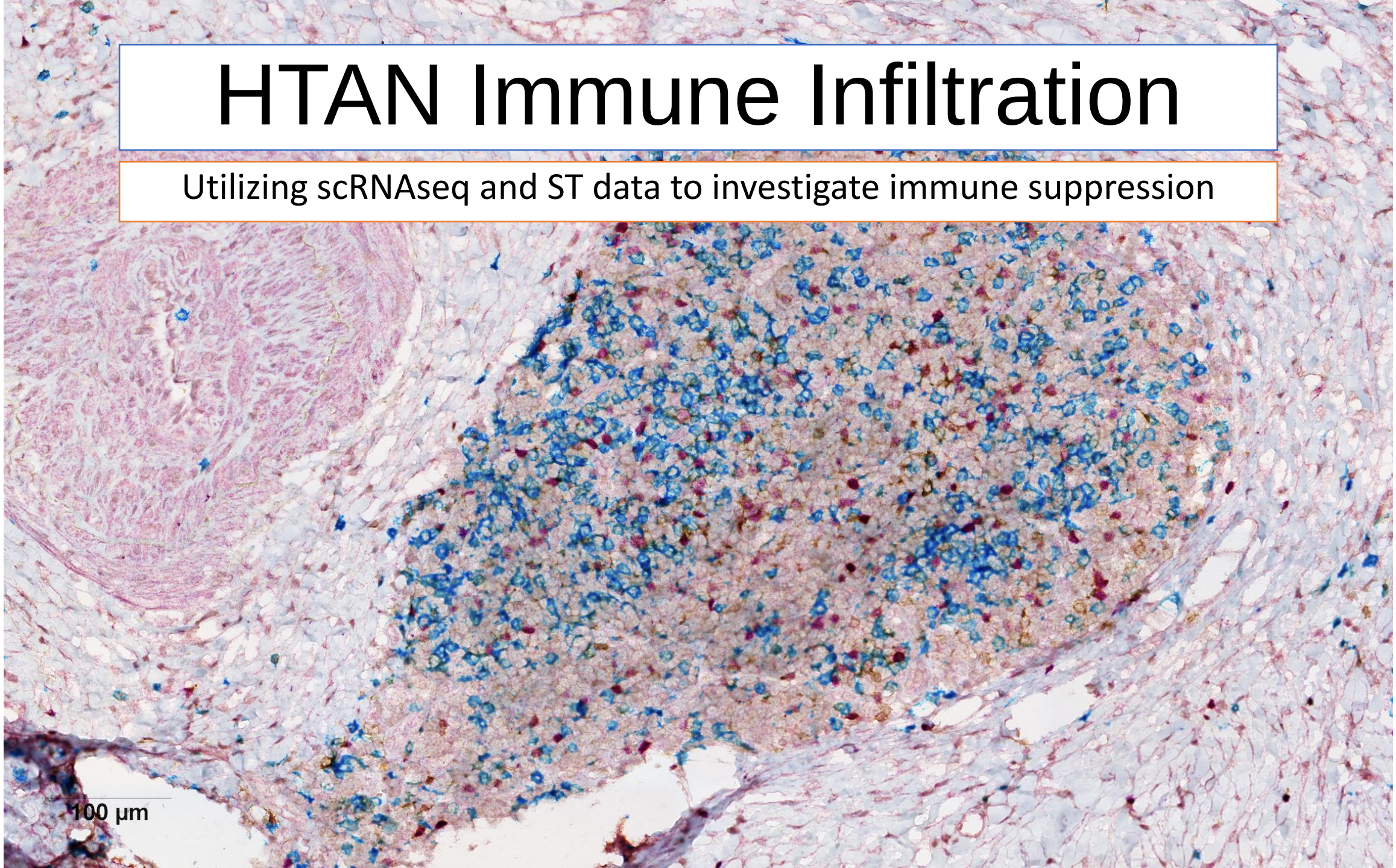
HTAN Immune Infiltration

Utilizing scRNAseq and ST data to investigate immune suppression



Background

- Pancreatic ductal adenocarcinoma (PDAC) is a lethal malignancy with a dismal 5 -year overall survival rate of less than 12% (seer.gov).
- PDAC is refractory to immune-targeting therapies, despite the observation that many patients do have infiltration of immune cells (including T cells).
- PDAC is a highly immunosuppressive tumor microenvironment, and more work is needed to determine the spatial organization of immune:tumor:stroma interactions

Project Goals

- Utilize Human Tumor Atlas Network (HTAN) spatial transcriptomics (ST) and single cell RNA sequencing (scRNAseq) datasets from pancreatic cancer patients to investigate potential mechanisms of immune suppression

- ST and scRNASeq was obtained from HTAN, publication Zhou et al., *Nature Genetics*, 2022

- Specific jamboree goal: download, process, and map immune infiltration of HTAN ST data (with using scRNAseq as reference for immune signatures)

