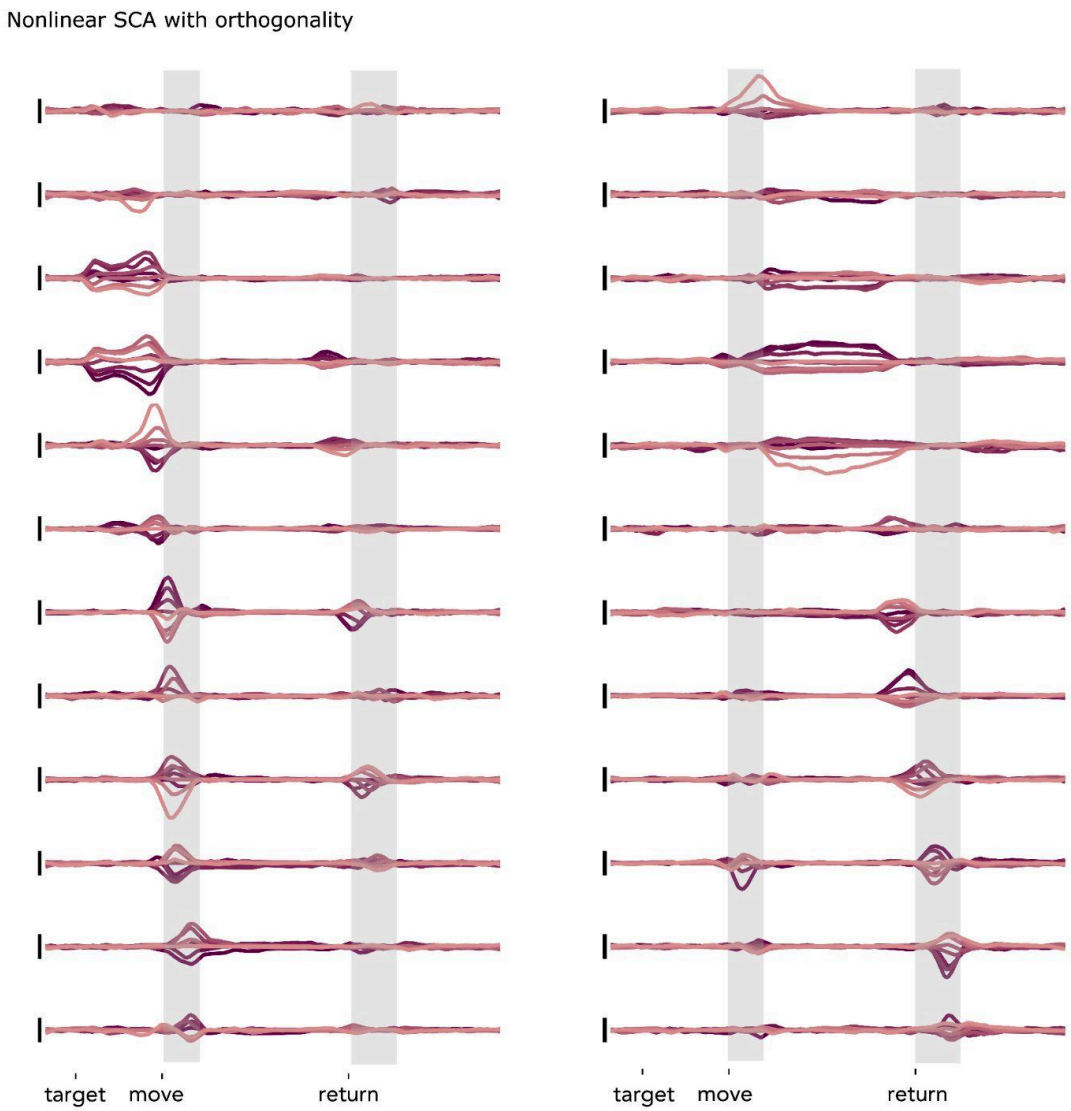


Figure S32. Nonlinear SCA provides demixed factors in reaching data. We fit nonlinear SCA to center-out reaching data (from Fig. 2). We show projections in twenty-four dimensions, ordered by the time point of the trial with maximum variance. These SCA dimensions share many of the same features as those plotted in Fig. 2, namely dimensions that are primarily occupied during preparatory, execution, or posture epochs.

