Final Project – SPARKpy

Yuyao Liu

December 2024

1 Introduction

With the rapid development of spatial transcriptomics, an increasing amount of spatially resolved transcriptomics (SRT) sequencing data is being generated [1]. This offers rich opportunities to study cellular differentiation and regulatory networks and provides possibilities for disease research and drug development. To fully extract biologically meaningful information from these datasets, different types of research methods are required depending on the research task. These include tasks such as cell type deconvolution [2–4], spatially variable gene (SVG) identification [5, 6], gene ontology (GO) analysis [7], and integration of normal and disease data [8].

One of the most fundamental tasks is to identify SVGs. SVGs are genes whose expression shows specific spatial patterns in the tissue. These genes do not simply exhibit a specific expression pattern in space for no reason; they usually play important regulatory roles in certain cell types within the tissue. Therefore, these genes are of particular interest in downstream biological research.

The importance of identifying SVGs can be summarized as follows. First, correctly identified SVGs are often genes that play a key role in the regulatory activities we are studying, helping with downstream tasks. Second, SRT data are high-dimensional and sparse. Direct manipulation of the entire data matrix is time-consuming and inefficient. In addition, the vast majority of genes may have little or no role in a specific tissue, and only a few transcripts contribute meaningful information. Including these white noise genes in any task can make methods less robust and negatively impact the analysis results. Therefore, introducing a feature selection step as preprocessing, where we select potentially interesting genes, is indispensable for most computational methods. A commonly used method is to select highly variable genes (HVGs), but this approach does not account for spatial information [8]. As a result, due to technical noise or batch effects, some genes that do not actually carry meaningful information may be selected as HVGs, which could complicate downstream experiments. In contrast, SVGs, which are determined with spatial locations taken into account, are more suitable for feature selection and may be more beneficial for research. Third, due to the high cost of sequencing technologies, designing an effective gene panel that can capture a small number of genes sufficient for biological research has become an important research question [9]. This can help save resources and reduce economic costs. Selecting the corresponding SVGs from existing data can provide guidance for gene panel design. Thus, an effective method for identifying SVGs is crucial.

In the paper *Statistical analysis of spatial expression patterns for spatially resolved transcriptomic studies* [5], the authors proposed SPARK, which is able to effectively identify SVGs.

2 Algorithm

2.1 Write at the Beginning

In this section, I will elaborate on the algorithmic details of SPARK and provide the necessary derivations. I will also point out the errors in the algorithm description in the supplementary method of SPARK and highlight them in red font. The SPARK package is primarily written in R, with some modules accelerated using C++. I rewrote the algorithm discussed in this section and implemented it in Python and named it SPARKpy, which can be found at https://github.com/yyLIU12138/SPARKpy.

While reviewing the SPARK code, I found that it includes plenty manual adjustments not mentioned in the original paper, and some algorithmic details have been modified, differing from what was described in the paper. Nevertheless, I implemented the algorithm based on the description provided in the paper.

2.2 Count Modeling

Suppose we have a SRT dataset with N cells and G genes, then we will have a data matrix $Y \in \mathbb{R}^{N \times G}$, which contains the transcript counts of each gene across different spots. Additionally, we have a matrix $S \in \mathbb{R}^{N \times 2}$ representing the spatial coordinates of each spot. The expression of the g-th gene across all spots is represented as $y_g \in \mathbb{R}^N$. Since we consider only one gene at a time when fitting the model, we do not use the subscript g and denote the transcript counts of the gene being analyzed as y_i for the i-th spot.

SPARK assumes that y_i follows a Poisson distribution, i.e., $y_i \sim \text{Poisson}(N_i \lambda_i)$, where N_i is the library size for the i-th spot and λ_i is an unknown parameter. We use a generalized linear spatial model (GLSM) [10, 11] to model the relationship as follows:

$$\ln \lambda_i = x_i^T \beta + b_i + \varepsilon_i$$

where x_i is the covariate vector corresponding to the i-th spot, β is a k-dimensional coefficient vector that includes an intercept term, b_i is the random effect associated with the spatial location, and ε_i is the noise independent of the spatial location.

In vector form, this model can be written as:

$$\ln \lambda = X\beta + b + \varepsilon$$

$$\lambda = (\lambda_1, \ldots, \lambda_N)^T$$

$$X = \left[x_1^T, \ldots, x_N^T\right]^T$$

$$b = (b_1, \ldots, b_N)^T \sim \mathcal{N}(0, \tau_1 K)$$

$$\varepsilon = (\varepsilon_1, \ldots, \varepsilon_N)^T \sim \mathcal{N}(0, \tau_2 I_N)$$

Here, K is a kernel function related to spatial location that is used to capture spatial patterns. After fitting the unknown parameters in this GLSM, determining whether a gene is SVG involves testing whether the null hypothesis $H_0 : \tau_1 = 0$ can be rejected.

