SpatialDE_analysis

June 24, 2019

1 SpatialDE Analysis

```
In [ ]: import numpy as np
        import pandas as pd
        import NaiveDE
        import SpatialDE
        from matplotlib import pyplot as plt
        import pickle
        from tqdm import tqdm
```

1.1 Import Datasets

```
In [ ]: def T_quality(x):
            return np.clip(1-np.log(1+x)/2.5,0,1)


        Q_th = 2.5
        min_count = 500
        barcodes_df = []
        tagList_df = pd.read_csv("../data/tagList_84-gene.csv", sep = ",", usecols = [0,1], hea
        datasets = ['170315_161220_4_1','161230_161220_3_1']
        for sample in datasets:
            df = pd.read_csv("../data/results/"+sample+"/barcodes.csv", sep = ",")
            df.seq_quality_min=df.seq_quality_min*df.general_stain_min.apply(T_quality)
            # Add gene names to dataframe
            d = pd.Series(tagList_df.Gene.values,index=tagList_df.Seq).to_dict()
            df["Gene"] = df['letters'].map(d)
            # Downsample barcode coordinate space by factor 8 for easier visualization
            df["global_X_pos"]=df.global_X_pos/8
            df["global_Y_pos"]=df.global_Y_pos/8
            # Remove reads not in the codebook
            df = df.dropna()
            # Filter reads by quality
            df = df[df.seq_quality_min>Q_th]
            # Filter reads by min count per gene
            df["count"] = 0
            for i,row in tagList_df.iterrows():
```

```
            df.loc[df["Gene"] == tagList_df.Gene[i],["count"]] = len(df[df["Gene"] == tagLi
        df = df[df["count"]>min_count]

        barcodes_df.append(df)
```

1.2 Generate Expression Tables

```
In [ ]: # Import and downsample by factor 8 image shape
        img_shape = np.round(np.array([[22508, 33566],[22563, 31782]])/8).astype(np.uint)

        # Create gene expression table
        expression_df = []
        sample_df = []
        for s_idx, df in enumerate(barcodes_df):
            x_min = 0; x_max= img_shape[s_idx,1];
            y_min = 0; y_max= img_shape[s_idx,0];
            batch_size_px=64
            overlap = 8

            express_table = pd.DataFrame(data={}, columns=df.Gene.unique(), index=list((str(x)
            for i in tqdm(range(x_min,x_max,batch_size_px)):
                for j in range(y_min,y_max,batch_size_px):
                    batch_df=df[(df.global_X_pos>=i-(batch_size_px/2)-overlap) & (df.global_X_
                    if len(batch_df):
                        batch_counts = batch_df['Gene'].value_counts()
                        express_table.loc[str(i)+'x'+str(j),batch_counts.index]=batch_counts

            express_table = express_table.fillna(0)

            # Create sample_info
            sample_info = pd.DataFrame(data={'x':list(x for x in range(x_min,x_max,batch_size_
            sample_info['total_counts'] = express_table.sum(axis=1)
            # Dropping empty batches
            express_table = express_table[sample_info.total_counts>10]
            sample_info = sample_info[sample_info.total_counts>10]

            expression_df.append(express_table)
            sample_df.append(sample_info)

In [ ]: # save dataframes
        for i,dataset in enumerate(datasets):
            expression_df[i].to_pickle('../data/results/'+dataset+'/SpatialDE_express_table.hd
            sample_df[i].to_pickle('../data/results/'+dataset+'/SpatialDE_sample_info.hdf5')

In [6]: # load dataframes
        img_shape = np.round(np.array([[22508, 33566],[22563, 31782]])/8).astype(np.uint)
        expression_df = []
        sample_df = []
```

