Genome Biology

METHOD

Open Access

Batch correcting single-cell spatial transcriptomics count data with Crescendo improves visualization and detection of spatial gene patterns

Nghia Millard[1,2,3,4,5,7,8], Jonathan H. Chen[7,8,9,10], Mukta G. Palshikar[2,3,7], Karin Pelka[8,9,12], Maxwell Spurrell[8,9,10], Colles Price[11], Jiang He[11], Nir Hacohen[6,7,8,9], Soumya Raychaudhuri[1,2,3,4,5,7,8*†] and Ilya Korsunsky[2,3,7*†]

†Soumya Raychaudhuri and Ilya Korsunsky are co-senior authors.

*Correspondence: soumya@broadinstitute.org; ikorsunsky@bwh.harvard.edu

[1] Division of Rheumatology, Inflammation and Immunity, Brigham and Women's Hospital, Boston, MA, USA
[2] Division of Genetics, Brigham and Women's Hospital, Boston, MA, USA
Full list of author information is available at the end of the article

Abstract

Spatial transcriptomics facilitates gene expression analysis of cells in their spatial anatomical context. Batch effects hinder visualization of gene spatial patterns across samples. We present the Crescendo algorithm to correct for batch effects at the gene expression level and enable accurate visualization of gene expression patterns across multiple samples. We show Crescendo's utility and scalability across three datasets ranging from 170,000 to 7 million single cells across spatial and single-cell RNA sequencing technologies. By correcting for batch effects, Crescendo enhances spatial transcriptomics analyses to detect gene colocalization and ligand-receptor interactions and enables cross-technology information transfer.

Keywords: Single-cell, Spatial transcriptomics, Batch correction, Crescendo, Patterns, Ligand-receptor interactions

Background

High dimensional single-cell technologies [1–3] enable the discovery and characterization of cellular heterogeneity and potential function of important cell states [4–8]. Data from single-cell RNA sequencing (scRNA-seq) and recently emerging spatial transcriptomics platforms that enable spatially resolved single-cell transcriptional profiling [9–13] can be used to examine expression patterns of individual genes. Identifying key genes is an essential part of defining cellular functions, building regulatory networks, and understanding cell–cell interactions [14–18]. In non-spatial scRNA-seq, it is common to visualize clusters of cells before quantitative data analysis using a uniform manifold approximation and projection (UMAP) [19], upon which we can overlay gene expression to identify cell-type-specific patterns. With spatial transcriptomics, we can go a step further and observe the expression of key genes in individual cells in the

context of their real physical location. Furthermore, spatial gene expression can be used to identify potential cell–cell communication via consistent colocalization of two genes with ligand-receptor analysis [16–18, 20–23]. However, understanding true spatial gene expression patterns is difficult because the measurements of many genes are sparse or not captured at sufficient levels [24, 25], exhibit systematic batch effects across samples [26–28], and can be expressed in a cell type that does not group together in physical space which makes effective visualization challenging [29–31].

We and others previously showed that for studies and datasets containing multiple batches or samples, it is extremely important to perform batch correction to correctly identify and profile cell types/states (and in some cases, rare cell types). However, single-cell batch correction algorithms such as Harmony [32], scVI [33], Seurat anchor integration [34], and mutual nearest-neighbors (MNN) [35] operate on a lower-dimensional representation of gene expression, rather than directly correcting the genes themselves. To facilitate the visualization of gene expression and identification of spatial gene patterns across batches, it is crucial to remove batch effects and provide a way to impute sparse or poorly captured gene expression across batches for individual genes. To our knowledge, only the bulk RNA-seq algorithm ComBat-Seq [36] is explicitly designed to batch correct individual gene counts, and no existing method does both batch correction and imputation.

Here, we present Crescendo, a novel solution that uses generalized linear mixed modeling to perform single-cell batch correction of gene counts. Crescendo is designed to work directly on count data and simultaneously corrects systematic batch variation across datasets and imputes low-expressed gene counts that result from technical confounders. In this manuscript, we focused on gene correction in the context of spatial transcriptomics, where it is critical to observe the expression of key genes in spatially defined individual cells, rather than in clusters of cells. First, we showed that Crescendo batch correction facilitates the tracking of 3-dimensional gene expression in spatial transcriptomics data containing three serial sections of a mouse brain [37]. To showcase Crescendo's scalability, we then performed temporal computational benchmarks on a 16-sample, 7-million-cell immuno-oncology spatial transcriptomics dataset [38]. Then, in a more challenging scenario, we demonstrated that Crescendo helps batch correct across technologies by integrating a scRNA-seq colorectal cancer (CRC) dataset [39, 40] with CRC spatial transcriptomics samples [41]. Finally, in proof-of-principle analyses, we illustrated that batch corrected gene expression enables the detection of spatial ligand-receptor interactions that were obscured by batch effects.

Results

Crescendo corrects batch variation in gene expression across datasets

Here, we showcase Crescendo in the context of spatial transcriptomics data. Crescendo is an extension of the Harmony algorithm, which removes batch effects in a lower-dimensional representation of data (Fig. 1A), such as principal components from principal components analysis (PCA). After Harmony fits linear models to PCA embeddings, Crescendo fits generalized linear models to gene expression counts (Fig. 1B, " Methods"). Both Harmony and Crescendo assume that batch effects are cell-type-specific. The result of Crescendo is batch corrected gene counts that can

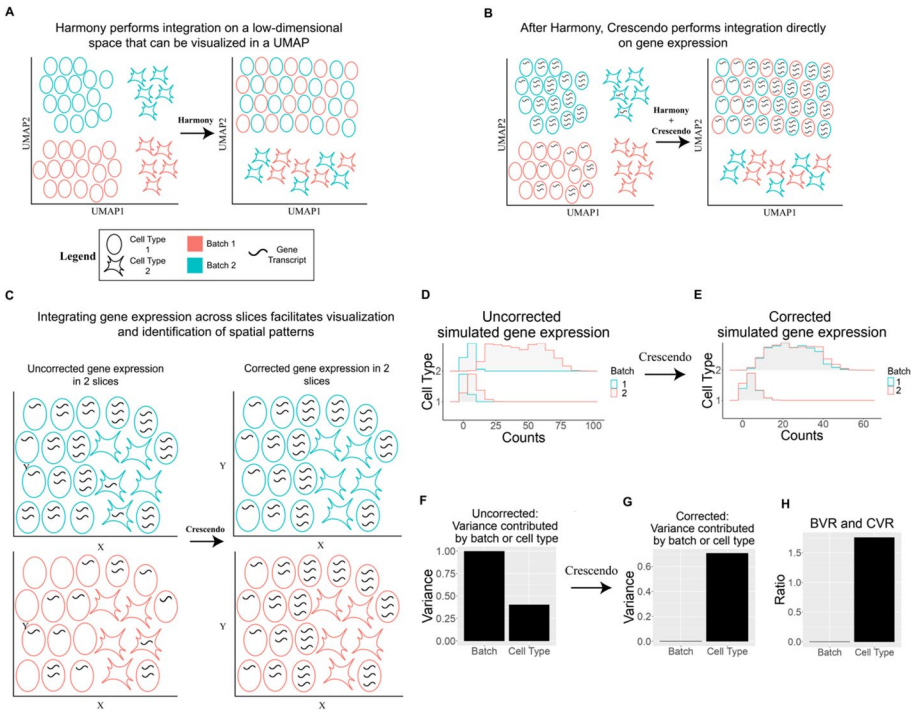

Fig. 1 Crescendo directly corrects gene expression. **A** Harmony batch corrects lower-dimensional embeddings like principal components that are visualized with a 2-dimensional UMAP. **B** Crescendo extends Harmony to batch correct genes expression, which can similarly be visualized in a UMAP. **C** Spatial transcriptomics allows for visualization of gene expression in the context of cellular locations. Due to batch effects, gene expression can be poorly expressed and spatial patterns can be obscured. Crescendo infers the gene expression of a cell, which facilitates the visualization and spatial pattern recognition of gene expression. Representative distributions of simulated gene expression before (**D**) and after (**E**) Crescendo batch correction. Batch-associated and cell-type-associated variance metrics before (**F**) batch correction and after (**G**) batch correction. **H** Calculated batch-variance ratio and cell-type-variance ratio metrics based on **F**–**G**

facilitate visualization of a gene across batches; in some cases, this may improve the ability to visualize and detect gene spatial patterns in a sample (Fig. 1C). Importantly, Crescendo preserves counts in the output expression matrix, making the final output amenable to count-based downstream analyses, such as visualization, differential expression, and spatial pattern analyses.

The inputs for Crescendo are a gene by counts matrix, cell-type information, and batch information; the output is a batch corrected gene by counts matrix. To facilitate scalability, we allow users to first perform a biased downsampling to reduce the number of cells while accounting for rare cell states and batches; this is used for model fitting, but we still perform batch correction on all cells (Additional file 1: Fig. S1A, " Methods"). After downsampling, we perform an estimation step in which we model how much variation in a gene's expression derives from intrinsic biological sources (such as cell-type identity) and confounding technical sources (batch effects such as sample or technology). We then perform a marginalization step, in which we use the model from the estimation step to infer a batch-free model of gene expression. Finally, we perform a matching step by using the original estimated model and the marginalized batch-free model to sample batch corrected counts (" Methods"). For lowly expressed genes or those assayed

with lower sensitivity, Crescendo can model gene expression assuming higher total read counts to perform imputation (" Methods").

Benchmarking gene-level batch correction with batch and cell-type variation metrics

Effective batch correction of gene expression must meet two objectives: (1) remove differences between cells that are driven by technical factors such as batch or technology and (2) preserve the biologically meaningful differences in gene expression, especially among cell types. Currently, there are limited metrics to evaluate an algorithm's performance in reducing batch variation while preserving biological variation. To evaluate Crescendo, we developed two metrics to quantify the performance of gene expression batch correction: the batch-variance ratio (BVR) and cell-type-variance ratio (CVR). The first metric quantifies batch effect removal as the ratio of batch-related variance in gene expression before versus after correction. Similarly, the second metric quantifies the preservation of cell-type-related differences as the ratio of cell-type-related variance in gene expression before versus after correction.

BVR and CVR are calculated based on counts; in brief, we fit generalized linear models in which we fit a gene's counts with random effects for batch and user-defined cell-type identity (" Methods"). For each gene, we fit this model on both the uncorrected and corrected data. To obtain the BVR, we calculate the ratio of the batch-related variances between these fitted models; similarly, we calculate the CVR from the cell-type-related variances from these models. Ideally, batch correction will decrease variance associated with batch, which lowers the post-correction batch variance to give a $BVR < 1$. Furthermore, we ideally want to maintain or increase cell-type variance after correction, which would give a $CVR \geq 1$; empirical observations from real data suggest that a $CVR \geq 0.5$ is generally good preservation of cell-type variability. We also note that if batch variance is initially low, batch correction may not be necessary.