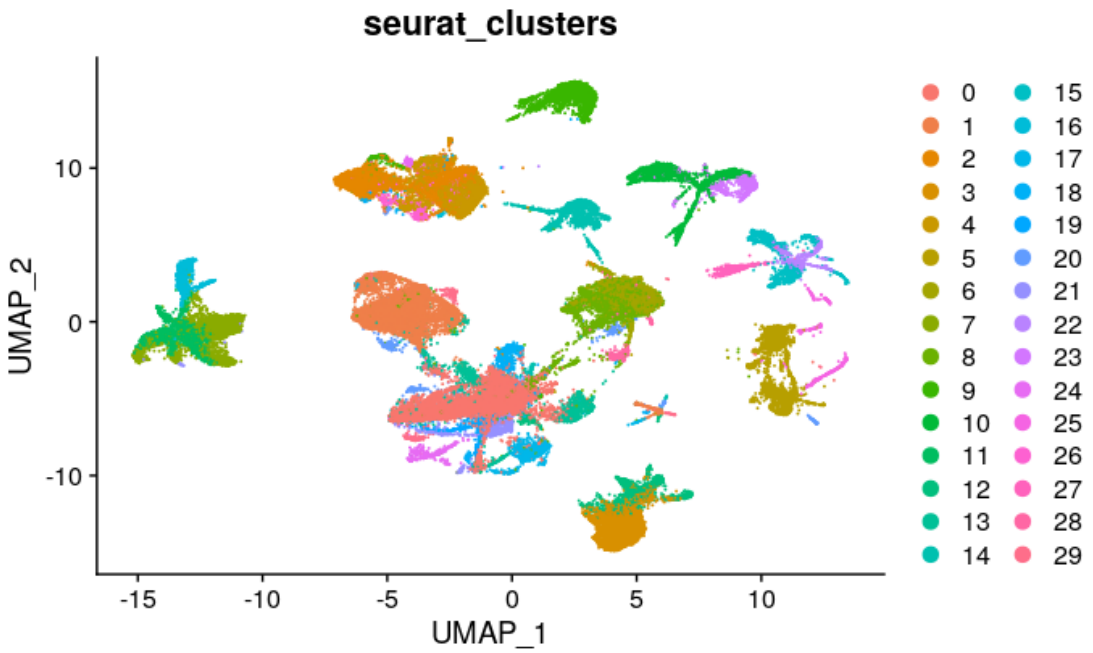
Hurdles to overcome

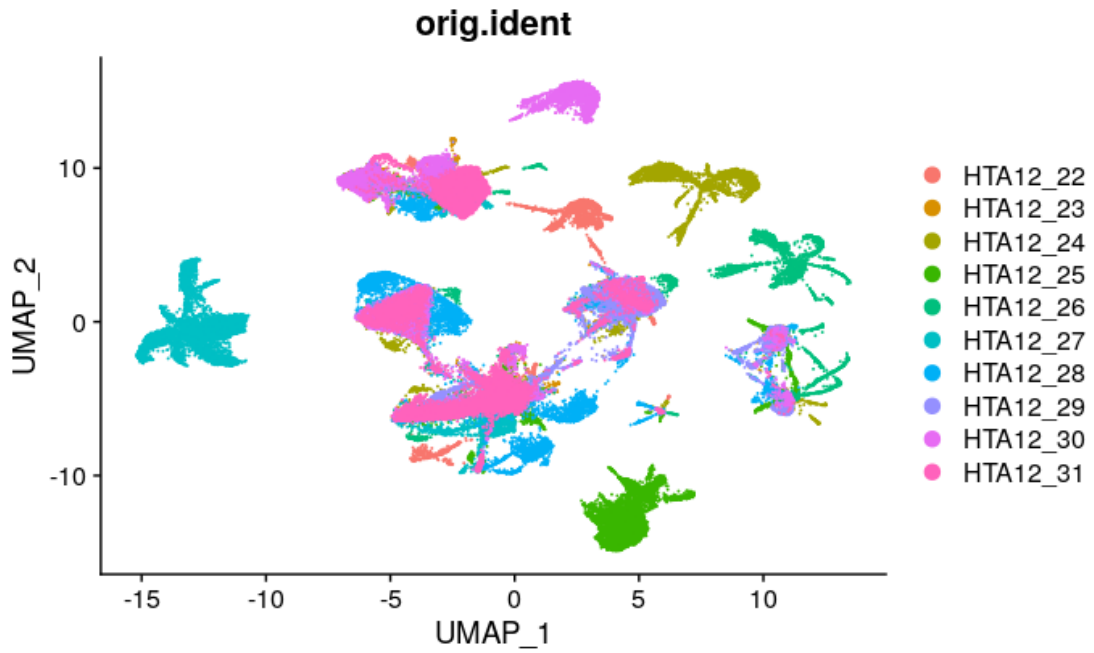
- Even from the manuscript language and HTAN manuals/manifests/metadata, it was **very difficult** to understand/interpret exactly **which samples matched with which patient and related to which technology (Visium ST vs scRNAseq vs snRNAseq vs IF), and associated metadata**

- **Saving of the seven bridges session was time-intensive**

- **Importing data from HTAN portal to seven bridges account was time-intensive and hard to determine best method to use**

Analysis of scRNAseq HTAN data





Hurdles: some issues with batch correction not collapsing the UMAPs efficiently

Labeling approaches

- GPT4 ai generated knowledge labeling of clusters→FAST, but errors…

Annotate Seurat's "FindMarkers" upregulated genes using GPT4 LLM

function calling allows us to use parasable json structured data

Add openai's python module

```
In [1]: !pip install -q openai

In [2]: import os
        import json
        import pandas as pd
        from getpass import getpass

        import openai
```

Get the user's API key

```
In [3]: # Prompt for secure key input
        api_key = getpass('Enter your OPENAI API key: ')

        # Set the key as an environment variable

        os.environ['OPENAI_API_KEY'] = api_key
```

Read the markers table

in this example we have saved rownames on export from R

```
In [4]: markers = pd.read_csv('/sbgenomics/project-files/scRNAseq_processing/seurat_scRNA_filtered_downsampled.harmonyJI
In [5]: markers.head()
```

Out[5]:

	p_val	avg_log2FC	pct.1	pct.2	p_val_adj	cluster	gene
chr3:93470147-93471055	0.0	4.395002	0.356	0.060	0.0	0	chr3:93470147-93471055
CD96	0.0	1.866024	0.375	0.152	0.0	0	CD96
SLFN12L	0.0	1.818384	0.317	0.132	0.0	0	SLFN12L
LINC01934	0.0	1.678011	0.276	0.114	0.0	0	LINC01934
CD247	0.0	1.598508	0.270	0.109	0.0	0	CD247

Set parameters for genes that will inform clusters

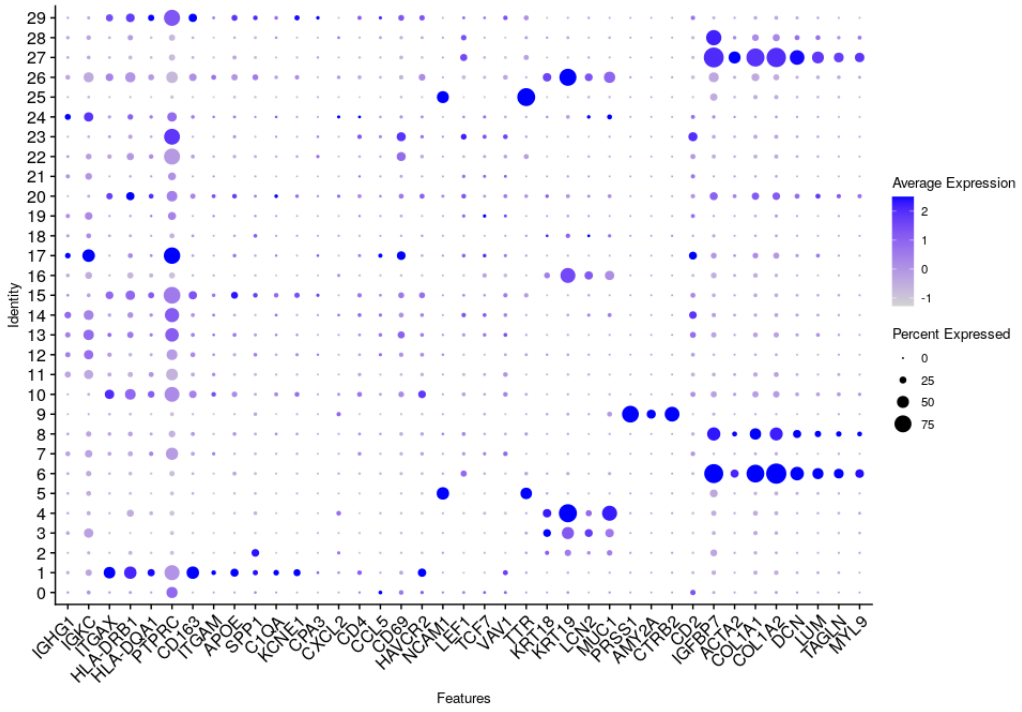
```
In [6]: n_genes = 20
        orderby_columns = ['avg_log2FC','p_val']
        orderby_ascending = [False,True]
        max_p_value_adj = 0.05
        min_avg_log2FC = 0.25
        drop_gene_strings = ['chr']

In [7]: ordered_markers = markers.sort_values(orderby_columns,ascending=orderby_ascending).reset_index(drop=True)
        for ds in drop_gene_strings:
            ordered_markers = ordered_markers.loc[~ordered_markers['gene'].str.contains(ds),:]
        ordered_markers = ordered_markers.loc[ordered_markers['p_val_adj'] < max_p_value_adj,:]
        print("Number of clusters:")
        print(len(ordered_markers['cluster'].unique()))
        cluster_obj = {}
        for i in sorted(ordered_markers['cluster'].unique()):
            sub = ordered_markers.loc[ordered_markers['cluster']==i,:]
            glist = list(sub.head(n=n_genes)['gene'])
            cluster_obj[i] = glist
```

