

Supplementary material for Infusing structural assumptions into dimensionality reduction for single-cell RNA sequencing data to identify small gene sets

Maren Hackenberg^{*,1,2}, Niklas Brunn^{*,1,2}, Tanja Vogel³, Harald Binder^{1,2,4}

¹ Institute of Medical Biometry and Statistics (IMBI),
Faculty of Medicine and Medical Center, University of Freiburg, Freiburg, Germany

² Freiburg Center for Data Analysis, Modeling and AI,
University of Freiburg, Freiburg, Germany

³ Institute of Anatomy and Cell Biology, Department Molecular Embryology,
Faculty of Medicine, University of Freiburg, Freiburg, Germany

⁴ Centre for Integrative Biological Signaling Studies (CIBSS),
University of Freiburg, Freiburg, Germany

Correspondence to maren.hackenberg@uniklinik-freiburg.de
and to niklas.brunn@uniklinik-freiburg.de

*These authors contributed equally.

Contents

Supplementary Note 1: Summary of the genes selected per latent dimension on cortical mouse data	4
Supplementary Note 2: BAE mappings of held out cortical mouse test data	4
Supplementary Note 3: Comparison of the BAE with supervised componentwise boosting on cortical mouse data	4
Supplementary Note 4: Gene selection stability analysis on cortical mouse data	5
Supplementary Note 5: Robustness analysis of structuring latent representations on cortical mouse data	5
Supplementary Note 6: Model comparison for gene selection on cortical mouse data	5
Supplementary Note 7: timeBAE gene selection comparison	6

Supplementary Note 8: Simulation of stair-like scRNA-seq data	6
Supplementary Note 9: Simulation study to illustrate how the BAE identifies distinct cell groups and corresponding explanatory genes	7
Supplementary Note 10: Model comparison of different autoencoder versions on simulated stair-like scRNA-seq data	7
Supplementary Fig. 1: Complementary UMAP plots colored by BAE latent dimensions on cortical mouse data	10
Supplementary Fig. 2: Cell type correspondence of BAE latent dimensions on cortical mouse data	11
Supplementary Fig. 3: Dynamics of BAE encoder weights across training epochs on cortical mouse data	12
Supplementary Fig. 4: BAE mappings of held out cortical mouse test data	13
Supplementary Fig. 5: UMAP plots of cortical mouse data colored by comp L_2 Boost predictions	14
Supplementary Fig. 6: Distribution of BAE gene selections on cortical mouse data	15
Supplementary Fig. 7: BAE robustness analysis of structuring latent representations on cortical mouse data	16
Supplementary Fig. 8: Colored UMAP embeddings of PCA representation of cortical mouse data	17
Supplementary Fig. 9: Comparison of different timeBAE encoder initialization strategies	18
Supplementary Fig. 10/11: Complementary UMAP plots for timeBAE analysis of embryoid body data	19
Supplementary Fig. 12: BAE application on simulated stair-like scRNA-seq data	21
Supplementary Fig. 13: Different BAE decoder parameter initializations on simulated stair-like scRNA-seq data	22
Supplementary Fig. 14: BAE results on simulated stair-like scRNA-seq data during different training epochs	23
Supplementary Fig. 15: BAE results without the disentanglement constraint on simulated stair-like scRNA-seq data during different training epochs	24
Supplementary Fig. 16: AE results on simulated stair-like scRNA-seq data during different training epochs	25
Supplementary Fig. 17: L_1 AE results on simulated stair-like scRNA-seq data during different training epochs	26

Supplementary Fig. 18: corAE results on simulated stair-like scRNA-seq data during different training epochs	27
Supplementary Fig. 19: VAE results on simulated stair-like scRNA-seq data during different training epochs	28
Supplementary Table 1: Summary of the genes selected per latent dimension on cortical mouse data	29
Supplementary Table 2: Change-point genes per latent dimension on cortical mouse data	29
Supplementary Table 3: Most frequently selected genes on cortical mouse data	30
Supplementary Table 4: Systematic replacement of informative genes with noise genes in cortical mouse data	31
Supplementary Table 5: Gene selection comparison of the BAE with sparsePCA and scPNMF	31
Supplementary Table 6: timeBAE gene selection comparison	32
Supplementary Table 7: Overlap patterns in simulated stair-like scRNA-seq data	32
Supplementary Algorithm 1: comp L_2 Boost algorithm	33
Supplementary Algorithm 2: Core BAE optimization algorithm	33
Supplementary Algorithm 3: Disentanglement constraint	34
Supplementary References	35

Supplementary Note 1: Summary of the genes selected per latent dimension on cortical mouse data

During training and starting from a zero initialization, the BAE selects weights in the encoder weight matrix to be updated. Nonzero weights represent a connection between genes and latent dimensions. After training the encoder weight matrix of a BAE is sparse, i.e. there are only a few connections from genes to each latent dimension. In the analysis of the cortical mouse data, the encoder weight matrix consisted of 10 latent dimensions and 1500 variables, i.e., genes. Out of the 15000 weights, 495 weights were nonzero (3.3%). The following tables provide information about the distribution of nonzero weights across the latent dimensions (Supplementary Table 1) and the top selected genes, i.e., change-point genes per latent dimension (Supplementary Table 2). Furthermore, the changes of the BAE encoder weight values can be tracked across training epochs. The dynamics of the encoder weights (coefficients) are presented in Supplementary Fig. 3.

Supplementary Note 2: BAE mappings of held out cortical mouse test data

We mapped the held-out test data (200 randomly selected cells from the 1,525 cortical mouse cells) to the latent space using the trained encoder of the BAE. To assess whether the test cells were accurately mapped to regions corresponding to their cell types in the training data, we predicted the test cell types by identifying the 10 closest cells in the latent space from the training data (based on Euclidean distance) and assigning the most frequent cell type among them (see Methods Section Label predictions of unseen data in the main text). This approach yielded a prediction accuracy of 94% when compared to the true labels of the test data. Supplementary Fig. 4 illustrates two-dimensional UMAP embeddings of the BAE training and test data representations.

Supplementary Note 3: Comparison of the BAE with supervised componentwise boosting on cortical mouse data

We compared the learned representation of the BAE with that of the supervised comp L_2 Boost algorithm, which is at the core of the optimization of the BAE. For applying the comp L_2 Boost algorithm to the cortical mouse data, we use the cell type annotations from [1] and created a one-hot encoded version of the cell labels that were stored in a matrix where the rows correspond to the cells and the columns to the different cell types. Then, for each cell type in the data we applied comp L_2 Boost for respectively 100 boosting steps with a step size of 0.02 using the standardized training data and the corresponding column of the one-hot matrix.

Supplementary Fig. 5 shows a scatter plot of a two-dimensional UMAP embedding of the neurons using the comp L_2 Boost representation. In each plot, the embedding is colored according to the model prediction for one specific cell type, i.e., where the corresponding column in the one-hot encoded cell type matrix was used as the responses.

Supplementary Note 4: Gene selection stability analysis on cortical mouse data

We analyzed the stability of gene selection in the BAE model using the cortical mouse data, as detailed in Methods Section Gene selection stability analysis in the main text. The results are presented in a bar plot (Supplementary Fig. 6). Additionally, we examined genes selected in 80% or more of the 30 runs (Supplementary Table 3). Across the 30 training runs, the BAE encoder weight matrices had an average of 96.25% zero elements, with a 95% confidence interval of [96.21%, 96.29%], rounded to two decimal places.

Supplementary Note 5: Robustness analysis of structuring latent representations on cortical mouse data

Details of the robustness analysis can be found in Methods Section Robustness analysis of structuring latent representations in the main text, while a summary of the structured gene replacement procedure is provided in Supplementary Table 4.

The learned structure of the BAE latent representation was assessed by visually inspecting 2D UMAP projections of cell representations to gain insights into the latent patterns (Supplementary Fig. 7). At a gene replacement rate of 2.87% (removal of the top 43 genes identified in the gene selection stability analysis, Supplementary Table 3), a significant portion of neural markers was already missing. At 13.53%, the BAE struggled to separate cells from different cortical layers into distinct clusters while still preserving the structure of GABAergic neurons. At 61.07%, layer-specific separation further deteriorated, though GABAergic neuron structure was roughly maintained. At the final replacement rate of 97.13%, the model barely distinguished GABAergic from glutamatergic neurons, yet interestingly, *Chodl*-expressing neurons (a subset of *Sst*-expressing neurons) often formed distinct clusters, even in the absence of the *Chodl* marker.

Overall, the BAE demonstrated a robust preservation of the core neural cell structure, even when a substantial fraction of informative genes was replaced with noise.

Supplementary Note 6: Model comparison for gene selection on cortical mouse data

The BAE was compared to sparsePCA [2] and scPNMF [3] in terms of gene selection, specifically the sparsity of the loading matrices and the BAE encoder weight matrix. Consistent with the gene selection stability analysis (see Supplementary Note 4), we trained the BAE 30 times under identical training conditions but with different initializations of the decoder parameters. Similarly, sparsePCA and scPNMF were run 30 times with varying seed parameters. The results, including sparsity levels (measured as the number of zero elements in the BAE encoder weight matrices and loading matrices), are summarized in Supplementary Table 5. For a detailed explanation of the analysis, refer to Methods Section Model comparison for gene selection in the main text.

Under the given conditions, the BAE produced encoder weight matrices with the highest sparsity level, achieving a mean percentage of zeros at 96.25%. On average, 448.57 genes were selected per run, representing 29.90% of the 1,500 genes. In contrast, sparsePCA and scPNMF exhibited (close to) zero variability across individual runs. The loading matrices for sparsePCA and

scPNMF had an average of approximately 41% and 45% zero elements, respectively, resulting in significantly more nonzero elements compared to the BAE encoder weight matrices. Consequently, almost all genes were connected to at least one component by a nonzero element in the loading matrices, with sparsePCA leaving only one gene unconnected. The average number of selected genes per run was 1,499 for sparsePCA and 1,500 for scPNMF.

Note, that the sparsity level of the different methods is controlled by model-specific hyperparameters for sparsePCA and scPNMF [2, 3]. For comparison, we used the default hyperparameter settings for each approach. Additionally, factors such as the number of training epochs or the step size of the boosting component for optimizing the encoder parameters can influence the sparsity level of the resulting BAE encoder weight matrix, as discussed in detail in Methods Section Componentwise likelihood-based boosting and Methods Section Core optimization algorithm of the BAE in the main text.

Supplementary Note 7: timeBAE gene selection comparison

In a setting, where both time point and cluster annotations of cells are available, the timeBAE results can be compared to a linear regression analysis. The time point and cluster annotations can be used to determine the most significant genes and compare them to the top genes selected by the timeBAE. The supervised linear modeling approach and comparison process are described in detail in Methods Section Gene selection comparison for the timeBAE in the main text. The results of the gene selection comparison are shown in Supplementary Table 6.

Supplementary Note 8: Simulation of stair-like scRNA-seq data

For demonstrating the BAE functionality in a controlled setting, we simulate a binary count matrix modeling 10 developmental stages using the simulation design from [4]. Specifically, a cell taking the value 1 for some gene indicates a gene expression level exceeding a certain threshold for that cell, whereas the value 0 indicates an expression level below that threshold. For $n = 1000$ cells and $p = 50$ genes, the data is modeled by i.i.d. random variables $(X_{i,j} \sim \text{Ber}(p_{i,j}))_{i=1,\dots,n, j=1,\dots,p}$, where $p_{i,j}$ denotes the probability of a value of 1 for a gene j in a cell i . In each developmental stage, the binarized expression levels of 100 cells are generated, where 5 genes are simulated as highly expressed, hence are characterizing for that cell stage, whereas the remaining genes are lowly expressed in that stage. Therefore, $p_{i,j}$ is set to 0.6 for highly expressed genes (HEGs) and to 0.1 for lowly expressed genes (LEGs). To model a smooth transition between two successive stages, HEGs of successive stages overlap, i.e., genes 1-5 characterize the first stage, genes 4-8 characterize the second stage, genes 7-11 the third stage, and so on, resulting in the stair-like structure shown in Supplementary Fig. 12, Panel b. Details about the overlap patterns can be found in Supplementary Table 7. In total there are 32 HEGs and 18 lowly expressed genes across all cell stages which are added as non-informative genes to make the variable selection task more challenging and realistic.

Supplementary Note 9: Simulation study to illustrate how the BAE identifies distinct cell groups and corresponding explanatory genes

To illustrate how the BAE works in a simple controlled scenario, we use the simulation design described in Supplementary Note 8. Specifically, our aim is to show that the BAE is capable of capturing distinct subgroups of cells in different dimensions together with characteristic sets of genes. The data was standardized before model training and non-informative genes were re-scaled by a scaling factor of $\frac{2}{3}$.

We train a BAE with a 2-layer decoder and 10 latent dimensions. For evaluating the ability in extracting meaningful genes related to some groups of cells, we inspect the encoder weight matrix of the BAE during the training, since each nonzero weight represents a link between a gene and one latent dimension. The results are shown in Supplementary Fig. 12. After the first training epoch, the BAE has selected one gene for each latent dimension, which serves as a guidance for further gene selection steps in the following epochs (Supplementary Fig. 12a). During the next epochs, the model then learns to link additional genes to the distinct latent dimensions, characterizing the same group of cells as the already selected genes. When training proceeds for a larger number of epochs, at some point, the BAE starts updating additional weights corresponding to still underrepresented patterns. Stopping the training after e.g. 15 training epochs results in a sparse encoder weight matrix capturing the main characteristic patterns of the data (Supplementary Fig. 12d). As overlap genes occur more frequently in the data than the genes exclusively abundant in one cell type, the model tends to preferentially select overlap gene pairs and each latent dimension captures the two cell types associated via the overlap genes (Supplementary Fig. 12c).