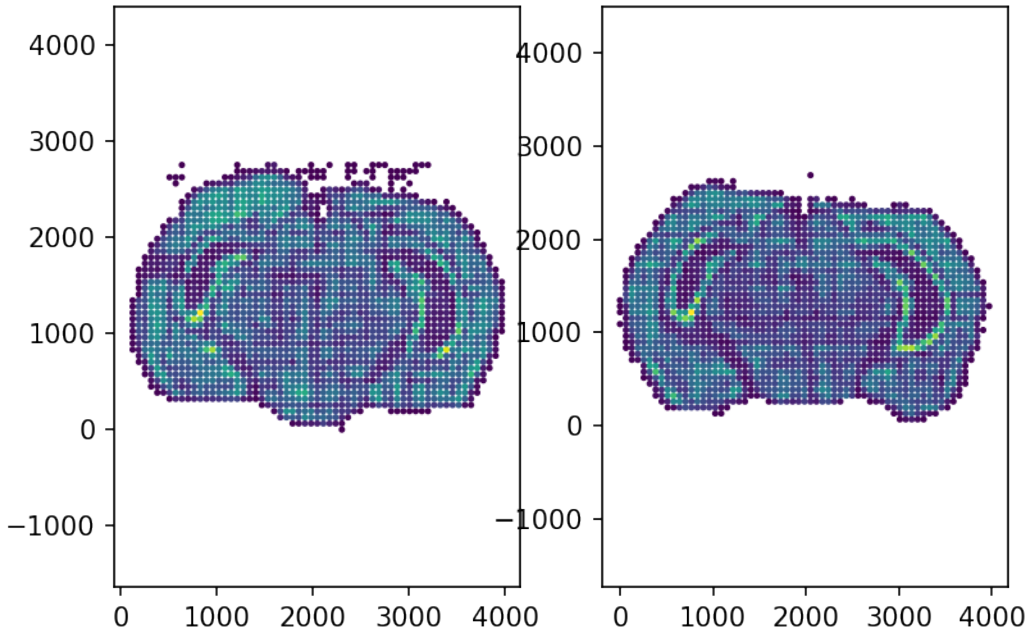
```
for i,dataset in enumerate(datasets):
    plt.rcParams["figure.dpi"] = 150
    plt.subplot(1,2,i+1)

    x_min = 0; x_max= img_shape[i,1];
    y_min = 0; y_max= img_shape[i,0];
    batch_size_px=16
    overlap = 16
    express_table = pd.read_pickle('../data/results/'+dataset+'/SpatialDE_express_table
    # Create sample_info
    sample_info = pd.DataFrame(data={'x':list(x for x in range(x_min,x_max,batch_size_
    sample_info['total_counts'] = express_table.sum(axis=1)
    # Dropping empty batches
    express_table = express_table[sample_info.total_counts>10]
    sample_info = sample_info[sample_info.total_counts>10]

    expression_df.append(express_table)
    expression_df[i] = expression_df[i].rename(('{}_'+str(i)).format)

    sample_df.append(sample_info)
    sample_df[i] = sample_df[i].rename(('{}_'+str(i)).format)
    sample_df[i]['s'] = i
    plt.scatter(sample_df[i]['x'], sample_df[i]['y'], c=sample_df[i]['total_counts'],s=
    plt.axis('equal');
```

```
/home/gapartel/miniconda3/lib/python3.7/site-packages/ipykernel_launcher.py:18: UserWarning: Bo
/home/gapartel/miniconda3/lib/python3.7/site-packages/ipykernel_launcher.py:18: UserWarning: Bo
```



1.3 Normalize Gene Expression Tables

```
In [7]: expression_df=pd.concat(expression_df,sort=True)
        expression_df=expression_df.dropna(axis=1)
        #expression_df=expression_df.fillna(0)
        sample_df=pd.concat(sample_df,sort=True)

        # Linear regression to account for library size and sequencing depth bias of each patc
        norm_expr = pd.concat([NaiveDE.stabilize(expression_df[sample_df.s==0].T).T,NaiveDE.sta
        resid_expr = pd.concat([NaiveDE.regress_out(sample_df[sample_df.s==0], norm_expr[sample
        resid_expr = NaiveDE.regress_out(sample_df, resid_expr.T, 'np.log(total_counts)').T
        #resid_expr = NaiveDE.regress_out(sample_df, norm_expr.T, 'np.log(total_counts)').T
        idx = resid_expr.var().sort_values(ascending=False).index
```

1.4 SpatialDE significance test

```
In [ ]: results = []
        for i,df in enumerate(datasets):
            X = sample_df.loc[sample_df.s==i,['x', 'y']]
            results.append(SpatialDE.run(X, resid_expr.loc[sample_df[sample_df.s==i].index,:]))
```

```
In [ ]: from IPython.display import display_html
        def display_side_by_side(*args):
            html_str=''
            for df in args:
                html_str+=df.to_html()
            display_html(html_str.replace('table','table style="display:inline"'),raw=True)

        display_side_by_side(results[0].sort_values('qval').head(10)[['g', 'l', 'qval']],resul
```