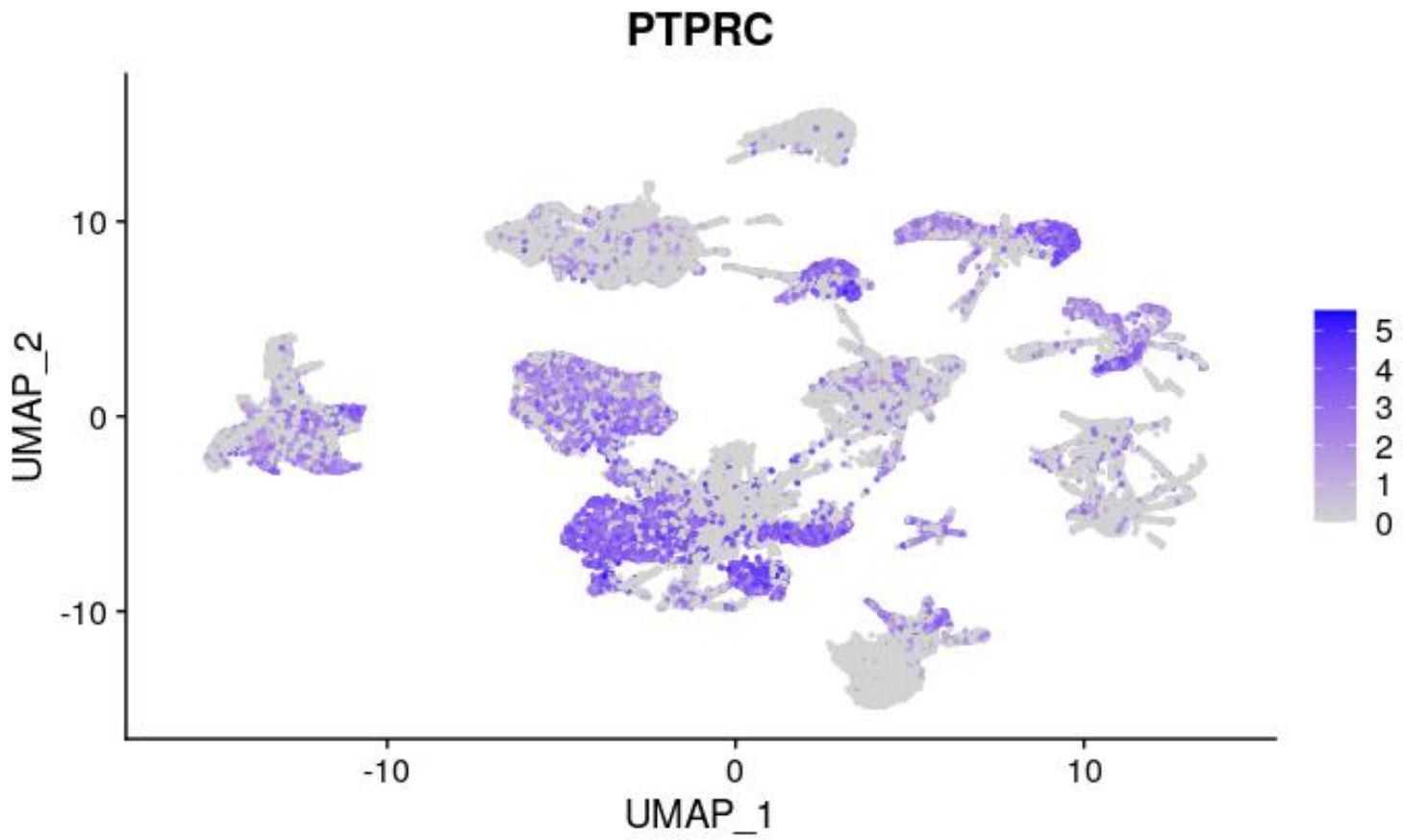
```
Number of clusters:
30
```

Set up our GPT function

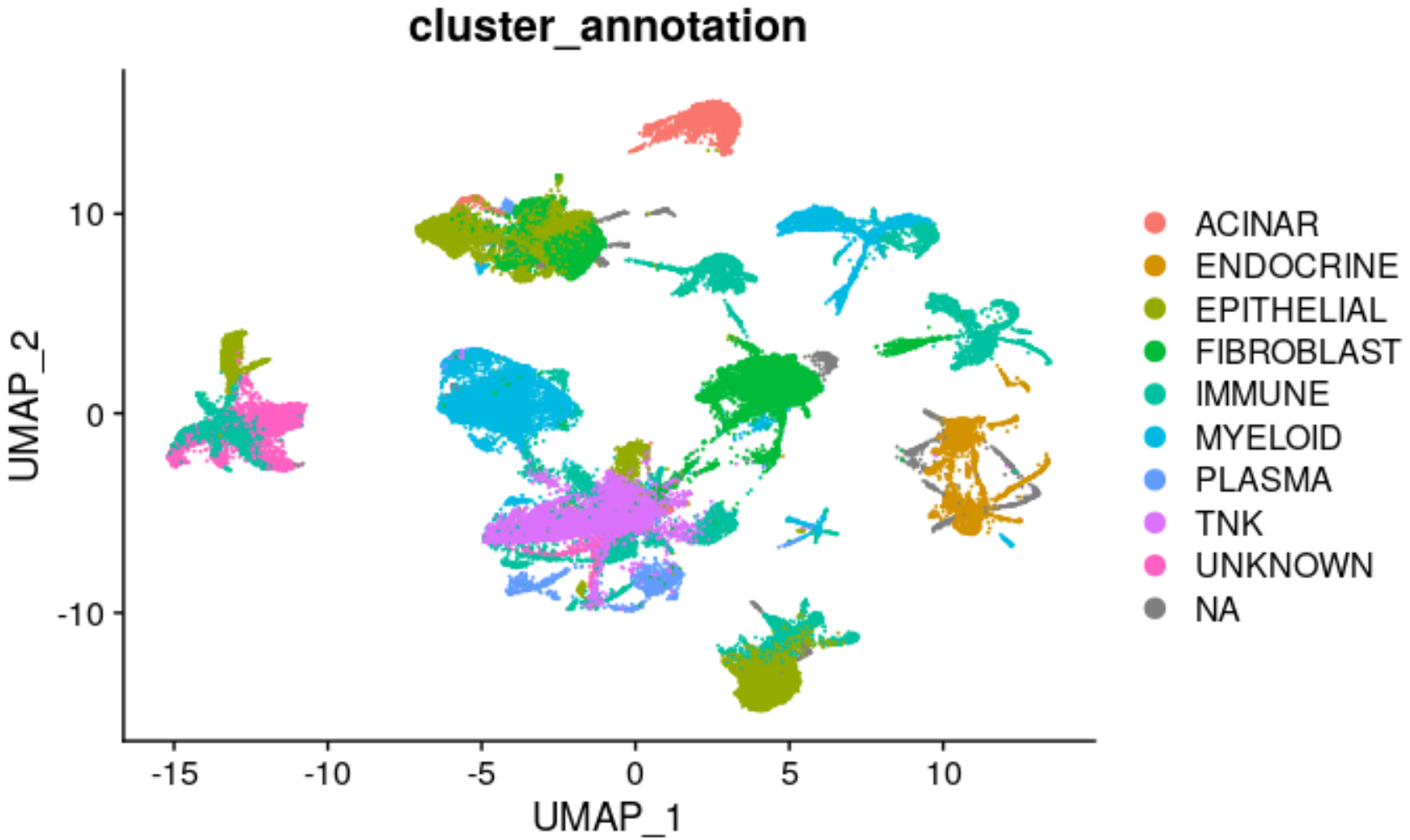
Manual Labeling with prior knowledge (publications)→SLOW, but potentially more accurate