Since for a single kernel matrix, whether we can reject the null hypothesis largely depends on how well the kernel matrix represents the spatial correlation pattern of the gene, SPARK tests each gene using ten different kernel matrices. These p-values are then combined into a single p-value using the Cauchy combination rule [12], and this final p-value is used to determine whether we reject the null hypothesis.

These ten kernel matrices consist of five Gaussian kernels and five cosine kernels with different parameters.

2.3 Fit the Null Model

We first estimate the parameters in the null GLSM model, i.e.

$$\ln(\lambda_i) = x_i^T\beta + \varepsilon_i$$

Given $x_i^T\beta$ and ε_1, the different y_i's are independent, and we can compute the mean and variance of y_i as follows:

$$E(y_i|\beta,\varepsilon) = \mu_i = N_i \exp(x_i^T\beta + \varepsilon_i)$$

$$\text{Var}(y_i|\beta,\varepsilon) = v(\mu_i) = \mu_i$$

The joint log-likelihood for the data is:

$$\begin{aligned} l(y|\beta,\tau_2) &= \ln\left[P(y|\beta,\varepsilon)P(\varepsilon|\tau_2)\right] \\ &= \ln\int \exp\left(\sum_{i=1}^{N}\ln P_i(\beta,\varepsilon) + \ln P(\varepsilon|\tau_2)\right)d\varepsilon \end{aligned}$$

We fit the model based on a penalized quasi-likelihood approach [13, 14]. First, we use the quasi-likelihood of the i-th spot as an approximation for its conditional log-likelihood, replacing $\ln P_i(y_i|\beta,\varepsilon)$, where the quasi-likelihood for the i-th spot is:

$$\text{ql}_i(\beta,\varepsilon) = \int_{y_i}^{\mu_i}\frac{y_i - t}{v(t)}dt = \int_{y_i}^{\mu_i}\frac{y_i - t}{t}dt$$

Thus, the overall quasi-likelihood is:

$$\text{ql}(\beta,\tau_2) = \ln\int \exp\left(\sum_{i=1}^{N}\text{ql}_i(\beta,\varepsilon) + \ln P(\varepsilon|\tau_2)\right)d\varepsilon$$

Next, we approximate $\text{ql}(\beta,\tau_2)$ using a Laplace approximation, which results in:

$$\tilde{\text{ql}}(\beta,\tau_2) = -\frac{1}{2}\ln|VD + I_N| + \sum_{i=1}^{N}\text{ql}_i(\beta,\tilde{\varepsilon}) - \frac{1}{2}\tilde{\varepsilon}^T V^{-1}\tilde{\varepsilon}\;(+\text{const})$$

where

$$V = \tau_2 I_N, \quad D = \text{diag}([\mu_1,\ldots,\mu_N]), \quad \tilde{\varepsilon} = \arg\max_{\varepsilon}\left(\sum_{i=1}^{N}\text{ql}_i(\beta,\varepsilon) + \ln P(\varepsilon|\tau_2)\right)$$

In the formulation provided by SPARK, the negative sign in $-\frac{1}{2}\ln|VD + I_N|$ is mistakenly omitted, but this does not seem to affect the subsequent derivations, so this is likely a typo.

The derivation is as follows: First, using the Laplace approximation, we get:

$$\text{ql}(\beta,\tau_2) = \ln I$$

where

$$I = \exp\left(\sum_{i=1}^{N} \mathrm{ql}_i(\beta, \tilde{\varepsilon}) + \ln P(\tilde{\varepsilon}|\tau_2)\right)\left(\frac{(2\pi)^N}{|-Hessian|}\right)^{1/2}$$

$$\tilde{\varepsilon} = \arg\max_{\varepsilon} f(\varepsilon), \quad \text{Hessian} = \frac{\partial^2 f(\varepsilon)}{\partial \varepsilon^2}$$

$$f(\varepsilon) = \sum_{i=1}^{N} \mathrm{ql}_i(\beta, \varepsilon) + \ln P(\varepsilon|\tau_2)$$

Then we define:

$$h(\mu_i) = \ln(\mu_i) = \ln N_i + x_i^T \beta + \varepsilon_i$$

Now, we compute the following derivatives:

$$\frac{\partial \mathrm{ql}_i(\beta, \varepsilon)}{\partial \varepsilon_i} = \frac{\partial \mathrm{ql}_i}{\partial \mu_i} \cdot \frac{\partial \mu_i}{\partial h(\mu_i)} \cdot \frac{\partial h(\mu_i)}{\partial \varepsilon_i} = \frac{y_i - \mu_i}{v(\mu_i)h'(\mu_i)}$$

$$\frac{\partial \mathrm{ql}_i(\beta, \varepsilon)}{\partial \varepsilon_j} = 0$$

$$\frac{\partial^2 \mathrm{ql}_i(\beta, \varepsilon)}{\partial \varepsilon_i^2} = -\frac{1}{v(\mu_i)[h'(\mu_i)]^2} + (y_i - \mu_i)\frac{\partial}{\partial \varepsilon_i}\left[\frac{1}{v(\mu_i)h'(\mu_i)}\right]$$

$$\frac{\partial^2 \mathrm{ql}_i(\beta, \varepsilon)}{\partial \varepsilon_i \partial \varepsilon_j} = 0$$

