

Supplementary Notes

Note S1 Benchmarking model scalability:

During benchmarking, we also monitored how running time and memory usage varied with changes in the number of cells, the number of genes, and their product across different methods. Since GraphST-PASTE is an extension of GraphST, only the performance metrics for GraphST are included here.

In terms of runtime, CN was the fastest among all methods. Although its running time increased with the number of cells, it consistently remained the shortest due to its simple design. Banksy was the second fastest, with runtime increasing gradually as the number of cells grew. In contrast, STAIG, SPIRAL and SpaDo exhibited the longest runtimes and were particularly sensitive to cell count, especially STAIG and SpaDo, which showed a sharp rise in runtime when processing datasets with more than 10,000 cells. The remaining methods fell in between. Notably, MENDER was more sensitive to cell count, showing a noticeable increase in runtime when processing datasets with over 100,000 cells while NicheCompass was particularly sensitive to the product of cell and gene counts (Additional file 2: Fig. S26a, c, e).

From a memory usage perspective, PRECAST and Banksy was the most efficient overall, though their memory consumption increased with the number of cells. CN, NicheCompass, MENDER, and CellCharter followed, with CN, NicheCompass and MENDER particularly affected by increases in the cell-gene product. When this product exceeded 40 million, their memory usage surpassed CellCharter. STAIG and GraphST were the most memory-intensive methods, with memory usage rising sharply once the number of cells exceeded 10,000. SpaDo also showed a substantial increase in memory usage as cell numbers increased. Additionally, due to these high demands, several methods, including GraphST, SpaDo, SPIRAL, STAIG and STAligner were limited to processing datasets with up to 40,000 cells or spots (Additional file 2: Fig. S26b, d, f).

In summary, Banksy demonstrated a clear advantage in both runtime and memory efficiency. CN, CellCharter, MENDER and PRECAST followed as second-tier performers. SpaDo and NicheCompass are better suited for smaller datasets, while other methods demand substantial memory and are limited in handling larger datasets.

Note S2 Datasets information:

In this benchmark study, we collated 19 datasets from various sequencing technologies, each containing at least two slices, to evaluate different methods. The datasets were categorized into five datasets based on sequencing technology and their application in this study: 10X Visium dataset (three datasets), the MERFISH dataset, comprising MERFISH Preoptic (1 dataset), MERFISH Brain (11 datasets), and the Large-scale dataset (1 dataset), STARMap dataset, BaristaSeq dataset, and TNBC dataset (MIBI-TOF sequencing). Below are the details of each dataset:

10X Visium dataset. This dataset includes spatial transcriptome data obtained using the 10X Visium platform. While the number of genes detected is comparable to single-cell transcriptome sequencing, each sequencing unit represents a spot rather than a cell. The dataset includes dorsolateral prefrontal cortex (DLPFC) tissue from three samples generated by the spatialLIBD project. Each sample consists of four slices, each with domain annotations. In this study, each sample was designated as DLPFC S1, DLPFC S2 and DLPFC S3. Specifically, DLPFC S1 contains the following slices: 151507 (4221 spots), 151508 (4381 spots), 151509 (4788 spots) and 151510 (4595 spots). DLPFC S2 contains the following slices: 151669 (3636 spots), 151670 (3484 spots), 151671 (4093 spots) and 151672 (3888 spots). DLPFC S3 contains four slices: 151673 (3611 spots), 151674 (3635 spots), 151675 (3566 spots) and 151676 (3431 spots). A total of 33,538 genes are detected in each slice. The original count matrix was downloaded, requiring normalization according to each method's specifications during the benchmarking process. This dataset is used for the evaluation of multi-slice integration, spatial clustering and spatial alignment.

MERFISH dataset. This dataset contains spatial transcriptome data generated using multiplexed error-robust fluorescence in situ hybridization (MERFISH) technology. Although MERFISH can achieve single-cell resolution, it has a limited number of detectable genes. The dataset includes data from three distinct studies: MERFISH Preoptic, MERFISH Brain, and the Large-scale datasets.

MERFISH Preoptic. This dataset comprises five consecutive slices from the same mouse preoptic region, with each slice identifying eight distinct spatial domains based on cell distribution. The slices are as follows: 0.04 (5488 cells), 0.09 (5557 cells), 0.14 (5926 cells), 0.19 (5803 cells), and 0.24 (5543 cells). Each slice detects 155 genes, and the dataset has been preprocessed, so no further normalization is needed for the benchmarking process. This dataset is used for the evaluation of multi-slice integration, spatial clustering and spatial alignment.