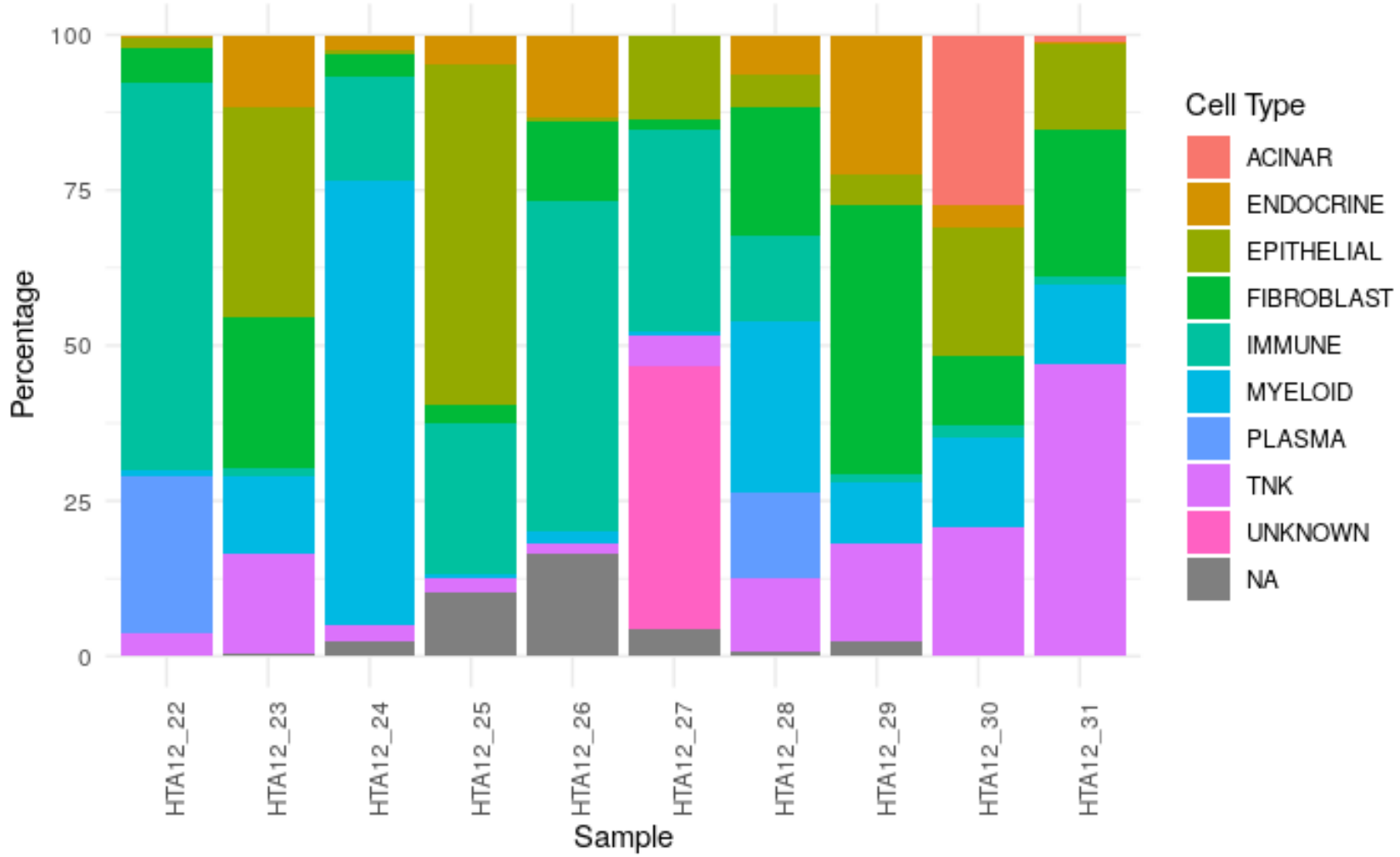
Leaky expression of PTPC (“CD45”=pan immune marker)



