**A Hidden Markov Random Field Model for Detecting Domain Organizations from Spatial Transcriptomic Data**

**Author**

Qian Zhu

**Affiliation**

Dana-Farber Cancer Institute, 360 Longwood Ave, Boston, MA, 02215, USA

**Corresponding author**

Qian Zhu, zqian@jimmy.harvard.edu

**Running Head**

HMRF for detecting domain organizations

**Abstract**

Cells in complex tissues are organized by distinct microenvironments and anatomical structures. This spatial environment of cells is thought to be important for division of labor and other specialized functions of tissues. Recently developed spatial transcriptomic technologies enable the quantification of expression of hundreds of genes while accounting for cells’ spatial coordinates, providing an opportunity to study spatially organized structures. Here, we describe a computational pipeline for detecting the spatial organization of cells based on a hidden Markov random field model. We illustrate this pipeline with data generated from multiplexed smFISH from the adult mouse visual cortex.

**Key words**

Hidden Markov random field, Spatial organization, Sequential fluorescence in situ hybridization, Multiplexed fluorescence in situ hybridization

**1 Introduction**

Determining cell types has been a crucial goal of single-cell transcriptomic profiling[1–4]. However, often within cell types, there is also cellular heterogeneity that is attributed to the cells’ distinct microenvironment and spatial context, which is often not well understood. The spatial environments come in various forms and scales in mammalian tissues. For example, the mammalian liver is divided into multiple lobules along the portal field where the metabolic tasks of the tissue are spatially assigned[5]. In the brain, cells are spatially distributed across various anatomical structures, or spatial domains, each of which is uniquely associated with distinct cognitive functions and behavior[6]. Understanding the principle of spatial division of cells requires spatially profiling technologies. Recently, development of single-molecule fluorescence in situ hybridization (smFISH)[7–11] has enabled the detection of mRNA transcripts while maintaining the spatial context. In contrast to scRNAseq studies which require cells to be dissociated from their physical context, the smFISH technique faithfully captures cells’ spatial coordinates by imaging. Furthermore, when smFISH is combined with sequential rounds of imaging (seqFISH[8, 11], MERFISH[9]), it is now possible to profile hundreds of genes for each cell while recording each cell’s coordinates. With spatial genomics technologies, we can now begin to examine distinct microenvironment niches and cells’ interactions within these microenvironments.

One critical question that can now be answered with spatial genomics data is to determine the spatial gene expression domains[11, 12]. Conventional methods, such as K-means, cluster cells solely on expression and ignore the spatial relationships between cells. Thus simply overlaying the cluster annotations on cells’ co-ordinates results in a noisy, inaccurate representation of spatial domain structure. Recently, we have developed an approach based on hidden Markov random field (HMRF) to dissect the spatial domain structure [12]. The approach balances intrinsic cellular expression and extrinsic neighborhood effects to probabilistically assign domain states to single cells. In this chapter, we describe a pipeline for performing a HMRF analysis using a mouse brain data set as an illustrating example.

**2 Materials**

**2.1 Prerequisites**

We require R version 3 and Python 2.7. The following Python prerequisite packages need to be installed: seaborn (0.7.0 or up), pandas, numpy, scipy, matplotlib. Use pip --user --install <package name> to install any missing packages. The following R packages are also required: lattice, misc3d, oro.nifti, pracma, Matrix, mvtnorm. We require JAVA (version 7 or 8) and GraphColoring package.

**2.2 R smfishHmrf package**

Obtain and install the smfishHmrf R package:

install.packages("devtools")

library(devtools)

install\_bitbucket("qzhudfci/smfishhmrf-r", ref="default")

**2.3 Python smfishHmrf package**

This contains the wrapper and interface functions for interacting with the R smfishHmrf package and is required for running HMRF and downstream visualizations. Install by:

pip install --user smfishHmrf

**2.4 Spatial single-cell data set**

Our pipeline is general to all types of spatial transcriptomic data. For this chapter, we focus on a mouse visual cortex data set generated by the seqFISH technology. This input data set is a mouse coronal brain slice that has been imaged and is composed of various sections of the hippocampus and visual cortex tissues (**Figure 1**). Each section, also called field, measures 1020x1020 units (each unit is equivalent to 220nm). These sections are imaged and processed together with two pieces of information provided: (1) cell coordinates (two versions: relative to each field, and relative to a stitched image which stitches adjacent fields in cortex), and (2) cellular gene expression for 125 genes.