Training longer, also noise genes are selected, but with small weights (Supplementary Fig. 12a). Supplementary Fig. 12e shows the absolute values of Pearson correlation coefficients between the 10 latent dimensions of the BAE after training for 15 epochs. All off-diagonal entries are small, confirming that latent dimensions of the BAE capture distinct factors of variation. Since the BAE focused on patterns driven by overlap genes, two neighboring latent dimensions have a slightly higher correlation.

One main source of uncertainty in learning the sparse encoder weight matrix are the randomly sampled decoder parameters. Before training the model, the decoder parameters are randomly sampled from a probability distribution, e.g. a common strategy is to sample the parameters according to the normalized initialization strategy [5], where the decoder weights are sampled from a uniform distribution, while the biases are set to zero. We investigated stability of the BAE latent representation with respect to different initializations of the decoder parameters across several training runs. We found the encoder weight matrices to display very similar patterns and sparsity levels after 15 training epochs (see Supplementary Fig. 13) and almost no non-informative genes are selected.

Supplementary Note 10: Model comparison of different autoencoder versions on simulated stair-like scRNA-seq data

To demonstrate the effectiveness of the different components in our modeling approach for identifying a sparse set of explanatory genes and latent dimensions that encode distinct groups of cells, we compared our BAE approach to 1) a BAE without the disentanglement constraint, i.e., with unconstrained negative gradients as responses; 2) a standard unregularized autoencoder;

3) an autoencoder with an additional L_1 penalty term to enforce sparsity; 4) an autoencoder with an additional correlation penalty term to enforce uncorrelated latent dimensions; 5) a variational autoencoder (VAE) [6]. Detailed results are shown in Supplementary Fig. 14-19).

Specifically, in our comparison we consider a vanilla autoencoder (AE) using the mean squared error as the reconstruction loss

$$L_{\text{rec}}(\mathbf{x}, \widehat{\mathbf{x}}) := \frac{1}{p} \sum_{j=1}^p (x_j - \widehat{x}_j)^2,$$

$$L_{\text{AE}}(\mathbf{X}, \widehat{\mathbf{X}}) := \frac{1}{n} \sum_{i=1}^n L_{\text{rec}}(\mathbf{x}_i, \widehat{\mathbf{x}}_i),$$

an AE where we added a weighted L_1 -penalty term for constraining the encoder parameters ($L_1\text{AE}$), such that the training objective which is to minimize becomes

$$L_{L_1\text{AE}}(\mathbf{X}, \widehat{\mathbf{X}}) := L_{\text{AE}}(\mathbf{X}, \widehat{\mathbf{X}}) + \alpha_{L_1\text{AE}} \cdot \sum_{j=1}^p \sum_{l=1}^d |\beta_{j,l}|,$$

and an AE where we added a weighted correlation penalty term utilizing the sample Pearson correlation coefficient $\widehat{\text{cor}}(\mathbf{x}, \mathbf{y})$ that penalizes the correlations between latent dimensions, such that the training objective which is to minimize becomes

$$L_{\text{corAE}}(\mathbf{X}, \widehat{\mathbf{X}}) := L_{\text{AE}}(\mathbf{X}, \widehat{\mathbf{X}}) + \alpha_{\text{corAE}} \cdot \sum_{k=1}^d \sum_{l=1}^d \widehat{\text{cor}}(f_{\text{enc}}(\mathbf{X}; \boldsymbol{\psi})_k, f_{\text{enc}}(\mathbf{X}; \boldsymbol{\psi})_l)^2,$$

where $\boldsymbol{\psi}$ denote the encoder parameters of the AE model.

In addition, we consider a VAE with a multivariate standard normal prior distribution for the latent space. The decoder learns the means of a multivariate normal distribution with a diagonal covariance matrix and fixed equal variance parameters σ_{dec}^2 . The loss function for training the VAE is

$$\hat{\mu}_{ij}, \hat{\sigma}_{ij}^2 = f_{\text{enc}}(\mathbf{x}_i; \boldsymbol{\psi})_j,$$

$$L_{\text{VAE}}(\mathbf{X}, \widehat{\mathbf{X}}) := \frac{1}{2\sigma_{\text{dec}}^2} \sum_{i=1}^n \sum_{j=1}^p (x_{ij} - \widehat{x}_{ij})^2 - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^p (1 + \log \hat{\sigma}_{ij}^2 - \hat{\mu}_{ij}^2 - \hat{\sigma}_{ij}^2),$$

where $\boldsymbol{\psi}$ denote the encoder parameters of the VAE model and the second term on the right side of the equation corresponds to the Kullback–Leibler divergence.

For a fair comparison, we used the same architecture and training hyperparameters for the different (V)AE versions and for the BAE, except that the (V)AE versions are allowed to have a trainable bias vector in their single-layer encoders but no activation function. For training the different AE versions, we are using the Adam optimizer [7] and set the learning rate for the optimization to 0.01. For training the VAE we use the AdamW optimizer [8], which incorporates weight decay regularization for the VAE encoder and decoder parameters. The weights for the different penalty terms were set to $\alpha_{L_1\text{AE}} = 0.1$, $\alpha_{L_1\text{AE}} = 0.075$ and $\alpha_{\text{VAE}} = 0.1$. The weight values were selected by empirically investigating the quality of the latent representation during the optimization process.

While there are no further restrictions made for the encoder of the AE, the L_1 -penalty term in the $L_1\text{AE}$ shrinks most of the weights towards zero during the optimization process, resulting in a

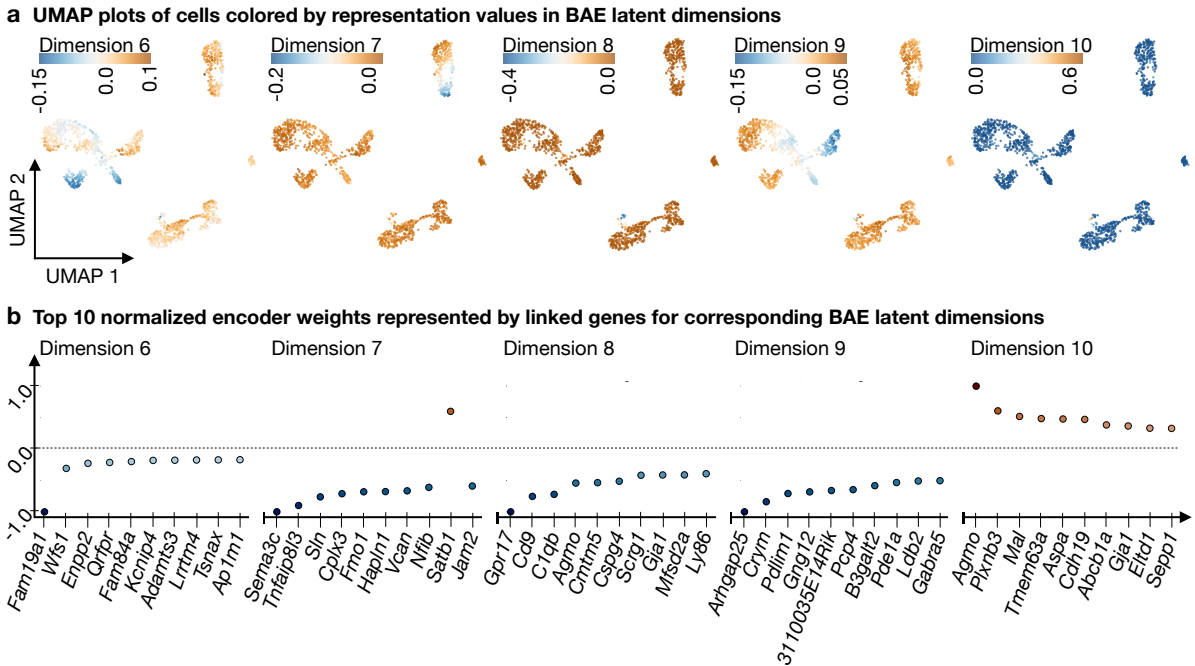
sparse encoder weight matrix consisting of many values close to, but not exactly zero. Empirically, however, that happened only after many training epochs, such that the L_1 AE is prone to overfitting, as opposed to the BAE, where starting from a zero initialization, meaningful variables are selected already during the first training epochs so that early stopping can be applied. The correlation penalty term incorporated in the loss function of the corAE enforces the model to learn a latent representation with uncorrelated latent dimensions.

Briefly, the BAE is the only model that identifies the cell group-characterizing genes already in the very first training epochs while maintaining uncorrelated latent dimensions that correspond to distinct groups of cells. The encoder weight matrix is sparse (Supplementary Fig. 14, 15), due to the stagewise selection of the boosting component in the encoder, which starts from a zero-initialized weight matrix. In contrast, all the other models start training from a nonzero random initialization of the encoder weight matrix. As a result, a sparse (and hence interpretable) encoder weight matrix that represents meaningful patterns is learned much later in the training process or not at all (Supplementary Fig. 16, 17, 18). The comparison with a standard BAE shows that the additional constraint on the selection criterion is essential for capturing distinct groups of cells in different latent dimensions and ensuring uncorrelated latent dimensions (Supplementary Fig. 15).

Of the other models, only the corAE provides uncorrelated latent dimensions, which is not surprising given that its training objective explicitly penalizes correlation between latent dimensions. Yet, it does not learn a clear structure for the latent representation (Supplementary Fig. 18). Similarly, the standard autoencoder does not learn structured latent representations (Supplementary Fig. 16). While the L_1 AE is able to find some cell stage characterizing patterns, which can be seen in the heatmaps of the encoder weight matrix and the latent representation after training for 1000 epochs (Supplementary Fig. 17), most of the encoder weights are not truly zero, compromising interpretability. For the VAE with weight decay regularization, the encoder weight matrix is dense in every stage of the optimization process. However, the model seems to be more robust in capturing patterns during later training epochs (1000 and 25000 epochs), but the latent dimensions tend to be more correlated compared to the other models (Supplementary Fig. 19).

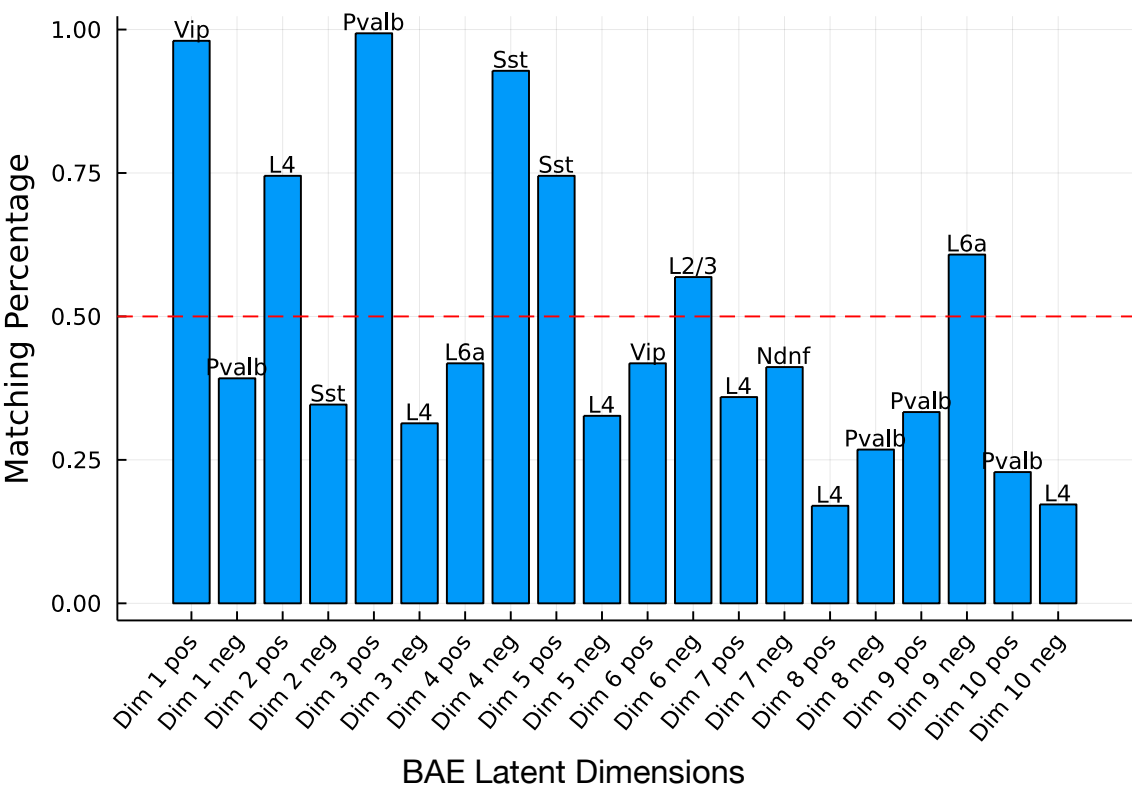
In summary, while the correlation and L_1 penalties are effective in the respective separate models, the BAE achieves both characteristics at the same time without an additional penalty term. The uncorrelated latent dimensions are enforced only via modification of the selection criterion, which can be flexibly tailored to encode also other constraints, such as time structure.