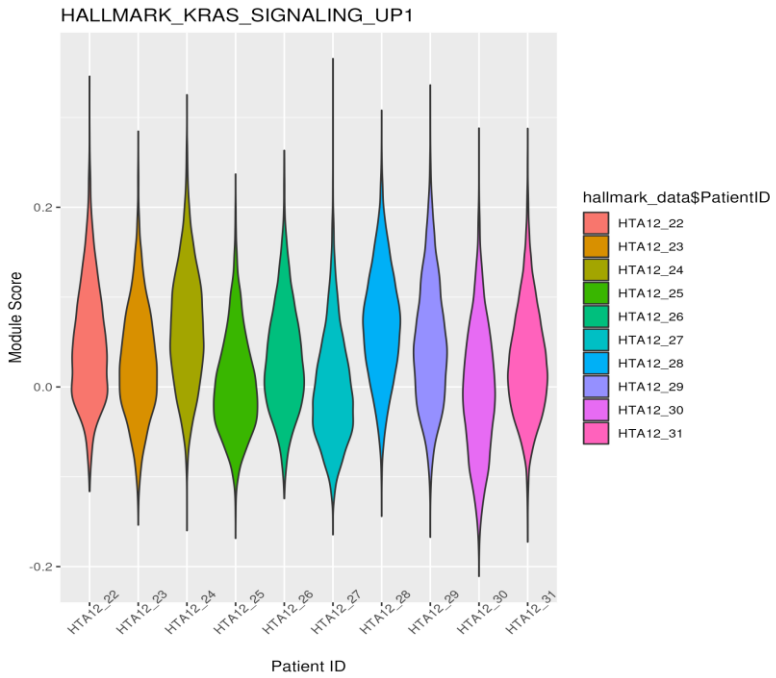
Manual Annotation of clusters

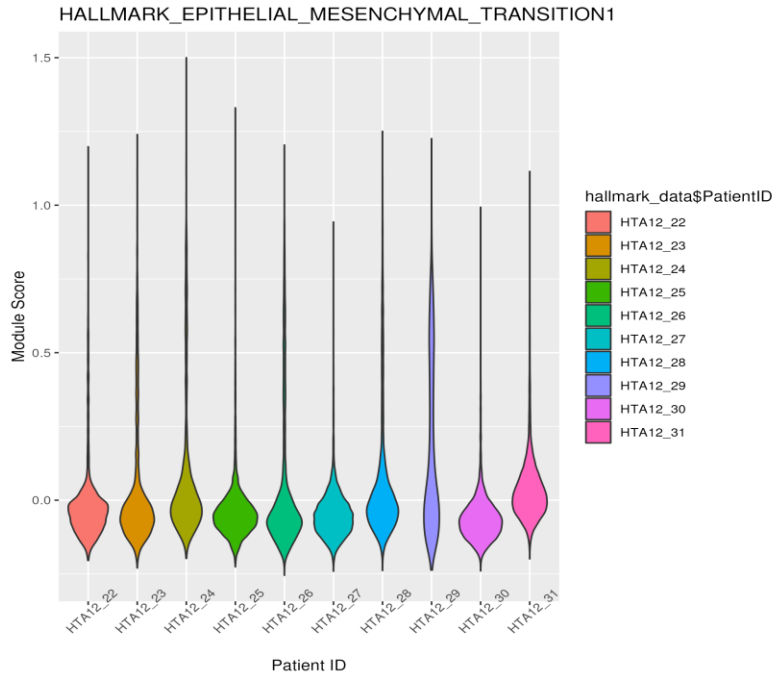


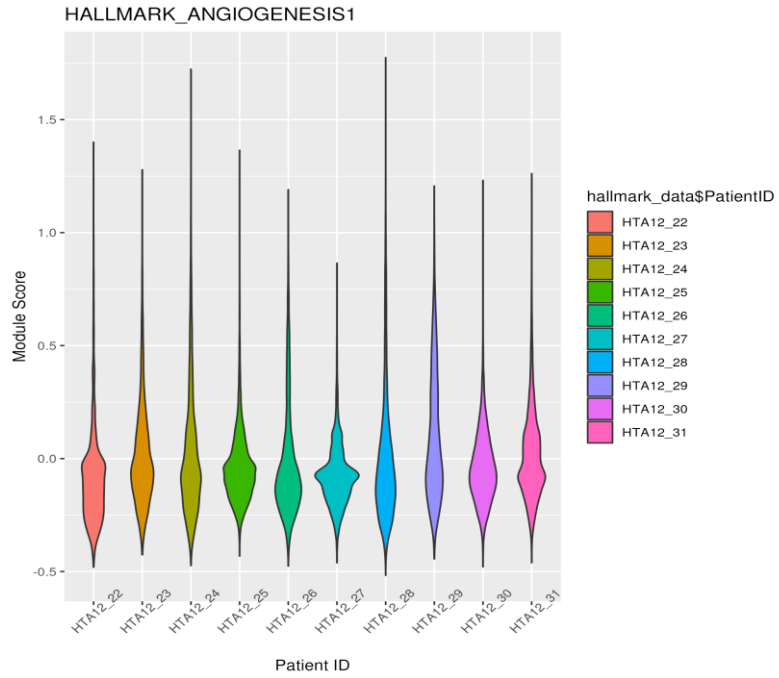
Relative frequency of cell populations



Module Scores- Examining if there are overall patient specific functional gene signatures







MsigDB hallmark 2023

Analysis of Spatial transcriptomics HTAN data

Sample: HT224P1-S1Fc2U1Z1Bs1-1

HTAN Participant Id: HTA12_22

Age at Dx	Primary Dx	Site of Biopsy	Tissue
61	Adenocarcinoma pancreatobiliary type	Head of pancreas	Pancreas NOS

tissue_lowres_image.png

Analysis Troubles

Seven Bridges / CGC:

- Re-opening a data studio requires the re-installation of all the R libraries from previous session
- Closing out a data studio takes a long time (on scale of hours)

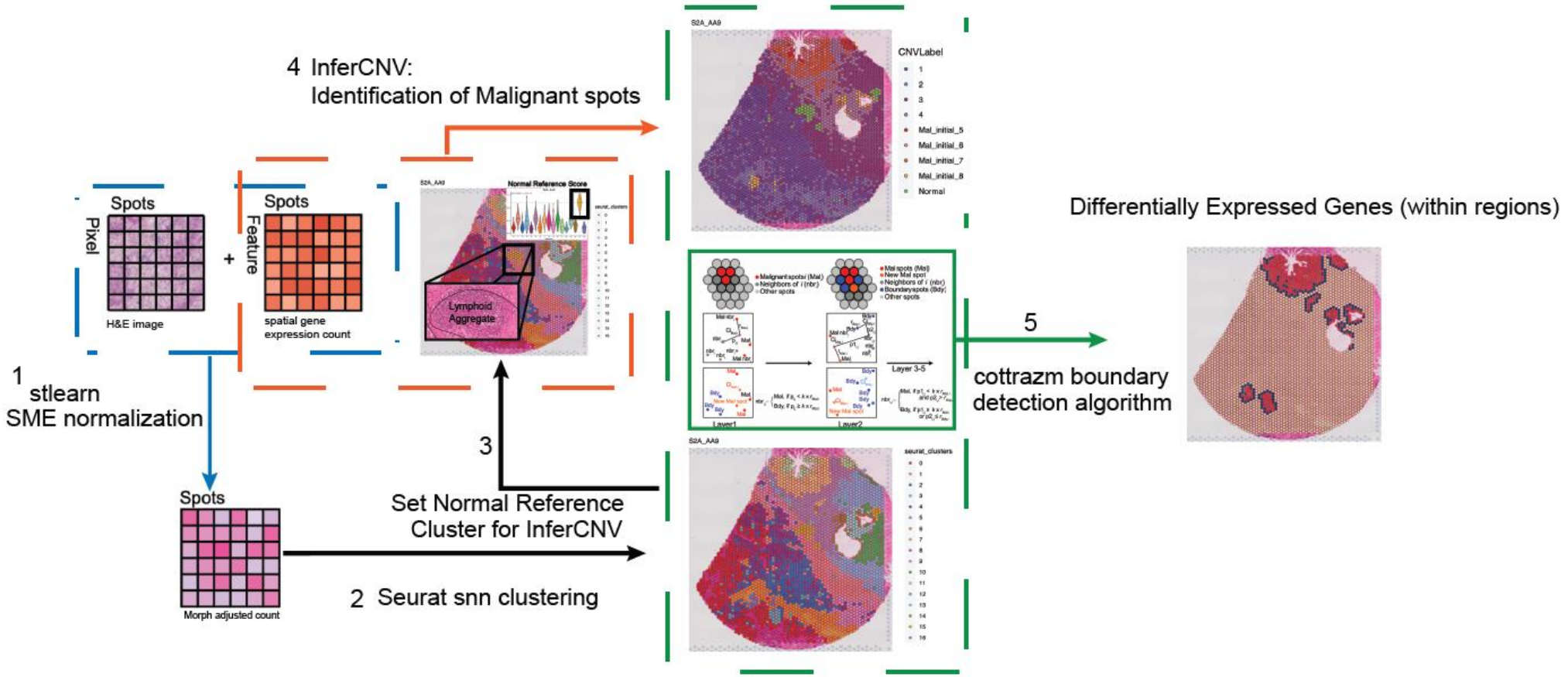
HTAN:

- Having a single .h5 for barcodes, features, matrix files would be easier than the files individually – as most read functions utilize the .h5 to read in visium/10x data

Cottrazm:

- Entire pipeline steps are performed within cottrazm functions, so there is little room for customization of parameters – also limited transparency
  - prevented loading of individual files (see .h5 issue above)
  - required direct edit to python helper script to fix error
- inferCNV with random forest parameter takes a long time to run, even with high RAM session.
  - would be better to run with leiden clustering, but did not have time to customize script to fit this output from inferCNV

Cottrazm Workflow



Analysis of Spatial transcriptomics HTAN data: Application of Cottrazm tool

Cottrazm analysis steps:

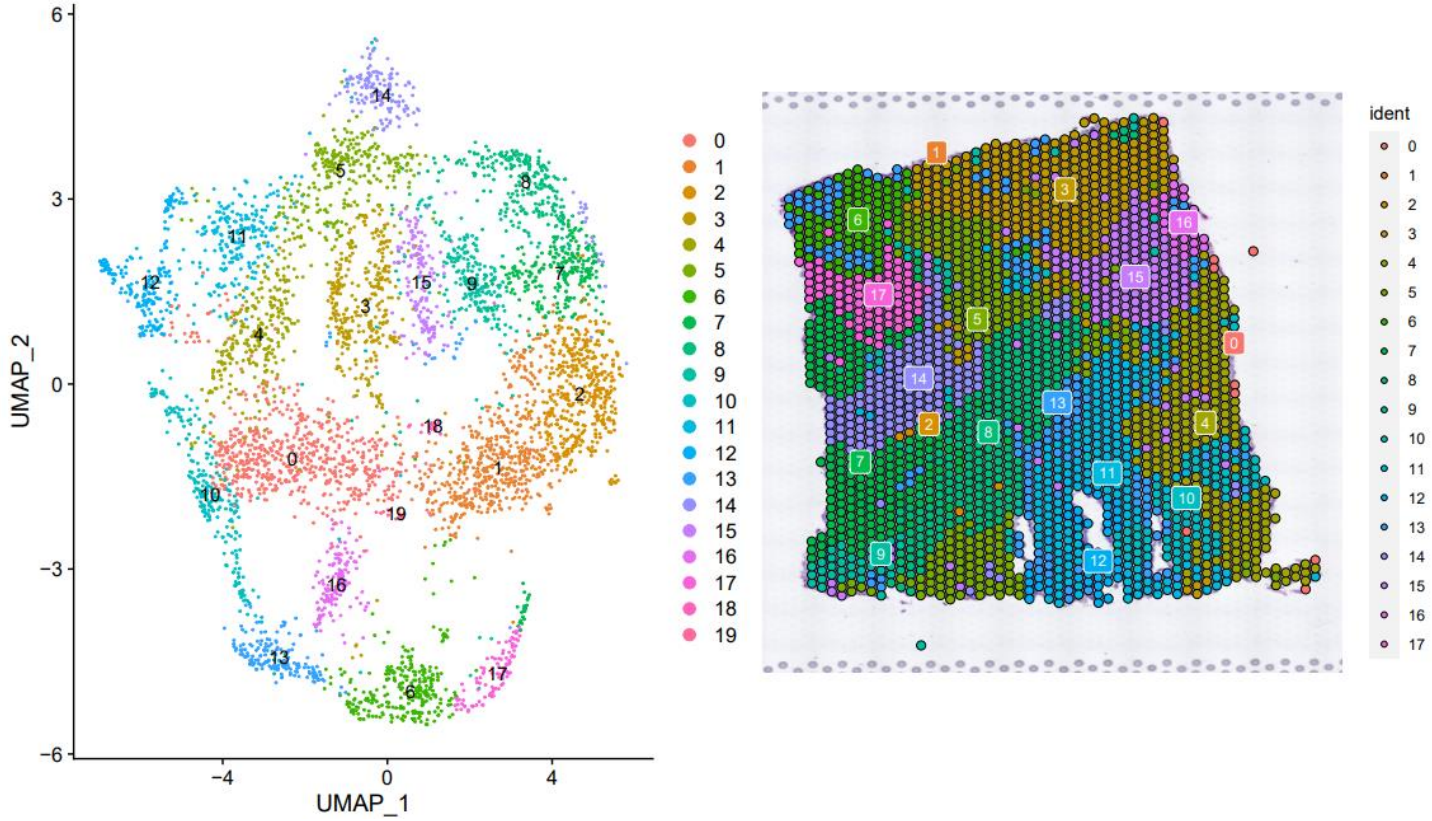
(1) Reading, pre-processing and quality control of tumor ST data

Created .h5 file for individual feature, barcode, matrix files for the sample in order to run *STEPreProcess*. This function generates qc metric plots and creates Seurat object with spatial transcriptomic and low-res image.

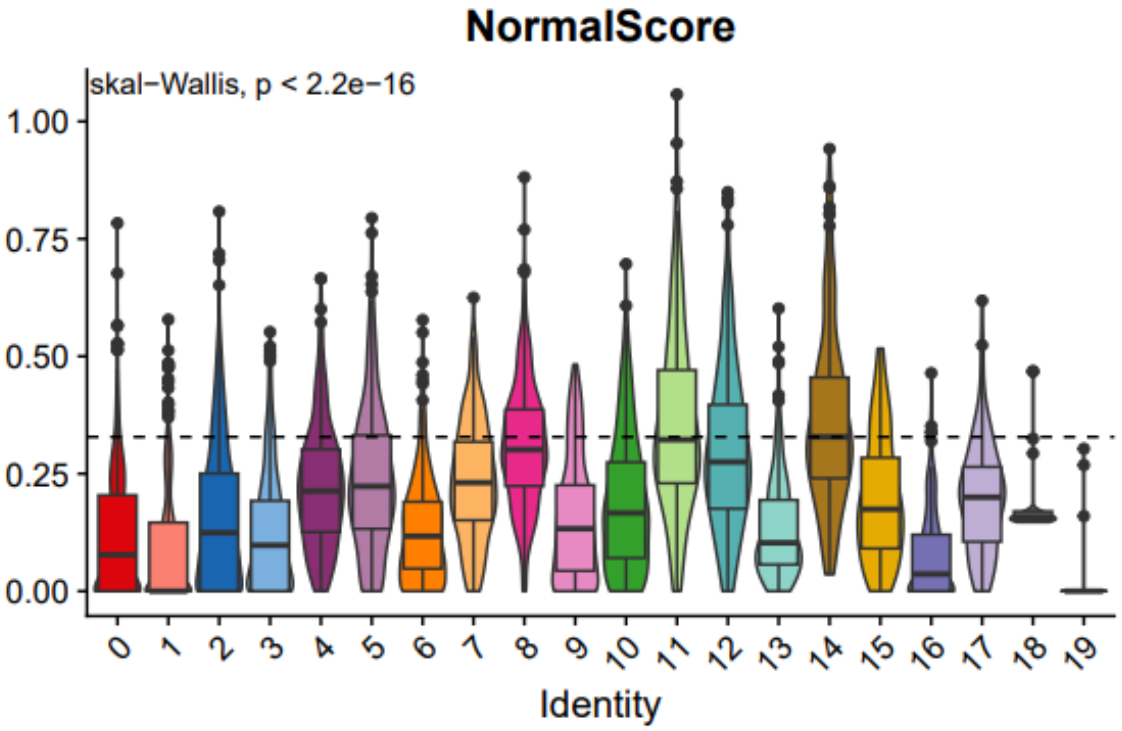
(2) Morphological adjusted cluster determination

Declare conda environment and source python script using *reticulate* package. Then run *ME_normalize_new*, custom python script that replaces one of the child functions of *STModiCluster*. Morphological information from tissue is used to adjust spatial feature expression (using functions from python *stlearn* library). The adjusted expression matrix is returned, then converted to Seurat assay object and added to main Seurat object as “Morph” asssay.