In general, we require the cell coordinate file and the gene expression file to carry out a spatial domain inference analysis. The specifications of these two files are as follows.

The cell coordinate file is made up of 4 headerless columns, separated by space: <cell index> <field ID> <x-coord> <y-coord>. All fields must be numerical. Cell index=1…N where N is the number of cells. Field ID specifies the field of view the cell is located in. X-coord, y-coord specify the coordinates of the cell in the respective field of view. Coordinates can be floating point decimals and can be negative. See a snippet of this file below:

1 0 675.080 -37.330

2 0 265.760 -231.140

3 0 753.460 -261.140

4 0 290.480 -261.520

5 0 991.430 -482.350

6 0 926.420 -675.880

7 0 414.500 -688.670

8 0 607.180 -773.680

9 0 715.720 -822.110

10 0 654.580 -896.760

11 0 472.450 -952.830

12 0 257.120 -133.350

13 0 700.010 -169.050

14 0 415.630 -252.450

The gene expression file is a space-separated matrix with rows being cells and the columns being genes. The order of cells (or rows) must be consistent with the order of cells in the coordinate file. Similarly, genes (or columns) are arranged in the order specified by a separate gene order file (genes.txt). First column is the row name, i.e. the cell index (equal to 1, 2, …, N). Note that there is no column header, so the first cell starts at the first line of the file. A snippet is shown below:

1 1.08 0.60 0.95 0.51 -1.67 0.65 1.14 0.79 0.61 0.18 0.74 0.62 0.86 0.61 0.59 0.34 -2.15 0.20 0.87 0.68 0.00 0.21 -0.00 0.49 -0.43 1.07 0.55 -1.32 0.10 1.31 1.28 0.18 0.63 0.03 0.52 0.47 0.55 0.98 0.33 0.64 -0.18 1.21 1.67 -0.37 1.04 0.11 -0.63 0.89 -0.39 -2.28 -0.02 0.66 0.92 -0.81 -0.39 -1.02 0.61 -0.05 -0.01 0.61 -0.01 -0.38 -0.05 1.52 -1.25 0.13 0.38 0.70 -0.02 0.11 -0.49 -0.35 -1.47 0.30 0.13 0.39 0.45 -2.69 0.14 0.74 0.24 -0.60 -1.08 -0.65 0.70 0.03 -1.56 -0.01 0.30 0.28 1.55 -0.25 -0.33 -0.10 -1.55 0.06 -0.77 0.41 -0.75 0.17 -1.35 -0.65 -2.06 1.42 -0.73 -2.55 -1.04 -1.35 -0.71 -0.76 -1.01 -1.69 -1.70 3.26 0.44 -1.65 -0.72 -1.40 -1.27 0.83 -0.74 -0.70 -0.05 2.65 1.76

2 1.82 1.60 2.38 -0.02 -0.26 1.94 0.08 0.07 0.91 -0.20 1.57 -0.08 0.06 1.53 0.40 0.18 0.41 1.44 0.19 0.01 0.19 -0.17 -0.26 0.06 1.18 0.02 0.01 0.06 -0.31 -0.05 -0.64 -0.01 -0.17 -0.58 1.31 0.94 0.59 1.58 -0.01 -0.10 -0.13 0.05 -0.20 0.00 -0.07 -0.07 0.19 -0.23 -0.02 -0.35 -0.68 1.27 -0.21 -0.23 -0.49 0.87 -0.33 0.41 1.25 -0.15 -0.70 -0.29 -0.47 1.98 0.01 -0.19 0.06 0.96 -0.23 -0.21 -0.19 -0.39 -0.85 -0.25 0.06 -0.40 -0.63 -0.30 -0.27 -0.29 -0.61 -0.08 -0.61 0.28 0.30 0.03 1.02 0.22 1.27 1.13 2.08 0.18 -0.22 1.31 -0.17 0.82 -0.15 1.74 -0.40 -0.40 -0.23 -2.36 -1.55 0.06 -0.97 -1.79 -1.38 -0.95 -1.69 -1.91 -2.38 -2.60 -1.49 2.32 0.16 0.05 -1.41 -2.14 -1.38 0.23 -2.20 -1.60 -1.40 2.42 0.54