Supplementary Fig. 1: Complementary UMAP plots colored by BAE latent dimensions on cortical mouse data



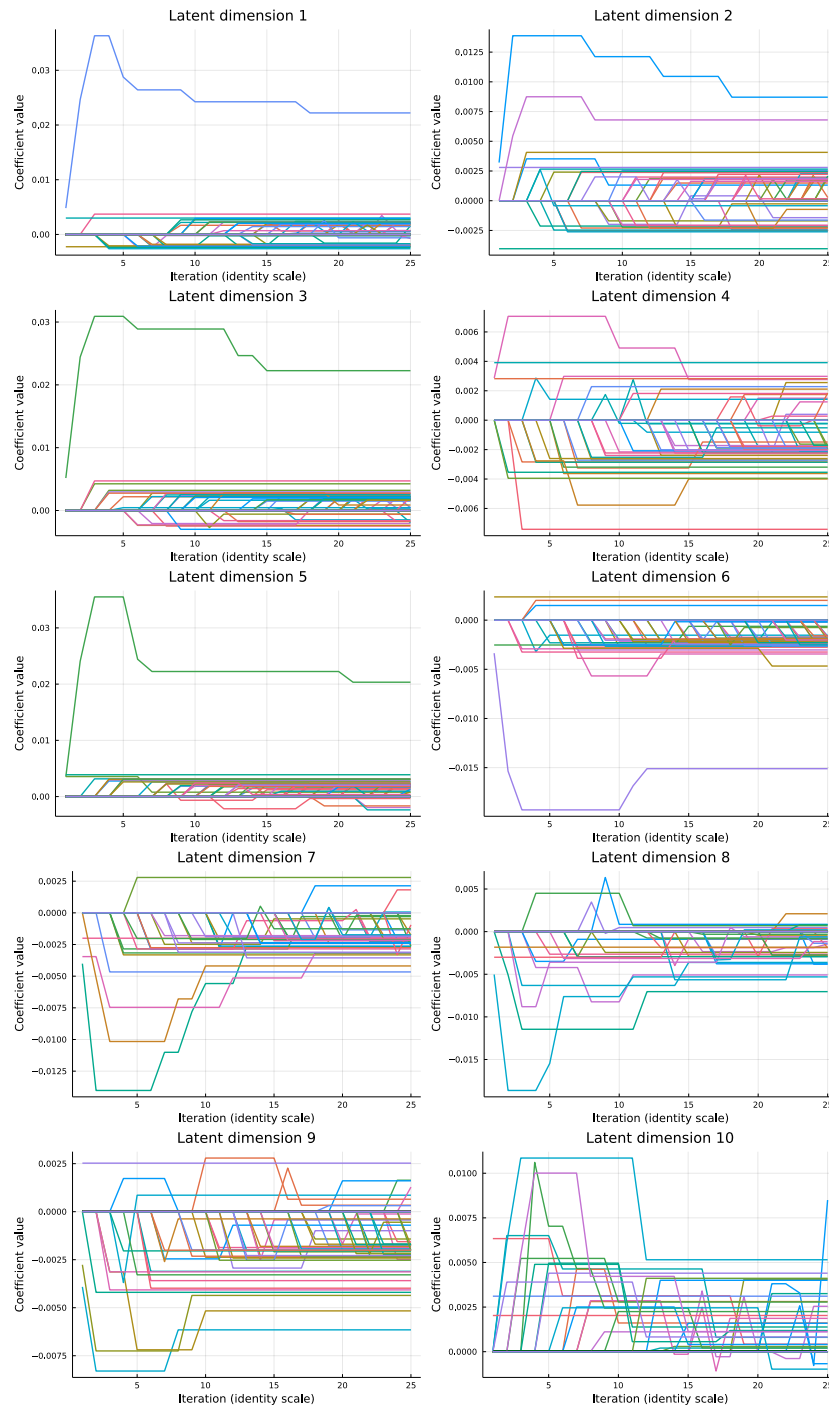
Supplementary Fig. 1: **Complementary UMAP plots colored by BAE latent dimensions on cortical mouse data.** **a** Scatter plots of a two-dimensional UMAP embedding computed using a ten-dimensional BAE latent representation of the cortical neurons. Cells are colored by their representation values in individual BAE latent dimensions (the last five dimensions are shown). **b** Scatter plots show the top ten normalized nonzero BAE encoder weights (represented by linked genes) in decreasing order of absolute value (from left to right) per latent dimension (the last five dimensions are shown). Normalization was performed by dividing each weight by the maximum absolute weight value for the corresponding latent dimension.

Supplementary Fig. 2: Cell type correspondence of BAE latent dimensions on cortical mouse data



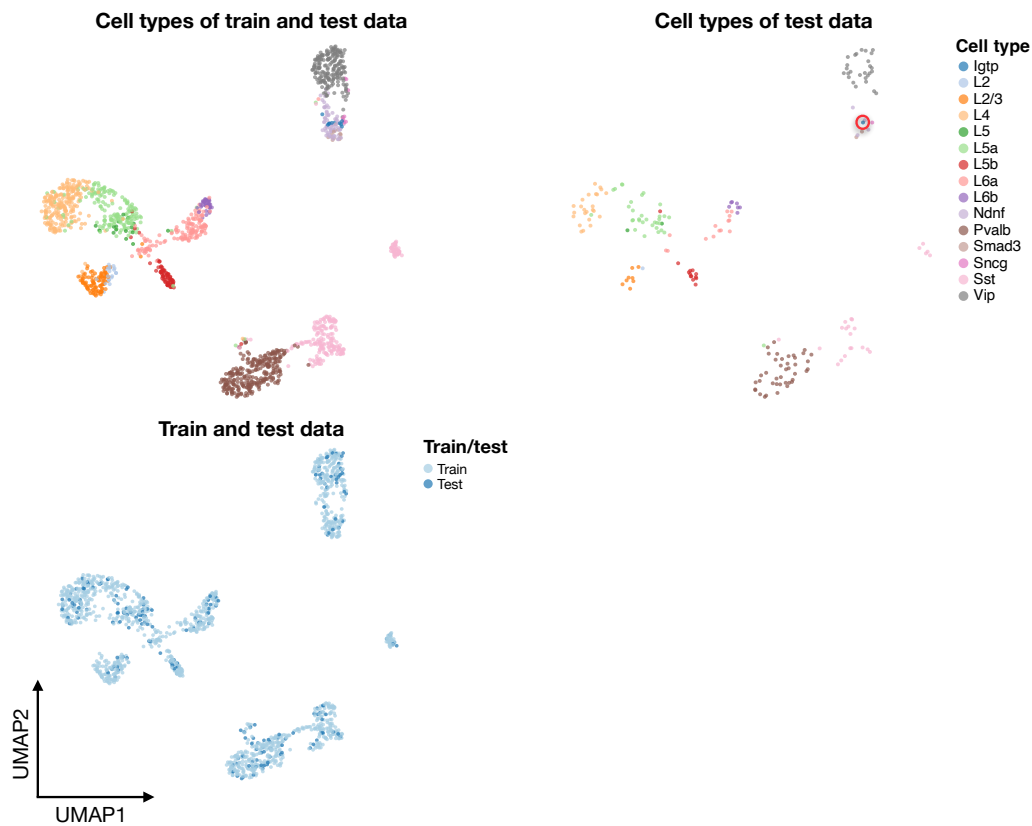
Supplementary Fig. 2: **Cell type correspondence of BAE latent dimensions on cortical mouse data.** The bar plot shows the percentage of cells belonging to the best matching cell type per latent dimension (see Methods Section Determining correspondence of latent patterns to cell types in the main text). Since cells can have either a positive or a negative representation value in each latent dimension, the positive and negative patterns are compared with the cell type patterns per latent dimension. For example, the most positive cell representations in latent dimension 1 strongly correspond to cells of the type Vip. The red horizontal line indicates the 50% line.

Supplementary Fig. 3: Dynamics of BAE encoder weights across training epochs on cortical mouse data



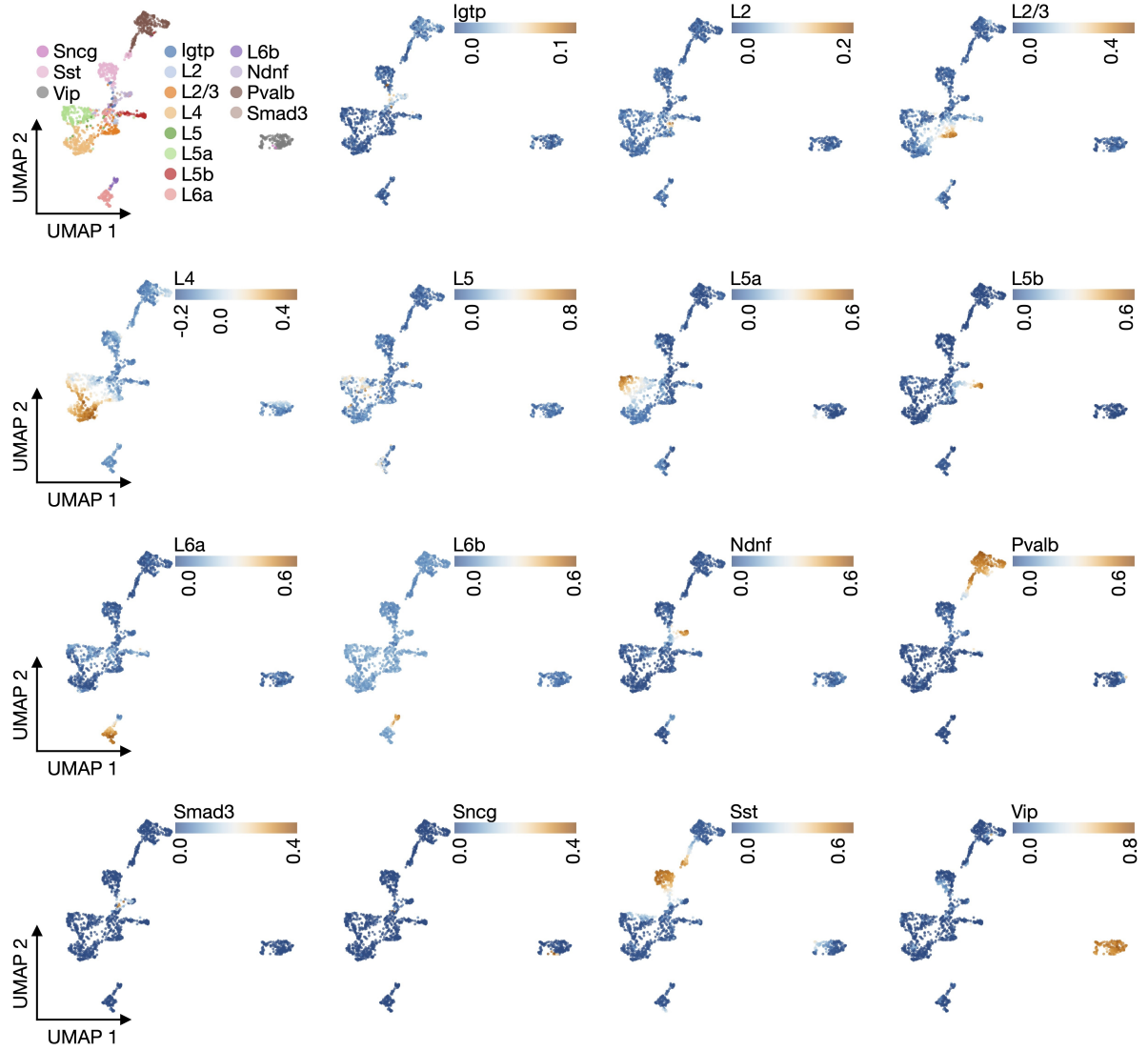
Supplementary Fig. 3: **Dynamics of BAE encoder weights across training epochs.** The plots show the evolution of individual BAE encoder weights (coefficients) across training epochs per latent dimension. Each line represents a specific weight corresponding to a gene.

Supplementary Fig. 4: BAE mappings of held out cortical mouse test data



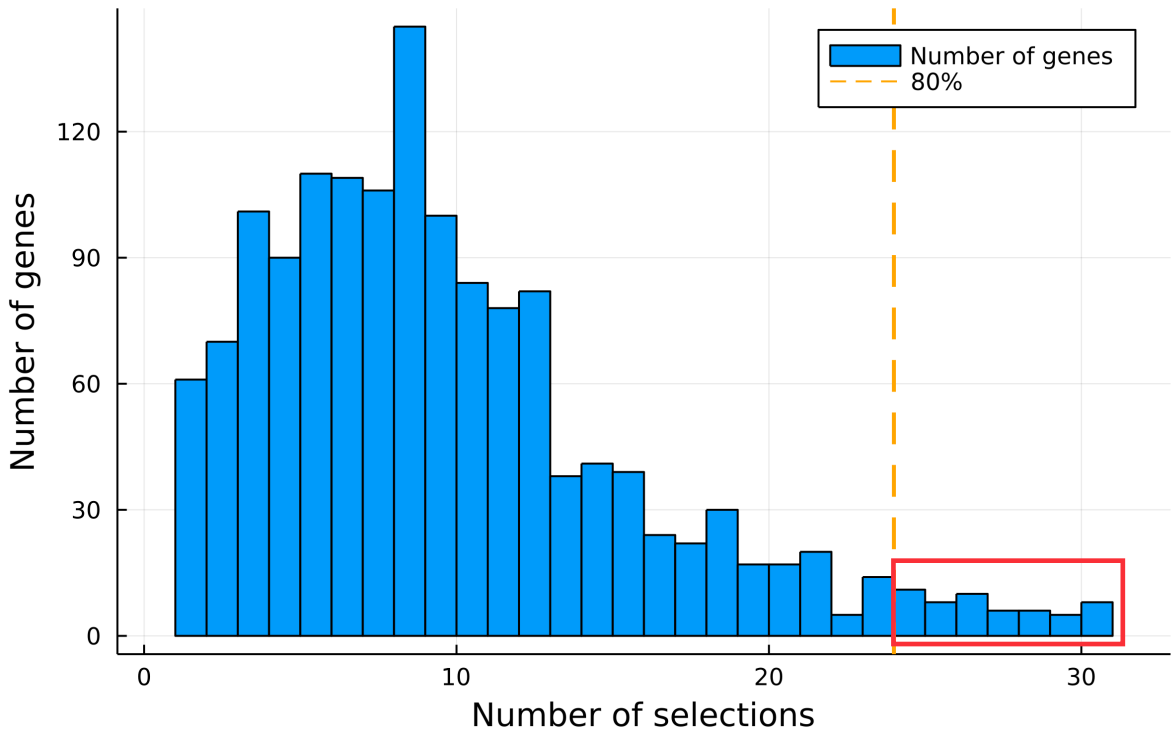
Supplementary Fig. 4: **BAE mappings of held out cortical mouse test data.** **Top left:** Scatter plot of a two-dimensional UMAP embedding of the BAE latent representation of the neurons colored by cell types (train and test data). **Top right:** Scatter plot of a two-dimensional UMAP embedding of the BAE latent representation of the neurons in the test data colored by cell types. An Igtp cell was added to the test data to ensure a consistent color coding. This cell is highlighted with a red circle. **Bottom left:** Scatter plot of a two-dimensional UMAP embedding of the BAE latent representation of the neurons in the test data, colored by training and test IDs.