Standard normalization/scaling/dim reduction workflow is performed using “Morph” assay that we’ve added to the object.



Analysis of Spatial transcriptomics HTAN data: Application of Cottrazm tool



An Immune score is calculated which represents the enrichment of expression of genes associated with immune cells (*PTPRC, CD2, CD3D, CD3E, CD3G, CD5, CD7, CD79A, MS4A1, CD19*).

(3) Run InferCNV on ST data

The algorithm declares cluster “14” as normal cluster due to highest scores in that cluster. However, clusters 8, 11, and 12 also have high enrichment of this metric. This is used as the input “reference” for *inferCNV* analysis.

This approach compares test populations (i.e., other non-normal clusters) to the reference, normal cluster expression, then estimates the likelihood of a CNV occurring at different regions of the chromosomes based on the difference in expression between observation and reference.

The Cottrazm function *STInferCNV* halts at step4. We are not able to move forward with this part of the analysis at this time, and therefore cannot declare the Tumor Boundary.

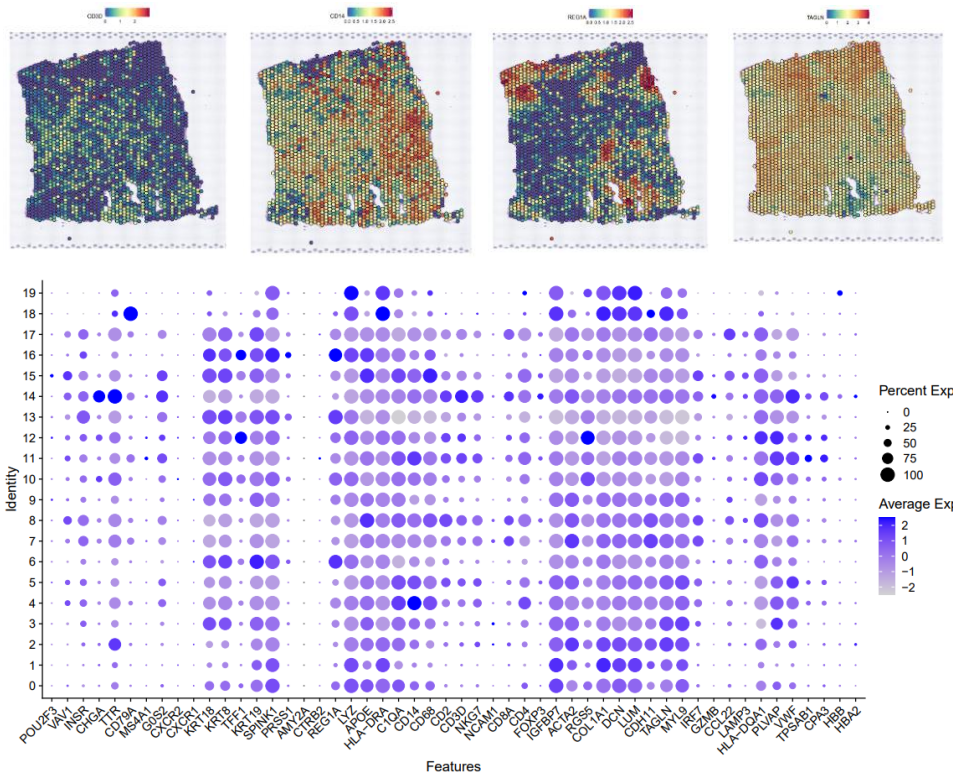
(4) Annotation of clusters

Based on my experience with single-cell, immune clusters mostly cluster apart from malignant. However, I don’t have experience annotating stromal and solid cancer cell types, so the annotation below may be incorrect.

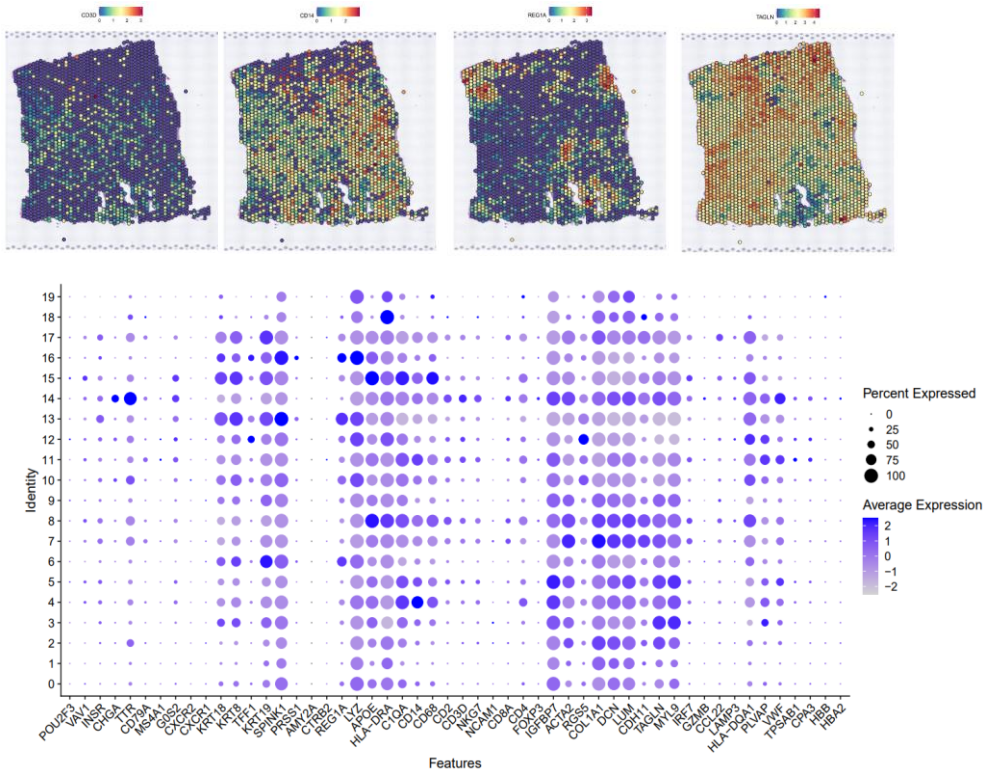
Based on immune “normal” scores and cluster proximity/overlap, the clusters in the purple region are *likely* immune clusters (8, 11, 12, 14, 5?) .