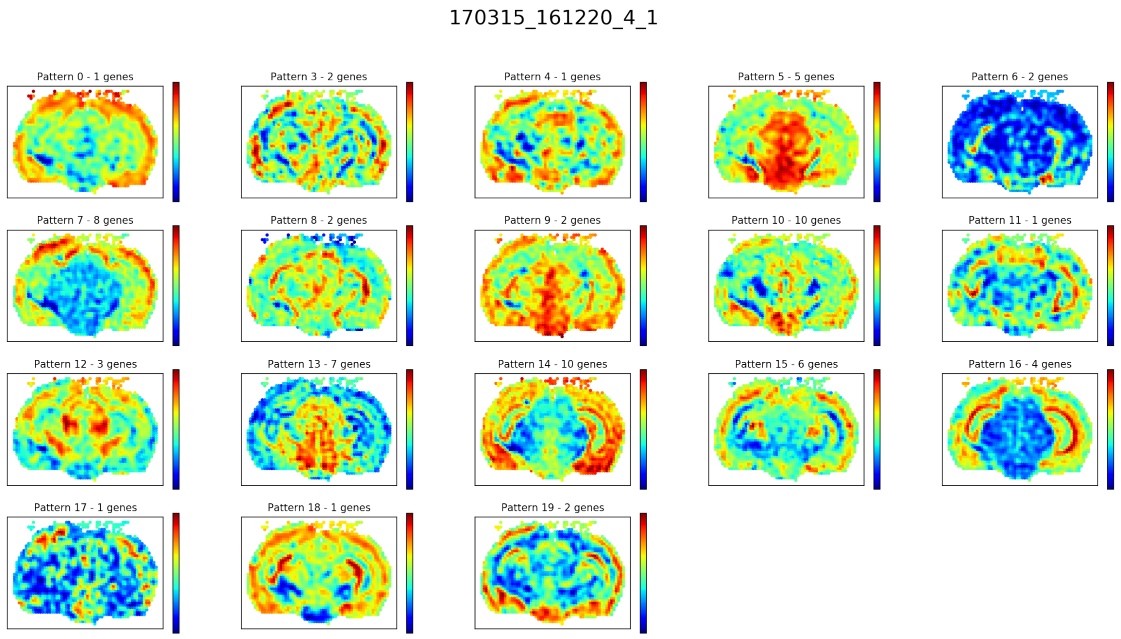
1.5 Automatic expression histology

```
In [ ]: res = []
        n_patterns = 20
        for i,df in enumerate(datasets):
            sign_results = results[i].query('qval < 0.05')
            X = sample_df.loc[sample_df.s==i,['x', 'y']]
            histology_results, patterns = SpatialDE.aeh.spatial_patterns(X, resid_expr.loc[samp
            res.append({'aeh': histology_results, 'patterns':patterns})
```

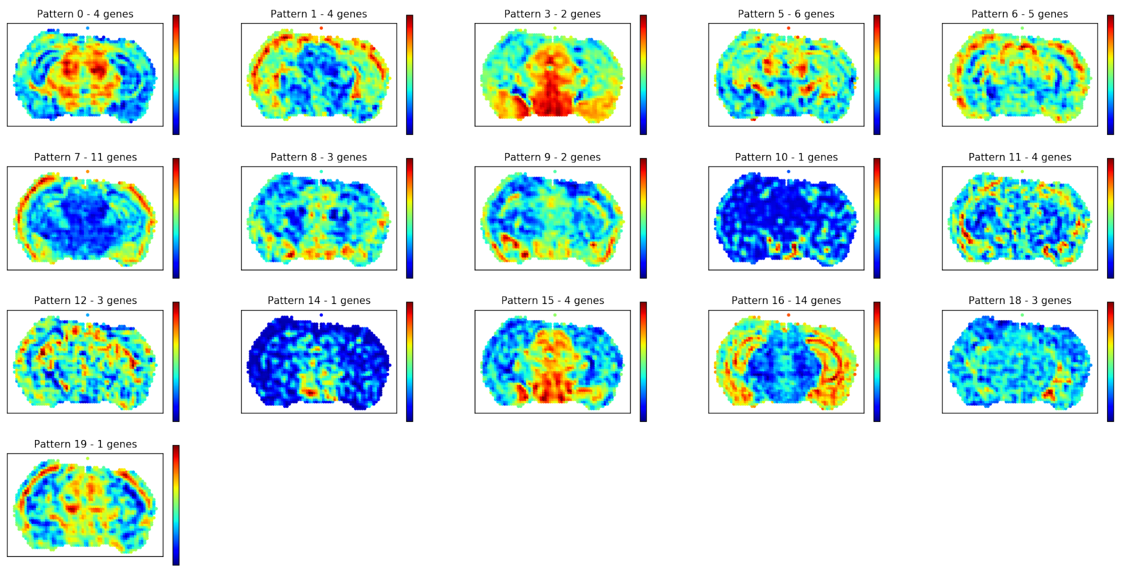
```
In [ ]: # Save results
        for s,df in enumerate(datasets):
            pickle.dump(res[s], open( "../data/results/"+df+"/SpatialDE_res.hdf5", "wb" ) )
```

```
In [17]: # Plot Histological Patterns
         res = []
```

```
n_patterns = 20
for s,df in enumerate(datasets):
    res.append(pickle.load(open( "../data/results/"+df+"/SpatialDE_res.hdf5", "rb")))
    histology_results = res[s]['aeh']
    patterns = res[s]['patterns']
    plt.figure(figsize=(20,10))
    plt.suptitle(df,fontsize=20)
    j=1
    for i in range(n_patterns):
        if len(histology_results[histology_results.pattern==i])>0:
            plt.subplot(4, 5, j)
            plt.scatter(sample_df.loc[sample_df.s==s,'x'], sample_df.loc[sample_df.s=
            plt.axis('scaled')
            plt.title('Pattern {} - {} genes'.format(i, histology_results.query('patte
            plt.colorbar(ticks=[]);
            plt.xticks([])
            plt.yticks([]);
            j = j+1
```