Supplementary Fig. 5: UMAP plots of cortical mouse data colored by $\text{comp}L_2\text{Boost}$ predictions



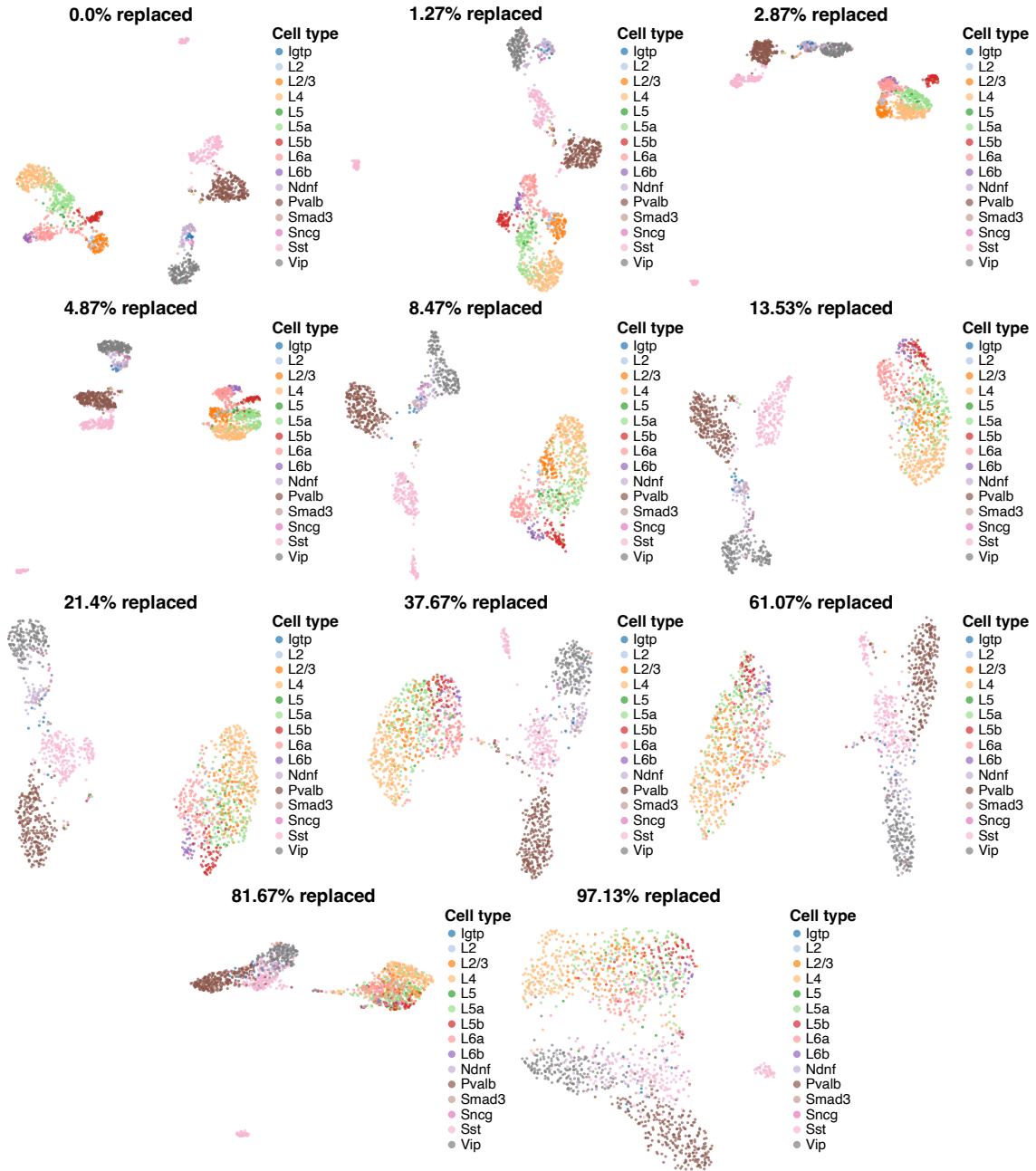
Supplementary Fig. 5: **UMAP plots of cortical mouse data colored by $\text{comp}L_2\text{Boost}$ predictions.** **Top left:** Scatter plot of a two-dimensional UMAP embedding of the neurons colored by cell types. The UMAP embedding was computed using the concatenated predictions of a componentwise boosting model applied sequentially to each column of a one-hot encoding of the cell labels. **Other plots:** Scatter plots show the same UMAP embedding colored by the predictions of the componentwise boosting model for each cell type. The plots are labeled by the cell type used to obtain the predictions, respectively.

Supplementary Fig. 6: Distribution of BAE gene selections on cortical mouse data



Supplementary Fig. 6: **Distribution of BAE gene selections on cortical mouse data.** Bar plot represents the number of individual genes (y-axis) with respect to the number of selections across different BAE training runs with different decoder parameter initializations (x-axis). The total number for which a gene could be selected was 30. The orange dashed vertical line reflects the 80% threshold of the number of selections and the red box marks the number of selections that lie above this threshold, reflecting genes that were selected in at least 24 of the 30 runs (see Supplementary Table 3).

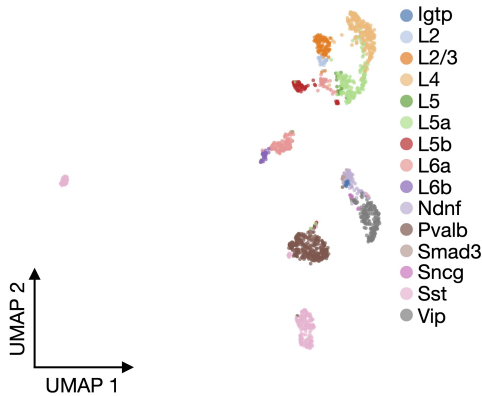
Supplementary Fig. 7: BAE robustness analysis of structuring latent representations on cortical mouse data



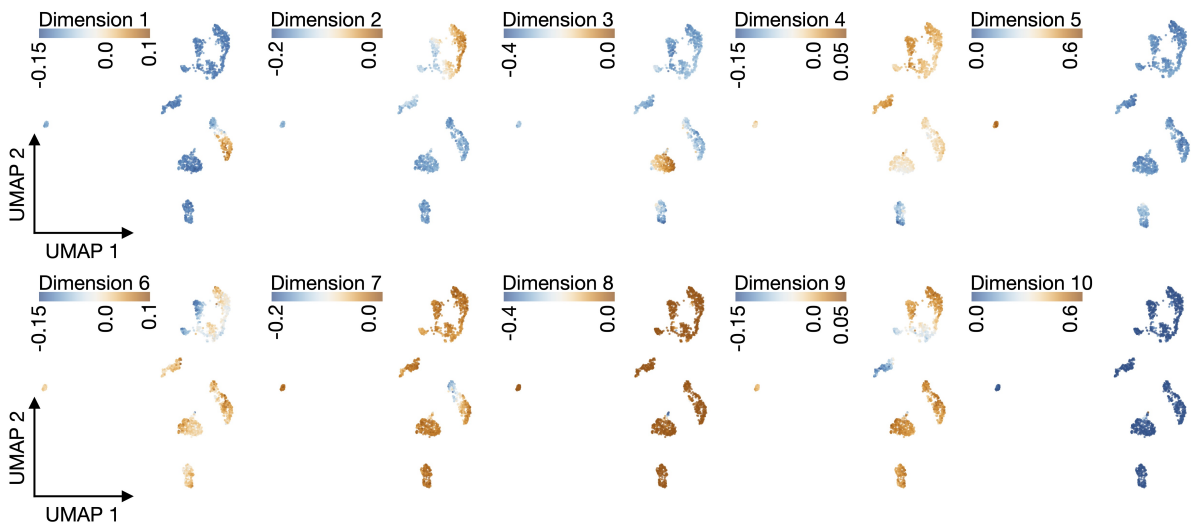
Supplementary Fig. 7: **BAE robustness analysis of structuring latent representations.** Scatter plots of 2D UMAP embeddings of BAE latent representations for the 1525 neural cells from the cortical mouse data are shown at varying levels of informative gene replacement with noise genes. The original dataset consists of 1500 highly variable genes (for details, see Methods Section Data preprocessing in the main text). All datasets consistently included 1,500 genes.

Supplementary Fig. 8: Colored UMAP embeddings of PCA representation of cortical mouse data

a UMAP plot of the first 10 PCs colored by cell types

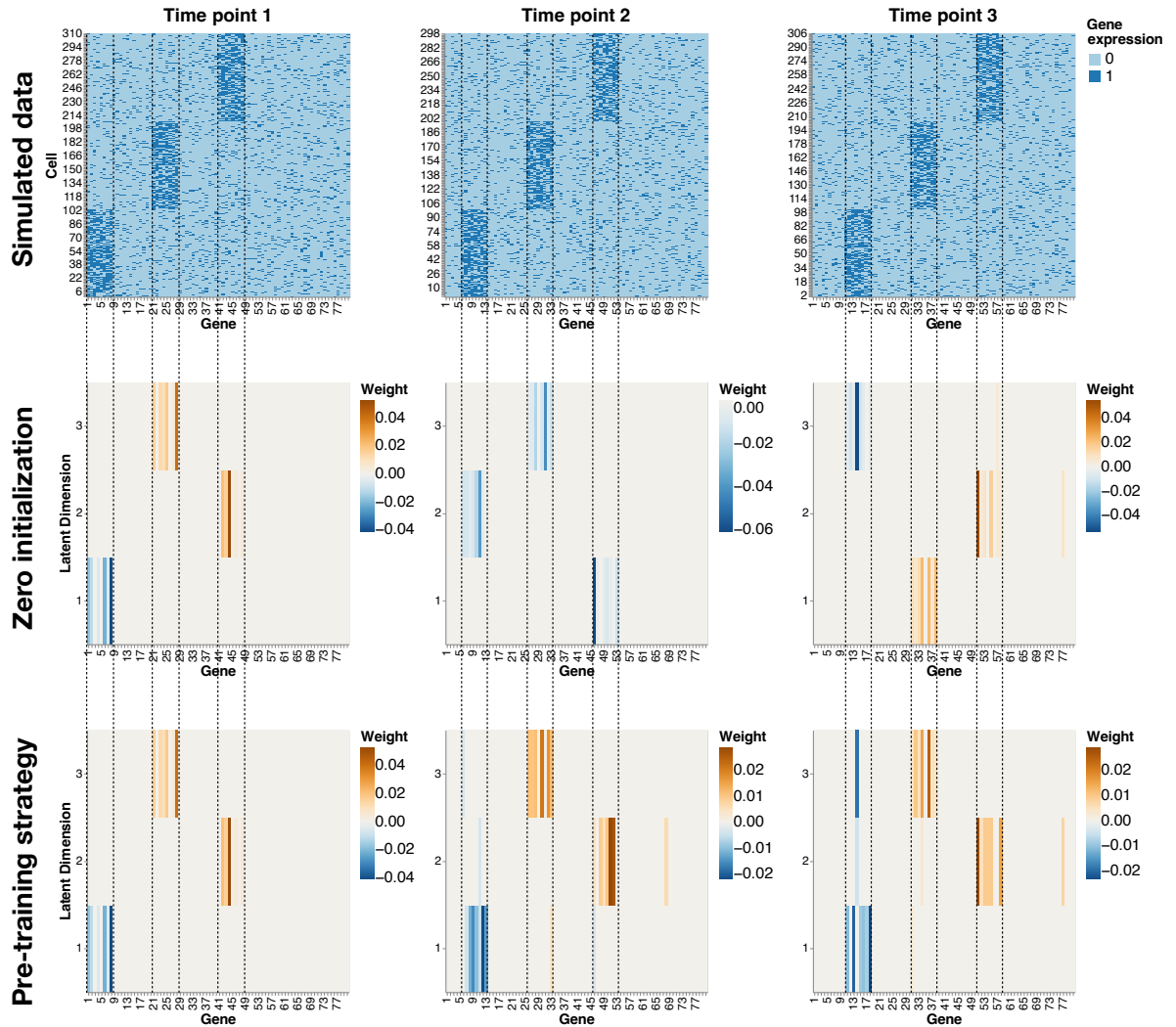


b UMAP plots of the first 10 PCs colored by representation values in BAE latent dimensions



Supplementary Fig. 8: **Colored UMAP embeddings of PCA representation of cortical mouse data.** **a** Scatter plot of a two-dimensional UMAP embedding of the first ten PCs of the neurons colored by cell types. **b** Scatter plots of the same UMAP embedding colored by the learned representation values of cells in the ten latent dimensions of the BAE.