To demonstrate these metrics, we show example genes that exhibit high or low BVRs/CVRs after we performed gene expression batch correction on 3 samples from a Vizgen mouse brain receptor map dataset [37] with both Crescendo (Additional file 1: Fig. S1B–C) and Seurat anchor integration [34] (Additional file 1: Fig. S1D–E). We then applied Crescendo on simulated gene expression data. To simulate a gene count distribution, we first simulated cells from different batches and cell types. We then simulated batch-specific and cell-type-specific gene expression rates to parameterize a Poisson distribution from which we sampled gene counts for each cell (Fig. 1D–E, " Methods"). For this representative gene, we performed Crescendo batch correction and calculated the BVR and CVR metrics (Fig. 1F–H). Over 10,000 gene simulations, we observed that Crescendo dramatically decreased batch effects in 100% of the simulated genes, with 98.64% of those genes also exhibiting $CVR \geq 0.5$ (Additional file 1: Fig. S1F).

Crescendo corrects batch effects across serial sections in whole mouse brain

We then designed an analysis to demonstrate the practical utility of Crescendo to correct batch-affected gene expression, and by doing so, improve visualization of gene expression in space. Batch correction can enable identification of gene expression patterns that were previously obscured by technical variation. We used a public spatial transcriptomics dataset of the mouse brain profiled by the Vizgen MERSCOPE platform

[37]. We performed a standard scRNA-seq analysis pipeline [42] to analyze and cluster three serial coronal slices (S3R1, S3R2, S3R3) from the same mouse brain that represent batches; in aggregate, this data contains in situ expression for 483 genes in 179,385 segmented cells (Fig. 2A). This dataset features inhibitory and excitatory neuronal subtypes, along with astrocytes, microglia, oligodendrocyte progenitor cells (OPCs), and endothelial cells (Fig. 2B). Batch effects were variable, with certain cell types (e.g., inhibitory and excitatory neuronal subtypes) exhibiting greater levels of batch effect than others (Additional file 1: Fig. S2A). In physical space, neurons tended to be well-organized, while cell types such as astrocytes and microglia were dispersed across the sections (Fig. 2C, Additional file 1: Fig. S2B).

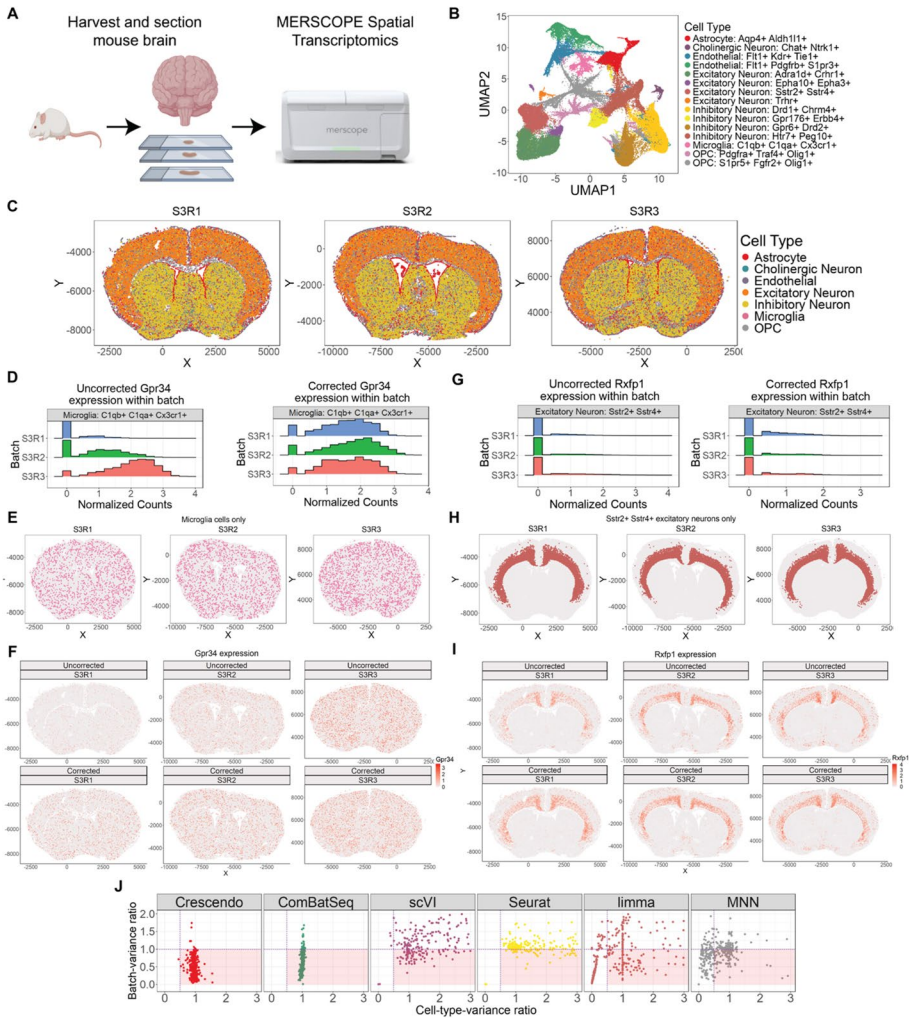

Fig. 2 Crescendo facilitates visualization of genes across spatial transcriptomics datasets of serial sections from whole mouse brain tissue. **A** The Vizgen MERSCOPE platform was used to assay three coronal mouse brain tissue slices [37]. **B** Cell state classifications of cells based on marker genes. **C** Spatial locations of broad cell types. Gene expression distributions across slices for *Gpr34* (**D**) and *Rxfp1* (**G**). Spatial locations of cell types with the highest expression *Gpr34* (**E**) and *Rxfp1* (**H**). Gene expression visualizations in physical space before and after Crescendo batch correction for *Gpr34* (**F**) and *Rxfp1* (**I**). **J** Scatter plots of batch-variance ratio (BVR) and cell-type-variance ratio (CVR) metrics calculated for all 483 genes across 5 different batch correction algorithms. Purple dashed vertical line is at CVR = 0.5 and the purple dashed horizontal line is at BVR = 1. Red at BVR < 1 and CVR ≥ 0.5 is the target zone for genes that were batch corrected well

Because these slices represent a z-stack of serial sections in a similar area of the brain, we expected genes to be expressed at consistent levels across the slices. However, we observed that several genes exhibited noticeable batch-related variance, though it tended to be smaller in magnitude compared to cell-type variance (Additional file 1: Fig. S3A). To begin, we analyzed the effect of Crescendo on three genes that were cell-type specific: *Gpr34* in microglia cells [43, 44], *Rxfp1* [45, 46] in cortical excitatory neurons, and *Epha8* [47] in striatal inhibitory neurons. Each of these genes was subject to batch effects (Fig. 2D, G, Additional file 1: Fig. S4A–E). For each gene, we show that batch correction improves visualization for a gene by making expression more consistent across batches.

We first looked at the gene *Gpr34*, which is predominantly expressed by microglia, a cell type that clustered together tightly in the UMAP (Fig. 2B, Additional file 1: Fig. S4A). However, in physical space, both microglia (Fig. 2E) and the expression of *Gpr34* (Fig. 2F) are spread out across the slices, making visualization challenging. This visualization is even worse in slice S3R1, which has overall lower expression (Fig. 2D, F). After using Crescendo to batch correct *Gpr34* expression, we observed noticeably higher expression of *Gpr34* in S3R1 at levels relative to the other two slices, and more even expression across all slices (Fig. 2D, F).

We next looked at the gene *Rxfp1*, which is predominantly expressed by *Sstr2*+*Sstr4*+excitatory neurons (Fig. 2B, Additional file 1: Fig. S4B). Here, we observed *Rxfp1* expression at similar maximal levels across all slices but noticed that many *Sstr2*+*Sstr4*+excitatory neurons in S3R1 had noticeably lower levels of *Rxfp1* expression (Fig. 2G–I). In physical space, we observed that *Sstr2*+*Sstr4*+excitatory neurons tended to cluster in specific layers of the cortex (Fig. 2H), but visualization of *Rxfp1* expression showed that expression was not consistent across these neurons in the same tissue (Fig. 2I). Again, after using Crescendo to batch correct *Rxfp1* expression, we observed more even expression across all slices while importantly not increasing expression in other cell types such as excitatory neurons in the other cortical layers (Fig. 2G, I, Additional file 1: Fig. S5A–B).

Finally, we looked at the gene *Epha8*, which is predominantly expressed by some inhibitory neuron states (Fig. 2B, Additional file 1: Fig. S4C). *Epha8* expression was also subject to batch effects, with low expression in slice S3R3 (Additional file 1: Fig. S4C–E). After batch correction with Crescendo, we were again able to observe relatively even *Epha8* expression across all slices (Additional file 1: Fig. S4D–E).

Our analyses so far demonstrate the utility of Crescendo to improve gene visualization by ameliorating batch effects in three genes. We next quantified how well Crescendo removes batch effects while retaining biological variation in all 483 genes in the MERFISH panel. We applied Crescendo to each gene and calculated the BVR and CVR metrics (Fig. 2J). Of the 483 genes, Crescendo produced a BVR < 1, CVR ≥ 0.5 in 408 genes. We next compared Crescendo's ability to batch correct individual genes in this mouse brain spatial dataset to five representative state-of-the-art algorithms: ComBat-Seq, scVI, Seurat anchor integration, MNN, and limma (" Methods"). With the caveat that these methods were not explicitly intended to be used for this purpose, we observed that scVI, Seurat, MNN, and limma struggled to remove batch variation from gene expression (sometimes even increasing of batch variation) or resulted in a dramatic decrease

in biological cell-type variation (Fig. 2J). Their poor performance is likely due to their assumptions of Gaussian structure in data rather than the count-based structure of gene expression. Of the 483 genes, ComBat-Seq, scVI Seurat, MNN, and limma produced a BVR < 1, CVR ≥ 0.5 in 364, 95, 142, 160, and 104 genes, respectively, compared to Crescendo's 408 genes. Head-to-head comparisons of algorithms using BVR/CVR showed that Crescendo outperformed alternative algorithms by consistently decreasing the most batch variation while preserving biological variation (Additional file 1: Fig. S6A–B). Highly variable gene conservation [48] metrics showed that Crescendo and ComBat-Seq outperform the alternative algorithms (Additional file 1: Table S1A, " Methods").

Crescendo scales efficiently to millions of cells

Single-cell datasets are increasing in size, with experiments regularly profiling 100,000 + cells per experiment, the creation of large single-cell atlases on the order of millions of cells, and spatial transcriptomics experiments potentially profiling 100,000 + cells per slice [49, 50]. This leap in data size makes computational efficiency critical for gene-level batch correction. We tested the ability of Crescendo and other methods to scale to both many cells and many batches by using an Immuno-oncology FFPE dataset produced by the Vizgen MERSCOPE platform [38]. The Immuno-oncology dataset features a custom 500-gene panel designed to profile immune, stromal, and malignant cells across 9 different tissue types. This collection contains 7,020,548 post-QC cells across 16 individual slices spanning 8 tissue types (" Methods").

We first attempted to batch correct all 500 genes with each method (Additional file 1: Fig. S7A). ComBat-Seq, Seurat, and MNN failed to complete due to memory requirements while limma took 6.6 h. Unlike alternative methods, Crescendo fits and batch corrects a gene independent of others, which allows users to fit genes individually and reduces the risk of running into memory complications. Overall, Crescendo performed best by batch correcting all 500 genes in 3.3 h. For the sake of comparison, we also summed processing time across genes to simulate downsampling each gene individually (" Methods"), which took 6.1 h.