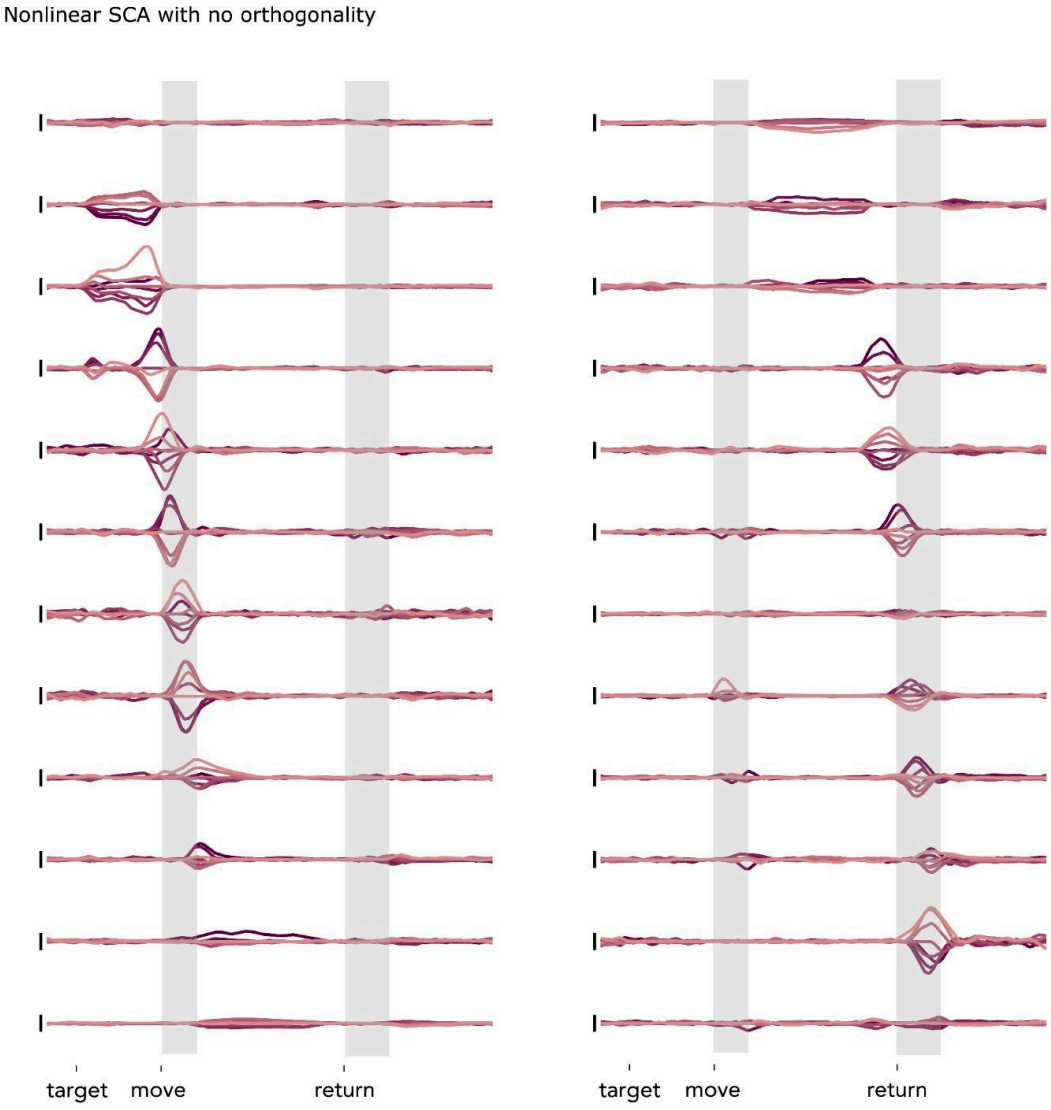
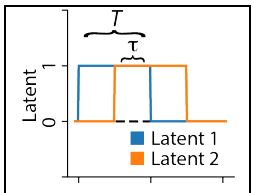


