Supplementary Information for SpaKnit: correlation subspace learning for integrating spatial multi-omics data

Contents

Supplementary Notes

Benchmarking methods

To evaluate the performance of SpaKnit, we compare it against six state-of-the-art methods, including spatial multi-omics data integration method (SpatialGlue[1]), single-cell multi-omics data integration methods (Seurat WNN[2], MultiVI[3], and MultiMAP[4]), spatial transcriptome methods (STAGATE[5] and SpaGCN[6]).

After completing the preprocessing steps, we proceeded to evaluate each benchmarking method following the instructions provided in their respective vignettes. Below, we describe the details of each method.

SpatialGlue. Feature graphs and spatial graphs are constructed using the *construct_neighbor_graph()* function based on the preprocessed feature and spatial information. Subsequently, the model is trained on the neighborhood graph with default parameters to obtain an integrated latent representation.

Seurat WNN. The *FindMultiModalNeighbors()* function is used to identify the nearest neighbors of each cell based on a weighted combination of the two modalities. The method then constructs a shared nearest neighbor (SNN) graph and performs clustering, incorporating the computed modality weights for each spot.

MultiVI. We first align and order the multi-omics data using *organize_multiome_anndatas()* function, consolidating different modalities into a single AnnData object. We then set batch_key="modality" when initializing the dataset with *scvi.model.MULTIVI.setup_anndata()* function. The model is trained using default parameters, and the integrated latent space representation is obtained.

MultiMAP. The integration is performed using *MultiMAP.Integration()* function, which generates the integrated latent representation under the default parameter Settings.

SpaGCN. We construct an adjacency matrix from spatial coordinates using

the *SpaGCN.calculate_adj_matrix()* function to calculate. The number of clusters is specified, and the optimal resolution is determined using the *spg.search_res()* function. Finally, the model is built and spatial clustering is performed.

STAGATE. We first merge the data from the two omics modalities. Then a spatial graph is computed using *STAGATE.Cal_Spatial_Net()* function with rad_cutoff=10. The model is trained using *STAGATE.train_STAGATE()* with alpha=0, generating integrated feature representations.

Simulations

Simulated Data for Different Spatial Patterns

To evaluate the performance of individual models in SpaKnit, we generated simulated datasets representing four spatial patterns with varying complexity. These spatial structures reflect common biological tissue organizations, either individually or as combinations. The four spatial patterns include quadrant structure, stripe structure, arc structure, and layered structure.

We first generate coordinate values using the *numpy.linspace()* function with parameters start=0, stop=60 and num=20:

$$x_i = y_i = a + i \cdot \frac{b-a}{n-1}, \; for \; i = 0,1,\ldots,19 \tag{1}$$

where:

- a is the starting value of the sequence and set to 0,
- b is the ending value of the sequence and set to 60,
- n is the number of elements in the sequence and set to 20.

This process generates 400 coordinates, corresponding to the number of spots in a single batch. We store these values in a 400x2 coordinate matrix. Since the samples are sorted based on their categories, we can simply rearrange the coordinate matrix to allocate spots from different categories into distinct spatial regions.

Quadrant Pattern. Firstly, we determine the center point of the coordinate matrix at $(30,30)$, and divide the entire space into four quadrants based on this

center point. By sorting and arranging coordinates according to their quadrant number, we assign spots of different cell types to distinct quadrants to form a clear regional segmentation.

Stripe Pattern. We sort the Y-axis coordinates in ascending order and divide them into four stripe regions along the Y-axis. This results in a vertically striped spatial pattern, where different cell types are arranged in sequential horizontal layers.

Arc Pattern. To construct an arc-shaped spatial pattern, we calculate the Euclidean distance of each coordinate from the origin and sort the coordinates accordingly. This process groups the space into multiple concentric arcs, where each level represents a distinct distance interval from the origin.

Layered Pattern. Similar to the Arc Structure, we calculate the Euclidean distance of each coordinate from the geometric center $(30,30)$ and sort these distances by their distances. This approach forms a nested layered organization where cells are distributed in spatial hierarchies based on their proximity to the center.

Simulated Data for Different Noise Combinations and Levels

Spatial multi-omics technologies enable the simultaneous capture of multiple molecule modalities within a single tissue slice, offering a comprehensive molecular perspective on tissue structure and composition. However, the process of capturing one modality can sometimes compromise the molecular integrity required of another, reducing its sensitivity in subsequent rounds of molecular sequencing or imaging[7]. Moreover, some modalities may experience "dropout" events[8].

To simulate this real-world adverse effect on molecular integrity and assess the robustness of SpaKnit under noisy conditions, we introduce three types of noise to the real datasets: Gaussian noise, pepper noise (dropout events), and mixed noise (combination of Gaussian and pepper noise).

Gaussian Noise Simulation. To simulate measurement errors and

random fluctuations in real biological signals, we introduce Gaussian noise into the spatial multi-omics data. For a dataset with modalities, let

- $\mathrm{X}_1 \in R^{n \times m_1}$ and $\mathrm{X}_2 \in R^{n \times m_2}$, representing the original matrices for the two modalities,
- n be the number of measured spots,
- m_1 and m_2 be the number of features in each modality.