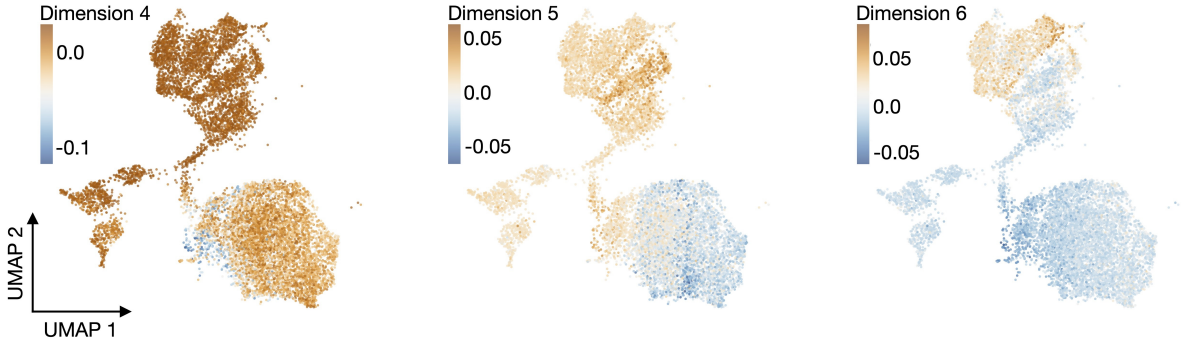
Supplementary Fig. 9: Comparison of different timeBAE encoder initialization strategies



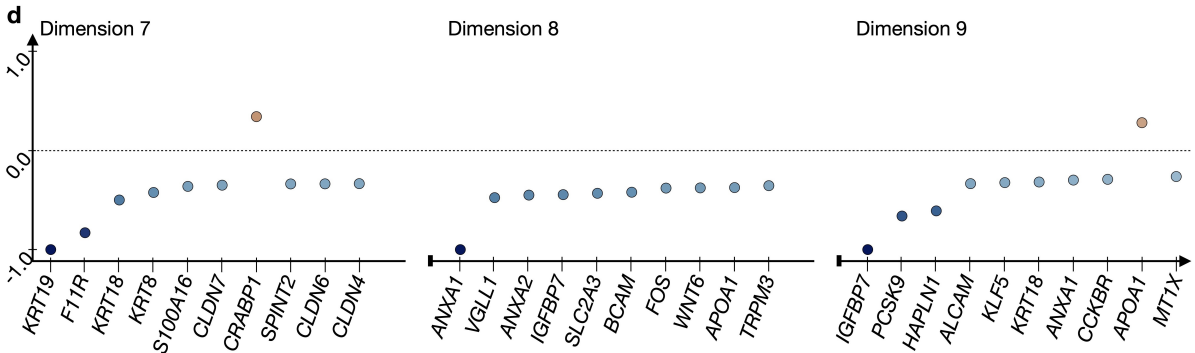
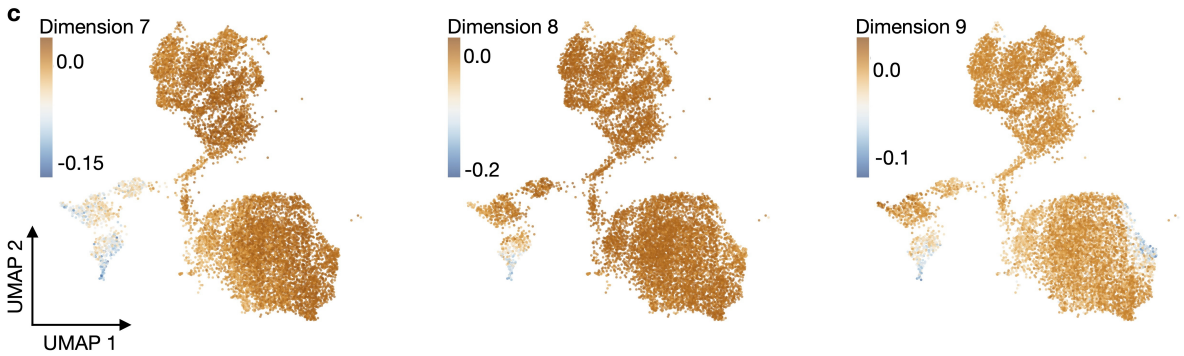
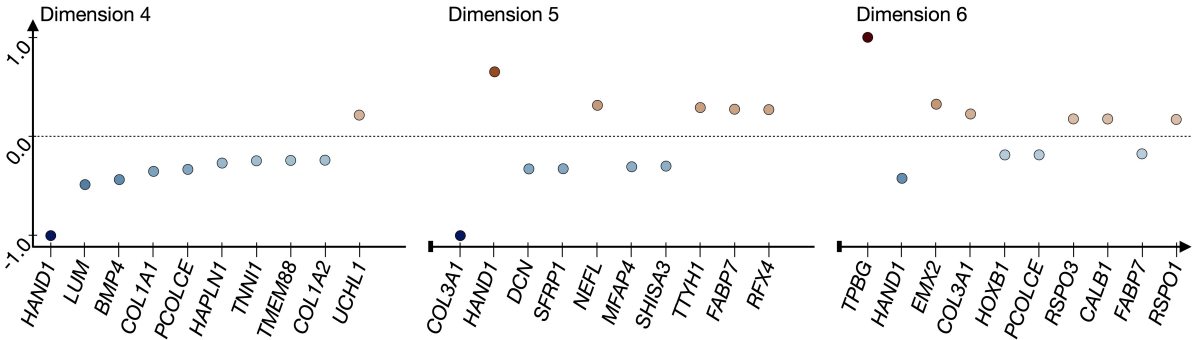
Supplementary Fig. 9: **Comparison of different timeBAE encoder initialization strategies.** **Top row:** Simulated binary count data representing three different cell groups evolving over three different time points. **Middle row:** BAE encoder weight matrices learned at each time point if all encoder weights were set to zero at the beginning of training for each BAE model. Although cell group patterns are captured in disentangled latent dimensions at each time point, no linkage of latent dimensions across time is achieved. **Bottom row:** The timeBAE encoder weight matrices are learned at each time point when the pre-training strategy is used. In this case, the model is able to capture cell group patterns in disentangled latent dimensions, while also linking corresponding latent dimensions across time.

Supplementary Fig. 10/11: Complementary UMAP plots for timeBAE analysis of embryoid body data

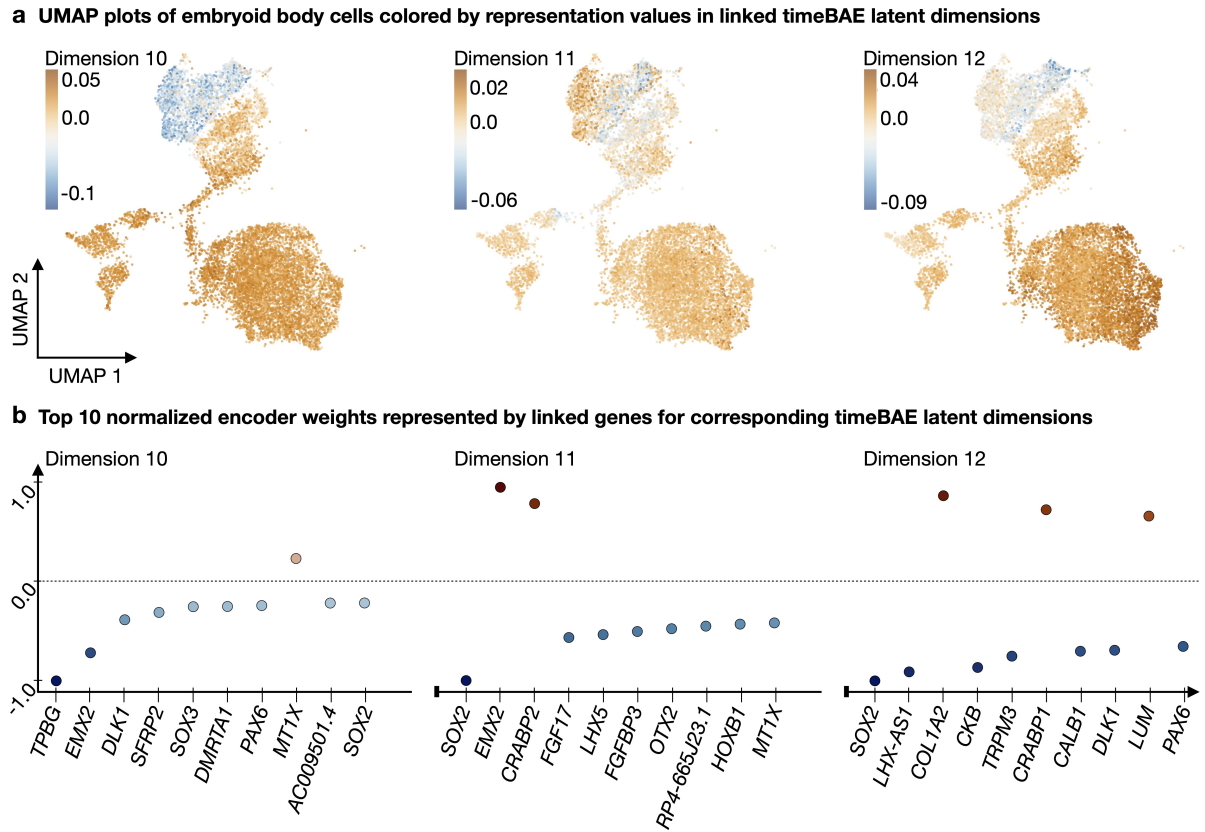
a UMAP plots of embryoid body cells colored by representation values in linked timeBAE latent dimensions



b Top 10 normalized encoder weights represented by linked genes for corresponding timeBAE latent dimensions

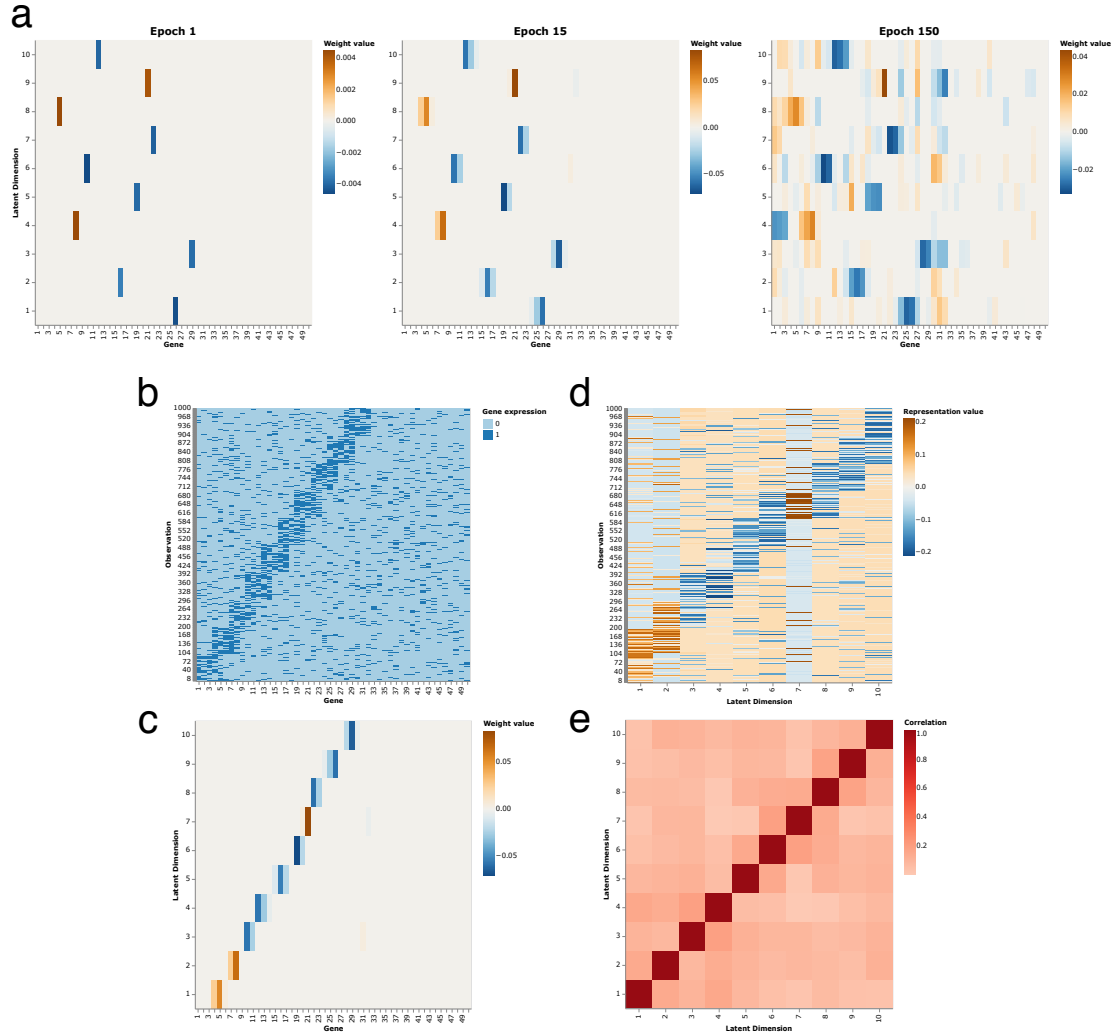


Supplementary Fig. 10: **Additional latent dimensions (4-9) for the timeBAE approach on embryoid body data.** **a** Scatter plots of a 2D UMAP embedding of the PCA representation of the trimmed embryoid body data. Plots are colored by the representation values of cells in linked timeBAE latent dimensions across time (dimensions 4-6 are shown). **b** Scatter plots show the top ten normalized nonzero timeBAE encoder weights (represented by linked genes) in decreasing order of absolute value (from left to right) per latent dimension (dimensions 4-6 are shown). Normalization was performed by dividing each weight by the maximum absolute weight value for the corresponding latent dimension. The first dimension has only seven corresponding nonzero weights. **c** Colored scatter plots of the UMAP embedding (dimensions 7-9 are shown). **d** Scatter plots of normalized coefficients (dimensions 7-9 are shown)