Figure S33. Orthogonality in nonlinear SCA is important for not overly sparsifying factors. We fit nonlinear SCA without orthogonality to center-out reaching data (from Fig. 2). We show projections in twenty-four dimensions, ordered by the time point of the trial with maximum variance. Relative to when using orthogonality (Fig. S32), most movement-related dimensions are only occupied prior to/during either outward or return reaches, rather than factors relating to both outward and return reaches. The orthogonality in Fig. S32 helped to prevent such an 'oversparsification,' providing an interpretation in line with our existing understanding.

Supplementary Text 1

Question: We here consider conditions under which encouraging sparsity via an L1 loss can automatically separate distinct neural processes into separate latents, using simplified 'ground truth' scenarios. Assume we have two true underlying latents (each corresponding to a neural process) that occur at times that are not completely overlapping. Let's say each is a square pulse of magnitude 1, which occurs for duration *T*, and the two pulses overlap for duration τ (Supp Text 1 Fig. 1). Our question is, in what scenarios will finding the sparsest latents, in terms of L1 loss, lead to finding the ground truth latents?



Supplementary Text 1 Fig 1. Setup.

Background: Assume we have *K* underlying latents: $\boldsymbol{Z_{true}} = [\boldsymbol{Z_1}, \ldots, \boldsymbol{Z_k}] \in \mathbb{R}^{TxK}$, and $\boldsymbol{X} = \boldsymbol{Z_{true}}\boldsymbol{V_{true}}$, where $\boldsymbol{X}$ is the neural activity and $\boldsymbol{V_{true}}$ is a loading matrix with rows of unit norm.

When inferring the latents, $\mathbf{Z}^*$, there are an infinite number of solutions with equally good reconstruction accuracy: $\boldsymbol{X} = \mathbf{Z}^*\boldsymbol{V}^* = \boldsymbol{Z_{true}}\boldsymbol{R}\boldsymbol{R^{-1}}\,\boldsymbol{V_{true}}$ where $\mathbf{Z}^* = \boldsymbol{Z_{true}}\boldsymbol{R}$ and $\boldsymbol{V}^* = \boldsymbol{R^{-1}}\,\boldsymbol{V_{true}}$.

In SCA, we constrain the rows of $\boldsymbol{V}^*$ to be unit norm, so that the L1 sparsity penalty on the latents, $|\mathbf{Z}^*|_1$, can't be arbitrarily decreased by simply increasing the magnitude of $\boldsymbol{V}^*$. We thus just consider $\boldsymbol{R}$ matrices that are rotation matrices (which will keep $\boldsymbol{V}^*$ to be unit norm). Still, there are infinite rotations of $\boldsymbol{Z_{true}}$ (and counter-rotations of $\boldsymbol{V_{true}}$), that will produce equal reconstructions of the data. In this framework, our question then becomes, for what underlying latents (i.e., for what $\boldsymbol{Z_{true}}$), does $\mathbf{Z}^* = \boldsymbol{Z_{true}}$ (as opposed to any other $\mathbf{Z}^* = \boldsymbol{Z_{true}}\boldsymbol{R}$), have the smallest $|\mathbf{Z}^*|_1$?