Note that the gene expression values above have undergone 1) log-transformations, and 2) gene- and cell-wise z-scoring, from the raw mRNA counts. Thus, the values are expression z-scores and are approximately normally distributed. We recommend this normalization for our spatial transcriptomic data set, but it also works on other data sets such as MERFISH [9], tissue microarray technologies[13].

**3 Methods**

**3.1 Background: hidden Markov random field model**

We recently developed a hidden Markov random field (HMRF) method to detect major spatial patterns of expression from single cell data. HMRF is a probabilistic method for pattern recognition[14–16]. Given a user input of states, the technique classifies each cell as belonging to one of the states based on gene expression and spatially neighboring cells.

Briefly, let represent the cells in the image domain. Let be the local neighborhood graph which defines the nodes that are neighbors of each other. Every cell is associated with expression value 1…125. Let be a classification function = {, for each cell }. Overall, the posterior probability of class assignment is given by:

(1)

where defines the conditional probability given the class configuration of the neighbors . The term is the probability of observing the expression xi given the class ’s Gaussian distribution function . Detailed mathematical explanations can be found in the original paper [12]. It suffices to say that models the extrinsic influence, or the environment made up of ’s surrounding cells, while is the intrinsic component (i.e. assigns probability based on the expression identity ). The energy function that is used to model has the Pots’ model, where the energy potential V is the sum of compatible pairwise interactions in the node ’s immediate neighborhood:

(2)

is a weighting constant. Parameters of the model including those in and are jointly estimated by expectation maximization procedure.

**3.1.1 Gene selection**

Selection of spatially coherent genes can aid the HMRF modeling. To help us find spatially coherent genes, we define a score as follows. For each gene we divide all the cells based on its bias-corrected gene expression into 2 classes: 1 - expressed class which corresponds to 90-th percentile in the gene’s expression distribution, and 0 - the remaining cells. We use silhouette metric to measure how spatially coherent is the 1-marked cells. Here, we used the rank-normalized, exponentially transformed distance to emphasize the local physical distance between cells. For a pair of cells, and , this distance is defined as:

(3)

where is the mutual rank[17] of and in the vectors of Euclidean distances and . is a rank-weighting constant set between 0.95 and 0.99. The spatial coherence of the gene is calculated as the Silhouette coefficient of the spatial distance between the two cell sets:

(4)

Where for a given cell in , is the average distance between and any cell in , and is defined as the average distance between and any other cell in .

**3.1.2 Neighborhood graph construction**

The spatial relationship between the cells is represented as an undirected graph, where each node is a cell and each edge connects a pair of neighboring cells if they are no further than a certain Euclidean distance apart. This distance threshold is a parameter of the graph construction function. We recommend setting the distance such that on average there are around 5-10 neighbors per cell.

**3.1.3 Number of clusters**

We initialize HMRF according to the K-means clustering results. The value of is selected on the basis of gap-statistics.

**3.2 Application: mouse visual cortex data set**

1. We first import the relevant packages.

import sys

import math

import os

import numpy as np

import scipy

import scipy.stats

import pandas as pd

from scipy.stats import zscore

from scipy.spatial.distance import euclidean,squareform,pdist

import smfishHmrf.reader as reader

from smfishHmrf.HMRFInstance import HMRFInstance

from smfishHmrf.DatasetMatrix import DatasetMatrix, DatasetMatrixSingleField

from smfishHmrf.bias\_correction import calc\_bias\_moving, do\_pca, plot\_pca

import smfishHmrf.visualize as visualize

import smfishHmrf.spatial as spatial

2. Illumination bias correction is recommended for this data set. Illumination bias refers to biases arising from the optical instrument which may produce images with skewed intensity in certain regions of the imaged field. We systematically detect and remove such imaging bias using a multiple field, smoothing average approach. The idea starts by realizing that the tissue cannot be imaged all in one time, but rather it is imaged in small subsections (fields). By comparing the intensity of the same spot across multiple fields, we can deduce an average bias for that spot for correction. This approach computes an average field bias vector by first overlaying all field images and apportioning cells into bins on a regular grid. The bias for a bin and a gene is the total expression of cells in and the 4 neighboring bins of across all images, divided by the number of cells in those bins, producing a smoothed estimate.