Since we have:

$$P(\varepsilon|\tau_2) = (2\pi)^{-N/2}|V|^{-1/2}\exp\left(-\frac{1}{2}\varepsilon^T V^{-1}\varepsilon\right)$$

we obtain:

$$-Hessian = D + V^{-1} + R$$

where

$$D = \mathrm{diag}\left(\frac{1}{v(\mu_1)[h'(\mu_1)]^2}, \ldots, \frac{1}{v(\mu_N)[h'(\mu_N)]^2}\right) = \mathrm{diag}([\mu_1, \ldots, \mu_N])$$

$$R = -\sum_{i=1}^{N}\left((y_i - \mu_i)\frac{\partial}{\partial \varepsilon}\left[\frac{1}{v(\mu_i)h'(\mu_i)}\right]\right)$$

The expectation of R is zero. Thus, we have:

$$-Hessian = D + V^{-1}$$

Therefore,

$$\tilde{\mathrm{ql}}(\beta, \tau_2) = -\frac{1}{2}\ln|VD + I_N| + \sum_{i=1}^{N} \mathrm{ql}_i(\beta, \tilde{\varepsilon}) - \frac{1}{2}\tilde{\varepsilon}^T V^{-1}\tilde{\varepsilon}(+\text{const})$$

Assume the iterative weights vary slowly with respect to the conditional mean, i.e.,

$$\frac{\partial D}{\partial \mu_i} \approx 0.$$

Based on previous studies [13, 14], we can construct the pseudo data as follows:

$$\tilde{y} = X\beta + \varepsilon + \Delta(y - \mu),$$

where

$$\Delta = \operatorname{diag}([h'(\mu_1), \ldots, h'(\mu_N)]).$$

We can update β and ε by solving the following linear system:

$$\begin{bmatrix} X^T DX & X^T D \\ DX & D + V^{-1} \end{bmatrix} \begin{bmatrix} \beta^{\text{new}} \\ \varepsilon^{\text{new}} \end{bmatrix} = \begin{bmatrix} X^T D\tilde{y} \\ D\tilde{y} \end{bmatrix}.$$

The derivation is as follows. First, we compute the partial derivatives:

$$\frac{\partial \text{ql}(\beta, \tau_2)}{\partial \beta} = X^T D\Delta(y - \mu),$$

$$\frac{\partial \text{ql}(\beta, \tau_2)}{\partial \varepsilon} = D\Delta(y - \mu) - V^{-1}\varepsilon,$$

where

$$\Delta = \operatorname{diag}([h'(\mu_1), \ldots, h'(\mu_N)]).$$

The second derivatives are:

$$\frac{\partial^2 \text{ql}(\beta, \tau_2)}{\partial \beta^2} = -X^T DX,$$

$$\frac{\partial^2 \text{ql}(\beta, \tau_2)}{\partial \beta \partial \varepsilon} = -X^T D,$$

$$\frac{\partial^2 \text{ql}(\beta, \tau_2)}{\partial \varepsilon^2} = -D - V^{-1}.$$

Using the Newton-Raphson method, we write the linear system as:

$$\begin{aligned}
\begin{bmatrix} X^T DX & X^T D \\ DX & D + V^{-1} \end{bmatrix} \begin{bmatrix} \beta^{\text{new}} \\ \varepsilon^{\text{new}} \end{bmatrix} &= \begin{bmatrix} X^T DX & X^T D \\ DX & D + V^{-1} \end{bmatrix} \begin{bmatrix} \beta \\ \varepsilon \end{bmatrix} + \alpha \begin{bmatrix} X^T D\Delta(y - \mu) \\ D\Delta(y - \mu) - V^{-1}\varepsilon \end{bmatrix} \\
&= \begin{bmatrix} X^T D(X\beta + \varepsilon) \\ D(X\beta + \varepsilon) + V^{-1}\varepsilon \end{bmatrix} + \alpha \begin{bmatrix} X^T D\Delta(y - \mu) \\ D\Delta(y - \mu) - V^{-1}\varepsilon \end{bmatrix} \\
&= \begin{bmatrix} X^T D(X\beta + \varepsilon + \alpha\Delta(y - \mu)) \\ D(X\beta + \varepsilon + \alpha\Delta(y - \mu) + (1 - \alpha)V^{-1}\varepsilon) \end{bmatrix}.
\end{aligned}$$

If we choose $\alpha = 1$, we have:

$$\begin{bmatrix} X^T DX & X^T D \\ DX & D+V^{-1} \end{bmatrix} \begin{bmatrix} \beta^{\text{new}} \\ \varepsilon^{\text{new}} \end{bmatrix} = \begin{bmatrix} X^T D[X\beta + \varepsilon + \Delta(y-\mu)] \\ D[X\beta + \varepsilon + \Delta(y-\mu)] \end{bmatrix}.$$