We add Gaussian noise as follows:

$$\begin{aligned} x_{ij}^{noised} &= x_{ij} + \mathcal{N}(\mu_1, \sigma_1^2), x_{ij} \in X_1 \\ x_{ij}^{noised} &= x_{ij} + \mathcal{N}(\mu_2, \sigma_2^2), x_{ij} \in X_2 \end{aligned} \tag{2}$$

where:

- $\mathcal{N}(\mu_1, \sigma_1^2)$ and $\mathcal{N}(\mu_2, \sigma_2^2)$ represent Gaussian distributions with given mean and variance,
- x_{ij} represents the original expression data of the location (i, j),
- x_{ij}^{noised} represents the noisy data of the location (i, j),
- To ensure the non-negativity of the data, we set $x_{ij}^{noised} = 1e-10, if\ x_{ij}^{noised} < 0$.

Pepper Noise Simulation ("Dropout" Events). To simulate "dropout" events, we introduce pepper noise using a Boolean mask:

$$x_{ij}^{noised} = x_{ij} \times mask_{ij}^{dropout} \tag{3}$$

where $mask_{ij}^{dropout}$ is a binary dropout mask, defined as:

$$mask_{ij}^{dropout} = \begin{cases} 0 & with\ probability\ \alpha \\ 1 & with\ probability\ 1-\alpha \end{cases} \tag{4}$$

where the dropout rate α controls the proportion of missing values in the dataset.

Mixed Noise Simulation

To assess SpaKnit's resilience under realistic noise conditions, we combine Gaussian noise and pepper noise to simulate the compound effects of

measurement errors and dropout events in biological data.

Quantifying Noise with Signal-to-Noise Ratio (SNR)

To quantify the impact of added noise, we calculate the signal-to-noise Ratio (SNR):

$$\mathrm{SNR}\left(\mathrm{dB}\right)=10\log_{10}\left(\frac{P_{expression}}{P_{\mathrm{noise}}}\right)=10\log_{10}\left(\frac{\frac{1}{n}\sum_{i,j}^{n}(x_{ij})^{2}}{\frac{1}{n}\sum_{i,j}^{n}(x_{ij}^{noised}-x_{ij})^{2}}\right) \quad (5)$$

where:

- $P_{expression}$ represents the power of the original expression data,
- P_{noise} represents the power of the data added noise,
- x_{ij} represents the original expression data of the location (i,j),
- x_{ij}^{noised} represents the noisy data of the location (i,j).

The SNR measures the relative strength of the signal compared to noise and we use it to quantify the amount of noise added.

Downstream analysis

Spatial Clustering. We applied our proposed framework to integrate latent representations from multimodal data and performed spatial clustering analysis. After evaluating clustering performance, we chose the 'leiden' algorithm[9] for the mouse brain dataset and the 'mclust' algorithm[10] for all other datasets. We conducted extensive experiments with different cluster numbers, comparing them against known biological structures or cell types to determine the most appropriate clustering parameters.

Differentially Expressed Gene (DEG) Analysis. Using scanpy package[11], we performed differential expression analysis based on the spatial domain identification results. Specifically, We first calculated a hierarchical cluster tree using the *sc.tl.dendrogram()* function, and then identified differentially expressed genes in each cluster using the *sc.tl.rank_genes_groups()* function.

Trajectory Inference using Partition-based Graph Abstraction[12]

(PAGA). After obtaining the integrated multimodal embeddings, we performed the trajectory inference task using Partition-based Graph Abstraction (PAGA). We first computed the Nearest neighbor graph using *sc.pp.neighbors()*, then performed PAGA analysis using *sc.tl.paga()* with default parameters.

Evaluation metrics

Adjusted Rand Index (ARI). ARI evaluates clustering performance by quantifying the similarity between predicted and true labels, adjusting for randomness:

$$ARI = \frac{RI - E(RI)}{max(RI) - E(RI)}$$
$$RI = \frac{a+b}{n(n-1)/2} \quad (6)$$

where:

- n denotes the total number of samples,
- a is the number of clusters that belong to the same cluster as the clustering result,
- b is the number of different categories clustered as different clusters.

The ARI ranges from -1 to 1, where 1 indicates perfect clustering, 0 suggests random clustering, and -1 represents the worst case.

Adjusted Mutual Information (AMI). AMI measures the amount of shared information between clustering results and true labels, adjusting for randomness:

$$\mathrm{AMI} = \frac{I(X;Y) - E[I(X;Y)]}{max(H(X),H(Y)) - E[I(X;Y)]}$$
$$H(X) = -\sum_i p(x_i) log p(x_i),\ H(Y) = -\sum_i p(y_i) log p(y_i) \quad (7)$$

where:

- $I(X;Y)$ denotes the mutual information of the clustering results and the true labels,
- $\mathrm{H}(X), H(Y)$ are the entropy values for predicted and true labels.

The AMI ranges from -1 to 1, where 1 indicates perfect agreement, 0

suggests random, and -1 represents the worst case.

Normalized Mutual Information (NMI). Similar to AMI but simpler, NMI for easy comparison across clustering methods. The NMI is described as follows:

$$NMI(X;Y) = 2\frac{I(X;Y)}{H(X)+H(Y)} \tag{8}$$

the definitions of $I(X;Y), \mathrm{H}(X)$ and $H(Y)$ are the same as those defined in equation (7).

Moran's I score. It measures spatial autocorrelation, assessing the degree to which a given label or feature is significantly associated with the clustering of its spatial distribution[13]. Specifically, Moran's I score is described as follows:

$$Moran's\ I\ score = \frac{N}{W}\frac{\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(x_i-\bar{x})(x_j-\bar{x})}{\sum_{i=1}^{N}(x_i-\bar{x})^2} \tag{9}$$

where:

- N is the total number of spatial coordinates,
- w_{ij} are the elements of a matrix of spatial weights and $W = \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}$,
- x are the given labels or features.