MERFISH Brain. This dataset includes slices from the frontal cortex and striatum regions of 12 mice, designated MERFISH Brain S1-12. MERFISH Brain S1 was excluded from the study as it contained only a single slice. The remaining samples include two to three slices, with approximately 25,000 to 40,000 cells per sample and 374 genes per slice. The original count data was downloaded and needs to be normalized in the benchmarking process. Additionally, all 374 genes are treated as high-variation genes if high-variation genes need to be identified. This dataset is used for the evaluation of multi-slice integration, spatial clustering and only MERFISH Brain S1 is used for the evaluation of spatial alignment.

Large-scale dataset. This dataset includes data from two mice, with a series of coronal slices encompassing the primary motor cortex. Since only mouse 1 has domain labels, only the data from mouse 1 were used for benchmarking in this study. A total of 33 slices were examined from mouse 1, with approximately 2,000 to 7,700 cells per slice and 254 genes per slice. The dataset has been preprocessed, so no further normalization is required for the benchmarking process. This data is used for the evaluation of spatial

clustering.

STARMap dataset. This dataset includes spatial transcriptome data generated using spatially resolved transcript amplicon (STARMap) technology, which also achieves single-cell resolution but detects fewer genes. This dataset consists of a set of data that is publicly available in a STARMap technology publication. It contains three consecutive slices from the medial prefrontal cortex of a single mouse, each with domain annotations. The slices are designated BZ5 (1049 cells), BZ9 (1053 cells), and BZ14 (1088 cells), with 166 genes detected in each slice. The original count data was downloaded and needs to be normalized in the benchmarking process. Additionally, all 166 genes are treated as high-variation genes if high-variation genes need to be identified. This dataset is used for the evaluation of multi-slice integration, spatial clustering and spatial alignment.

BaristaSeq dataset. This dataset consists of spatial transcriptome data obtained using barcode in situ targeted sequencing (BaristaSeq). While it achieves single-cell resolution, the number of detectable genes is limited. The dataset is part of the SpaceTx project and includes three consecutive slices from the primary cortex region of a single mouse. The slices are labeled 1 (1525 cells), 2 (2042 cells), and 3 (1690 cells), with 79 genes detected in each slice. The data has been preprocessed, so no further normalization is needed for the benchmarking process. This dataset is used for the evaluation of multi-slice integration, spatial clustering and spatial alignment.

TNBC dataset. The TNBC dataset is based on multiplexed ion beam imaging by time-of-flight (MIBI-TOF) technology and was derived from a study of triple-negative breast cancer (TNBC), examining protein expression abundance in pathological sections from 41 patients. One slice was obtained from each patient, and the patients were categorized into three groups based on spatial organization: cold, mixed, and compartmentalized. After filtering, 40 slices were retained for analysis. Each slice contains between 2,000 and 8,000 cells, with 36 proteins profiled per slice. The data has been preprocessed, so no further normalization is needed for the benchmarking process. This dataset is used for the evaluation of slice representation.

Note S3 Benchmarking method:

In this benchmark study, we evaluated twelve recently published methods for multi-slice integration. Each method was executed according to the code provided in the original studies, and any encountered issues were corrected properly. The following section provides detailed information on the specific parameters and hyperparameter settings used for each method.

Banksy. Banksy represents each cell by incorporating both its own expression and the features of its surrounding neighbors, using spatial kernels to integrate transcriptomic and spatial information for more precise cell type or domain identification. The method incorporates the Harmony algorithm to integrate embeddings from multiple slices, followed by the Leiden clustering algorithm to identify cell types or domains. A key hyperparameter,

λ , controls the relative importance of a cell's intrinsic features and its neighbors in the embedding space. The λ value is chosen based on the downstream task, whether for cell type or domain identification. Another hyperparameter, K_{geom} , influences the selection of k-nearest neighbors during neighbor information retrieval. For 10X Visium data, we set $\lambda=0.2$ and $K_{\text{geom}}=18$, while for image-based data like MERFISH, we set $\lambda=0.8$, $K_{\text{geom1}}=15$, and $K_{\text{geom2}}=30$. Data was normalized using the Seurat package as recommended in the original study, and an additional resolution search module was used to identify the optimal resolution for accurate domain detection.

CellCharter. CellCharter integrates expression profile data across multiple slices by using variational autoencoders (VAEs) to obtain new cell representations. The model then embeds each cell based on both its own representation and the representations of its neighboring cells, thereby incorporating spatial information. A Gaussian mixture model is used for clustering cells to identify domains. Key parameters include *hvg*s (highly variable genes), *nlayes* (number of neural network layers), and *nhood_dists* (neighborhood distances). As recommended by the original paper, a grid search strategy was used to determine the optimal hyperparameters for each data. Additionally, data normalization and identification of highly variable genes were carried out using the Scanpy package, consistent with the original approach.