In contrast with the alternative methods that use the information from all genes (which means removing a gene changes results), Crescendo allows users to batch correct genes independently and prioritize specific genes of interest. To better understand the scaling behavior of Crescendo based on the number of genes and cells corrected, we repeatedly batch corrected random samples of genes in increasingly larger subsamples of the 7 million cells. In each run, we batch corrected 1, 2, 5, 10, or 50 random genes 100 different times for each subsample of 10K, 25K, 100K, 250K, 500K, 750K, 1M, 2M, 3M, 4M, 5M, 6M, or all 7M cells (" Methods"). Crescendo was able to consistently correct 50 genes across 7 million cells in less than 7 min (Additional file 1: Fig. S7B). For each run, we also isolated the amount of time Crescendo takes to perform the downsampling, estimation, marginalization, and matching steps (Additional file 1: Fig. S7C). Computational runtime for the downsampling step was dependent on the number of cells while runtime for the estimation step scaled relatively linearly based on the number of genes being corrected. The marginalization and matching steps also depended on the number of genes but scaled slightly better than linearly.

We also performed singular runs in which we batch corrected all 500 genes at once for each dataset size. For all 500 genes at once, Crescendo was able to process a 3 million-cell dataset in 1310 s (30 m; Additional file 1: Fig. S7D); the estimation step ran into memory issues when fitting GLMs on dataset sizes of greater than 4 million.

Crescendo corrects technology effects by integrating paired colorectal cancer scRNA-seq and spatial transcriptomics datasets

We hypothesized that Crescendo could be used to integrate and impute gene expression between non-spatial technologies, which captures the full transcriptome, and spatial technologies, which gives physical locations of transcripts. To demonstrate this, we used Crescendo to correct technology batch effects by integrating two human colorectal cancer (CRC) spatial transcriptomics slices with a CRC 10X scRNA-seq dataset [39–41]. The scRNA-seq dataset contains 69,153 cells from 29 CRC tissue samples while the two spatial transcriptomics slices (PFA_A6 and PFA_A11) were both generated from the same CRC tissue sample taken from a donor in the scRNA-seq dataset (Fig. 3A, " Methods"). The two technologies share 477 common genes, reflecting the smaller gene panel in the spatial transcriptomics datasets. Integrating these two technologies represents a more challenging scenario because the technology batch effects are noticeably larger than the batch effects between the spatial transcriptomics samples (Fig. 3B, " Methods"). Human CRC tissue also contains much less organization than the highly structured quality of the brain, further complicating gene visualization. Moreover, cells in primary human tissue tend to be packed close to each other, making overplotting even more problematic. To address this, we plotted gene expression similar to the mouse brain section by plotting gene-expressing cells over non-expressing cells to represent a best-case scenario of visualizing gene expression across slices.

Similar to the author-defined cell types in the scRNA-seq data, we identified epithelial cancer cells, fibroblasts, endothelial cells, T cells, B cells, plasma cells, and myeloid cells in the spatial transcriptomics slices (Fig. 3C; " Methods"). In physical space, all cell types were relatively spread out, though some cell types such as epithelial cells and fibroblasts occasionally formed small aggregates.

After performing batch correction with Harmony (Fig. 3B, " Methods"), we compared gene expression between the spatial slices and the scRNA-seq data; we found that several genes were expressed in most cells of a cell type in the scRNA-seq data but not well-expressed in that same cell type in the spatial transcriptomics slices. For instance, the gene *MS4A1* (*CD20*) is a marker for B cells and was well-expressed in the scRNA-seq data but was not expressed at high levels in the spatial slices (Fig. 3D). After Crescendo batch correction, we observed increased expression of *MS4A1* in the spatial slices on a level more similar to the scRNA-seq data; this also provided easier visualization of *MS4A1* expression in the spatial slices that was consistent with the locations of B cells (Fig. 3D–F). We observed similar trends for the T-cell-specific gene *CD3D*. Unlike *MS4A1*, expression of *CD3D* was visible in physical space, but the level of expression is still lower than scRNA-seq (Fig. 3I). After batch correction, we observed more even *CD3D* expression across all three datasets and strengthened *CD3D* expression, particularly in PFA_A6, in the spatial slices that was consistent with the locations of T cells (Fig. 3G–I).

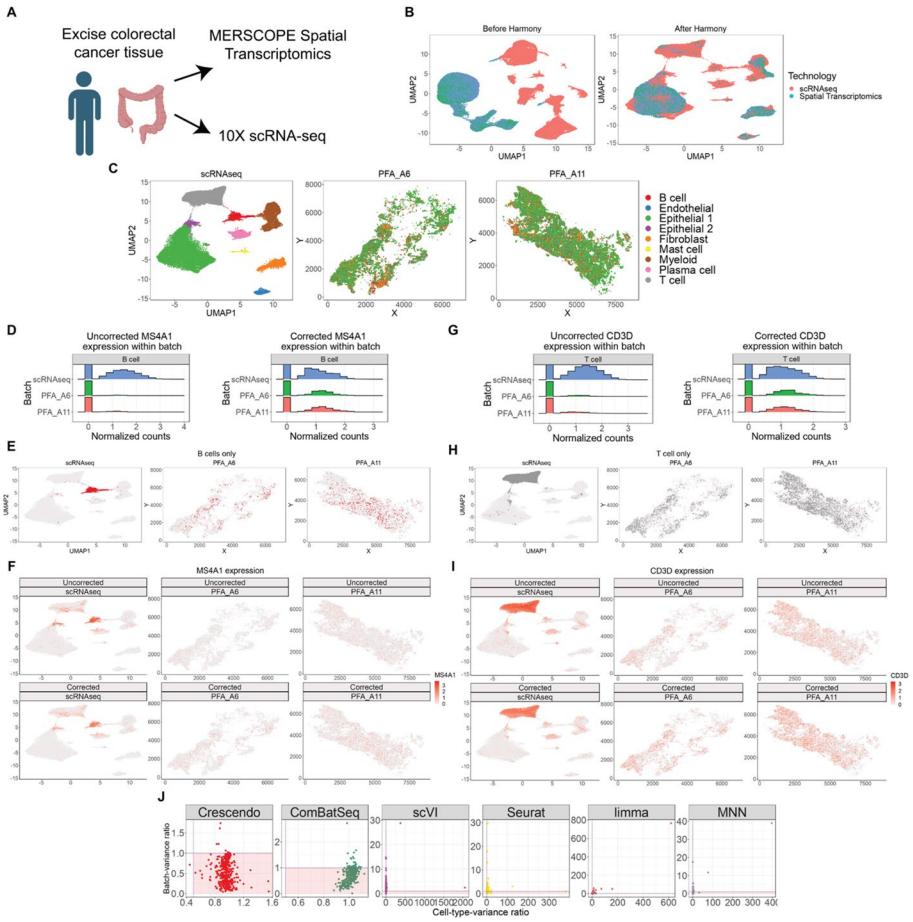

Fig. 3 Crescendo batch corrects technology effects between a colorectal cancer (CRC) scRNA-seq dataset and two CRC spatial transcriptomics samples. **A** Colorectal cancer samples were assayed with scRNA-seq and spatial transcriptomics. These datasets shared 477 genes. **B** UMAP embedding of cells from scRNA-seq and spatial transcriptomics before and after batch correction with Harmony (correction performed on a batch variable where the scRNA-seq dataset and each spatial slice was considered a batch). **C** Broad cell type classification of cells and spatial locations of cell types in spatial slices (middle, right). Gene expression distributions across slices for *MS4A1* (**D**) and *CD3D* (**G**). **E**, **H** In these and following plots, scRNA-seq is plotted in UMAP space, while spatial slices are plotted in physical space. Spatial locations of cell types with the highest expression of *MS4A1* (**E**) and *CD3D* (**H**). Gene expression visualizations in physical space before and after Crescendo batch correction for *MS4A1* (**F**) and *CD3D* (**I**). **J** Scatter plots of batch-variance ratio (BVR) and cell-type-variance ratio (CVR) metrics calculated for all 477 genes across 5 different batch correction algorithms. Purple dashed vertical line is at CVR = 0.5 and the purple dashed horizontal line is at BVR = 1. Red at BVR < 1 and CVR ≥ 0.5 is the target zone for genes that were batch corrected well

Subsequently, we performed batch correction and calculated the BVR and CVR metrics on all 477 genes in the CRC scRNA-seq and spatial datasets with Crescendo and the other 4 benchmarking methods (" Methods"). We observed that of the 477 genes, 439 exhibited a batch variance greater than 0.001 (Additional file 1: Fig. S3B, " Methods"). Of these 439 genes, Crescendo, ComBat-Seq, scVI, Seurat, MNN, and limma provided a BVR < 1, CVR ≥ 0.5 in 423, 372, 49, 78, 2, and 89 genes, respectively (Fig. 3J). We had to plot on different scales since the ranges of BVR and CVR varied widely by method, with Seurat having the notably poor maximum BVR of 30. Again, head-to-head comparisons showcased Crescendo's superior performance over alternative algorithms (Additional file 1: Fig. S8A–B), and highly variable gene

conservation showed that Crescendo and ComBat-Seq outperform the alternative algorithms (Additional file 1: Table S1B).

Batch correcting spatial transcriptomics gene expression facilitates the identification of spatial ligand-receptor interactions via gene–gene correlations

We next looked at the spatial patterns of gene expression in physical space. A powerful aspect of single-cell spatial transcriptomics is the ability to simultaneously look at gene expression of a cell in the context of its physical neighbors. This view lets us hypothesize about potential interactions between neighboring cells through gene–gene interactions, particularly ligand-receptor interactions. To evaluate the ability of Crescendo to inform and improve the power to detect gene–gene interactions, we analyzed the correlation between a cell's gene expression with that of its spatially neighboring cells.

Many investigators want to find spatial patterns between genes in specific cell types of interest. Thus, in these analyses, we calculated a spatial cross-correlation index (SCI) between genes in a cell-type-aware manner (" Methods"). Briefly, we subsetted cells within a slice to two cell types; for cell-type 1, we identified its nearest neighbors within a 30 μm Euclidean distance that were from cell-type 2 and calculated a weighted sum of nearest-neighbor gene expression to obtain a nearest-neighbor expression matrix. We repeated this procedure for all combinations of cell types in each slice independently. Two genes in two different cell types that share similar spatial expression patterns exhibit a positive CI, dissimilar patterns exhibit a negative SCI, and an SCI of zero indicates no consistent spatial pattern between the genes (Fig. 4A). SCI provides a way to quantitatively evaluate how batch correction affects the spatial patterns of gene expression.

Overall, we observed that for most gene–gene pairs in a cell-type pair, the SCI did not noticeably change after batch correction (Fig. 4B, Additional file 2, Additional file 3) in either slice. When classifying these gene–gene pairs based on literature-derived ligand-receptor pairs [17, 51, 52] (Fig. 4B, red points), we again observed minimal changes in SCI. However, we did observe several pairs that had a low uncorrected SCI change to a higher corrected SCI. We chose two such examples that are well-studied ligand-receptor pairs to observe how Crescendo batch correction affects both visualization and SCI.