We calculated the Moran's I score using squidpy package[13] with the function *squidpy.gr.spatial_autocorr().*

CHAOS score. It measures the spatial continuity of given labels[14]. Specifically, for each categorical label encompassing a dataset of more than two samples, we initially calculated the Euclidean distances among all intra-class spots, then constructed a one-nearest neighbor (1NN) graph and its corresponding adjacency matrix *A*. We denote a_{ij} is element of *A*, which is calculated as:

$$a_{ij} = \begin{cases} dist(i,j) & if\ spot\ i\ is\ spot\ j's\ nearest\ neighbor \\ 0 & otherwise \end{cases} \tag{10}$$

where $dist(i,j)$ is the Euclidean distance between spot *i* and spot *j*.

Finally, for each classification category, the mean of all non-zero entries within the corresponding adjacency matrix is taken as the CHAOS score for that particular category. For labels that consist of a solitary sample, the CHAOS score is systematically set to zero:

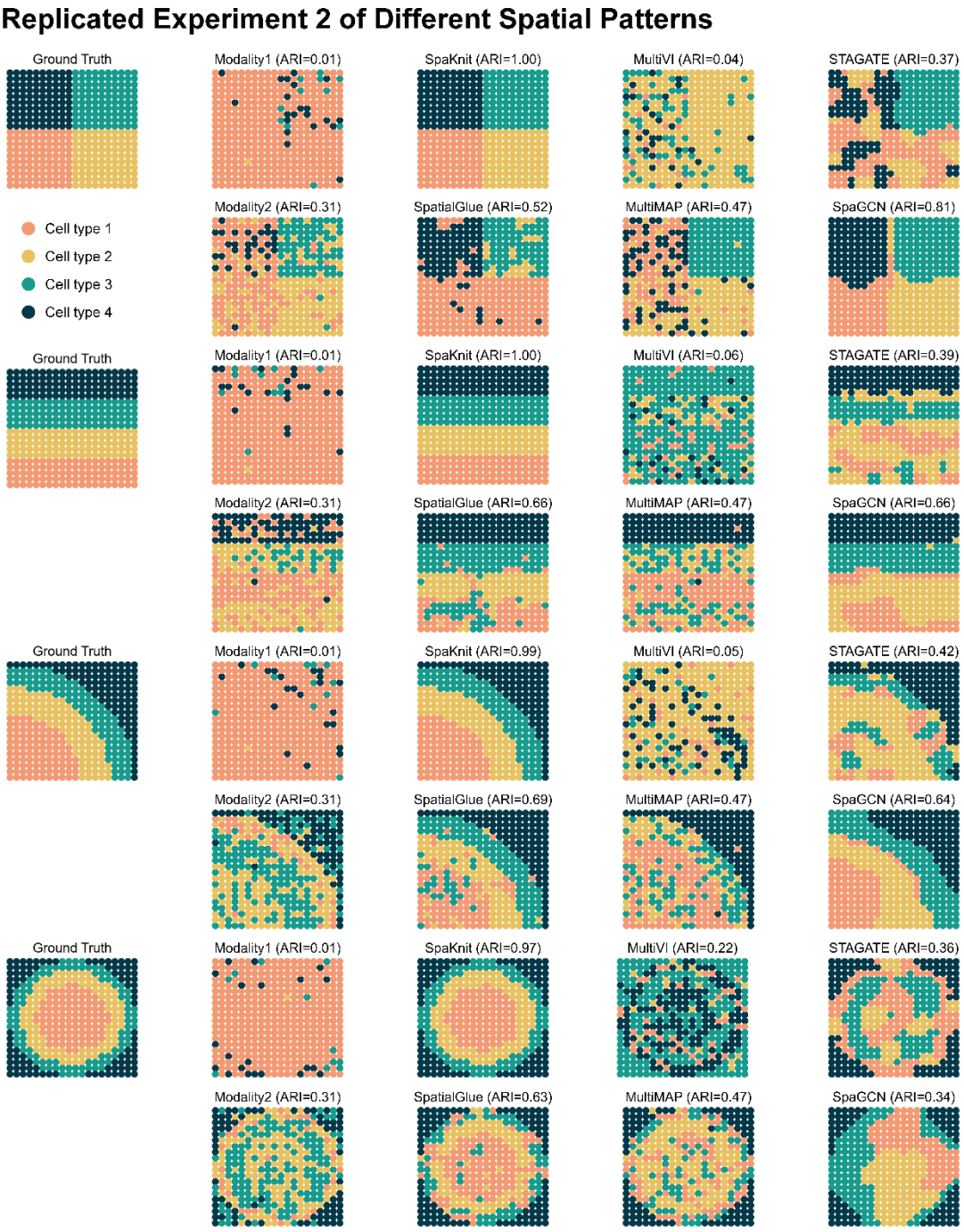
$$CHAOS\ score_k = \begin{cases} \dfrac{\sum_{ij}^{n_k} a_{ij}}{n_k} & if\ n_k > 1 \\ 0 & otherwise \end{cases} \tag{11}$$

where n_k is the spots total number of cluster *k*. A smaller $CHAOS\ score_k$ indicates better spatial continuity of cluster *k*.