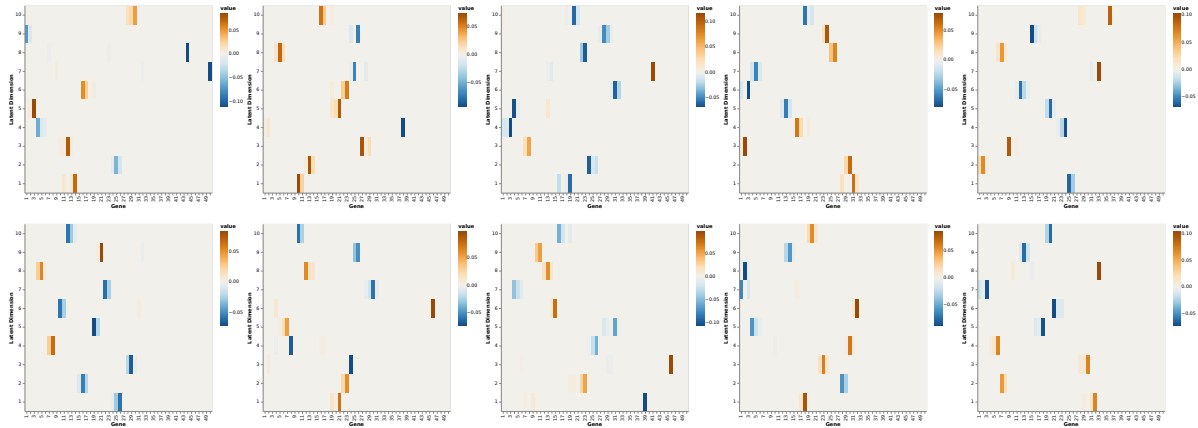
Supplementary Fig. 11: **Additional latent dimensions (10-12) for the timeBAE approach on embryoid body data.** **a** Scatter plots of a 2D UMAP embedding of the PCA representation of the trimmed embryoid body data. Plots are colored by the representation values of cells in linked timeBAE latent dimensions across time (dimensions 10-12 are shown). **b** Scatter plots show the top ten normalized nonzero timeBAE encoder weights (represented by linked genes) in decreasing order of absolute value (from left to right) per latent dimension (dimensions 10-12 are shown). Normalization was performed by dividing each weight by the maximum absolute weight value for the corresponding latent dimension. The first dimension has only seven corresponding nonzero weights.

Supplementary Fig. 12: BAE application on simulated stair-like scRNA-seq data



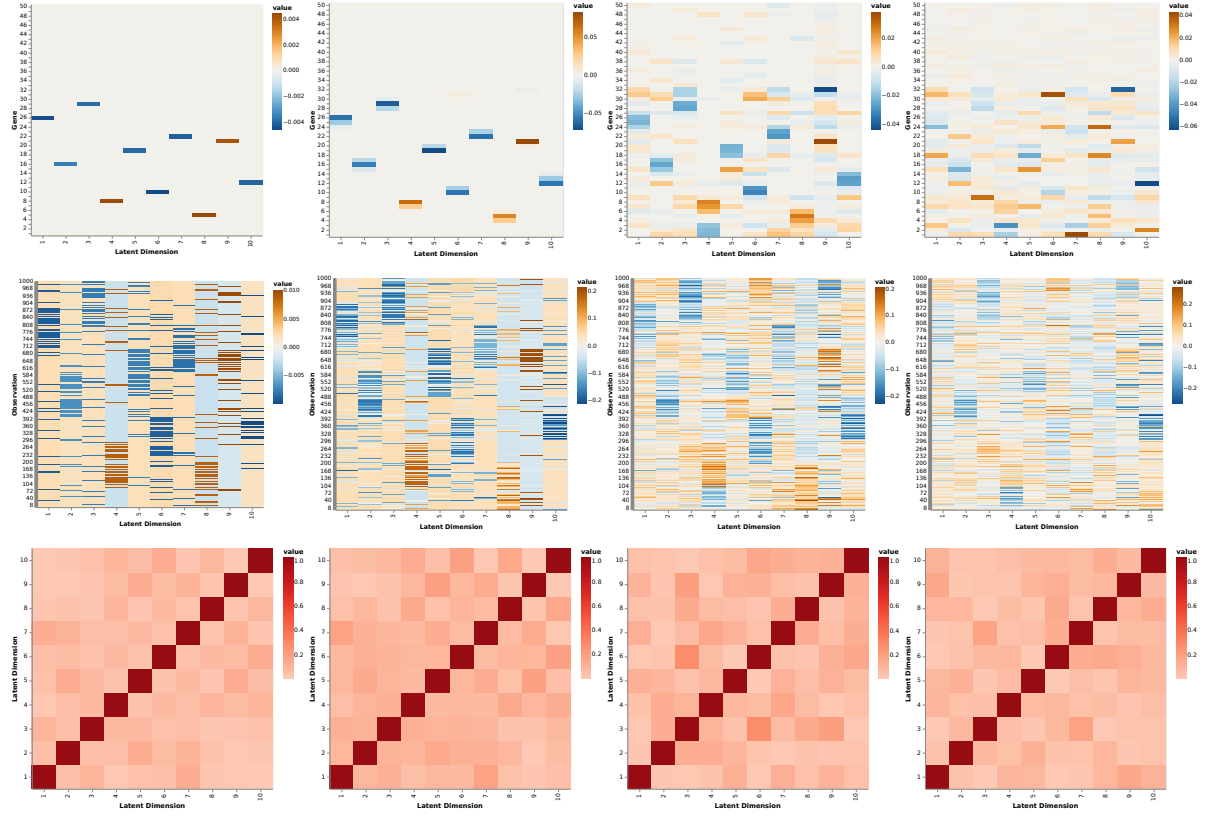
Supplementary Fig. 12: **BAE application on simulated stair-like scRNA-seq data.** **a** Heatmaps display the evolution of the BAE encoder weight matrix starting from a zero-initialization after training for 1, 15, 150 epochs (from left to right). **b** Heatmap of the simulated binary count matrix consisting of groups of cells in 10 different developmental stages. **c** We manually rearrange the latent dimensions of the encoder weight matrix after 15 epochs, so that learned patterns match the patterns in the count matrix for clarity. The heatmap displays the latent representation produced by the trained BAE after 15 epochs. **d** Heatmap shows the permuted encoder weight matrix of the BAE after training for 15 epochs. **e** Heatmap illustrates the absolute values of the sample Pearson correlation coefficients between the permuted latent dimensions of the BAE after 15 training epochs.

Supplementary Fig. 13: Different BAE decoder parameter initializations on simulated stair-like scRNA-seq data



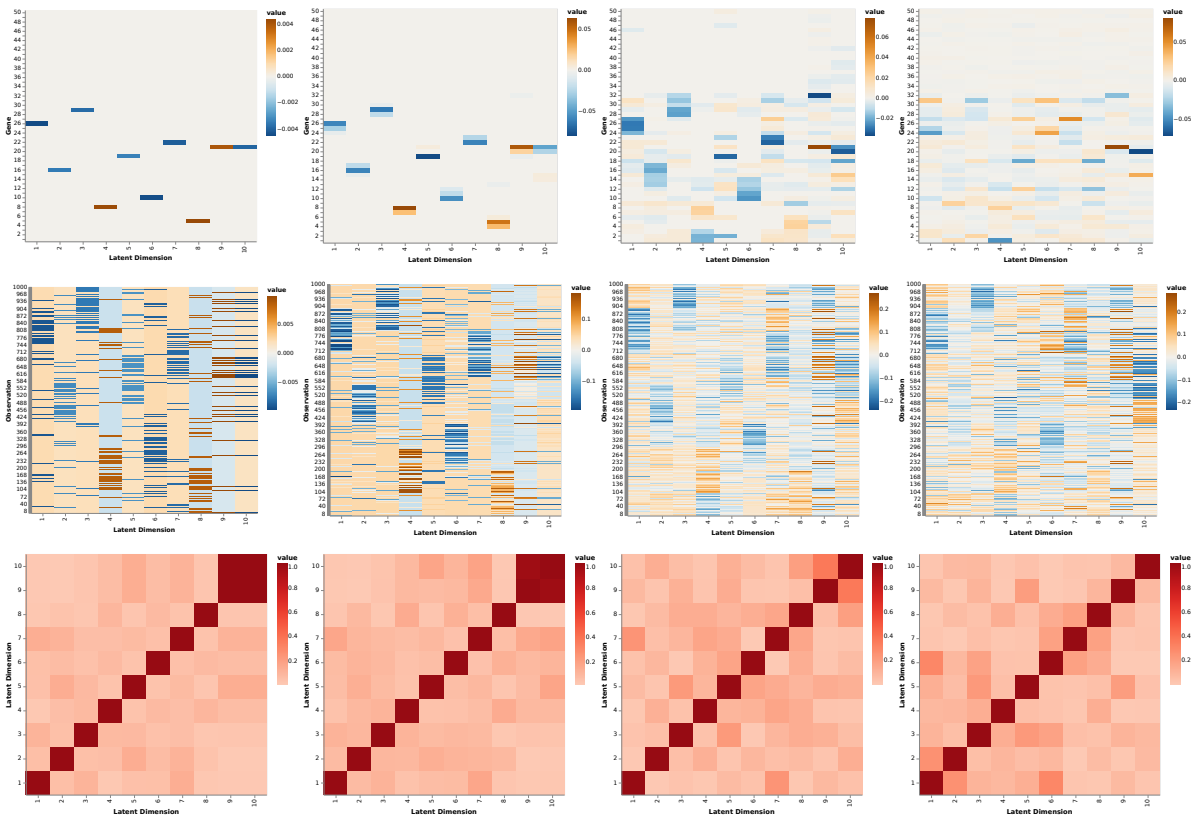
Supplementary Fig. 13: **BAE encoder weight matrices for different decoder parameter initializations.** Heatmaps represents the learned encoder weight matrices of BAEs after 15 training epochs with differently initialized decoder parameters. Training of the BAEs was performed using the same simulated dataset and hyperparameters.

Supplementary Fig. 14: BAE results on simulated stair-like scRNA-seq data during different training epochs



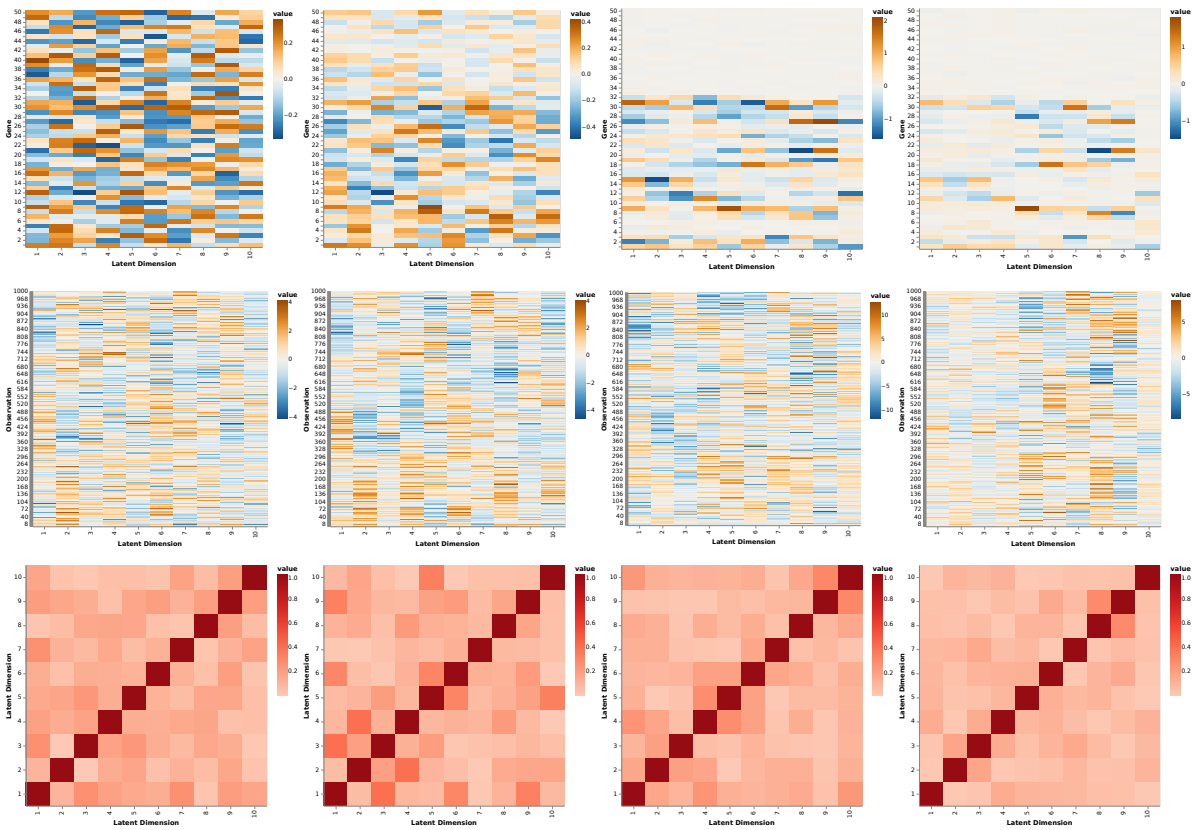
Supplementary Fig. 14: **BAE results on simulated stair-like scRNA-seq data during different training epochs.** Different rows display the heatmaps of the encoder weight matrix (top row), the latent representation (middle row) and the absolute sample Pearson correlation between the different latent dimensions (bottom row). Each column represents the results after training the model for a certain number of training epochs: 1, 15, 1000, 25000 (from left to right).