161230_161220_3_1



```
In [18]: for i,df in enumerate(datasets):
             histology_results = res[i]['aeh']
             print(df)
             for i in histology_results.sort_values('pattern').pattern.unique():
                 print('Pattern {}: '.format(i))
                 print(', '.join(histology_results.query('pattern == @i').sort_values('membersl
             print("\n")
```

```
170315_161220_4_1
Pattern 0:
Arpp21
Pattern 3:
Rgs12, Plcxd2
Pattern 4:
Cpne5
Pattern 5:
Atp1b1, Calb2, Scg2, Slc6a1, Zcchc12
Pattern 6:
Nos1, Npy2r
Pattern 7:
Cxcl14, Cck, Hapln1, Lamp5, Nov, Rgs4, Satb1, Vip
Pattern 8:
Aldoc, Ndnf
Pattern 9:
Pvrl3, Gap43
Pattern 10:
Adgrl2, Cnr1, Gad1, Penk, Rasgrf2, Sema3c, Serpini1, Tac2, CdA3, Fxyd6
```

```
Pattern 11:
Kit
Pattern 12:
Chrm2, Pvalb, Slc24a2
Pattern 13:
Enpp2, Grin3a, Rgs10, Sulf2, Plp1, Sncg, Tac1
Pattern 14:
Calm2, Fam19a1, Htr3a, Nr4a2, Rprm, Crhbp, Enc1, Gda, Pde1a, Snca
Pattern 15:
Reln, Rorb, Cox6a2, Gabrd, Id2, Pcp4
Pattern 16:
Bcl11b, 3110035E14Rik, Crym, Neurod6
Pattern 17:
Fos
Pattern 18:
Nrn1
Pattern 19:
Calb1, Wfs1


161230_161220_3_1
Pattern 0:
Aldoc, Enpp2, Chrm2, Plp1
Pattern 1:
Cox6a2, Pvrl3, Neurod6, Satb1
Pattern 3:
Gap43, Zcchc12
Pattern 5:
Atp1b1, Hapln1, Sulf2, Gad1, Pvalb, Slc24a2
Pattern 6:
Fos, Pcp4, Reln, Id2, Rorb
Pattern 7:
Rasgrf2, Cpne5, Fam19a1, Vip, Arpp21, Calb1, Gabrd, Gda, Lamp5, Nov, Rgs4
Pattern 8:
CdA3, Nos1, Fxyd6
Pattern 9:
Penk, Wfs1
Pattern 10:
Tac2
Pattern 11:
Adgrl2, Rgs10, Rprm, Serpini1
Pattern 12:
Cxcl14, Ndnf, Sema3c
Pattern 14:
Sncg
Pattern 15:
Grin3a, Calb2, Scg2, Tac1
Pattern 16:
```

```
Calm2, Cnr1, Crhbp, Kit, Nr4a2, Nrn1, Rgs12, Snca, 3110035E14Rik, Bcl11b, Cck, Crym, Enc1, Pde
Pattern 18:
Htr3a, Slc6a1, Npy2r
Pattern 19:
Plcxd2
```

```
In [ ]:
```