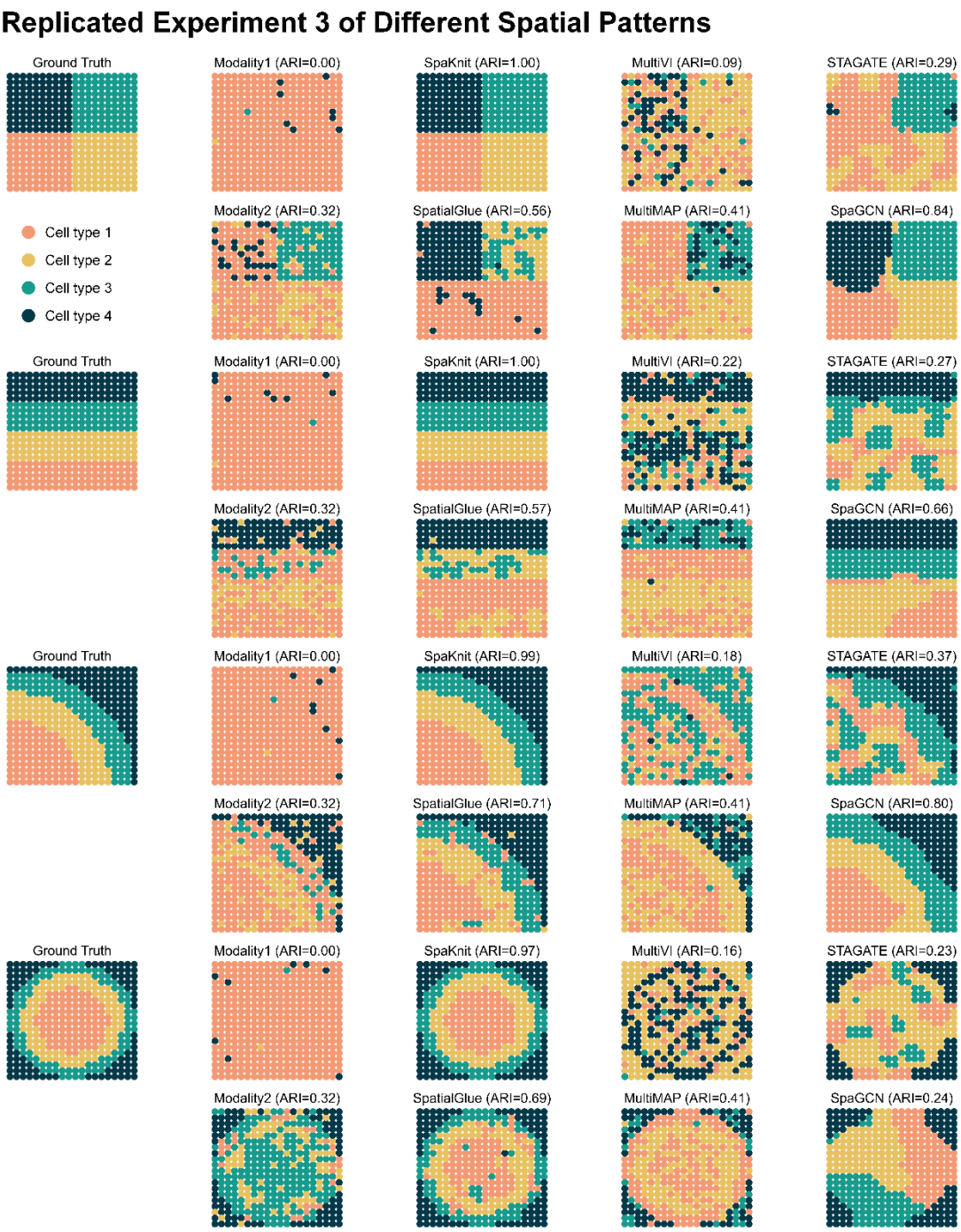
Ablation studies

To evaluate the contribution of different components in SpaKnit, we performed ablation experiments: removed the INR module and evaluated performance, removed the DCCAE module and analyzed its effect, and individually removed loss functions (CCA Loss and Reconstruction Loss). Each variant was tested on four simulated spatial patterns, comparing spatial domain identification accuracy against the full SpaKnit.