Define

$$\tilde{y} = X\beta + \varepsilon + \Delta(y-\mu),$$

then

$$\begin{bmatrix} X^T DX & X^T D \\ DX & D+V^{-1} \end{bmatrix} \begin{bmatrix} \beta^{\text{new}} \\ \varepsilon^{\text{new}} \end{bmatrix} = \begin{bmatrix} X^T D\tilde{y} \\ D\tilde{y} \end{bmatrix}.$$

Next, we introduce $H = D^{-1} + V$, and we can compute:

$$\hat{\beta} = (X^T H^{-1} X)^{-1} X^T H^{-1} \tilde{y},$$

$$\hat{\varepsilon} = VH^{-1}(\tilde{y} - X\hat{\beta}).$$

To simplify computation, we compute:

$$\hat{\beta} = \left[\sum_{i=1}^{N} \frac{1}{([D^{-1}]_{ii} + \tau_2)} x_i x_i^T \right]^{-1} \left[\sum_{i=1}^{N} \frac{1}{([D^{-1}]_{ii} + \tau_2)} x_i \tilde{y}_i \right].$$

where x_i is the i-th row of X, but we regard it as column vector.

Substituting $\hat{\beta}$ and $\hat{\varepsilon}$ into the quasi-likelihood and evaluating D at $(\hat{\beta}, \hat{\varepsilon})$, and ignoring the dependence of D on τ_2, we replace $\sum_{i=1}^{N} \text{ql}_i(\hat{\beta}, \tilde{\varepsilon})$ with the Pearson chi-squared statistic:

$$\sum_{i=1}^{N} \frac{(y_i - \mu_i)^2}{v(\mu_i)}.$$

Thus, we have:

$$\text{ql}(\hat{\beta}, \tau_2) \approx -\frac{1}{2} \ln |H| - \frac{1}{2} (\tilde{y} - X\hat{\beta})^T H^{-1} (\tilde{y} - X\hat{\beta}) + \text{const}.$$

Since $\hat{\beta}$ is estimated, we account for the loss of degrees of freedom using the restricted maximum likelihood estimate (RMLE) [13, 15]:

$$\text{ql}_R(\hat{\beta}, \tau_2) = -\frac{1}{2} \ln |H| - \frac{1}{2} \ln |X^T H^{-1} X| - \frac{1}{2} (\tilde{y} - X\hat{\beta})^T H^{-1} (\tilde{y} - X\hat{\beta}) + \text{const}.$$

Define

$$P = H^{-1} - H^{-1} X (X^T H^{-1} X)^{-1} X^T H^{-1}.$$

In the original study, P was incorrectly defined as $P = H^{-1} - H^{-1} X^T (X^T H^{-1} X)^{-1} X H^{-1}$.

Using the Newton-Raphson method, we update τ_2 with the average information (AI) algorithm [15]:

First, we have

$$\frac{\partial \,\mathrm{ql}_R(\hat{\beta},\tau_2)}{\partial \tau_2} = \frac{1}{2}\left(\tilde{y}^T P I_N P \tilde{y} - \mathrm{tr}(P I_N)\right) = \frac{1}{2}\left(\tilde{y}^T P P \tilde{y} - \mathrm{tr}(P)\right)$$

and

$$\frac{\partial^2 \,\mathrm{ql}_R(\hat{\beta},\tau_2)}{\partial \tau_2^2} = -\tilde{y}^T P I_N P I_N P \tilde{y} + \frac{1}{2}\mathrm{tr}(P I_N P I_N) = -\tilde{y}^T P P P \tilde{y} + \frac{1}{2}\mathrm{tr}(PP).$$

Then we can calculate the AI as

$$I_{\mathrm{AI}} = \frac{1}{2}\left[-\frac{\partial^2 \,\mathrm{ql}_R(\hat{\beta},\tau_2)}{\partial \tau_2^2} - \mathbb{E}\left[\frac{\partial^2 \,\mathrm{ql}_R(\hat{\beta},\tau_2)}{\partial \tau_2^2}\right]\right] = \frac{1}{2}\tilde{y}^T P P P \tilde{y}.$$

In the original study, I_{AI} was incorrectly defined as $I_{\mathrm{AI}} = -\tilde{y}^T P P P \tilde{y}$. While the missing $\frac{1}{2}$ coefficient is less critical, the incorrect sign leads to non-convergence. This is likely a typo.

Based on the above, we propose the following algorithm:

1. Initialize β and τ_2, and compute pseudo data $\tilde{y}$.
2. Update τ_2 using the AI algorithm.
3. Update β and ε based on τ_2 and $\tilde{y}$.
4. Update $\tilde{y}$ based on β and ε.
5. Repeat steps 2–4 until convergence.

2.4 Score Test

First, we construct the score statistic for each kernel K as follows:

$$S_0 = \frac{1}{2}\tilde{y}^T P K P \tilde{y}.$$

Under the null hypothesis, S_0 follows a mixture of chi-square distributions with 1 degree of freedom, i.e.,

$$\sum_{i=1}^{t} \psi_i \chi_{1i}^2,$$

where ψ_i are the non-zero eigenvalues of $\frac{PK}{2}$. For convenience in calculation, we use the Satterthwaite method [16]. We approximate this mixture using a scaled chi-square distribution $\kappa\chi_\nu^2$, and estimate the two unknown parameters using matching moments.