Answer: We'll start by considering the scenario when there is no temporal overlap between latent processes ($\tau = 0$). If we view the activity of these two latents in state space (as in Main text Fig. 1G), when only the first latent, $\boldsymbol{Z_1}$ is active, $\mathbf{Z}^*$ lies on the x-axis of a circle (Main text Fig. 1G, solid black line), and when only $\boldsymbol{Z_2}$ is active, $\mathbf{Z}^*$ lies on the y-axis of a circle (Main text Fig. 1G, solid red line). For any such time point of $\boldsymbol{Z_{true}}$ that lies on an axis, $|\boldsymbol{Z_{true}}|_{\boldsymbol{1}} = 1$. Any rotation (e.g. dashed lines in Main text Fig. 1G) will increase $|\mathbf{Z}^*|_1$ above 1: $|\mathbf{Z}^*_{\mathbf{1}}|_1 + |\mathbf{Z}^*_{\mathbf{2}}|_1 \geq 1$. This is because of the geometrical fact that the hypotenuse is always shorter than the sum of sides of a triangle.

Next, we consider the more general scenario when there is temporal overlap between latent processes ($\tau > 0$). At timepoints in $\boldsymbol{Z_{true}}$ where both $\boldsymbol{Z_1}$ and $\boldsymbol{Z_2}$ are active ('overlapping timepoints' both with values of 1), any rotation of $\mathbf{Z}^*$ away from $\boldsymbol{Z_{true}}$ will actually decrease $|\mathbf{Z}^*|_1$. For instance, consider rotating $[1,1]$, with $|\mathbf{Z}^*|_1 = 2$, with a -45 degree rotation to $[\sqrt{2}, 0]$ with $|\mathbf{Z}^*|_1 = \sqrt{2}$ (in fact, a 45 degree rotation will most decrease $|\mathbf{Z}^*|_1$ for overlapping points). Thus, depending how many overlapping vs non-overlapping timepoints there are, rotations may increase or decrease the total $|\mathbf{Z}^*|_1$. Analytically solving for the τ at which the decrease in $|\mathbf{Z}^*|_1$ from non-overlapping timepoints balances out the increase in $|\mathbf{Z}^*|_1$ from overlapping timepoints (deriving equations shown below), we get $\tau = (2 - \sqrt{2})T \approx 0.586T$. Further overlap above this value leads to the ground truth latents not being the sparsest.

Deriving equations: We calculate the maximum amount of overlap τ for which two latents of duration T will have the smallest L1 value when not rotated, versus when the latents are rotated 45 degrees, as 45 (or -45) degrees is the rotation that will lead to maximum sparsity for overlapping timepoints in this scenario (Main text Fig. 1J). To do so, we find when the L1 term

with a 45 degree rotation becomes smaller than the L1 term with no rotation (when the ground truth latents are the sparsest).

The L1 term with no rotation is $2T$. This is because, for T time points, for 2 latents, each has an L1 of 1.

The L1 term with a 45 degree rotation is $\sqrt{2}\tau + 2\sqrt{2}(T-\tau)$. Term 1 in this sum is because, for the τ overlapping timepoints, $[1,1]$ gets rotated to $[0,\sqrt{2}]$, with L1=$\sqrt{2}$. Term 2 in the sum is because, for the T-τ overlapping timepoints, for the 2 latents, $[1,0]$ and $[0,1]$ get rotated to $[\sqrt{2}/2,\sqrt{2}/2]$ and $[-\sqrt{2}/2,\sqrt{2}/2]$, respectively, each with an L1 of $\sqrt{2}$, resulting in a total L1 of $2\sqrt{2}(T-\tau)$.

Finding when the L1 term without rotation is smaller than with rotation:

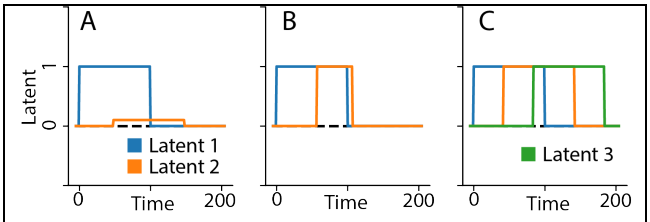
$$\sqrt{2}\tau + 2\sqrt{2}(T-\tau) \geq 2T$$