CN. The CN method features a straightforward design that defines a "window" as a set of *k* neighboring cells. The composition of different cell types within each window is used to characterize the window, and the MiniBatchK-Means method is employed to cluster these windows, identifying cell domains. The primary hyperparameter, *k*, determines the number of neighboring cells in each window. In this study, we set *k*=10, as recommended in the original paper. Additionally, since this method requires cell type information, if such information was unavailable, the data was normalized using the Scanpy pipeline. Cells were then clustered with the `sc.tl.leiden` function with *resolution*=2 and the resulting clusters were used as the cell type labels.

GraphST and GraphST-PASTE. GraphST uses graph neural networks and self-supervised contrastive learning to integrate spatial and expression data while removing batch effects across slices. The method employs domain identification algorithms such as *mclust*, Leiden, and Louvain for cell clustering. GraphST-PASTE, an extension of GraphST, first uses PASTE to align the slices before running GraphST, improving integration performance. Both methods followed the original code, with *mclust* used for clustering. Data normalization and identification of highly variable genes were performed using the Scanpy package. Moreover, a resolution search module was also implemented to ensure that detected domains align with the underlying data.

MENDER. Like CN, MENDER characterizes cells by considering the composition of neighboring cells across multiple ranges, aiming to integrate expression and transcriptomic information. Additionally, MENDER uses the Harmony algorithm to address batch effects and the Leiden algorithm to cluster cells in the absence of cell type labels. Key parameters

include the scale, which defines the number of ranges used in constructing multi-range neighborhoods, and nn or radius, which control the size of each range. The original article notes that data generated by different sequencing technologies require different hyperparameter settings. Therefore, sequencing technology was used as a hyperparameter in the modified model, which then applies the recommended settings for each technology. Additionally, data was normalized using the Scanpy package, in accordance with the original recommendations.

NicheCompass. NicheCompass is a graph deep learning algorithm that integrates inter-cell and intra-cell interactions as prior knowledge to create effective cell representations. The Leiden algorithm is used for domain identification based on these representations. The original code was followed, and data was normalized using the Scanpy package. Moreover, a resolution search module was employed to ensure that the detected domains align with the true data. In addition, the original paper recommends combining spot deconvolution with the analysis of spot-level data, such as 10X Visium, to improve the accuracy. Since our study does not incorporate a spot deconvolution module, NicheCompass was excluded from the evaluation on the 10X Visium dataset to ensure fair and accurate comparisons.

PRECAST. PRECAST integrates multiple spatial transcriptomics slides directly from normalized gene expression matrices, bypassing traditional multi-step batch correction pipelines based on spatial principal components. It factorizes each expression matrix into a latent embedding with a shared distribution across cell or domain clusters, while simultaneously achieving spatial dimension reduction, clustering, and cross-slide alignment. The code, including data normalization, integration and spatial clustering, followed the code in the original paper.

SpaDo. SpaDo is a multi-slice integration method that characterizes cells using neighborhood information. It involves cell type annotation, the calculation of Spatially Adjacent Cell type Embedding (SPACE), Jensen-Shannon divergence (JSD)-based hierarchical clustering, and multi-slice spatial domain analysis. The method uses different schemes for cell annotation and characterization at the single-cell and spot resolution levels. Due to the requirement for reference datasets for cell deconvolution at spot resolution, we analyzed all data at the single-cell resolution in this study. The method was executed according to the original code, with data normalization conducted using the Seurat package. Additionally, since this method requires cell type information, if cell type annotations were unavailable, the Seurat pipeline was used to normalize the data, and FindClusters with resolution=2 was employed to cluster cells. These clusters were then used as cell type labels.

SPIRAL. SPIRAL consists of two modules: SPIRAL-integration for multi-slice data integration and SPIRAL-alignment for spatial alignment across slices. In this study, we evaluate the SPIRAL-integration module. This module integrates data across slices by combining GraphSAGE with a domain adaptation network to correct batch effects and unify biological signals. The Louvain algorithm is employed for spatial clustering. As in the

original implementation, data normalization and identification of highly variable genes were performed using the Scanpy package. Additionally, a resolution search module was incorporated to ensure that the identified domains reflect the underlying data structure.