Supplementary Figures

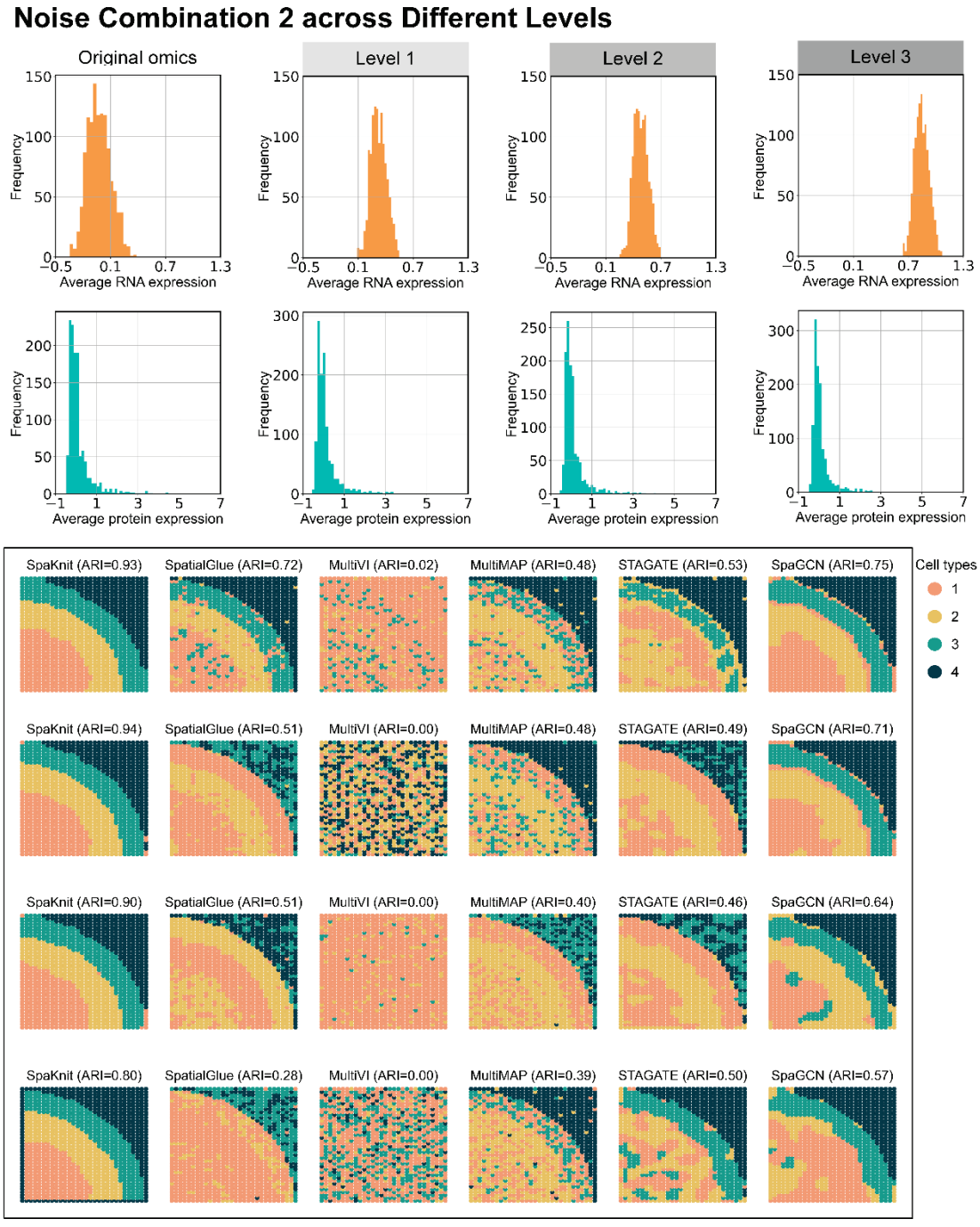


See next page

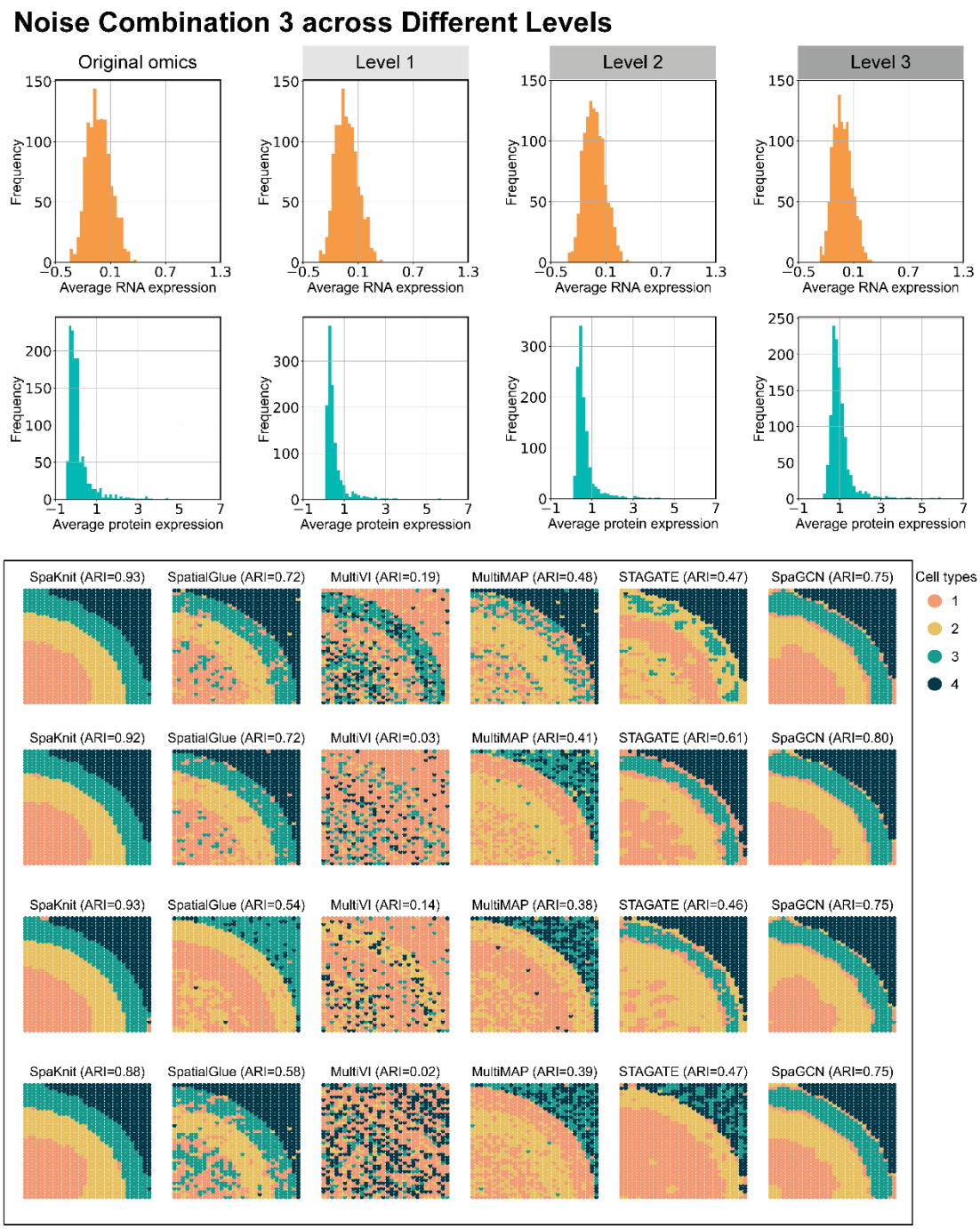


Supplementary Figure S1. Spatial Clustering Performance Comparison Among Six Methods Across Four Spatial Patterns of Replicated Experiments 2 and 3. For each of the four spatial patterns, the figure presents **ground truth** labels, clustering results using **mono-omics**, and the performance of **various integration methods**. These patterns represent different levels of spatial complexity, modeled after real biological structures. The clustering outcomes illustrate the ability of each method to preserve spatial

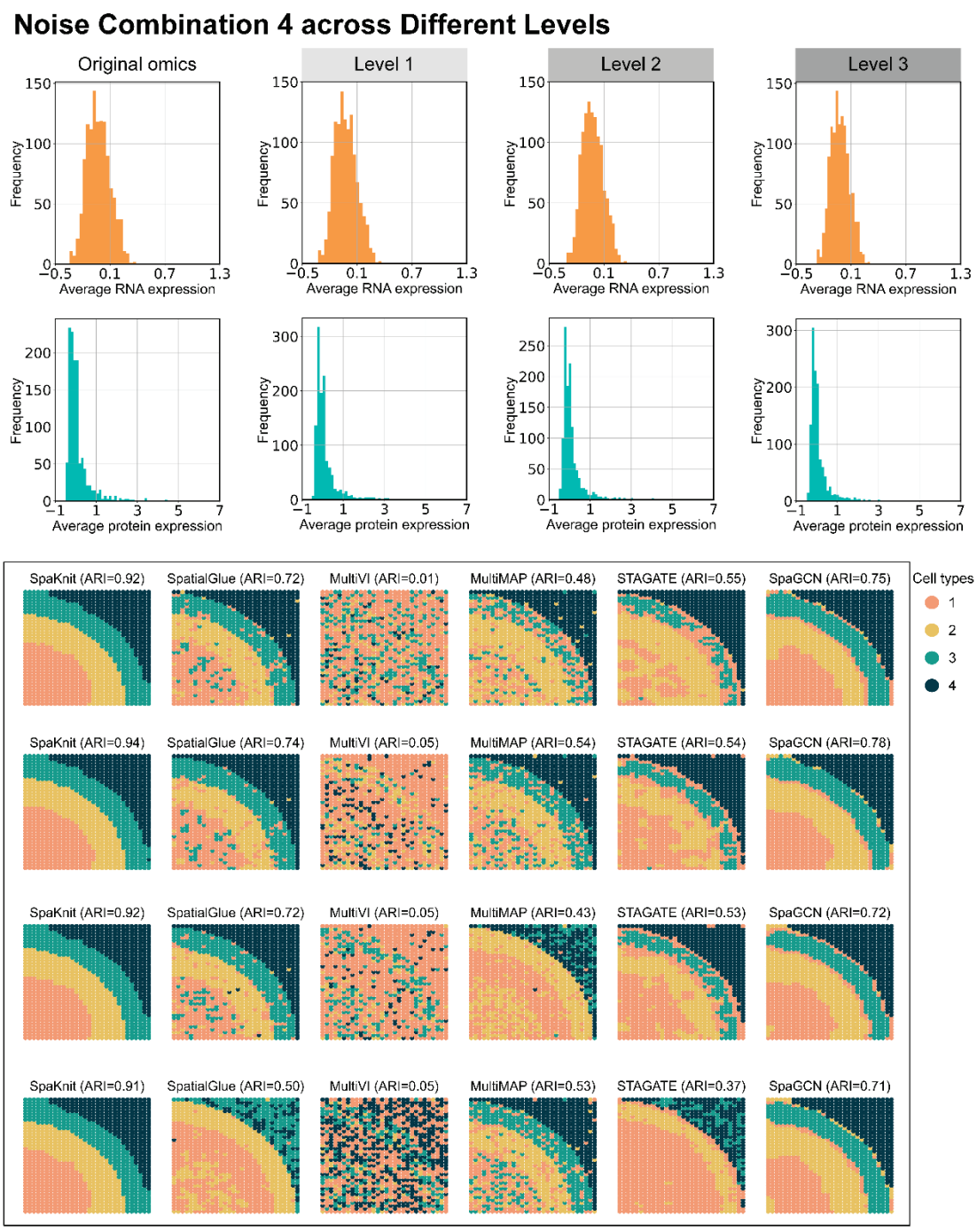
organization and accurately identify spatial domains. **Spatial Pattern 1:** a simple spatial distribution of quadrant structure with well-separated regions; **Spatial Pattern 2:** a moderate spatial complexity of strip structure with gradual transitions between regions, forming stripe-like patterns; **Spatial Pattern 3:** a more intricate spatial pattern of art structure with gradual transitions between regions, exhibiting curved patterns; **Spatial Pattern 4:** the most complex spatial layered structure, characterized by concentric, progressively changing regions. All replicate experiments (1-3) across four spatial patterns quantitively evaluated Using AMI and NMI are presented in **Extended Data Fig. 2e.**