First, we looked at how *JAG2* expression in endothelial cells and *NOTCH3* expression in fibroblasts formed spatial coherent patterns. Previous studies [53] show that *NOTCH3* signaling can drive transcriptional and spatial gradients in fibroblasts after interacting with Notch ligands, like Jagged-2, from vascular endothelial cells. In physical space, we observed areas of colocalization between endothelial cells and fibroblasts, and that the SCI for *JAG2* in endothelial cells and *NOTCH3* in fibroblasts was initially 0.276 in PFA_A6 and 0.361 in PFA_A11 (Fig. 4C). However, *JAG2* expression in some cells was difficult to visualize due to batch effects. After batch correction with Crescendo, we observed more visible expression of *JAG2* in endothelial cells in both slices, which made identification of colocalizing *JAG2*-expressing endothelial cells and *NOTCH3*-expressing fibroblasts easier (Fig. 4C). Statistically, the SCI increased to 0.324 in PFA_A6 and 0.379 in PFA_A11.

Next, we looked at *CCR3* expression in myeloid cells and expression of its ligand *CCL11* in fibroblasts, involved in the chemotaxis of leukocytes [54, 55] (Fig. 4D). Notably, we observed that *CCR3* was mainly expressed by a subset of myeloid cells, with

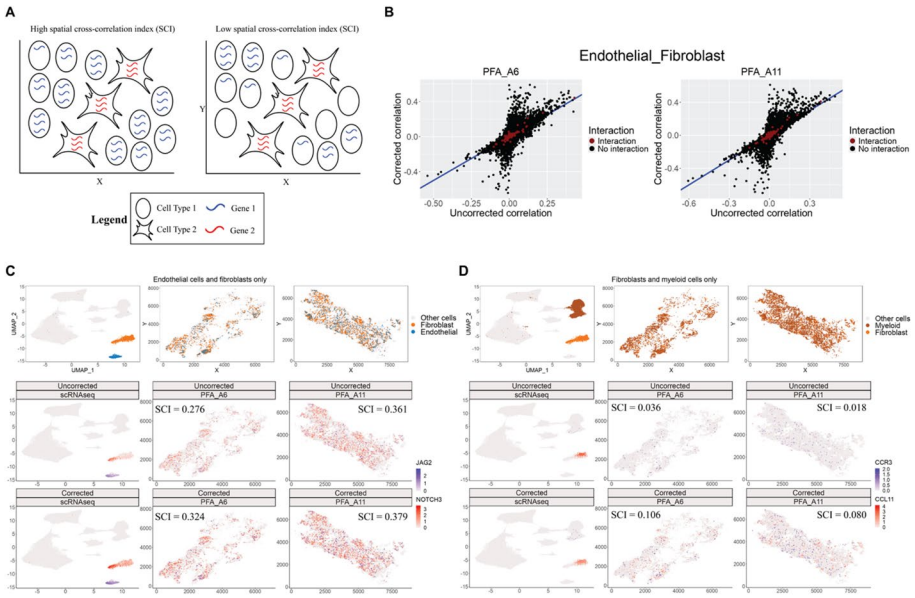

Fig. 4 Crescendo batch correction increases ability to visualize and detect spatial gene–gene correlations. **A** Example schematics of gene–gene pairs that have a high spatial cross-correlation index (SCI) and a low SCI. **B** Comparison of SCIs for all fibroblast and myeloid cell gene–gene pairs before batch correction vs. after batch correction. **C** In these and following plots, scRNA-seq is plotted in UMAP space, while spatial slices are plotted in physical space. Spatial locations of fibroblasts and endothelial cells (top). Gene expression visualization of *JAG2* in endothelial cells and *NOTCH3* in fibroblasts (bottom). SCIs are listed for each spatial sample before and after batch correction. **D** Spatial locations of myeloid cells and fibroblasts (top). Gene expression visualization of *CCR3* in myeloid cells and *CCL11* in fibroblasts (bottom). SCIs are listed for each spatial sample before and after batch correction

much better expression in the spatial transcriptomics slices. Conversely, *CCL11* expression was much higher in the scRNA-seq dataset. With batch-specific low expression of both genes, it was perhaps unsurprising that we saw low SCIs of 0.036 in PFA_A6 and 0.018 in PFA_A11, suggesting almost non-existent colocalization of these genes (Fig. 4D). However, after batch correction with Crescendo, visualization of this gene–gene pair showed a modest increase in *CCR3* expression in some myeloid cells and a dramatic increase in *CCL11* expression such that areas where these genes colocalize are now visible (Fig. 4D). Statistically, SCI noticeably increased to 0.106 in PFA_A6 and to 0.080 in PFA_A11. We note that the lower SCI values for this gene–gene pair is due to colocalization of these genes' expression being limited to certain areas of the tissue while the *JAG2-NOTCH3* pair was more ubiquitously expressed within the specified cell types. Overall, these results suggest that Crescendo can help recover spatial patterns that were previously obscured by batch effects.

We then looked at *COL1A2* expression in fibroblasts and *CXCL14* in myeloid cells, which have no previously known interactions. *CXCL14* expression was noticeably low in both the spatial datasets and the scRNA-seq while *COL1A2* was well-expressed primarily in the scRNA-seq dataset. With such low expression in both genes in the spatial datasets, the SCI was notably low in both slices: − 0.009 for PFA_A6 and − 0.004 for PFA_A11 (Additional file 1: Fig. S9A–B). After batch correction, we observed that *COL1A2* expression was noticeably increased in the spatial datasets but *CXCL14* was

still low; this resulted in a dramatically decreased SCI in both slices to − 0.455 in PFA_A6 and − 0.506 in PFA_A11 (Additional file 1: Fig. S9B).

Finally, we reasoned that if two cell types colocalize, then the SCI between their markers should be relatively high. Fibroblasts and T cells are abundant cell types in the spatial slices and visually appear close to each other in many areas (Additional file 1: Fig. S9C). However, the SCI between a pair of their markers, *FN1* in fibroblasts and *CD3E* in T cells, was relatively low at 0.024 in the first slice and 0.016 in the second (Additional file 1: Fig. S9D). Visualization of these marker genes showed that the low spatial cross-correlation is explained by the low expression of these genes. After batch correction with Crescendo, we observed much more visible expression of both *FN1* and *CD3E* in both slices (Additional file 1: Fig. S9D) with an accompanying increase in SCI to 0.248 in the first slice and to 0.290 in the second slice.

Discussion

Identifying genes or features of interest is an important aspect of generating hypotheses from single-cell data. In spatial transcriptomics data, visualizing a gene's spatial patterns can help infer the role of a gene in the function of a cell type and the localization of cell types to specific niches. Thus, it is important to batch correct and impute gene expression in order to accurately visualize it. Here, we introduced Crescendo, which accepts Harmony outputs and gene counts as input and returns batch corrected counts as output. We showed that Crescendo can remove batch effects from a vast majority of genes in spatial transcriptomics data, which facilitated better visualization of a gene's expression across batches and overall gene spatial patterns. Gene-level batch correction of spatial transcriptomics data is naturally visualized on tissue slices; however, while visualization offers a qualitative sense of results, we emphasize the need for quantitative metrics. Thus, we developed the BVR and CVR metrics to quantify the level of batch correction and biological conservation for gene-level batch correction procedures, which demonstrated that Crescendo outperforms alternative methods. We also developed SCI to quantify how spatial gene expression patterns change after correction.

Furthermore, Crescendo is scalable to millions of cells, which enables it to accommodate the large number of cells featured in modern single-cell spatial transcriptomic datasets [56–59] and single-cell atlases [7, 60, 61]. We showcased Crescendo's scalability by batch correcting genes in 7 million cells across 16 batches. We predict that spatial datasets will continually grow to incorporate more individuals and multiple samples from the same individual, thus making scalability even more important.

Batch correcting gene expression with Crescendo (or any correction algorithm) has notable fundamental technology-based limitations. The first is that we observed that a gene can be expressed at extremely low levels in all batches; if this is the case, then batch correction will not rescue its expression. Poor expression of certain genes like cytokines can be observed in many technologies [24, 25, 62–65], so investigators should consider how highly a gene is expressed before attempting to use batch correction to impute gene expression. This only occurred in select cases, while Crescendo was able to reduce batch variation in most genes. Another significant limitation for batch correcting gene expression in some spatial transcriptomics datasets is that fluorescence in situ hybridization (FISH)-based spatial datasets require cell segmentation, which is a significant challenge

[31, 66–68]. Inaccurate segmentation can erroneously assign certain transcripts to the wrong cell. Since the number of unique genes expressed and the transcripts per cell in spatial data tends to be significantly lower than scRNA-seq [25, 67], erroneous assignment of transcripts to a cell makes the data considerably noisier and batch correction of genes more difficult. Indeed, in all spatial datasets we showcased, we observed several instances of a cell type containing transcripts of markers for other cell types (e.g., a B cell marker in T cells). Theoretically, segmentation could cause systematic errors in transcript assignments; for example, if B cells and T cells tend to colocalize, there is a higher chance for their transcripts to be erroneously assigned among each other. If transcripts are systematically erroneously assigned, it is possible that batch correction may increase expression of an erroneous gene. We speculate that as segmentation performance increases, the effectiveness of batch correction should increase as well.

Algorithmic limitations of Crescendo include inability to impute missing genes from batches, potentially inaccurate parameter estimation of a gene's expression within a batch within a cell type deviates too far from a Poisson distribution, and sensitivity to small sample sizes. Crescendo's current implementation limits batch correction to genes expressed in all batches; prediction and imputation for missing genes in a batch (when it is present in others) will require a different algorithm such as a k-nearest neighbor-based averaging of gene expression. Due to Crescendo's parameter estimation relying on Poisson generalized linear mixed models (GLMMs), estimation may be inaccurate if a gene's expression exhibits significantly higher sparsity than expected by a Poisson model or high overdispersion—these Poisson GLMMs could theoretically be substituted for zero-inflated or negative binomial models, respectively. We note gene expression within a batch, within a sample, and within a cell type usually follows a Poisson distribution and that zero-inflation and overdispersion are usually observed due to heterogeneity between batches, samples, and cell types rather than underlying technical artifacts [69]. The usage of GLMMs also introduces potential sensitivity to small sample sizes. Crescendo fits GLMMs with regularization to stabilize parameter estimates, but if a cell type within a batch is very rare (< 10 cells), Crescendo may struggle to perform accurate estimation of a gene's expression.

Conclusion

Crescendo batch corrects gene expression to aid visualization of spatial gene expression patterns across batches and facilitate downstream analyses such as gene colocalization and ligand-receptor analysis. The Crescendo framework has other potential applications because it models counts, which are present in data generated from other technologies. In this manuscript, we focused on the batch correction of FISH-based spatial transcriptomics data; however, Crescendo is also compatible with spatial transcriptomics data generated from "spot"-based protocols (instead of cells containing counts, it would be spots containing counts). We caution users looking to perform batch correction from spot-based spatial transcriptomics data because spots can potentially contain transcripts from multiple cell types and confound biological signal. Further potential applications include batch correcting counts for single-cell ATAC-seq data [70–72] or genomic data. Batch correcting genomic counts may be useful for quantitative trait loci (QTL) analyses [73–75] if they are confounded by technical noise. Due to the visual benefits of batch

correcting gene counts to be more even across batches, we envision that investigators will utilize Crescendo to aid in gene visualization and hypothesis generation in scRNA-seq or spatial transcriptomics datasets.