The mean of S_0 is given by:

$$\kappa\nu = \sum_{i=1}^{t} \psi_i = \frac{1}{2}\mathrm{tr}(PK),$$

and its variance is:

$$2\kappa^2 \nu = 2\sum_{i=1}^{t} \psi_i^2 = \frac{1}{2}\mathrm{tr}(PKPK).$$

Thus, we can compute:

$$\hat{\kappa} = \frac{\mathrm{tr}(PKPK)}{2\mathrm{tr}(PK)},$$

and

$$\hat{\nu} = \frac{\mathrm{tr}(PK)^2}{\mathrm{tr}(PKPK)}.$$

In the original paper, the mean of S_0 was incorrectly written as $\mathrm{tr}(PK)$, but in the code, I found that they corrected this error.

Next, we rescale S_0 as $S_0/\hat{\kappa}$, and compute the p-value for each kernel using the chi-square distribution with $\hat{\nu}$ degrees of freedom.

Thus, we obtain a set of p-values corresponding to different kernels, and use the Cauchy combination rule [12] to combine them into a single test statistic:

$$T = \sum_{k=1}^{n_k} w_k \tan\left((0.5 - p_k)\,\pi\right),$$

where n_k is the number of kernels (set to 10 in this case), w_k is the weight for the k-th Cauchy statistic (defaulted to $\frac{1}{n_k}$), and p_k is the p-value associated with the k-th kernel.

The advantage of the Cauchy combination rule is that it can combine p-values from different tests (or kernels) in a way that accounts for the different sensitivities of each test, often improving the power of the overall test.

Finally, we compute the corresponding p-value for T as follows:

$$p_T \approx \frac{1}{2} - \frac{\arctan\left(\frac{T}{\sum_{k=1}^{n_k} w_k}\right)}{\pi}.$$

For each gene, we obtain a p-value and use the Benjamini-Yekutieli (B-Y) [17] procedure to control the False Discovery Rate (FDR).

3 Results

First, I validated the correctness of my implemented SPARKpy using data sequenced by spatial transcriptomics (ST) [18] from a mouse main olfactory bulb (MOB) section. This dataset was also used in the original SPARK study to explore the performance of SPARK.