expr\_bias = calc\_bias\_moving(expr=expr, position=Xcen, field=field, interest\_field=FDs, num\_bin=50)

We next model the bias pattern of all genes using PCA (do\_pca function). The contributions of the top principal components are subtracted from the expression matrix (expr). The result is contained in corrected\_expr.

corrected\_expr = do\_pca(expr=expr, expr\_bias=expr\_bias, centroid=Xcen, field=field, interest\_field=FDs, num\_bin=50, top\_comp\_remove=5)

We plot the top PCs representing the orthogonal bias patterns (**Figure 2**).

plot\_pca(expr\_bias=expr\_bias, num\_bin=50, top\_pc=5, out\_file="bias\_component.png")

Note that if the expression matrix is already corrected for this bias, skip this step and go to step 3.

3. Load the bias corrected gene expression matrix, and the relevant coordinate file. To save time, we provide the bias-corrected expression and coordinate files in a URL for download <https://bitbucket.org/qzhudfci/smfishhmrf-py/src/default/data/>. Download the content in a directory. Change the variable directory to point to the directory containing these files.

directory = "workdir"

genes = reader.read\_genes("%s/genes" % directory)

Xcen, field = reader.read\_coordinates("%s/fcortex.coordinates.txt" % directory)

expr = reader.read\_expression\_matrix("%s/fcortex.expression.txt" % directory)

The expression matrix (expr, a 2D numpy array) consists of cells from the visual cortex region. Xcen and field are encoded as numpy 2D array and numpy 1D array respectively.

4. Construct a DatasetMatrixSingleField instance. DatasetMatrixSingleField is a class that encapsulates all information about a spatial transcriptomic data set. We initiate an instance with expression, gene names, cell annotations (set to None) and coordinates (Xcen).

this\_dset = DatasetMatrixSingleField(expr, genes, None, Xcen)

5. Construct a local neighborhood graph. We compute the cell pairwise Euclidean distance matrix using the cell coordinates in Xcen. Then we determine a cut-off on Euclidean distance such that a pair of cells separated by less than the cut-off distance is assigned an edge. This cut-off can be expressed as top X-percentile of Euclidean distances. For example, 0.30 means a cutoff that is equal to 0.30% of all Euclidean distance values. In this example, we settle on 0.30% as this cutoff produces on average 5 neighbors per cell (see output in **Note 1**). Use test\_adjacency\_list to test a number of cut-off values (see output in **Note 1**).

this\_dset.test\_adjacency\_list([0.3, 0.5, 1], metric="euclidean")

this\_dset.calc\_neighbor\_graph(0.3, metric="euclidean")

6. We compute the independent regions of the graph (see **Note 2**) - this step is required for the node update during the iterative HMRF parameter estimation.

this\_dset.calc\_independent\_region()

7. Spatial gene selection. As a sanity check, we are interested in knowing which genes might have a spatial pattern. A spatial coherence score has been defined in Equation 4. We first calculate a dissimilarity matrix based on the rank, exponential transformed Euclidean distance matrix between all cells’ coordinates. This gives more emphasis in the distances between cells that are close to each other (Equation 4).

euc = squareform(pdist(Xcen, metric="euclidean"))

dissim = spatial.rank\_transform\_matrix(euc, reverse=False, rbp\_p=0.95)

Then we perform the silhouette calculation. Essentially, for each gene, expressions are divided into 1’s (expressed) or 0’s (not expressed). Silhouette metric is calculated to assess the spatial distribution for the expressed cell group (see **Note 3** for advanced usage of spatial.calc\_silhouette\_per\_gene). A statistical P-value is reported per gene.