Methods

Crescendo

Overview

The goal of Crescendo is to account for and remove the effects of technical covariates on gene counts, while keeping all other effects (e.g., cell type, cell cycle, and other latent biological processes) intact. To achieve this, we first use regression to fit the expression of each gene as a function of user-specified technical factors, latent biological factors, and cell-specific residuals. We then remove the modeled effects of technical factors and keep those of biological factors and residuals. In a linear model, this process is achieved by directly subtracting the fitted technical effects from the gene expression levels. In a count-based model, such as Poisson regression, this is not possible and requires more sophisticated mathematical methods. In the remaining sections, we build intuition and derive formulas for each component of the algorithm separately in "Removing the effects of covariates in count-based regression," "Predicting batch-free counts," and " Modeling the interaction between technical and biological effects" sections, show how these components piece together in "Putting it together: Crescendo algorithm," and finally describe our strategy to scale Crescendo to big data in " Scaling to large data with robust downsampling." The Crescendo software is available as an R package at https://github.com/immunogenomics/crescendo.

Removing the effects of covariates in count-based regression

In this section, we first detail the regression model that estimates the effect of technical confounders on gene expression and then specify how to analytically remove the effect of these confounders from the regression model. Generalized linear regression models are designed to model the effect of covariates on count-based response data. In scRNA-seq analysis, Poisson GLMs have become the tool of choice, sometimes augmented by priors to account for overdispersion (i.e., negative binomial GLM in DEseq2 [76]), and random effects (i.e., Poisson GLMMs in lme4 [77]). In a Poisson GLM, we model the effect of a technical covariate Y on the expected number of counts μ_{gi} of gene g in cell i with the following formula:

$$\mu_{gi} = \text{nUMI}_\text{i} \times \exp\left(\beta_g + Y_i\gamma_g\right) + \epsilon_{gi} \tag{1}$$

Here, the gene frequency has a baseline expression level $\exp(\beta_g)$ plus a multiplicative offset $\exp(Y_i\gamma_g)$ defined by which batch Y_i cell i belongs to. The offset term nUMI_i, which is the total number of UMIs in cell i, multiplies the gene frequency into an expected number of counts. The difference between the observed and expected gene counts is explained by the additive residual offset ϵ_{gi}, assumed to be normally distributed: $\epsilon_{gi} \sim N(0, \sigma)$. We next ask how to remove the effect of the $Y_i\gamma_g$ term on μ_{gi}. Because $Y_i\gamma_g$ is inside the exponent, we cannot subtract it. Nor can we divide by $\exp\left(Y_i\gamma_g\right)$, because that would rescale the residual term. Instead, we use expectation to integrate out the effect of $Y_i\gamma_g$.

$$\mathrm{E_Y}\ \mu_{\mathrm{gi}} = \mathrm{E_Y}\left[\mathrm{nUMI}\ \times\ \exp\left(\beta_{\mathrm{g}} + \mathrm{Y_i Y_g}\right) + \in_{\mathrm{gi}}\right] \tag{2}$$

Using the linearity property of expectation and the independence of nUMI, ϵ_{gi}, and β_g on Y, we simplify to:

$$\mathrm{E_Y}\ \mu_{\mathrm{gi}}\ = \mathrm{nUMI}\ \times\ \exp\left(\beta_{\mathrm{g}}\right)\ \times\ \mathrm{E_Y}\left[\exp\left(\mathrm{Y_i Y_g}\right)\right] + \in_{\mathrm{gi}} \tag{3}$$

Now, we need to simplify the term $\mathrm{E_Y}\left[\exp\left(\mathrm{Y_i Y_g}\right)\right]$ which is an expectation over a non-linear function. This term does not have a general closed form. However, if we model γ_g as a random effect, letting $\gamma_g \sim N(0, \sigma_Y)$, then $\exp\left(\mathrm{Y_i}\gamma_{\mathrm{g}}\right)$ has a log-normal distribution, with a closed form expectation: $\mathrm{E_Y}\left[\exp\left(\mathrm{Y_i}\gamma_{\mathrm{g}}\right)\right] = \exp\left(\sigma_{\mathrm{Y}}/2\right)$. Plugging this back in, we get a closed form expression for the "batch-free" expected number of counts:

$$\mathrm{E_Y}\ \mu_{\mathrm{gi}}\ = \mathrm{nUMI}\ \times\ \exp\left(\beta_{\mathrm{g}} + \frac{\sigma_{\mathrm{Y}}^2}{2}\right) + \in_{\mathrm{gi}} \tag{4}$$

To recap, in this section, we started with a Poisson-based regression model of gene counts and a confounding batch effect, we formulated "removing" the batch effect as an expectation, and we derived a closed form for this expectation by modeling the confounder as a random effect, rather than a fixed effect. This result tells us what the expected gene expression is with (Eq. 1) and without (Eq. 4) the batch effect. The steps that utilize Eqs. 1 and 4 are the estimation and marginalization steps, respectively. However, this expectation is a real number and does not tell us how many gene counts of g to assign to cell i. The next section does.

Predicting batch-free counts

In this section, we derive the framework we use to assign a "batch-corrected" count for gene g in cell i. This framework is built on inverse cumulative distribution function (iCDF) matching, a tool from probability theory that essentially aligns two distributions through their respective CDFs. To build intuition, let us consider a simple scenario. For random variables $A \sim N(2, 4)$ and $B \sim N(0, 3)$, we want to find the value of $B = b$ that is equivalent to $A = 3$. Here, we define equivalence in terms of equal probabilities. Given the two CDFs, find b such that $\Pr(A < 3) = \Pr(B < b)$. To find b, we use the inverse CDF of B, also known as the quantile function, defined such that $F_B^{-1}(\Pr(B < b)) = b$. To find the value of $B = b$ that is equally likely as $A = 3$, we find the probability of $A = 3$ and plug it into the quantile function of B: $b = F_B^{-1}(\Pr(A < 3))$. Now let us connect this procedure to our task of finding "batch-free" counts for a gene g in cell i. Let us define Eq. 1 above as the *full model* and Eq. 4 as the batch-free model. Our task is to find the value of "batch-free" gene counts X_{gi}^* that are equally probable to observe under the batch-free model (Eq. 1) as the observed gene counts X_{gi} are under the full model (Eq. 4). Plugging in these equations, we get the solution for X_{gi}^*:

$$\mathrm{X_{gi}^*}\ = \mathrm{F^{-1}}\left(\Pr\left(\mathrm{X_{gi}}\ < \mu_{\mathrm{gi}}\right),\ \mathrm{E_Y}\ \mu_{\mathrm{gi}}\right) \tag{5}$$

Above, F^{-1} is the quantile function of a Poisson distribution with mean $\mathrm{E_Y}\ \mu_{\mathrm{gi}}$, while $\Pr(X_{gi} < \mu_{gi})$ is the CDF of a Poisson with mean μ_{gi}. This step constitutes the matching

step. This procedure ensures that the distribution of corrected counts X_g for some gene g remains conditionally Poisson, automatically adjusting all moments of the distribution appropriately. In contrast, if this procedure was used with Gaussian distributions, the iCDF transformation is equivalent to arithmetic subtraction to match a change in the expected means. In the Gaussian case, the mean is independent from variance and all higher order moments and can thus be changed without accounting for these moments.

Modeling the interaction between technical and biological effects

The models above all assume that gene expression is well-modeled by mean gene expression β_{0g} and a batch offset $Y_i\beta_g$. However, it is well known that these batch effects are different for different cell types. Batch correction methods for scRNA-seq data account for this dependence between cell type and batch effect in different ways. MNNCorrect and Seurat CCA learn local correction factors based on a nearest neighbor graph, scVI learns non-linear batch effects using deep neural networks, and Harmony explicitly assigns cells to latent clusters and uses linear regression to model cluster-specific batch effects. In Crescendo, we must also account for batch effects that vary across cell types. Because both Crescendo and Harmony use regression, we use Harmony's framework to model the interaction between latent biological factors and explicit batch factors. Following Harmony's notation, cells are probabilistically assigned to 1 of K clusters, with probabilities $\mathrm{R_{ik}} \geq 0$, $\sum_{\mathrm{k}=1\ldots\mathrm{K}}\mathrm{R_{ik}} = 1$. Because R can be considered just another covariate in a GLMM, we include these latent biological clusters into Eq. 1 directly:

$$\mu_{gi} = \mathrm{nUMI_i} \times \exp\left(\sum\nolimits_k R_{ik}\left(\beta_{kg} + Y_i\gamma_{kg}\right)\right) + \epsilon_{gi} \tag{6}$$

In the equation above, the baseline expected expression of gene g is specified by an expected value of that gene over $\sum_k R_{ik}\beta_{kg}$, and the batch effect is also an expectation over the batch effect terms $\sum_k R_{ik}Y_i\gamma_{kg}$ for all the clusters that cell i probabilistically belongs to. For completeness, let us also specify the marginalized, batch-free equation under this mixture model.

$$\mathrm{E_Y}\mu_{\mathrm{gi}} = \mathrm{nUMI_i} \times \exp\left(\sum\nolimits_{\mathrm{k}}\mathrm{R_{ik}}\left(\beta_{\mathrm{kg}} + \frac{\sigma_{\mathrm{Y}}^2}{2}\right)\right) + \epsilon_{\mathrm{gi}} \tag{7}$$

In both Eqs. 6 and 7, the latent cluster assignment matrix R is given by running the Harmony algorithm on cell's PCA embeddings. Note that we chose to model the total counts as a linear mixture of rates model rather than a Poisson mixture model. This formulation lends itself better to inference and interpretation, as a cell's gene counts do not come from two independent processes. Instead, the probabilistic clusters reflect uncertainty about a cell's biological identity and allow us to infer a more robust estimate of that cell's batch effects and hence underlying gene expression generative model.

Putting it together: Crescendo algorithm

The Crescendo algorithm puts together the components built and motivated in "Removing the effects of covariates in count-based regression," "Predicting batch-free counts," and " Modeling the interaction between technical and biological effects" sections, into three essential steps: estimation (Eq. 6), marginalization (Eq. 7), and matching (Eq. 5).

Algorithm 1 Crescendo

```
Input: gene-by-cell counts matrix X
Input: cell-by-cluster probabilistic cluster matrix R
Input: batch identity vector Y
Initialize empty gene-by-cell matrix X*
for gene g in 1,…,G do
  for cell I in 1,…,N do
    Fit β_kg and γ_kg for full model (eqn 6)
    Compute expected counts E_Y μ_gi (eqn 7)
    Infer counts X*_gi (eqn 5)
  end
end
Output: X*
```

The most expensive step of Crescendo is to fit the β_{kg} and γ_{gk} effects for the full model. Because γ_{kg} is a random effect, we could use the popular lme4 R package to fit the model. However, we found that the glmnet R package was faster at fitting the same model. We used glmnet as a suitable replacement, because of the mathematical equivalence between ridge regression (i.e., glmnet) and random effect models (i.e., lme4) with one random effect [78]. The remainder of the steps were implemented with custom R and C++ code, relying heavily on the C++ Boost libraries for the iCDF computations. The code for Crescendo is available as an open-source R package on github.com/immunogenomics/crescendo.