See next page

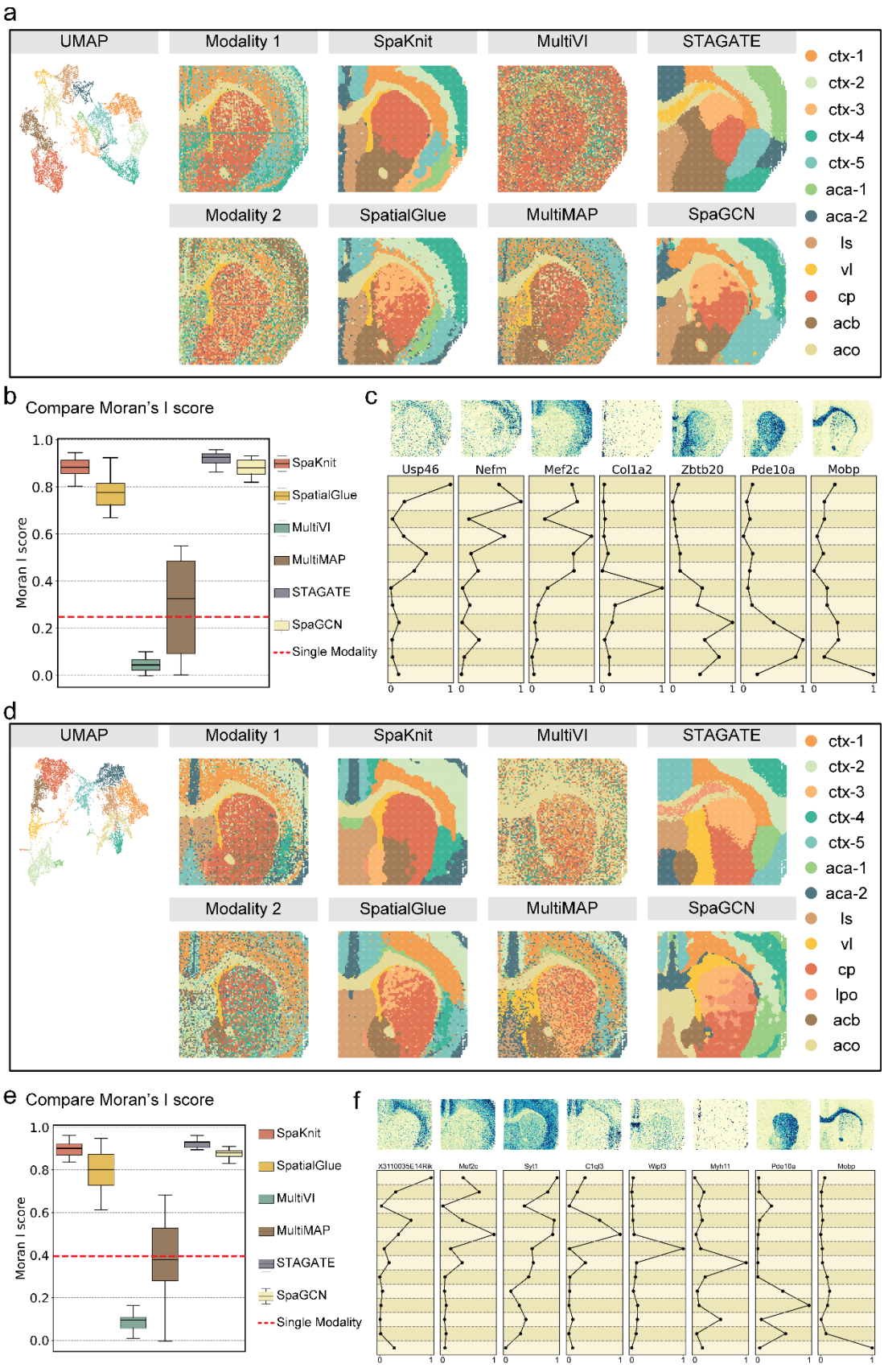


See next page



Supplementary Figure S2. Spatial Clustering Performance Comparison Among Six Methods Across Four Levels of Noise Combination 2-4. The figure presents the original expression frequency distribution, the data distributions following the addition of three different levels of noise, and the performance of various integration methods. These noises reflect the different levels of noise interference of real spatial multi-omics technologies. **Noise Combination 2:** Gaussian Noise added to modality 1 and pepper Noise added

to Modality 2; **Noise Combination 3:** Pepper Noise added to modality 1 and Gaussian Noise added to modality 2; **Noise Combination 4:** Pepper Noise added to two modalities. All noise combinations across different noise levels quantitively evaluated Using AMI and NMI are presented in **Extended Data Fig.3.**



Supplementary Figure S3. Spatial Domain Identification in H3K27me3 and H3K27ac Slices of Mouse Brain Dataset. (a) Spatial Domain Identification

in H3K27me3 Slice. This panel compares the mono-omics representation with the clustering performance of different methods in the H3K27me3 slice. **(b) Quantitively Evaluation Using Moran's I Score in H3K27me3 Slice.** To quantify performance, **Moran's I score** was computed for each method. Boxplot displays the Moran's I score distributions across all six methods. The red dashed line marks the median performance of mono-omics methods, serving as a baseline reference. **(c) Marker Genes Identification in H3K27me3 Slice.** The accurate spatial domain identifications of SpaKnit effectively highlight the marker genes of each cell type. The scanpy package was subsequently employed for further identification in H3K27me3 Slice. **(d) Spatial Domain Identification in H3K27ac Slice.** This panel compares the mono-omics representation with the clustering performance of different methods in the H3K27ac slice. **(e) Quantitively Evaluation Using Moran's I Score in H3K27ac Slice.** To quantify performance, **Moran's I score** was computed for each method. Boxplot displays the Moran's I score distributions across all six methods. The red dashed line marks the median performance of mono-omics methods, serving as a baseline reference. **(f) Marker Genes Identification in H3K27ac Slice.** The accurate spatial domain identifications of SpaKnit effectively highlight the marker genes of each cell type. The scanpy package was subsequently employed for further identification in H3K27ac Slice.