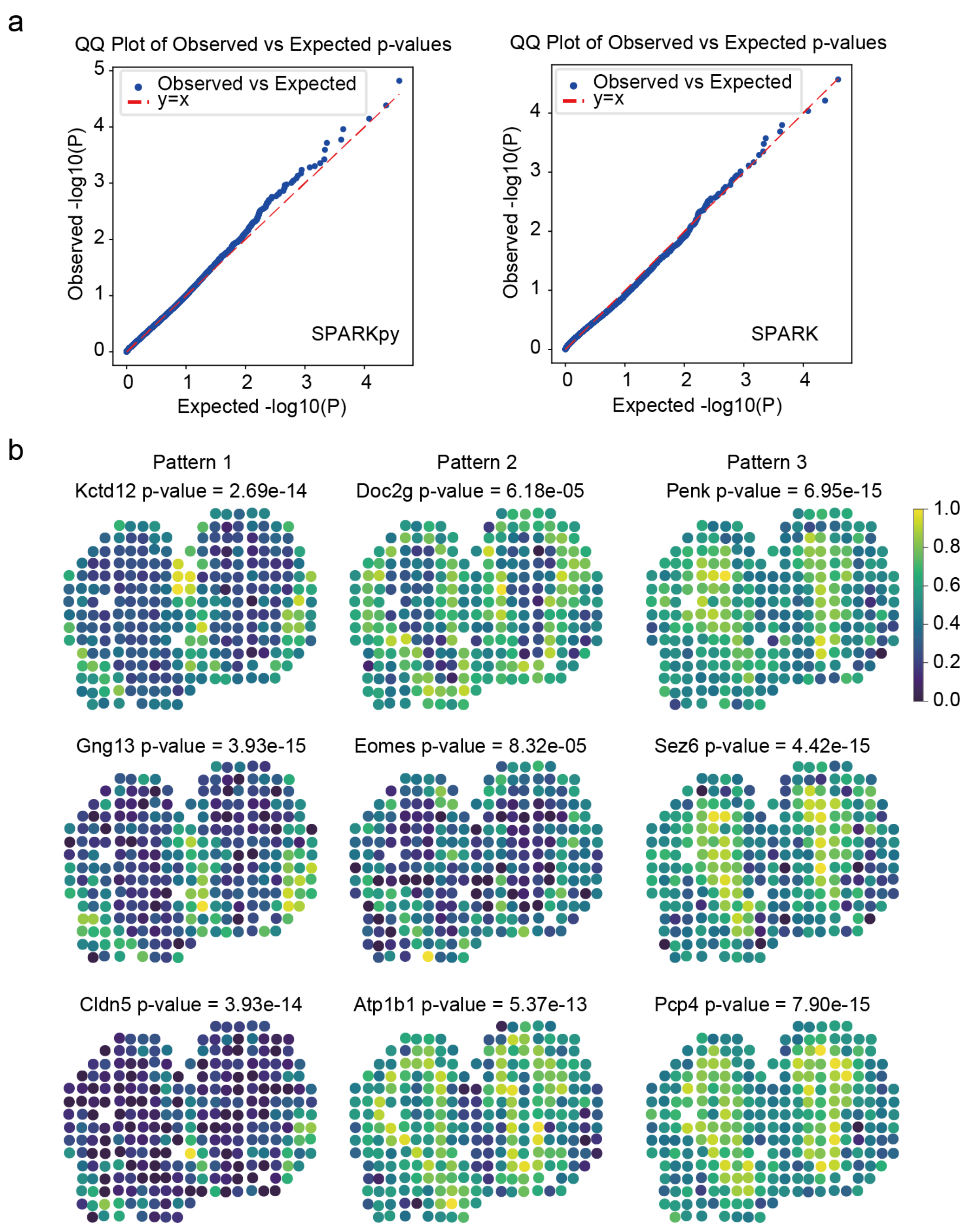

Figure 1: **a**, Quantile–quantile plot of the observed $-\log_{10}(P)$ from SPARKpy and SPARK against the expected $-\log_{10}(P)$ under the null condition in the permuted data. P values were combined across five permutation replicates. **b**, Spatial expression pattern for representative genes across the three layers.

To begin with, a good score statistic should produce well-calibrated p-values that are uniformly distributed between 0 and 1 under the null condition. To verify this, I used the same method as in the SPARK paper to perturb the real data, which effectively disrupted the potential spatial correlation between genes. The perturbed data, satisfying the null condition, was then used to calculate the p-values for each gene using SPARKpy. This process was repeated multiple times, and the results were combined. As shown, under the null condition, the observed p-values were uniformly distributed between 0 and 1, indicating that SPARKpy is capable of producing well-calibrated p-values [Figure 1a]. Similarly, I also performed the same test on SPARK in R using the same settings, and unsurprisingly, I obtained well-calibrated p-values [Figure 1a].

It is worth noting that since the diameter of each spot in this ST dataset is 100 μm, and the distance between the centers of adjacent spots is 200 μm, the resolution is relatively low. Therefore, the MOB region cannot be finely segmented by spots. Instead, it can only be roughly divided into three regions: the glomerular layer (pattern 1), the mitral cell layer (pattern 2), and the granular cell layer (pattern 3) [5]. The SVGs identified by SPARKpy accurately delineate these three patterns [Figure 1b], which include several known marker genes [18]. For example, *Kctd12* is highly expressed in the glomerular layer, *Doc2g* is highly expressed in the mitral cell layer, and *Penk* is highly expressed in the granular cell layer. These genes were all identified as SVGs, and their adjusted p-values were very low, demonstrating the model's confidence in these identifications.

Subsequently, I used a breast cancer dataset [18] [Figure 2a] from the same sequencing technology to explore the capabilities of SPARKpy. This dataset consists of four slices, and SPARK also used this dataset, but only on one slice. However, joint analysis of multiple slices can enhance statistical performance. Since the covariate matrix X can contain more than just an intercept column, but also include batch covariates, I used two slices from this dataset to explore whether adding a batch covariate could effectively address batch effects between slices. This is an important aspect for joint analysis of multiple slices.

First, to establish a relationship between the two slices, they needed to be aligned to a common coordinate system (CCS). I used PASTE2 [19] to align the two slices, setting the overlap value to 0.99. As can be seen, although PASTE2 only performs rigid transformations of the coordinates, it was sufficient to align the two slices well [Figure 2c]. It is worth mentioning that since the distance between spots is 200 μm, the thickness of each slice is 16 μm, and the original experiment selected one slice every four slices, the vertical distance between the slices was approximately 64 μm [18]. Based on this proportion, I assigned appropriate z-coordinates to the two slices to avoid excessively small distances between points.

Although no obvious batch effects were observed between the two slices through UMAP

[Figure2b], when I used SPARKpy to identify SVGs separately in the two slices, I found that the overlap of the SVGs selected from the same gene set was not particularly large [Figure 2d]. Subsequently, I tested whether adding a batch covariate while simultaneously selecting SVGs from both slices would improve the results [Figure 2d]. I found that some marker genes were selected as SVGs in all four experiments, such as the gene *FN1*. However, for the gene *CD44*, while it was selected as an SVG in all four experiments, the *p*-value was significantly smaller in the experiment with the batch covariate compared to the other three groups. Moreover, *ARPC5* and *COX7C*, which were not selected as SVGs in the first three experiments due to higher *p*-values, were identified as SVGs when the batch covariate was included [Figure 2e]. The spatial expression patterns of these genes were also very pronounced, proving that adding a batch covariate indeed helps with the integration of multiple slices.