Scaling to large data with robust downsampling

Single-cell studies now often include more than 100,000 cells, while spatial transcriptomics datasets that include multiple slices may include millions of cells. While fitting count-based models for gene expression is more accurate than Gaussian models, they can take substantially longer to fit, especially as the dataset size increases. Furthermore, users may desire to fit more than one gene, which can mean fitting multiple models across millions of cells. To reduce the required computational resources and time for fitting, we allow users the option to downsample their data in a batch and cell-type aware manner. This downsampling is only for the purposes of fitting the GLMM for a gene, which constitutes the bulk of the computational runtime in Crescendo—all cells will be sampled batch corrected counts regardless of whether downsampling was utilized. For downsampling in a batch and cell-type aware manner, we designate a minimum number of cells m so that we do not downsample too few cells. In this manuscript, we downsample the input dataset such that there are at least m cells within each cell-type within each batch in the downsampled dataset. If there are fewer than m cells within a cell-type within a batch, all cells of that type are kept. For more complicated data structures such as nested batch structures, we suggest downsampling such that the lowest-level groups have at least m cells; these would require a custom downsampling function depending on the data structure. By default, we utilize Harmony soft-cluster assignments, which means that a cell may have membership in multiple clusters. For the purposes

of downsampling, we assign each cell a discrete cell-type label by creating a probability distribution from its soft-cluster membership probabilities, and then we sample a cluster label from this distribution. We allow users to specify a proportion, which proportionally downsamples the number of cells within a cell-type within a batch (e.g., a proportion of 0.25 will try to sample 25% of cells in a cell-type in a batch, unless there are fewer than m such cells). In general, we tended to fit on around 20,000 cells total for each dataset. By default, we set $m = 100$, but it is likely that fewer cells are required to obtain relatively similar coefficients to the full dataset.

BVR and CVR performance metrics

In scRNA-seq, the performance of a batch correction algorithm is often evaluated by how they change the structure of the data in a low-dimensional latent space. Typically, batch correction algorithms increase the diversity of batches in a local area of the latent space, which is quantified with a metric. Because these latent spaces are summarizations of many genes, we cannot directly apply previously created metrics to quantifying batch correction performance in a single gene. Thus, we now describe two metrics which can be used to evaluate the performance of batch correction in genes.

Effective batch effect correction of gene expression must meet two objectives: (1) remove differences (variation) between cells of the same cell type that are driven by technical factors such as batch and (2) preserve the biologically meaningful differences in gene expression among cell types. With these objectives in mind, we developed two metrics that each addresses one of these objectives. The first metric, which we call the batch-variance ratio (BVR), quantifies how much batch effect was removed from a gene count distribution after batch correction, while the second metric, cell-type-variance ratio (CVR) quantifies the preservation of cell-type variation after batch correction.

To calculate the BVR and CVR metrics, we fit Poisson generalized linear models (GLMs) that estimate the batch variance and the cell-type variance present in a given count distribution. We calculate these variances by fitting batch and cell-type as independent random effects, as well as an independent interaction term between batch and cell-type to estimate cell-type-specific batch variance. In practice, we utilize user-defined discrete clusters (e.g., T cell, B cell). To fit Poisson GLMs, we used the R package "presto," which utilizes the "glmer" function from the R package "lme4." For the observed counts X, we fit the following formula:

$$X \sim 1 + \left(1|celltype\right) + (1|batch) + (1|celltype : batch) \tag{8}$$

For the batch corrected counts X*, we similarly fit

$$X^* \sim 1 + \left(1|celltype\right) + (1|batch) + (1|celltype : batch) \tag{9}$$

For fitting the Poisson GLMs in Eqs. 8 and 9, we use the observed nUMI for cells as the offset.

To calculate the BVR, we obtain the variance estimates for the batch and cell-type-specific batch terms. For simplicity, we calculate the overall batch variance estimate as the sum of the cell-type-specific batch and batch estimates. We obtain an overall batch-variance estimate from the batch corrected model in Eq. 9 in the same way. Let B_{pre} be

the pre-correction batch-variance estimate obtained from Eq. 8 and let B_{post} be the post-correction batch-variance estimate obtained from Eq. 9. We calculate BVR as:

$$\text{BVR} = \frac{B_{post}}{B_{pre}} \tag{10}$$

In a similar manner, we obtain cell-type variance estimates from both Eqs. 8 and 9. Let C_{pre} be the pre-correction cell-type-variance estimate from Eq. 8 and let C_{post} be the post-correction cell-type-variance estimate from Eq. 9. We calculate CVR as:

$$\text{CVR} = \frac{C_{post}}{C_{pre}} \tag{11}$$

The BVR metric quantifies how much batch-related variance was removed after batch correction, while the CVR metric quantifies how much cell-type-related variance was preserved after batch correction. Based on the two objectives we outlined at the beginning of this section, ideal batch correction will decrease batch variance resulting in a BVR < 1, while preserving or increasing cell-type variance resulting in a CVR ≥ 1. In practice, batch correction usually features a trade-off—the more aggressively batch effects are removed, the more cell-type variance tends to be removed (although sometimes cell-type variance is also increased if a gene becomes more specific to a cell-type after batch correction). Empirically, a CVR ≥ 0.5 was a reasonable trade-off if the BVR was lowered.

We also note that the batch-related variance value before correction may be a useful value for users, as it can help determine which genes have higher levels of batch effects and might need correction.

Gene count simulations

We simulated gene count distributions by sampling from Poisson distributions parameterized by different rates based on the cell-type or batch a cell is from. To simulate a single gene, we designate the number of batches, as well as the number of cells we will simulate for each cell type per batch. Each cell belongs to one cell type and one batch. We then arbitrarily set a base rate for each cell type (e.g., a rate of 1 for cell-type 1 and a rate of 3 for cell-type 2). To simulate batches, we sampled a batch-specific rate for each batch from a standard normal distribution, and then centered all batch-specific rates around 0. For simplicity and visualization, we simulated from two cell types in two batches, though this framework is compatible with an arbitrary number of cell types and batches. After sampling batch-specific rates, we add them to the base rate for each cell type. For example, batch 1 will add a batch-specific rate of 0.405 to cell-type 1's rate of 1 to result in a unique rate of 1.405, while batch 2 will add a batch-specific rate of -0.405 to cell-type 1's rate of 1 to result in a unique rate of 0.595. Thus, each cell type within each batch has its own unique rate that represents a batch effect.

We then assign each cell a probability membership for each cell type, which represents soft-cluster membership. For two cell types, we sampled from a beta distribution parameterized with $\alpha = 0.5$, $\beta = 0.5$ to get the probability p of a cell belonging to one cell-type; to calculate the probability q or a cell belonging to the other cell-type, we simply take $1 - p$. We also set each cell to have the same constant number of unique molecular

identifiers (nUMI), though this framework is compatible with variable nUMIs (recommend sampling from a lognormal distribution).

We then created a design matrix that contains the cell-type probabilities and the batch identities of each cell, and then matrix-multiplied the design matrix with a matrix containing the batch-specific rates for each cell type. After, we multiplied the resulting product with a matrix containing the cell-type probabilities to recover a rate for each cell based on its batch and cell-type identity. To represent read depth, we add a log-transformed nUMI constant to each cell's rate (in our simulations, we set the constant for each cell to be equal at 10,000). Finally, we use the resulting rates to parameterize a Poisson distribution for each cell, which we then sample a count from.

For Additional file 1: Fig. S1F, we simulated 10,000 genes, with the base cell-type rates set at 1 for cell-type 1 and 3 for cell-type 2.

Plotting gene expression visualizations

To plot gene expression across batches, we utilize the "facet_wrap" function from the R package "ggplot2." This function allows us to visualize the same gene's expression across all batches together on the same scale. For visualization purposes, we plot cells that express a gene on top of other cells that do not express the gene. This represents a best-case scenario in which we should be able to see every instance of gene expression, and is extremely forgiving if the gene is poorly expressed. In practice, most visualization of data is performed with cells being randomly mixed such that gene-expressing cells are not always on top. In such scenarios, we observed that Crescendo dramatically improves visualization even more than the best-case scenario, which is already a significant improvement.

Benchmarking and comparison to other algorithms

To perform benchmarking in Fig. 2, Additional file 1: Fig. S5, and Fig. 3, we compared Crescendo with the following algorithms: ComBat-Seq, scVI, Seurat anchor integration, limma, and mutual nearest-neighbors (MNN) correction.

To batch correct for ComBat-Seq, we used the "ComBat_seq" function from the R Bioconductor package "sva" with default parameters. ComBat-Seq is designed to fit and output counts, so we calculated BVR and CVR metrics based on fitting Poisson models of the raw gene expression counts and the ComBat-Seq batch corrected gene expression counts ("BVR and CVR performance metrics" section).

For batch correcting with scVI, we utilized the "get_normalized_expression" function from the "scvi" module where we set the library size parameter to 10,000 and the other parameters as default. We were unable to finish running scVI on the large 7-million cell cancer dataset. SCVI's integration returned normalized gene expression, so we calculated BVR and CVR metrics based on fitting Gaussian models of the normalized gene expression counts and the Seurat-corrected counts ("BVR and CVR performance metrics" section).

For batch correcting with Seurat, we used Seurat version 4.3.0 in R. For each batch, we created a Seurat object and normalized them with the "NormalizeData" function. We then integrated the datasets with the "FindIntegrationAnchors" and "IntegrateData" functions with default parameters and dims = 1:20. To access corrected counts from the

integration, we accessed the object's "@assays$integrated@data" slot. Seurat's integration works on normalized gene expression and returns gene expression in a similar normalized space, so we calculated BVR and CVR metrics based on fitting Gaussian models of the normalized gene expression counts and the Seurat-corrected counts ("BVR and CVR performance metrics" section).

For limma, we used the "removeBatchEffect" function from the R package "limma" with default parameters. Limma's integration works on normalized gene expression and returns gene expression in a similar normalized space, so we calculated BVR and CVR metrics based on fitting Gaussian models of the normalized gene expression counts and the limma-corrected counts ("BVR and CVR performance metrics" section).

For MNN, we used the "fastMNN" function from the R package "batchelor" with default parameters. MNN's integration works on cosine-normalized gene expression and returns gene expression in a similar normalized space, so we calculated BVR and CVR metrics based on fitting Gaussian models of the cosine-normalized gene expression counts and the MNN-corrected counts ("BVR and CVR performance metrics" section).

Vizgen Mouse Brain Receptor analysis details

We downloaded the Vizgen Mouse Brain Receptor metadata and count matrices from the Vizgen Data Release Program [37]. This dataset contains a panel of 483 genes. For Fig. 2, we subsetted the data to only include cells from slice 3 (S3R1, S3R2, S3R3), which represent serial sections from the same mouse brain (186,910 total cells). Following Vizgen recommendations, we filtered out cells with fewer than 50 total expressed transcripts or fewer than 50 uniquely expressed genes, resulting in 179,385 remaining cells: 53,269 from S3R1, 64,476 from S3R2, and 61,640 from S3R3. For the following steps, we used all 483 genes. We library-normalized cells with standard log-normalization with the median read counts as the scale factor and scaled genes with z-score scaling. We then utilized PCA to reduce the dimensionality of the data to the top 20 PCs and performed batch correction with the Harmony algorithm. To cluster cells, we utilized Leiden clustering with resolution = 0.2. Finally, we used Crescendo to batch correct all genes using S3R1, S3R2, and S3R3 as batches. We used the observed nUMI as the initial offset, and then used the median nUMI as the final offset for imputation. For visualization in physical space, we rotated each slice's coordinates such that they are in the same orientation.