res = spatial.calc\_silhouette\_per\_gene(genes=genes, expr=expr, dissim=dissim, examine\_top=0.1, permutation\_test=True, permutations=100)

print "gene", "sil.score", "p-value"

for i,j,k in res:

print i,j,k

See the silhouette score output in **Note 4**. We select a P value cutoff in choosing spatial genes, to give about 90 genes. Though, users can impose additional gene restriction criterion, which we did in this example (see next step for additional restriction).

res\_df = pd.DataFrame(res, columns=["gene", "sil.score", "pval"])

res\_df = res\_df.set\_index("gene")

new\_genes = res\_df[res\_df.pval<=0.05].index.get\_values().tolist()

print new\_genes

Some examples of spatial genes are below and we can visualize them using the following code (see output in **Figure 3**).

for g in ["calb1", "acta2", "tbr1"]:

visualize.gene\_expression(this\_dset, goi=g, vmax=2.0, vmin=0, \

title=True, colormap="Reds", size\_factor=5, dot\_size=20, \

outfile="%s.png" % g)

8. Optionally, users may further remove cell-type specific genes from the spatial gene list. This is particularly helpful if certain cell types are not known to form any spatial patterns, such as astrocytes and microglia, or if users wish to remove cell-type variations so as to focus solely on spatial variation in the data. In our case, we have determined a list of cell-type specific genes strongly associated with the 8 major cell types in the cortex (based on the Tasic et al scRNAseq data[18]) (see **Note 5**), and removed these genes from the spatial gene list from the previous step. This results in a 69-gene list.

Optionally, users may also select genes based on which genes are correlated to top principal components from PCA analysis. See the spatial.pc\_genes() function, which computes the significant genes of each component with the jacksaw algorithm [19]. Users can combine this criterion and the spatial criterion in defining suitable genes for HMRF.

9. Once we decide on the genes to use for HMRF, we load this list (HMRF.genes file), and we create a data subset using these genes with the subset\_genes function

new\_genes = reader.read\_genes("%s/HMRF.genes" % directory)

new\_dset = this\_dset.subset\_genes(new\_genes)

10. Initiate HMRF instance. We create an output directory. Then we construct an instance of the HMRFInstance class, which is a special class encapsulating all information about a HMRF analysis. Constructor of HMRFInstance requires <run name> <output directory> <DatasetMatrix instance> <K> <initial beta> <beta increment> <number of betas> <tolerance>. This construction is designed to iterate HMRF over many betas.

print "Running HMRF..."

outdir = "spatial.jul20"

if not os.path.isdir(outdir):

os.mkdir(outdir)

this\_hmrf = HMRFInstance("cortex", outdir, new\_dset, 9, 0, 0.5, 30, tolerance=1e-20)

As the above shows, HMRF is set to run for 30 times, starting at beta=0, and at 0.5 increment (this covers beta=0 up to and excluding beta=15.0). Repeatedly running HMRF helps users see the changes in the spatial domain structure as the smoothing parameter beta is increased. Because at this stage it is uncertain what is the best beta, we need to assess all betas. Then, selecting the best beta depends on the actual data set and how the spatial pattern looks, and we discuss a beta selection guideline in step 14.

11. Run HMRF.

this\_hmrf.init(nstart=1000)

this\_hmrf.run()

Init() and run() are wrapper functions for R scripts where the core of HMRF is implemented. Init determines initial conditions by running K-means to determine cluster centroids and covariance matrices, and initializing HMRF to these settings. Run performs the HMRF modeling, including an expectation-maximization procedure to iteratively estimate the parameters of the HMRF model, until convergence criteria is met. Run HMRF will iterate over all K’s and all beta’s. At the end, files will be automatically generated in the output directory (see **Note 6**), with a copy of the results loaded in the class instance.