Supplementary Fig. 15: BAE results without the disentanglement constraint on simulated stair-like scRNA-seq data during different training epochs



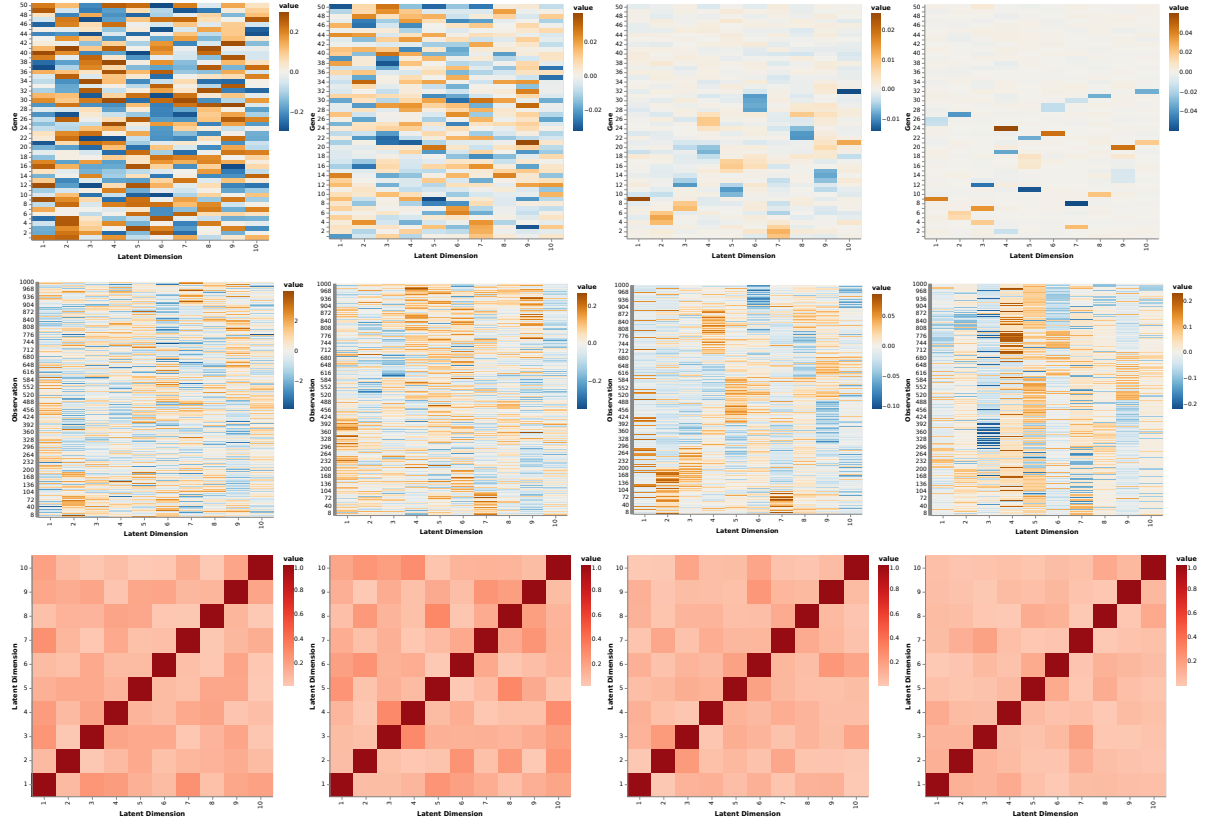
Supplementary Fig. 15: BAE results without the disentanglement constraint on simulated stair-like scRNA-seq data during different training epochs. Different rows display the heatmaps of the encoder weight matrix (top row), the latent representation (middle row) and the absolute sample Pearson correlation between the different latent dimensions (bottom row). Each column represents the results after training the model for a certain number of training epochs: 1, 15, 1000, 25000 (from left to right).

Supplementary Fig. 16: AE results on simulated stair-like scRNA-seq data during different training epochs



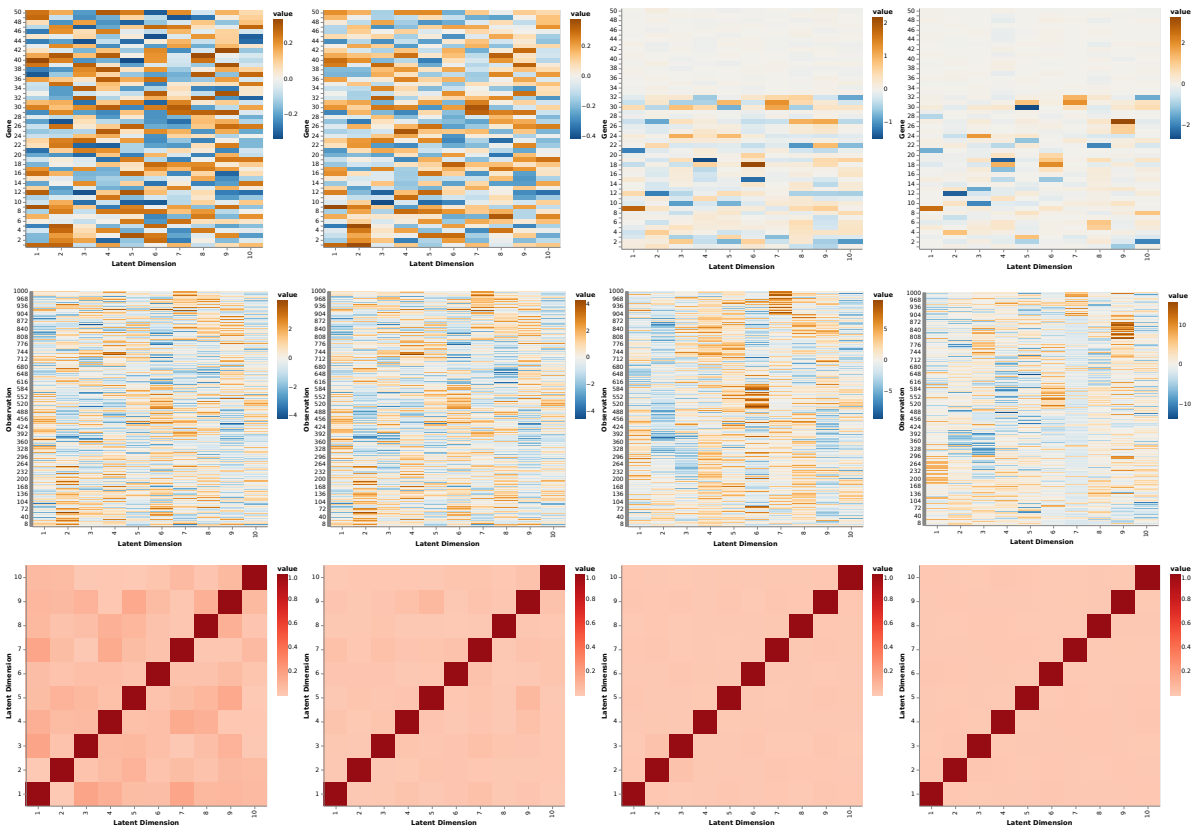
Supplementary Fig. 16: AE results on simulated stair-like scRNA-seq data during different training epochs. Different rows display the heatmaps of the encoder weight matrix (top row), the latent representation (middle row) and the absolute sample Pearson correlation between the different latent dimensions (bottom row). Each column represents the results after training the model for a certain number of training epochs: 1, 15, 1000, 25000 (from left to right).

Supplementary Fig. 17: L_1 AE results on simulated stair-like scRNA-seq data during different training epochs



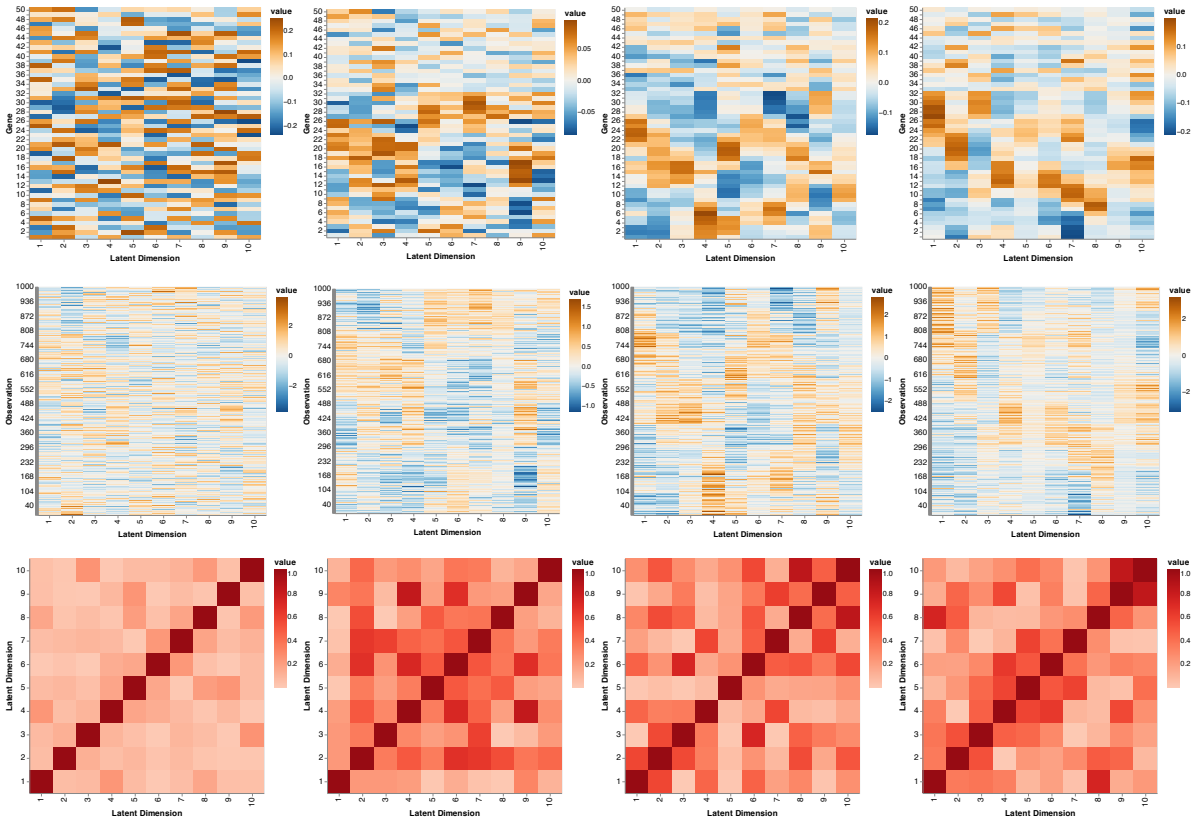
Supplementary Fig. 17: L_1 AE results on simulated stair-like scRNA-seq data during different training epochs. Different rows display the heatmaps of the encoder weight matrix (top row), the latent representation (middle row) and the absolute sample Pearson correlation between the different latent dimensions (bottom row). Each column represents the results after training the model for a certain number of training epochs: 1, 15, 1000, 25000 (from left to right).

Supplementary Fig. 18: corAE results on simulated stair-like scRNA-seq data during different training epochs



Supplementary Fig. 18: **corAE results on simulated stair-like scRNA-seq data during different training epochs.** Different rows display the heatmaps of the encoder weight matrix (top row), the latent representation (middle row) and the absolute sample Pearson correlation between the different latent dimensions (bottom row). Each column represents the results after training the model for a certain number of training epochs: 1, 15, 1000, 25000 (from left to right).

Supplementary Fig. 19: VAE results on simulated stair-like scRNA-seq data during different training epochs



Supplementary Fig. 19: VAE results on simulated stair-like scRNA-seq data during different training epochs. Different rows display the heatmaps of the encoder weight matrix (top row), the latent representation (middle row) and the absolute sample Pearson correlation between the different latent dimensions (bottom row). Each column represents the results after training the model for a certain number of training epochs: 1, 15, 1000, 25000 (from left to right).

Supplementary Table 1: Summary of the genes selected per latent dimension on cortical mouse data

Latent dimension	Number of BAE selected genes (nonzero weights)	Percentage of selected genes (nonzero weights)	Number of change-point genes (nonzero weights)	Percentage of change-point genes (nonzero weights)
1	47	3.13	1	0.07
2	54	3.6	2	0.13
3	53	3.53	1	0.07
4	57	3.8	1	0.07
5	56	3.73	1	0.07
6	58	3.87	1	0.07
7	47	3.13	2	0.13
8	36	2.4	1	0.07
9	57	3.8	1	0.07
10	30	2.0	1	0.07

Supplementary Table 1: **Summary of the genes selected per latent dimension on cortical mouse data.** **First column:** Number of genes selected out of the total number of genes (1500) during training per latent dimension. **Second column:** Percentage of genes selected. **Third column:** Number of change-point genes indicating the top selected genes w.r.t. the absolute values of the corresponding encoder weights per latent dimension (see Methods Section Identification of top selected genes per latent dimension in the main text). **Fourth column:** Percentage of change-point genes.