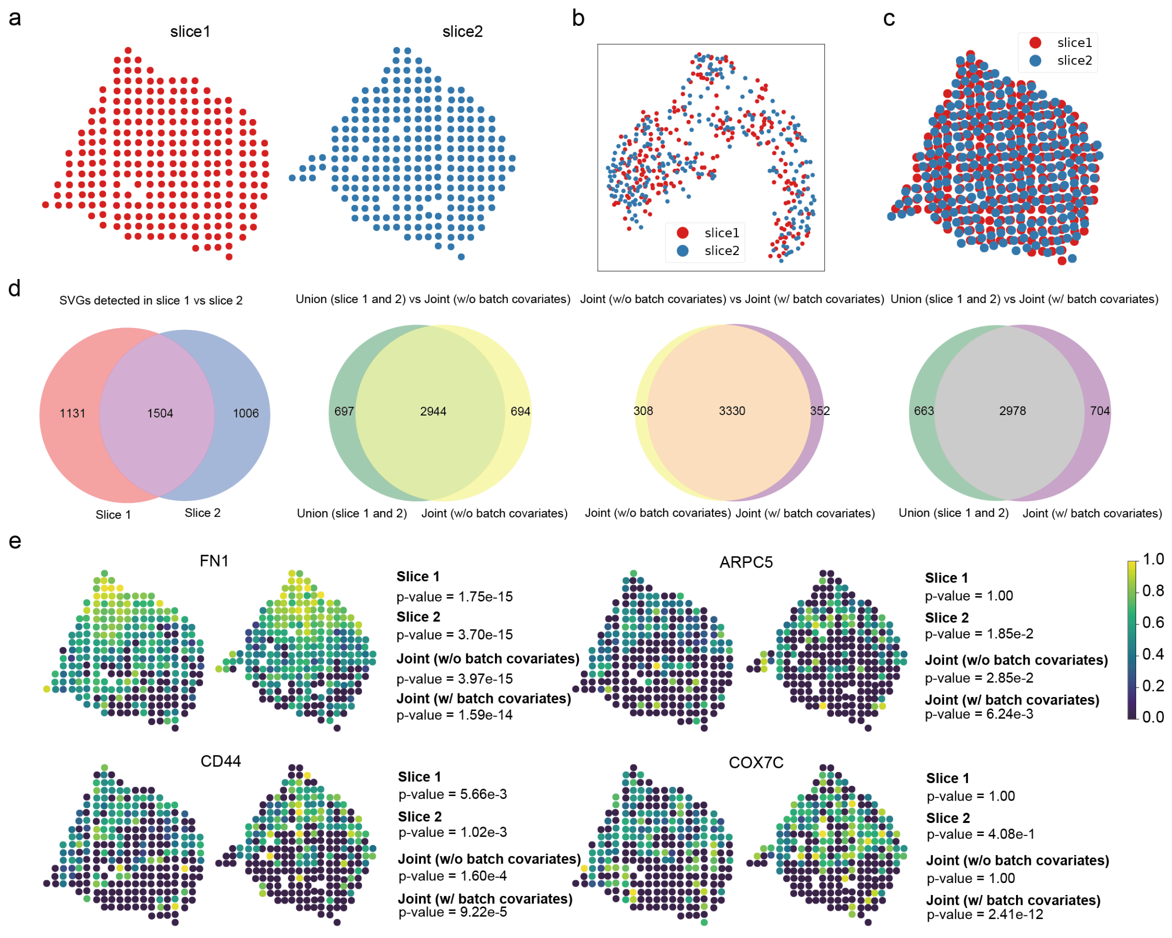

Figure 2: **a**, Two breast cancer tissue slices. **b**, UMAP of two breast cancer tissue slices. **c**, Slices aligned to a common coordinate system using PASTE2. **d**, Venn diagram of the overlap between SVGs identified in different groups. **e**, Spatial expression patterns of some identified SVGs and their p-values across different groups.

4 Some Comments

Overall, this method indeed has many advantages. For instance, the score statistic of this method can produce well-calibrated p-values, which the authors of the original paper highlighted as a feature that many other methods fail to achieve. This is, in fact, a necessary prerequisite for valid hypothesis testing. Moreover, the interpretability of this method is strong, and it can identify SVGs with practical significance.

However, this method also has some problems. The most significant problem is that the authors introduced many manual modifications into the corresponding code of this method, which ideally should not exist in this method. These modifications are essentially applied externally to the method itself, actively altering the results. I wrote the code mainly according to the methods described in the supplementary material of the paper, with very few additional manual modifications. One major modification I made was in calculating the parameters for the approximate scaled chi-square distribution. The original code had a correction for the variance term of the score statistic. I found that if this correction was not added, the p-values obtained under the null distribution would not be well-calibrated. Therefore, I included this correction in the code. However, this correction was not mentioned in the methods section of the original paper.