STAIG. STAIG integrates gene expression, spatial coordinates, and histological images by combining graph-contrastive learning with the self-supervised Bootstrap Your Own Latent (BYOL) framework, thereby enhancing multi-slice integration through image-guided alignment. A variant of STAIG without histological images is also available. In this study, since some datasets lacked histological images, the image-free version was used consistently. The mclust algorithm was applied for spatial clustering. Following the original implementation, data normalization and identification of highly variable genes were performed using the Scanpy package. Additionally, a resolution search module was incorporated to ensure that the identified domains align with the underlying data structure.

STAligner. STAligner employs a graph attention autoencoder to embed spots or cells based on gene expression and spatial coordinates. It then identifies spot triplets—groups of similar and dissimilar spots across slices—to guide the integration process by aligning biologically comparable regions. The mclust algorithm is used for spatial clustering. In accordance with the original implementation, data normalization and identification of highly variable genes were carried out using the Scanpy package.

The following table summarizes the benchmark tasks participated in by each method:

Table S1: Summary of benchmark tasks participated in by each method.

Methods	Multi-slice integration	Spatial clustering	Spatial alignment	Slice representation
Banksy	All datasets	All datasets	All alignment datasets	TNBC dataset
CellCharter	All datasets	All datasets	All alignment datasets	TNBC dataset
CN	Excluded due to lack of integration output.	All datasets	All alignment datasets	TNBC dataset
GraphST	Not run on the Large-scale dataset due to memory limitations.	Not run on the Large-scale dataset due to memory limitations.	All alignment datasets	Not run due to memory limitations.
GraphST-PASTE	Not run on the Large-scale dataset due to memory limitations.	Not run on the Large-scale dataset due to memory limitations.	Excluded due to use of a built-in alignment tool.	Not run due to memory limitations.

MENDER	All datasets	All datasets	All alignment datasets	TNBC dataset
NicheCompass	Not run on 10X Visium dataset datasets due to model incompatibility.	Not run on 10X Visium dataset datasets due to model incompatibility.	Not run on 10X Visium dataset datasets due to model incompatibility.	TNBC dataset
PRECAST	All datasets	All datasets	All alignment datasets	TNBC dataset
SpaDo	Not run on the Large-scale dataset due to memory limitations.	Not run on the Large-scale dataset due to memory limitations.	All alignment datasets	Not run due to memory limitations.
SPIRAL	Not run on the Large-scale dataset due to memory limitations.	Not run on the Large-scale dataset due to memory limitations.	All alignment datasets	Not run due to memory limitations.
STAIG	Not run on the Large-scale dataset due to memory limitations.	Not run on the Large-scale dataset due to memory limitations.	All alignment datasets	Not run due to memory limitations.
STAligner	Not run on the Large-scale dataset due to memory limitations.	Not run on the Large-scale dataset due to memory limitations.	All alignment datasets	Not run due to memory limitations.

Note S4 Resolution searching module:

In methods like Banksy and NicheCompass, the number of identified domains is influenced by the cluster resolution. To ensure consistency across different methods, a resolution searching module has been integrated into these methods. This module is designed to determine the optimal resolution based on the test data, following the approach used in MENDER. Specifically, the resolution range is set between 0 and 5. And the module employs a dichotic search algorithm, which iteratively identifies the optimal resolution setting.

Note S5 Domain matching during plotting and landmark domain choosing:

To visualize the results of spatial clustering and spatial alignment, we plotted the spatial distribution and 3D stack maps of cells, comparing both the ground truth and the outcomes of various methods. To ensure consistency and clarity, a uniform color scheme was applied to represent the ground truth and the results of each method. Each cell was assigned a

color based on its domain type. However, because the domain results identified by different methods do not directly align with the ground truth, we implemented a domain matching module before plotting to establish a correspondence between the predicted results and the ground truth.

Specifically, for each method, we first constructed a confusion matrix between the ground truth and the method's identified domains. Starting with the predicted domain that had the largest number, we identified the most corresponding ground truth domain and assigned it as the label for that predicted domain. We then removed the corresponding records from the confusion matrix and continued the matching process for the remaining predicted domains in a sequential manner. This approach ensures that the domains identified by the ground truth and the methods are consistently labeled, facilitating clear visualization. However, in some cases, the most ground truth domains corresponding to some predicted domains may have been assigned to other domains, leading to the assignment of grey colors for such domains in the plots.

Additionally, when performing coordinate correction based on the landmark domain, it is essential that the same domain be selected as the landmark across all methods. The results of the domain matching process are therefore useful in selecting a consistent landmark domain for this task.

It is important to note that the domain matching process is performed solely for visualization purposes and to aid in the selection of a unified landmark domain. The initial recognition outcomes are used to calculate performance metrics related to domain identification and spatial coordinate correction for each method.