12. Next, we visualize the spatial clusters in 2D. **Figure 4** shows the result for K=9, beta=9.0 and indicates an resemblance of the structure to the visual cortex layers.

visualize.domain(this\_hmrf, 9, 9.0, dot\_size=45, size\_factor=10, outfile="visualize.beta.%.1f.png" % 9.0)

13. To check if the detected spatial domains are significant, we compare it to a case where the spatial positions of the cells are fully shuffled (or randomly permuted). We first create a randomly permuted data set by shuffling the cells in the original matrix. The parameter 0.99 in the instance method shuffle() is the shuffling proportion (see **Note 7**).

print "Running pertubed HMRF..."

outdir = "perturbed.jul20"

if not os.path.isdir(outdir):

os.mkdir(outdir)

perturbed\_dset = new\_dset.shuffle(0.99)

This returns a new data set of DatasetMatrix class. We initiate a new HMRF instance using the same parameters as the original case, and run HMRF on this instance.

perturbed\_hmrf = HMRFInstance("cortex", outdir, perturbed\_dset, 9, 0, 0.5, 30, tolerance=1e-20)

perturbed\_hmrf.init(nstart=1000)

perturbed\_hmrf.run()

14. We compare the log-likelihood of the model in the random case and original case.

k=9

betas = np.array(range(0, 90, 5) + range(90, 150, 10)) / 10.0

lik\_data, diff\_data = [], []

for b in betas:

lik\_data.append((b, "observed", this\_hmrf.likelihood[(k,b)]))

lik\_data.append((b, "random", perturbed\_hmrf.likelihood[(k,b)]))

diff\_data.append((b, "obs - rand", this\_hmrf.likelihood[(k,b)] – \

perturbed\_hmrf.likelihood[(k, b)]))

a\_lik = pd.DataFrame(data={"label":[v[1] for v in lik\_data], "beta":[v[0] for v in lik\_data], "log-likelihood":[v[2] for v in lik\_data]})

d\_lik = pd.DataFrame(data={"label":[v[1] for v in diff\_data], "beta":[v[0] for v in diff\_data], "log-likelihood":[v[2] for v in diff\_data]})

axn = sns.lmplot(x="beta", y="log-likelihood", hue="label", data=a\_lik, fit\_reg=False)

axn = sns.lmplot(x="beta", y="log-likelihood", hue="label", data=d\_lik, fit\_reg=False)

Here the output shows the log-likelihood of randomized and observed domain patterns independently (**Figure 5**). Log-likelihood is a measure of model fitting and is a function of the number of parameters in the model, beta, among others. The log-likelihood generally increases as beta increases, showing that the model tries to aggressively smooth the state by using the neighboring cell states. As log likelihood by itself is less interpretable, we compare it to the shuffled case, and check which beta is the difference largest. The difference between observed and random cases is largest at around beta=6.0 to 9.0 (**Figure 5**). This result likely suggests the following phenomenon: at low beta, the model predominantly uses the cell’s own expression to determine the cell state, and is no different from K-means. As beta reaches a critical point, neighboring cells collective influence the cell sate to favor a coherent domain like pattern, thereby significantly increases the probability compared to the random case (which is unable to form domains at this critical point due to the lack of spatial relationships). Overall, this critical point may serve as a selection guideline for beta.

15. We provide the entire pipeline as a script for download (see file test\_jul20.py at <https://bitbucket.org/qzhudfci/smfishhmrf-py>). To further facilitate usage and streamline the process, we allow users to define a settings file, which would include input and output files and directories, and various HMRF settings. This settings file is to be used with the pipeline script to obviate the need to modify the script for a new data set (see smfishHmrf-py manual for details).

**4 Notes**

1. Output:

cutoff:0.30% #nodes:1597 #edges:3852 avg.nei:4.82

cutoff:0.50% #nodes:1597 #edges:6384 avg.nei:7.99

cutoff:1.00% #nodes:1597 #edges:12745 avg.nei:15.96

cutoff:2.00% #nodes:1597 #edges:25489 avg.nei:31.92

cutoff:5.00% #nodes:1597 #edges:63721 avg.nei:79.80

1. Getting the independent regions of the neighborhood graph is required for the parameter estimation step. This s so that neighboring nodes are not updated one after the other. We turn to the graph coloring problem in computer science to determine the color of the nodes in the graph where neighbors have different colors[20]. The node colors determine the node update order in the EM procedure. By finding equivalent nodes across different independent regions, the aim is to improve consistency of estimates and parallelization.
2. The spatial coherence scoring function calc\_silhouette\_per\_gene has a parameter examine\_top to control the proportion of cells (0-1.0) on which to measure coherence. By default, this is 0.1, meaning that the top 10% of cells expressing the gene are used for spatial coherence calculation. It can be increased if the user thinks the spatial pattern is located within a larger proportion.
3. Silhouette output:

gene sil.score p-value

amigo2 0.0555237 0.0

cldn5 0.0262171 0.0

calb1 0.0189745 0.0

kcnip 0.0187823 0.0

tbr1 0.0173751 0.0

pax6 0.0169212 0.0

nes 0.0156901 0.0

gda 0.0150629 0.0

col5a1 0.0148561 0.0

loxl1 0.0120843 0.0

sox2 0.011049 0.0

slc5a7 0.00993408 0.0

nov 0.00985005 0.0

itpr2 0.00915686 0.0

cpne5 0.00913211 0.0

Nell1 0.00875134 0.0

mrc1 0.00864791 0.0

rhob 0.00830748 0.0

acta2 0.00802404 0.0

...

Foxa1 0.000510314 0.28

Zfp715 0.000449031 0.34

Galnt3 0.000186029 0.49

Blzf1 -0.000113328 0.64

Laptm5 -0.000492621 0.81

Gm6377 -0.000751389 0.87

Zfp90 -0.00092005 0.91

1. Tasic et al[18] is a complementary scRNAseq data set (GSE71585) which defined 8 major cell types in the visual cortex, including astrocytes, microglia, endothelial cells, 3 oligodendrocyte clusters, GABA-ergic and glutamatergic neurons. We use this data set to determine cell-type specific genes to be removed from HMRF gene list. The goal is to remove these genes from HMRF they are already explained by their cell-type variation. For each gene that is overlapping with our seqFISH data set (i.e. 125 genes) and which is differentially expressed (DE) across cell types, we calculate the gene’s average expression z-scores per cell type, as shown in the heat-map (**Figure 6**) for 43 DE genes. Star indicates strongly DE genes (avg. expr z-score > 2.0), and are selected to be removed from HMRF.
2. The output directory contains many files organized by the beta value, such as the probability estimates, the converged centroids, and the covariance matrices. The file \*.prob.txt is a matrix of domain probabilities where each row is a cell and each column is a domain. The assigned domain for each cell is the domain with highest probability value of each row. Results are divided among directories based on K, then within directory by beta.
3. The shuffling proportion (0-1.0) is the proportion of cells that swap positions. For example, 0.99 means that the shuffling rate is such that 99% of cells exchanged position compared to the original data. A low shuffling proportion is good for testing the robustness against spatial noise. When shuffle\_prop is set close to 1, all spatial correlations would be destroyed in the data and this is a useful random control case.

**Figure legends**

Figure 1 Example of imaged sections in the mouse visual cortex. Each blue box shows a section, also called a field. Sections are stitched together to form a global tissue view.

Figure 2 Top 4 principal components associated with the illumination bias. Each panel is a 50 bins by 50 bins heatmap showing the bias level associated with each region of the microscope field of view. Left to right: PC1, 2, 3, 4.

Figure 3 Spatial gene expression of a few spatially selected genes.

Figure 4 The spatial domain organization of the visual cortex, revealed by our HMRF modeling of the spatial data. Each color indicates cells belonging to a spatial domain. There are 9 spatial domains (K=9) and beta is set to 9.0.

Figure 5 Spatial perturbation analysis. We fully shuffled the spatial positions such that 100% of cells’ positions are exchanged. (a) Log-likelihood of the HMRF model is calculated and plotted for each value of beta from 0 to 20.0. (b) Difference in log-likelihood between observed and randomly shuffled data is plotted.

Figure 6 Cell type specific genes from the scRNAseq visual cortex data set. Average expressions for 44 cell type specific genes that were selected from MAST[21] are plotted for each cell type. Star indicates strongly differentially expressed genes (avg. expr z-score > 2.0) that were removed from HMRF.