$$\dots$$

$$\tau \leq T(2-\sqrt{2}) \approx 0.586T$$

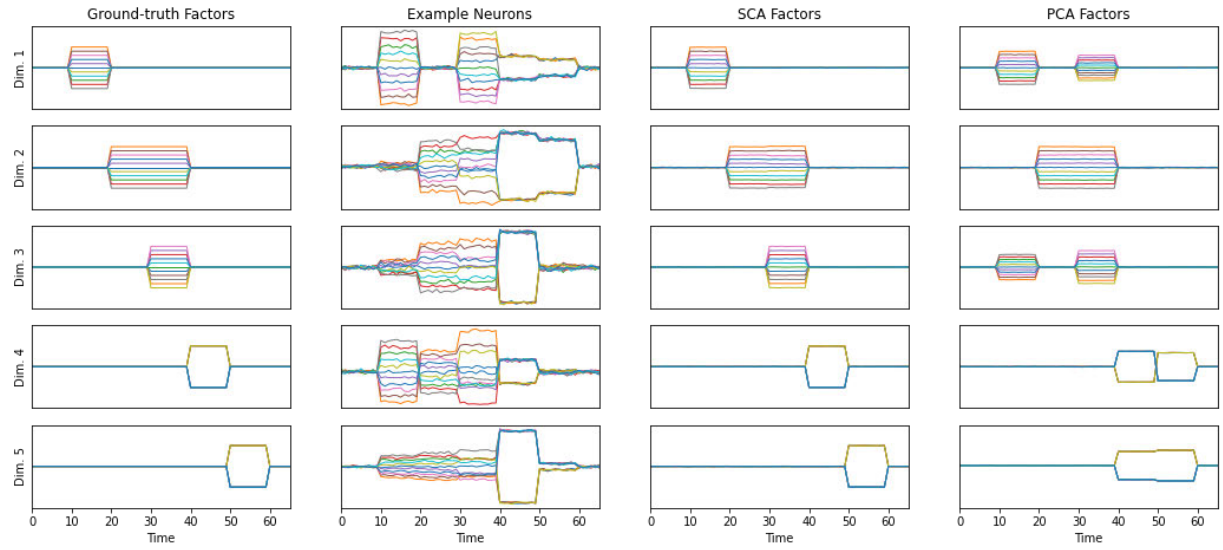
This analytical value is consistent with that empirically found in Main text Fig. 1J.

Generality: The general result that $\boldsymbol{Z_{true}}$ is commonly the sparsest solution is robust to when we change the magnitude of latents, duration of latents, or number of latents (Supplementary Text 1 Fig. 2), although the exact conditions on τ will slightly change.



Supplementary Text 1 Fig 2. Robustness. We show the maximal overlap that leads to the ground truth latents being the sparsest solution when **(A)** one latent has magnitude 0.1, **(B)** one latent has half the duration, and **(C)** there are 3 latents.

Further Example: Going beyond examples with primarily two latents, we provide one additional empirical example of SCA discovering ground truth latents that are non-overlapping or partially overlapping. Let's say there is an experiment in which two stimuli of differing magnitudes occur sequentially, and then the subject needs to report which has a greater magnitude after a go cue. There are latent factors (rows in Supplementary Text 1 Fig. 3) corresponding to neural processes related to 1) the first stimulus, 2) holding the first stimulus in memory until the 2nd stimulus is presented, 3) the second stimulus, 4) the decision based on the comparison, and 5) the report of the decision after the go cue. In this toy example, neurons (column 2) are mixtures of these latent factors (column 1). SCA (column 3) is able to effectively recover the ground truth, non-coactive processes.



Supplementary Text 1 Fig 3. Further empirical example. (Column 1) Ground-truth factors, that are either non-overlapping or partially overlapping in time. Each color represents a different condition. **(Column 2)** Example neurons generated as orthogonal mixtures of these factors. **(Column 3)** Learned SCA factors from the simulated neurons, which recover the ground truth factors. **(Column 4)** Learned PCA factors from the simulated neurons, which mix the underlying ground truth factors.