To be honest, although the algorithm I wrote also produces well-calibrated p-values, when testing on actual data, I found that SPARKpy identified far more SVGs than SPARK at the same FDR level. Since manual modifications are not limited to just a few places in the original code, I am unsure which part is responsible for this difference. I believe that methods requiring so many manual modifications should be avoided. Therefore, SPARKpy cannot actually be considered a replacement for SPARK in Python, although there is no evidence to suggest that one method produces better results than the other.

Secondly, this method consumes considerable time, especially when incorporating the covariate matrix. Although some parts of the code have been accelerated using C++, it still requires a long time to run. This issue is also pointed out in the later version, SPARK-X [6].

Nevertheless, after reviewing this method, I genuinely admire the authors' capabilities in finding appropriate methods for each step that needs to be executed.

References

[1] Dario Bressan, Giorgia Battistoni, and Gregory J Hannon. The dawn of spatial omics. *Science*, 381(6657):eabq4964, 2023.

[2] Dylan M Cable, Evan Murray, Luli S Zou, Aleksandrina Goeva, Evan Z Macosko, Fei Chen, and Rafael A Irizarry. Robust decomposition of cell type mixtures in spatial transcriptomics. *Nature biotechnology*, 40(4):517–526, 2022.

[3] Ying Ma and Xiang Zhou. Spatially informed cell-type deconvolution for spatial transcriptomics. *Nature biotechnology*, 40(9):1349–1359, 2022.

[4] Gefei Wang, Jia Zhao, Yan Yan, Yang Wang, Angela Ruohao Wu, and Can Yang. Construction of a 3d whole organism spatial atlas by joint modelling of multiple slices with deep neural networks. *Nature Machine Intelligence*, 5(11):1200–1213, 2023.

[5] Shiquan Sun, Jiaqiang Zhu, and Xiang Zhou. Statistical analysis of spatial expression patterns for spatially resolved transcriptomic studies. *Nature methods*, 17(2):193–200, 2020.

[6] Jiaqiang Zhu, Shiquan Sun, and Xiang Zhou. Spark-x: non-parametric modeling enables scalable and robust detection of spatial expression patterns for large spatial transcriptomic studies. *Genome biology*, 22(1):184, 2021.

[7] Zhuoqing Fang, Xinyuan Liu, and Gary Peltz. Gseapy: a comprehensive package for performing gene set enrichment analysis in python. *Bioinformatics*, 39(1):btac757, 2023.

[8] Xiang Zhou, Kangning Dong, and Shihua Zhang. Integrating spatial transcriptomics data across different conditions, technologies and developmental stages. *Nature Computational Science*, 3(10):894–906, 2023.

[9] Ian Covert, Rohan Gala, Tim Wang, Karel Svoboda, Uygar Sümbül, and Su-In Lee. Predictive and robust gene selection for spatial transcriptomics. *Nature Communications*, 14(1):2091, 2023.

[10] Yi Li, Haicheng Tang, and Xihong Lin. Spatial linear mixed models with covariate measurement errors. *Statistica Sinica*, 19(3):1077, 2009.

[11] K Ben-Ahmed, A Bouratbine, and M-A El-Aroui. Generalized linear spatial models in epidemiology: A case study of zoonotic cutaneous leishmaniasis in tunisia. *Journal of Applied Statistics*, 37(1):159–170, 2010.

[12] Yaowu Liu, Sixing Chen, Zilin Li, Alanna C Morrison, Eric Boerwinkle, and Xihong Lin. Acat: a fast and powerful p value combination method for rare-variant analysis in sequencing studies. *The American Journal of Human Genetics*, 104(3):410–421, 2019.

[13] Norman E Breslow and David G Clayton. Approximate inference in generalized linear mixed models. *Journal of the American statistical Association*, 88(421):9–25, 1993.

[14] Shiquan Sun, Jiaqiang Zhu, Sahar Mozaffari, Carole Ober, Mengjie Chen, and Xiang Zhou. Heritability estimation and differential analysis of count data with generalized linear mixed models in genomic sequencing studies. *Bioinformatics*, 35(3):487–496, 2019.

[15] Arthur R Gilmour, Robin Thompson, and Brian R Cullis. Average information reml: an efficient algorithm for variance parameter estimation in linear mixed models. *Biometrics*, pages 1440–1450, 1995.

[16] Franklin E Satterthwaite. An approximate distribution of estimates of variance components. *Biometrics bulletin*, 2(6):110–114, 1946.

[17] Yoav Benjamini and Daniel Yekutieli. The control of the false discovery rate in multiple testing under dependency. *Annals of statistics*, pages 1165–1188, 2001.

[18] Patrik L Ståhl, Fredrik Salmén, Sanja Vickovic, Anna Lundmark, José Fernández Navarro, Jens Magnusson, Stefania Giacomello, Michaela Asp, Jakub O Westholm, Mikael Huss, et al. Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. *Science*, 353(6294):78–82, 2016.

[19] Xinhao Liu, Ron Zeira, and Benjamin J Raphael. Paste2: partial alignment of multi-slice spatially resolved transcriptomics data. *bioRxiv*, 2023.