Supplementary Table 2: Change-point genes per latent dimension on cortical mouse data

Latent dimension	Changepoint gene 1	Changepoint gene 2
1	<i>Vip</i>	
2	<i>Rorb</i>	<i>Pamr1</i>
3	<i>Pvalb</i>	
4	<i>Galnt16</i>	
5	<i>Chodl</i>	
6	<i>Fam19a1</i>	
7	<i>Sema3c</i>	<i>Tnfrsf8l3</i>
8	<i>Gpr17</i>	
9	<i>Arhgap25</i>	
10	<i>Agmo</i>	

Supplementary Table 2: **Change-point genes per latent dimension.** The table shows the change-point genes per latent dimension, sorted by the corresponding absolute values of the encoder weights (see Methods section Identification of top selected genes per latent dimension in the main text). The maximum number of change-point genes per latent dimension in the mouse cortical data is 2 and the patterns observed in the latent dimensions are predominantly driven by the change-point genes.

Supplementary Table 3: Most frequently selected genes on cortical mouse data

	Gene	Number of selections	Percentage of selections	Gene type
1	<i>Sst</i>	30	100	Neural marker
2	<i>Cplx3</i>	30	100	Neural marker
3	<i>Fmo1</i>	30	100	
4	<i>Slco1a4</i>	30	100	Nonneural marker
5	<i>Cyba</i>	30	100	
6	<i>Chodl</i>	30	100	Neural marker
7	<i>Dgkb</i>	30	100	Neural marker
8	<i>Agmo</i>	30	100	
9	<i>Tnfrsf8l3</i>	29	96.66	Neural marker
10	<i>Lrrtm4</i>	29	96.66	
11	<i>Pdlim1</i>	29	96.66	
12	<i>Cd9</i>	29	96.66	Nonneural marker
13	<i>A930038C07Rik (Ndnf)</i>	29	96.66	Neural marker
14	<i>Pvalb</i>	28	93.33	Neural marker
15	<i>Vtn</i>	28	93.33	Nonneural marker
16	<i>Fam84a</i>	28	93.33	
17	<i>Reln</i>	28	93.33	Neural marker
18	<i>Plxnb3</i>	28	93.33	
19	<i>Calb1</i>	28	93.33	Neural marker
20	<i>2310042E22Rik (Tedd3)</i>	27	90	Neural marker
21	<i>Myh4</i>	27	90	Neural marker
22	<i>Cspg4</i>	27	90	Nonneural marker
23	<i>Rorb</i>	27	90	Neural marker
24	<i>Olfml1</i>	27	90	
25	<i>Gpr37l1</i>	27	90	
26	<i>Anax3</i>	26	86.66	
27	<i>Ly86</i>	26	86.66	Nonneural marker
28	<i>C1qb</i>	26	86.66	Nonneural marker
29	<i>Gpr17</i>	26	86.66	Nonneural marker
30	<i>Gng12</i>	26	86.66	
31	<i>Vip</i>	26	86.66	Neural marker
32	<i>Mas1</i>	26	86.66	
33	<i>Foxp2</i>	26	86.66	Neural marker
34	<i>Eltf1</i>	26	86.66	Nonneural marker
35	<i>Wfs1</i>	26	86.66	Neural marker
36	<i>Enpp2</i>	25	83.33	Neural marker
37	<i>Nfia</i>	25	83.33	
38	<i>Mag</i>	25	83.33	Nonneural marker
39	<i>Myh13</i>	25	83.33	Neural marker
40	<i>Myh8</i>	25	83.33	Neural marker
41	<i>Cpne6</i>	25	83.33	
42	<i>Gng11</i>	25	83.33	
43	<i>Slc38a3</i>	25	83.33	
44	<i>Gabra1</i>	24	80	Neurotransmitter receptor
45	<i>Gja1</i>	24	80	Nonneural marker
46	<i>Ecscr</i>	24	80	
47	<i>Slco1c1</i>	24	80	Nonneural marker
48	<i>6330527O06Rik (Lamp5)</i>	24	80	Neural marker
49	<i>Olig2</i>	24	80	
50	<i>Pcp4</i>	24	80	
51	<i>Spock1</i>	24	80	
52	<i>Cmtm5</i>	24	80	
53	<i>Arhgap25</i>	24	80	Neural marker
54	<i>Jam2</i>	24	80	

Supplementary Table 3: **Most frequently selected genes.** The table lists genes selected by the BAE in at least 80% of the 30 runs. Columns display the gene names, number of selections, percentage of selections, and their classification based on [1]. Genes without a classification were not listed in [1].

Supplementary Table 4: Systematic replacement of informative genes with noise genes in cortical mouse data

Selection frequency threshold	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Number of genes replaced	1457	1225	916	565	321	203	127	73	43	19	0
Percentage of genes replaced	97.13%	81.67%	61.07%	37.67%	21.4%	13.53%	8.74%	4.87%	2.87%	1.27%	0.0%

Supplementary Table 4: **Systematic replacement of informative genes with noise genes in cortical mouse data.** The table below summarizes the number of highly variable genes (out of 1500) in the cortical mouse dataset that were replaced with noise genes. The replacement was based on the selection frequency threshold, which represents the percentage of runs in which a gene was selected during the gene selection stability analysis (see Supplementary Note 4 and Supplementary Table 3). Genes with selection frequencies above the threshold were replaced. In total, 1457 of the 1500 highly variable genes were selected by the BAE across all runs.

Supplementary Table 5: Gene selection comparison of the BAE with sparsePCA and scPNMF

Model	Mean percentage of zeros (loading matrix/ encoder weight matrix)	95% CI of percentage of zeros	Mean number of selected genes	Mean percentage of selected genes	Number of genes selected in all runs
BAE	96.25%	[96.21%, 96.29%]	448.57	29.90%	1457
sparsePCA	40.85%	–	1499	99.93%	1499
scPNMF	44.76%	–	1500	100%	1500

Supplementary Table 5: **Gene selection comparison of the BAE with sparsePCA and scPNMF.** The table summarizes the percentages of zero elements in the BAE encoder weight matrix and the loading matrices for sparsePCA and scPNMF across 30 training runs. Additionally, it reports the number of selected genes, where a gene was considered selected in a run if it had at least one nonzero element across the components in the corresponding BAE encoder weight matrix or loading matrices. 95% confidence intervals for sparsePCA and scPNMF are omitted because there was no variability in the nonzero elements across runs.

Supplementary Table 6: timeBAE gene selection comparison

C = Cluster T = Time Point	Visually matching timeBAE latent dimension	Number of selected genes timeBAE	Number of selected genes linear model	Number of gene overlap	Percentage of gene overlap
All	–	87	124	59	67.82%
C1 T1	4	10	36	5	50.0%
C1 T2	5	12	56	7	58.33%
C1 T3	6	11	39	5	45.45%
C2 T1	10	12	33	4	33.33%
C2 T2	11	11	56	7	63.64%
C2 T3	12	15	40	7	46.67%
C3 T1	1	7	30	3	42.86%
C3 T2	2	14	33	8	57.14%
C3 T3	3	14	22	7	50.0%

Supplementary Table 6: **timeBAE gene selection comparison**. The second row consists of the collected gene selection information for all latent dimensions of the timeBAE and for all fitted linear models (latent dimensions were reorganized as described in Methods Section Adaptation for time series). The other rows represent the comparisons between the linear model for the respective cluster and time point and the visually corresponding timeBAE latent dimension. timeBAE latent dimensions 7, 8, and 9 were excluded. The percentages of gene overlap were calculated relative to the number of genes selected for the timeBAE latent dimensions.

Supplementary Table 7: Overlap patterns in simulated stair-like scRNA-seq data

	Stage 1-2	Stage 2-3	Stage 3-4	Stage 4-5	Stage 5-6
Overlap genes	4, 5	7, 8	10, 11	13, 14	16, 17
	Stage 6-7	Stage 7-8	Stage 8-9	Stage 9-10	
Overlap genes	19, 20	22, 23	25, 26	28, 29	

Supplementary Table 7: **Overlap patterns in simulated stair-like scRNA-seq data**

Supplementary Algorithm 1: comp L_2 Boost algorithm

Algorithm 1: comp L_2 Boost algorithm

Input: Initial estimate $\hat{\beta}^{(0)} \in \mathbb{R}^p$, standardized observation matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, mean centered response vector $\mathbf{y} \in \mathbb{R}^n$, step size $\varepsilon \in (0, 1)$, number of boosting iterations $M \in \mathbb{N}$

Output: Estimated solution β^*

for $m = 1, \dots, M$ **do**

$$\begin{aligned} \hat{\eta}^{(m)} &= \mathbf{X} \hat{\beta}^{(m-1)}; \\ \mathbf{r}^{(m)} &= \mathbf{y} - \hat{\eta}^{(m)}; \\ \hat{\gamma}^{(m)} &= \frac{1}{n-1} \cdot \mathbf{X}^\top \mathbf{r}^{(m)}; \\ j^* &= \arg \max_{j=1 \dots p} |\hat{\gamma}_j^{(m)}|; \\ \hat{\beta}^{(m)} &= \begin{cases} \hat{\beta}_j^{(m-1)} + \varepsilon \hat{\gamma}_j^{(m-1)} & , j = j^* \\ \hat{\beta}_j^{(m-1)} & , \text{else.} \end{cases} \end{aligned}$$

end

$$\beta^* = \hat{\beta}^{(M)}$$

Supplementary Algorithm 2: Core BAE optimization algorithm

Algorithm 2: Core BAE optimization algorithm

Input: Standardized observation matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$, BAE with d latent dimensions, zero-initialized transposed encoder weight matrix \mathbf{B} and randomly initialized decoder parameters θ , number of boosting iterations $M \in \mathbb{N}$, step size $\varepsilon \in (0, 1)$ learning rate $\nu \in (0, 1)$, batch size $m \leq n$

Output: Encoder weights \mathbf{B}^*

while *stopping criterion is not met* **do**

$$\begin{aligned} \mathbf{Z} &= \mathbf{X} \mathbf{B}; \\ \mathbf{g}^{(l)} &= -\nabla_{\mathbf{z}^{(l)}} \frac{1}{n} \sum_{i=1}^n L_{\text{rec}}(\mathbf{x}_i, f_{\text{dec}}(\mathbf{z}_i; \theta)), \quad l = 1, \dots, d; \\ \mathbf{for} \quad l \quad \mathbf{in} \quad 1, \dots, d \quad \mathbf{do} \\ & \quad \hat{\mathbf{g}}^{(l)} = \text{standardize}(\mathbf{g}^{(l)}); \\ & \quad \beta^{(l)} = \text{comp}L_2\text{Boost}(\beta^{(l)}, \mathbf{X}, \hat{\mathbf{g}}^{(l)}; \varepsilon, \text{steps} = M); \\ \mathbf{end} \\ \mathbf{B} &= (\beta^{(1)}, \dots, \beta^{(d)}); \\ \theta &= \text{optimize}(\theta, m, \nu; \mathbf{B}) \end{aligned}$$

end

$$\mathbf{B}^* = \mathbf{B}$$

Supplementary Algorithm 3: Disentanglement constraint

Algorithm 3: Disentanglement constraint

Input: Number of latent dimension $l \in \{1, \dots, d\}$, standardized observation matrix X , transposed encoder weight matrix $B \in \mathbb{R}^{p \times d}$, standardized pseudo response vector $\hat{g}^{(l)} \in \mathbb{R}^n$

Output: Constraint pseudo response vector $\hat{y}^{(l)}$

$\mathcal{I} := \{l\} \cup \{k = 1, \dots, d \mid \beta^{(k)} = \mathbf{0}_{\mathbb{R}^p}\};$

if $|\mathcal{I}| = d$ **then**

$\hat{y}^{(l)} = \hat{g}^{(l)};$

end

else

$Z = XB;$

$\alpha = ((Z^{(-\mathcal{I})})^\top Z^{(-\mathcal{I})})^{-1} (Z^{(-\mathcal{I})})^\top \hat{g}^{(l)};$

$y^{(l)} = \hat{g}^{(l)} - Z^{(-\mathcal{I})} \alpha;$

$\hat{y}^{(l)} = \text{standardize}(y^{(l)});$

end

Supplementary References

1. Tasic B, Menon V, Nguyen TN, Kim TK, Jarsky T, Yao Z, Levi B, Gray LT, Sorensen SA, Dolbeare T, Bertagnolli D, Goldy J, Shapovalova N, Parry S, Lee C, Smith K, Bernard A, Madisen L, Sunkin SM, Hawrylycz M, et al. Adult mouse cortical cell taxonomy revealed by single cell transcriptomics. *Nature Neuroscience* 2016;19:335–46.
2. Mairal J, Bach F, Ponce J, and Sapiro G. Online dictionary learning for sparse coding. In: *Proceedings of the 26th annual international conference on machine learning*. 2009:689–96.
3. Song D, Li K, Hemminger Z, Wollman R, and Li JJ. scPNMF: sparse gene encoding of single cells to facilitate gene selection for targeted gene profiling. *Bioinformatics* 2021;37:i358–i366.
4. Hess M, Hackenberg M, and Binder H. Exploring generative deep learning for omics data by using log-linear models. *Bioinformatics* 2020;36:5045–53.
5. Glorot X and Bengio Y. Understanding the difficulty of training deep feedforward neural networks. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010:249–56.
6. Kingma DP and Welling M. Auto-Encoding Variational Bayes. In: *2nd International Conference on Learning Representations (ICLR), Conference Track Proceedings*. Ed. by Bengio Y and LeCun Y. 2014.
7. Kingma DP and Ba J. Adam: A Method for Stochastic Optimization. In: *3rd International Conference on Learning Representations (ICLR), Conference Track Proceedings*. Ed. by Bengio Y and LeCun Y. 2015.
8. Loshchilov I. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 2017.