**References**

1. Deng Q, Ramsköld D, Reinius B, Sandberg R (2014) Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells. Science (80- ) 343:193–196 . doi: 10.1126/science.1245316

2. Jaitin DA, Kenigsberg E, Keren-Shaul H, et al (2014) Massively parallel single cell RNA-Seq for marker-free decomposition of tissues into cell types. Science (80- ) 343:776–779 . doi: 10.1126/science.1247651.Massively

3. Macosko EZ, Basu A, Regev A, et al (2015) Highly Parallel Genome-wide Expression Profiling of Individual Cells Using Nanoliter Droplets. Cell 161:1202–1214 . doi: 10.1016/j.cell.2015.05.002

4. Schiffenbauer YS, Kalma Y, Trubniykov E, et al (2011) Droplet Barcoding for Single-Cell Transcriptomics Applied to Embryonic Stem Cells. Cell 161:1187–1201 . doi: 10.1016/j.cell.2015.04.044

5. Halpern KB, Shenhav R, Matcovitch-Natan O, et al (2017) Single-cell spatial reconstruction reveals global division of labour in the mammalian liver. Nature 542:1–5 . doi: 10.1038/nature21065

6. Lein ES, Hawrylycz MJ, Ao N, et al (2006) Genome-wide atlas of gene expression in the adult mouse brain. Nature 445:168–176 . doi: nature05453 [pii]\n10.1038/nature05453

7. Raj A, van den Bogaard P, Rifkin SA, et al (2008) Imaging individual mRNA molecules using multiple singly labeled probes. Nat Methods 5:877–879 . doi: 10.1038/nmeth.1253

8. Lubeck E, Cai L (2012) Single-cell systems biology by super-resolution imaging and combinatorial labeling. Nat Methods 9:743–748 . doi: 10.1038/nmeth.2069

9. Chen KH, Boettiger AN, Moffitt JR, et al (2015) Spatially resolved, highly multiplexed RNA profiling in single cells. Science (80- ) 348: . doi: 10.1126/science.aaa6090

10. Moffitt JR, Hao J, Bambah-Mukku D, et al (2016) High-performance multiplexed fluorescence in situ hybridization in culture and tissue with matrix imprinting and clearing. Proc Natl Acad Sci 113:14456–14461 . doi: 10.1073/pnas.1617699113

11. Shah S, Lubeck E, Zhou W, Cai L (2016) In Situ Transcription Profiling of Single Cells Reveals Spatial Organization of Cells in the Mouse Hippocampus. Neuron 92:342–357 . doi: 10.1016/j.neuron.2016.10.001

12. Zhu Q, Shah S, Dries R, et al (2018) Decomposing spatially dependent and cell type specific contributions to cellular heterogeneity. Nat Biotechnol. doi: 10.1101/275156

13. St\r ahl PL, Salmén F, Vickovic S, et al (2016) Visualization and analysis of gene expression in tissue sections by spatial transcriptomics. Science (80- ) 353:78–82 . doi: 10.1126/science.aaf2403

14. Wang Q (2012) HMRF-EM-image: Implementation of the Hidden Markov Random Field Model and its Expectation-Maximization Algorithm. arXiv Prepr

15. Li SZ (2009) Markov Random Field Modeling in Image Analysis

16. Li SZ (2003) Modeling image analysis problems using Markov random fields. Stoch Process Model Simul 473

17. Obayashi T, Kinoshita K (2011) COXPRESdb: a database to compare gene coexpression in seven model animals. Nucleic Acids Res 39:D1016–D1022

18. Tasic B, Menon V, Nguyen TN, et al (2016) Adult mouse cortical cell taxonomy revealed by single cell transcriptomics. Nat Neurosci 19:335–346 . doi: 10.1038/nn.4216

19. Chung NC, Storey JD (2015) Statistical significance of variables driving systematic variation in high-dimensional data. Bioinformatics 31:545–554 . doi: 10.1093/bioinformatics/btu674

20. Brélaz D (1979) New methods to color the vertices of a graph. Commun ACM 22:251–256 . doi: 10.1145/359094.359101

21. Finak G, McDavid A, Yajima M, et al (2015) MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. Genome Biol 16:278 . doi: 10.1186/s13059-015-0844-5