Supplementary Tables

Supplementary Table S1. Description of Experimental Datasets Used in This Study.

Platform	Name	Replicate	Feature size	Related figures
10x Genomics Visium	Simulation of different spatial patterns	1	400x3,000 400x31	Fig. 2a-c, Extended Data Fig. 1a,b, Extended Data Fig. 2 a-e, Fig. S1
		2	400x3,000 400x31	
		3	400x3,000 400x31	
		4	400x3,000 400x31	
10x Genomics Visium	Simulation of different noise combinations	1	1,200x3,000 1,200x31	Fig.2 d-f, Extended Data Fig. 3a-c, Fig. S2
		2	1,200x3,000 1,200x31	
		3	1,200x3,000 1,200x31	
		4	1,200x3,000 1,200x31	
Spatial-transcriptome-epigenome	Mouse Brain	ATAC P22	9,215x22,914 9,215x121,068	Fig. 3a-c, Extended Data Fig. 4a-f, Fig. S3
		H3K4me3	9,548x22,731 9,548x35,270	
		H3K27ac	9,370x23,415 9,370x104,162	
		H3K27me3	9,752x25,881 9,752x70,470	
SPOTS	Mouse Spleen	1	2,568x32,285 2,568x21	Fig. 3d-f, Extended Data Fig. 5a-f
		2	2,768x32,285 2,768x21	
Stereo-CITE-seq	Mouse Thymus	1	4,253x23,529 4,253x19	Fig. 4a-f, Extended Data Fig. 6a-d
		2	4,646x23,960 4,646x19	
		3	4,228x23,221 4,228x19	

Supplementary Table S2. Description of noise added to the simulated datasets of different noise conditions. Each noise combination set 4 noise level, the first one was the control group, and the noise of the other three groups increased in order. μ_1, μ_2 denote the mean values of two modalities.

Dataset	Noise level	omics 1			omics 2		
		Mean value	Standard deviation	Dropout rate	Mean value	Standard deviation	Dropout rate
Noise condition 1	1						
	2	μ_1	0.5		μ_2	0.5	
	3	μ_1	1		μ_2	1	
	4	μ_1	2		μ_2	2	
Noise condition 2	1						
	2	μ_1	0.5				0.05
	3	μ_1	1				0.1
	4	μ_1	2				0.2
Noise condition 3	1						
	2			0.05	μ_2	0.5	
	3			0.1	μ_2	1	
	4			0.2	μ_2	2	
Noise condition 4	1						
	2			0.05			0.05
	3			0.1			0.1
	4			0.2			0.2

Supplementary References

1. Long, Y., Ang, K.S., Sethi, R. et al. Deciphering spatial domains from spatial multi-omics with SpatialGlue. *Nat Methods* **21**, 1658–1667 (2024).
2. Hao, Y. et al. Integrated analysis of multimodal single-cell data. *Cell* **184**, 3573–3587 (2021).
3. Ashuach, T. et al. MultiVI: deep generative model for the integration of multimodal data. *Nat. Methods* **20**, 1222–1231 (2023).
4. Jain, M.S., Polanski, K., Conde, C.D. et al. MultiMAP: dimensionality reduction and integration of multimodal data. *Genome Biol* **22**, 346 (2021).
5. Dong, K., Zhang, S. Deciphering spatial domains from spatially resolved transcriptomics with an adaptive graph attention auto-encoder. *Nat Commun* **13**, 1739 (2022).
6. Hu, J., Li, X., Coleman, K. et al. SpaGCN: Integrating gene expression, spatial location and histology to identify spatial domains and spatially variable genes by graph convolutional network. *Nat Methods* **18**, 1342–1351 (2021).
7. Coleman, K., Schroeder, A. & Li, M. Unlocking the power of spatial omics with AI. *Nat Methods* **21**, 1378–1381 (2024).
8. Qiu, P. Embracing the dropouts in single-cell RNA-seq analysis. *Nat Commun* **11**, 1169 (2020).
9. Traag, V.A., Waltman, L. & van Eck, N.J. From Louvain to Leiden: guaranteeing well-connected communities. *Sci Rep* **9**, 5233 (2019).
10. Scrucca, L., Fop, M., Murphy, T. B. & Raftery, A. E. mclust 5: Clustering, Classification and Density Estimation Using Gaussian Finite Mixture Models. R J. 8, 289–317 (2016).
11. Wolf, F. A., Angerer, P. & Theis, F. J. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol*. **19**, 15 (2018).
12. Wolf, F.A., Hamey, F.K., Plass, M. et al. PAGA: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of

single cells. *Genome Biol* **20**, 59 (2019).

13. Palla, G., Spitzer, H., Klein, M. et al. Squidpy: a scalable framework for spatial omics analysis. *Nat Methods* **19**, 171–178 (2022).
14. Shang, L., Zhou, X. Spatially aware dimension reduction for spatial transcriptomics. *Nat Commun* **13**, 7203 (2022).