Scalability analysis

For the scalability analyses, we utilized the public Vizgen FFPE Immuno-oncology dataset [38]. We downloaded the metadata and count matrices from the Vizgen Data Release Program. This dataset contains a panel of 500 genes measured on 16 human cancer samples across 9 different tissue types (~8.7M total cells). Following Vizgen recommendations, we filtered out cells with fewer than 50 total expressed transcripts or fewer than 50 uniquely expressed genes, resulting in 7,020,548 remaining cells. We library-normalized cells with standard log-normalization with the median read counts as the scale factor and scaled genes with z-score scaling. We then utilized PCA to reduce the dimensionality of the data to the top 20 PCs and performed batch correction with the Harmony algorithm. To batch correct with Crescendo, we used sample identity (Lung Sample 1, Lung

Sample 2, Liver Sample 1, etc.) as batches. We used the observed nUMI as the initial offset, and then used the median nUMI as the final offset for imputation.

To accommodate the large memory required to load this dataset, batch correction and scalability analyses on this dataset were run on a server containing 24 cores and 128 GB of RAM for all algorithms.

Integrated colorectal cancer (CRC) scRNA-seq and spatial transcriptomics analysis

For the CRC scRNA-seq dataset, we downloaded the metadata and count matrices from GEO: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE178341 [40]. We first filtered for only cells from SPECIMEN_TYPE = T (cells taken from tumor samples) and SINGLECELL_TYPE = SC3Pv3 (only cells assayed with 10X v3), which resulted in 90,312 remaining cells. We further QCed to keep only cells that featured a total nUMI from 30 to 2000 counts and expression in at least 10 unique genes, resulting in 86,627 cells. Spatial samples were generated from one of the donors in this dataset—we kept all donors in the scRNA-seq dataset as the spatial donor only had ~ 1600 scRNA-seq cells.

The spatial transcriptomics tissues were produced in collaboration with Vizgen. These tissues derive from the same patient sample, which is also represented in the scRNA-seq data. Segmentation was performed with Baysor [68].

After combining scRNA-seq and spatial data, we library-normalized cells with standard log-normalization with the median read counts as the scale factor and scaled genes with z-score scaling. We then utilized PCA to reduce the dimensionality of the data to the top 20 PCs and performed batch correction with the Harmony algorithm. To cluster cells, we utilized Leiden clustering with resolution = 0.1. To batch correct with Crescendo, we represented scRNA-seq as its own batch and the two spatial transcriptomics slices as their own individual batch (scRNA-seq, PFA_A6, and PFA_A11 were the batches). We used the observed nUMI as the initial offset, and then used the median nUMI as the final offset for imputation.

Spatial cross-correlation index (SCI) calculations

To calculate an SCI in a cell-type-aware manner, we first subsetted a spatial transcriptomic dataset's count matrix to two (user-specified) cell types. We then calculated the 30 nearest-neighbors for each cell (excluding itself) with the "nn2" function from the R package "RANN" and retrieved a sparse distance matrix from the nn2 output with the "getDistMat" function provided by Crescendo. We next removed a cell's neighbors if they are the same cell-type (by setting its value to 0 in the distance matrix). We then removed neighbors with a distance > 30 μm from the cell. Finally, we binarized the matrix by setting all non-zero values to 1. This binarized matrix (K) contains information on whether another cell is a nearest-neighbor, a different cell type, and within a distance of 30. Thus, for a cell-type 1, we have its nearest-neighbors from cell-type 2 and vice-versa.

We then take the subsetted raw gene count matrix and log-normalize the counts with the median nUMI as the scale factor to produce a normalized gene counts matrix X. Then, we matrix-multiplied the raw gene count matrix with the binarized matrix (K) to produce a normalized gene nearest-neighbors gene counts matrix (XK). Thus, X contains the gene expression of cells while XK contains the gene expression of that cell's nearest-neighbors from the other cell type. Finally, we use the "cor" function from base R

with X and XK as input and with default parameters to obtain the correlations for each gene–gene pair. SCI calculations were performed in each slice independently.

Two genes that share similar spatial expression patterns will exhibit a higher SCI, while two genes whose spatial patterns are not correlated with exhibit a low SCI.

Defining ligand-receptor gene pairs

To classify a gene–gene pair as a ligand-receptor pair, we downloaded several ligand-receptor databases from the Lewis Lab Compendium of ligand-receptor pairs in literature (hosted on Github) [51]. In particular, we utilized two datasets from Browaeys et al. [17] and Ramilowski et al. [52], which encompassed a large set of reasonable ligand-receptor pairs. We took the union of these datasets resulting in 896 ligand-receptor pairs.

Highly variable gene conservation calculations

The highly variable gene (HVG) conservation coefficient for each algorithm is defined by Luecken et al. [48]. Its intention is to measure the conservation of biological signal and is calculated by identifying the highly variable genes of a dataset before and after correction. Similar to Luecken et al., we used the scanpy package to calculate the top 250 HVGs before and after correction, because the mouse brain and CRC datasets contained fewer than 500 genes. The HVG coefficient is then calculated as the intersection of the HVGs before and after batch correction, divided by the minimum number of HVGs before and after correction. A higher HVG coefficient indicates better conservation of biological signal.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s13059-025-03479-9.

Additional file 1: Contains supplementary figures (Figs. S1–S9) and supplementary Table 1.

Additional file 2: Contains spatial correlation indices for CRC slice PFA_A6. Each sheet contains the indices for a cell type-cell type pair.

Additional file 3: Contains spatial correlation indices for CRC slice PFA_A11. Each sheet contains the indices for a cell type-cell type pair.

Additional file 4: Review history.

Acknowledgements
We thank members of the Hacohen lab and the Vizgen research team for their contributions to generating the colorectal cancer (CRC) scRNA-seq and spatial transcriptomics datasets.

Peer review information
Johanna Klughammer and Veronique van den Berghe were the primary editors of this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Authors' contributions
N.M., I.K., and S.R. conceived the project. N.M. developed the method and performed the analyses under the guidance of S.R. and I.K. J.C. and P.K. generated the CRC scRNA-seq dataset. M.G.P. provided computational analysis of single-cell datasets. J.C., M.S., K.P., J.H., and C.P. designed the CRC spatial transcriptomics panel, constructed the cohort, and generated spatial transcriptomics datasets. All authors participated in the interpretation and writing of the manuscript.

Funding
This work was funded in part by the National Institutes of Health (5K01AR078355, 5U01HG012009, 1R01HG013083, 5UC2AR081023, and 5P01AI148102) and the Chan-Zuckerberg Initiative Data Insights Program.

Data availability
The R code for running Crescendo is available for download on Github at https://github.com/immunogenomics/crescendo under an MIT license [79]. The version of Crescendo used for the analyses in this manuscript has been deposited

on Zenodo at https://zenodo.org/records/14366602 [80]. The original Vizgen MERFISH Mouse Brain Receptor spatial transcriptomics and Vizgen MERSCOPE FFPE Human Immuno-oncology spatial transcriptomics datasets analyzed in this manuscript are publicly available under the Vizgen Data Release Program at https://vizgen.com/data-release-program/ [37, 38]. The original colorectal cancer (CRC) scRNA-seq dataset is publicly available on GEO at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE178341 [40]. The CRC spatial transcriptomics dataset has been deposited on Zenodo at https://zenodo.org/records/14602110 [81].

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
I.K. does bioinformatics consulting for Mestag Therapeutics. S.R. is a founder for Mestag Therapeutics and a scientific adviser for Pfizer, Janssen, and Nimbus Therapeutics. J.H. is a co-founder and stockholder of Vizgen, Inc. K.P. consults for Santa Ana Bio.

Author details
[1]Division of Rheumatology, Inflammation and Immunity, Brigham and Women's Hospital, Boston, MA, USA. [2]Division of Genetics, Brigham and Women's Hospital, Boston, MA, USA. [3]Center for Data Sciences, Brigham and Women's Hospital, Boston, MA, USA. [4]Department of Biomedical Informatics, Harvard Medical School, Boston, MA, USA. [5]Department of Medicine, Harvard Medical School, Boston, MA, USA. [6]Department of Immunology, Harvard Medical School, Boston, MA, USA. [7]Harvard Medical School, Boston, MA, USA. [8]Broad Institute of MIT and Harvard, Cambridge, MA, USA. [9]Massachusetts General Hospital (MGH) Cancer Center, Harvard Medical School, Boston, MA, USA. [10]Department of Pathology, MGH, Boston, MA, USA. [11]Vizgen, Inc, Cambridge, MA, USA. [12]UCSF Institute of Genomic Immunology, Gladstone Institutes, San Francisco, CA, USA.

Received: 8 March 2024 Accepted: 21 January 2025
Published online: 25 February 2025

References

1. Picelli S, et al. Smart-seq2 for sensitive full-length transcriptome profiling in single cells. Nat Methods. 2013;10:1096–100.
2. Macosko EZ, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. Cell. 2015;161:1202.
3. Stoeckius M, et al. Large-scale simultaneous measurement of epitopes and transcriptomes in single cells. Nat Methods. 2017;14:865.
4. Villani AC, et al. Single-cell RNA-seq reveals new types of human blood dendritic cells, monocytes, and progenitors. Science (1979). 2017;356:eaah4573.
5. Zhang F, et al. Defining inflammatory cell states in rheumatoid arthritis joint synovial tissues by integrating single-cell transcriptomics and mass cytometry. Nat Immunol. 2019;20:928.
6. Smillie CS, et al. Intra- and inter-cellular rewiring of the human colon during ulcerative colitis. Cell. 2019;178:714-730.e22.
7. Yazar S, et al. Single-cell eQTL mapping identifies cell type-specific genetic control of autoimmune disease. Science. 1979;2022(376):eabf3041.
8. Zhang F, et al. Deconstruction of rheumatoid arthritis synovium defines inflammatory subtypes. Nature. 2023. https://doi.org/10.1038/s41586-023-06708-y.
9. Codeluppi S, et al. Spatial organization of the somatosensory cortex revealed by osmFISH. Nat Methods. 2018;15(11):932–5.
10. Wang X, et al. Three-dimensional intact-tissue sequencing of single-cell transcriptional states. Science. 2018;361:eaat5691.
11. Moffitt JR, et al. Molecular, spatial, and functional single-cell profiling of the hypothalamic preoptic region. Science. 2018;362:eaau5324.
12. He S, et al. High-plex imaging of RNA and proteins at subcellular resolution in fixed tissue by spatial molecular imaging. Nat Biotechnol. 2022;40(12):1794–806.
13. Chen A, et al. Spatiotemporal transcriptomic atlas of mouse organogenesis using DNA nanoball-patterned arrays. Cell. 2022;185:1777-1792.e21.
14. Salehi N, Karimi-Jafari MH, Totonchi M, Amiri-Yekta A. Integration and gene co-expression network analysis of scRNA-seq transcriptomes reveal heterogeneity and key functional genes in human spermatogenesis. Sci Rep. 2021;11(1):1–13.
15. Iacono G, Massoni-Badosa R, Heyn H. Single-cell transcriptomics unveils gene regulatory network plasticity. Genome Biol. 2019;20:1–20.
16. Efremova M, Vento-Tormo M, Teichmann SA, Vento-Tormo R. Cell PhoneDB: inferring cell-cell communication from combined expression of multi-subunit ligand-receptor complexes. Nat Protoc. 2020;15:1484–506.

17. Browaeys R, Saelens W, Saeys Y. NicheNet: modeling intercellular communication by linking ligands to target genes. Nat Methods. 2020;17:159–62.
18. Jin S, et al. Inference and analysis of cell-cell communication using Cell Chat. Nat Commun. 2021;12:1088.
19. McInnes L, Healy J, Melville J. UMAP: uniform manifold approximation and projection for dimension reduction. 2018.
20. Kumar MP, et al. Analysis of single-cell RNA-Seq identifies cell-cell communication associated with tumor characteristics. Cell Rep. 2018;25:1458.
21. Tyler SR, et al. PyMINEr finds gene and autocrine-paracrine networks from human islet scRNA-Seq. Cell Rep. 2019;26:1951-1964.e8.
22. Cabello-Aguilar S, et al. SingleCellSignalR: inference of intercellular networks from single-cell transcriptomics. Nucleic Acids Res. 2020;48:e55.
23. Miller BF, Bambah-Mukku D, Dulac C, Zhuang X, Fan J. Characterizing spatial gene expression heterogeneity in spatially resolved single-cell transcriptomic data with nonuniform cellular densities. Genome Res. 2021;31:1843–55.
24. Zhao P, Zhu J, Ma Y, Zhou X. Modeling zero inflation is not necessary for spatial transcriptomics. Genome Biol. 2022;23:1–19.
25. Fang S, et al. Computational approaches and challenges in spatial transcriptomics. Genomics Proteomics Bioinformatics. 2023;21:24–47.
26. Robert C, Watson M. Errors in RNA-Seq quantification affect genes of relevance to human disease. Genome Biol. 2015;16:1.
27. Liu B, Li Y, Zhang L. Analysis and visualization of spatial transcriptomic data. Front Genet. 2022;12: 785290.
28. Liu W, et al. Probabilistic embedding, clustering, and alignment for integrating spatial transcriptomics data with PRECAST. Nat Commun. 2023;14(1):1–18.
29. Dong K, Zhang S. Deciphering spatial domains from spatially resolved transcriptomics with an adaptive graph attention auto-encoder. Nat Commun. 2022;13(1):1–12.
30. Wang, H. et al. Systematic benchmarking of imaging spatial transcriptomics platforms in FFPE tissues. bioRxiv. 2023.12.07.570603. 2023. https://doi.org/10.1101/2023.12.07.570603.
31. Hartman A, Satija R. Comparative analysis of multiplexed in situ gene expression profiling technologies. bioRxiv. 2024.01.11.575135. 2024. https://doi.org/10.1101/2024.01.11.575135.
32. Korsunsky I, et al. Fast, sensitive and accurate integration of single-cell data with Harmony. Nat Methods. 2019;16:1289–96.
33. Lopez R, Regier J, Cole MB, Jordan MI, Yosef N. Deep generative modeling for single-cell transcriptomics. Nat Methods. 2018;15(12):1053–8.
34. Stuart T, et al. Comprehensive integration of single-cell data. Cell. 2019;177:1888.
35. Haghverdi L, Lun ATL, Morgan MD, Marioni JC. Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors. Nat Biotechnol. 2018;36(5):421–7.
36. Zhang Y, Parmigiani G, Johnson WE. ComBat-seq: batch effect adjustment for RNA-seq count data. NAR Genom Bioinform. 2020;2:lqaa078.
37. Vizgen Data Release V1.0. May 2021. Mouse brain receptor map. 2021. https://info.vizgen.com/mouse-brain-map?submissionGuid=1f5c93f2-d904-4d15-b0bf-039fb2faa2b6.
38. Vizgen MERFISH FFPE human immuno-oncology data set, May 2022. https://info.vizgen.com/ffpe-showcase?submissionGuid=a33d0205-6315-46f1-8569-aa86813cdd8f.
39. Pelka K, et al. Spatially organized multicellular immune hubs in human colorectal cancer. Cell. 2021;184:4734.
40. Pelka, K, Chen, JH, Anderson, AC, Rozenblatt-Rosen, O, Regev, A and Hachoen, N. A single cell atlas of MMRd and MMRp colorectal cancer. Datasets. Gene expression omnibus. 2021. https://identifiers.org/geo:GSE178341.
41. Millard N, Chen JH, Palshikar M, Pelka K, Spurrell M, Price C, He J, Hacohen N, Raychaudhuri S, Korsunsky I. Colorectal cancer spatial transcriptomics and single-cell RNA-sequencing dataset. Datasets Zenodo. 2025. https://doi.org/10.5281/zenodo.14602110.
42. Luecken MD, Theis FJ. Current best practices in single-cell RNA-seq analysis: a tutorial. Mol Syst Biol. 2019;15:8746.
43. Schöneberg T, Meister J, Knierim AB, Schulz A. The G protein-coupled receptor GPR34 – the past 20 years of a grownup. Pharmacol Ther. 2018;189:71–88.
44. Preissler J, et al. Altered microglial phagocytosis in GPR34-deficient mice. Glia. 2015;63:206–15.
45. Gundlach AL, et al. Relaxin family peptides and receptors in mammalian brain. Ann N Y Acad Sci. 2009;1160:226–35.
46. Abboud C, et al. Analgesic effect of central relaxin receptor activation on persistent inflammatory pain in mice: behavioral and neurochemical data. Pain Rep. 2021;6:E937.
47. Cramer KS, Miko IJ. Eph-ephrin signaling in nervous system development. F1000Res. 2016;5:F1000.
48. Luecken MD, et al. Benchmarking atlas-level data integration in single-cell genomics. Nat Methods. 2021;19(1):41–50.
49. Tanay A, Regev A. Scaling single-cell genomics from phenomenology to mechanism. Nature. 2017;541:331–8.
50. Svensson V, Vento-Tormo R, Teichmann SA. Exponential scaling of single-cell RNA-seq in the past decade. Nat Protoc. 2018;13(4):599–604.
51. Armingol E, Officer A, Harismendy O, Lewis NE. Deciphering cell–cell interactions and communication from gene expression. Nat Rev Genet. 2020;22(2):71–88.
52. Ramilowski JA, et al. A draft network of ligand–receptor-mediated multicellular signalling in human. Nat Commun. 2015;6(1):1–12.
53. Wei K, et al. Notch signalling drives synovial fibroblast identity and arthritis pathology. Nature. 2020;582:259–64.
54. Huaux F, et al. Role of eotaxin-1 (CCL11) and CC chemokine receptor 3 (CCR3) in bleomycin-induced lung injury and fibrosis. Am J Pathol. 2005;167:1485–96.
55. Kindstedt E, et al. CCL11, a novel mediator of inflammatory bone resorption. Sci Rep. 2017;7(1):1–10.
56. Lake BB, et al. An atlas of healthy and injured cell states and niches in the human kidney. Nature. 2023;619(7970):585–94.
57. Zhang B, et al. A human embryonic limb cell atlas resolved in space and time. Nature. 2023;2023:1–11. https://doi.org/10.1038/s41586-023-06806-x.

58. Kanemaru K, et al. Spatially resolved multiomics of human cardiac niches. Nature. 2023;619(7971):801–10.
59. Ding J. et al. SpatialCTD: a large-scale TME spatial transcriptomic dataset to evaluate cell type deconvolution for immuno-oncology. bioRxiv. 2023.04.11.536333. 2023.https://doi.org/10.1101/2023.04.11.536333.
60. Cao J, et al. The single cell transcriptional landscape of mammalian organogenesis. Nature. 2019;566:496.
61. Cao J, et al. A human cell atlas of fetal gene expression. Science. 2020;370:eaba7721.
62. Qiu P. Embracing the dropouts in single-cell RNA-seq analysis. Nat Commun. 2020;11(1):1–9.
63. Tang W, Jørgensen ACS, Marguerat S, Thomas P, Shahrezaei V. Modelling capture efficiency of single-cell RNA-sequencing data improves inference of transcriptome-wide burst kinetics. Bioinformatics. 2023;39:btad395.
64. Kim TH, Zhou X, Chen M. Demystifying 'drop-outs' in single-cell UMI data. Genome Biol. 2020;21:1–19.
65. Jiang R, Sun T, Song D, Li JJ. Statistics or biology: the zero-inflation controversy about scRNA-seq data. Genome Biol. 2022;23(1):1–24.
66. Stringer C, Wang T, Michaelos M, Pachitariu M. Cellpose: a generalist algorithm for cellular segmentation. Nat Methods. 2020;18(1):100–6.
67. Atta L, Fan J. Computational challenges and opportunities in spatially resolved transcriptomic data analysis. Nat Commun. 2021;12(1):1–5.
68. Petukhov V, et al. Cell segmentation in imaging-based spatial transcriptomics. Nat Biotechnol. 2021;40(3):345–54.
69. Svensson V. Droplet scRNA-seq is not zero-inflated. Nat Biotechnol. 2020;38(2):147–50.
70. Buenrostro JD, et al. Single-cell chromatin accessibility reveals principles of regulatory variation. Nature. 2015;523:486.
71. Baek S, Lee I. Single-cell ATAC sequencing analysis: from data preprocessing to hypothesis generation. Comput Struct Biotechnol J. 2020;18:1429–39.
72. Fang R, et al. Comprehensive analysis of single cell ATAC-seq data with SnapATAC. Nat Commun. 2021;12(1):1–15.
73. Nathan A, et al. Single-cell eQTL models reveal dynamic T cell state dependence of disease loci. Nature. 2022;606(7912):120–8.
74. Zhang J, Zhao H. eQTL studies: from bulk tissues to single cells. J Genet Genomics. 2023;50:925–33.
75. Kang JB, Raveane A, Nathan A, Soranzo N, Raychaudhuri S. Methods and insights from single-cell expression quantitative trait loci. 2023. https://doi.org/10.1146/annurev-genom-101422-10043724,277-303.
76. Love MI, Huber W, Anders S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. 2014;15:1–21.
77. Bates D, Mächler M, Bolker BM, Walker SC. Fitting linear mixed-effects models using lme4. J Stat Softw. 2015;67:1–48.
78. Friedman J, Hastie T, Tibshirani R. Regularization paths for generalized linear models via coordinate descent. J Stat Softw. 2010;33:1–22.
79. Millard N, et al. Crescendo zenodo repository. 2024. Zenodo at https://zenodo.org/records/14366602.
80. Millard N. et al. Crescendo github repository. 2024. Github at https://github.com/immunogenomics/crescendo.
81. Millard, N. et al. Colorectal cancer spatial transcriptomics and single-cell RNA-sequencing dataset. 2025. Zenodo at https://zenodo.org/records/14602110.

Publisher's Note