Labeling clusters→ Normalization method matters

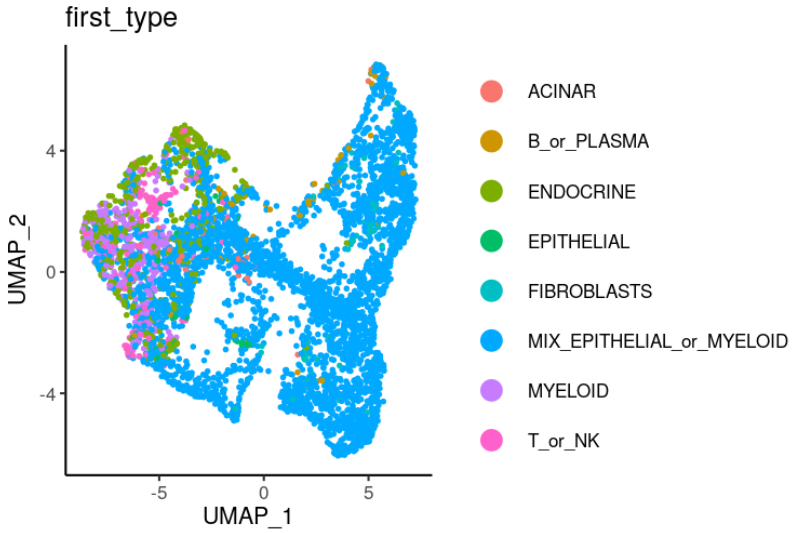
(4.1) Morph Assay

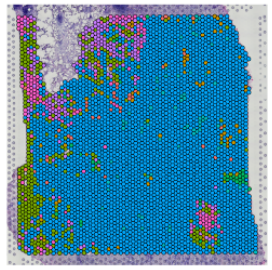


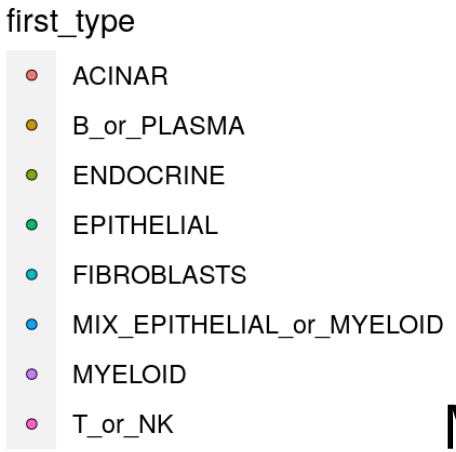
(4.2) Spatial – Seurat Norm Assay

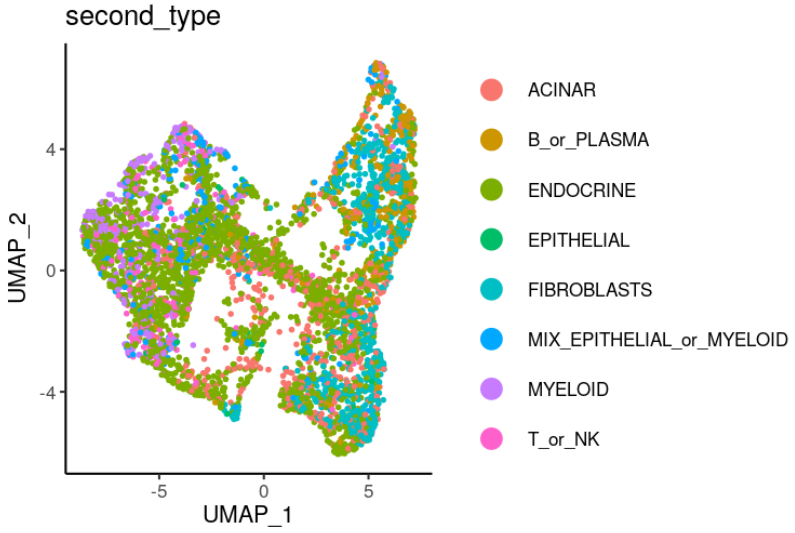


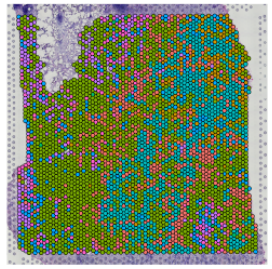
Spatial data deconvolution of cell types

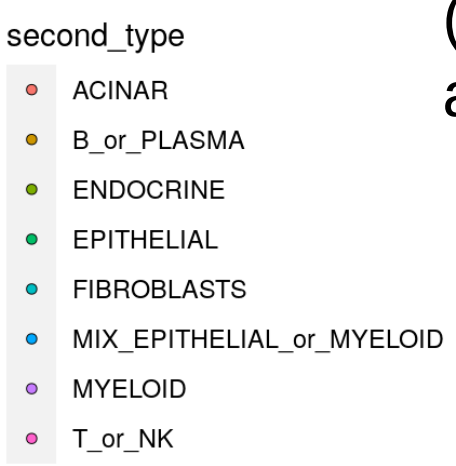








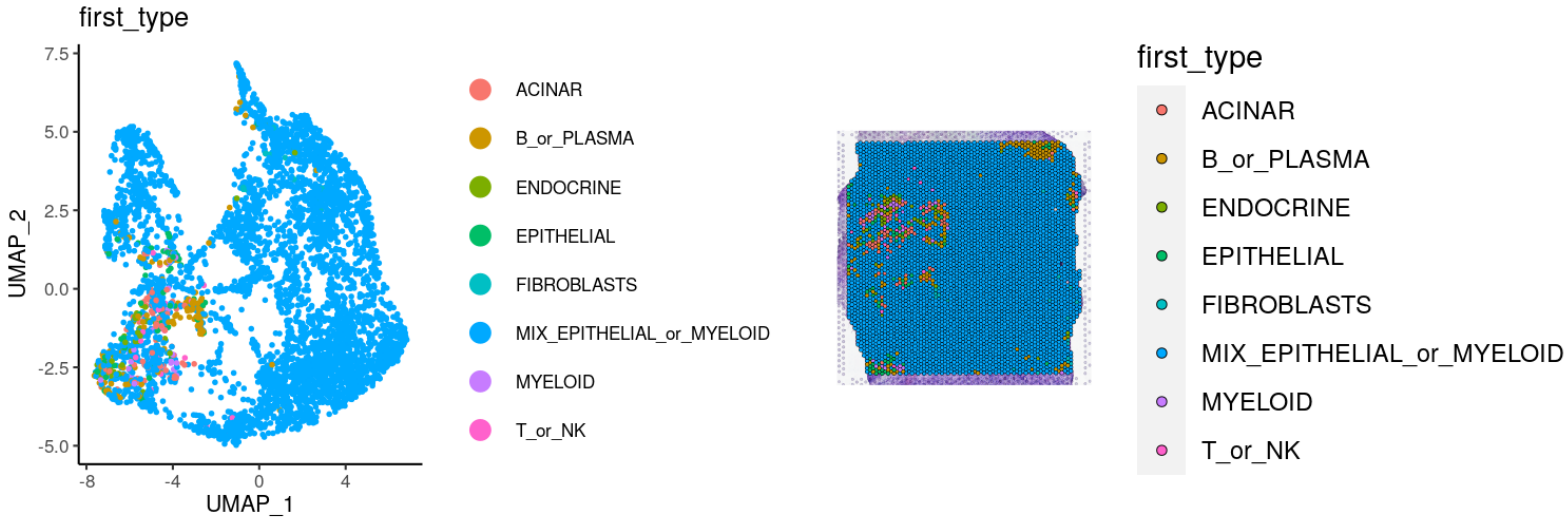


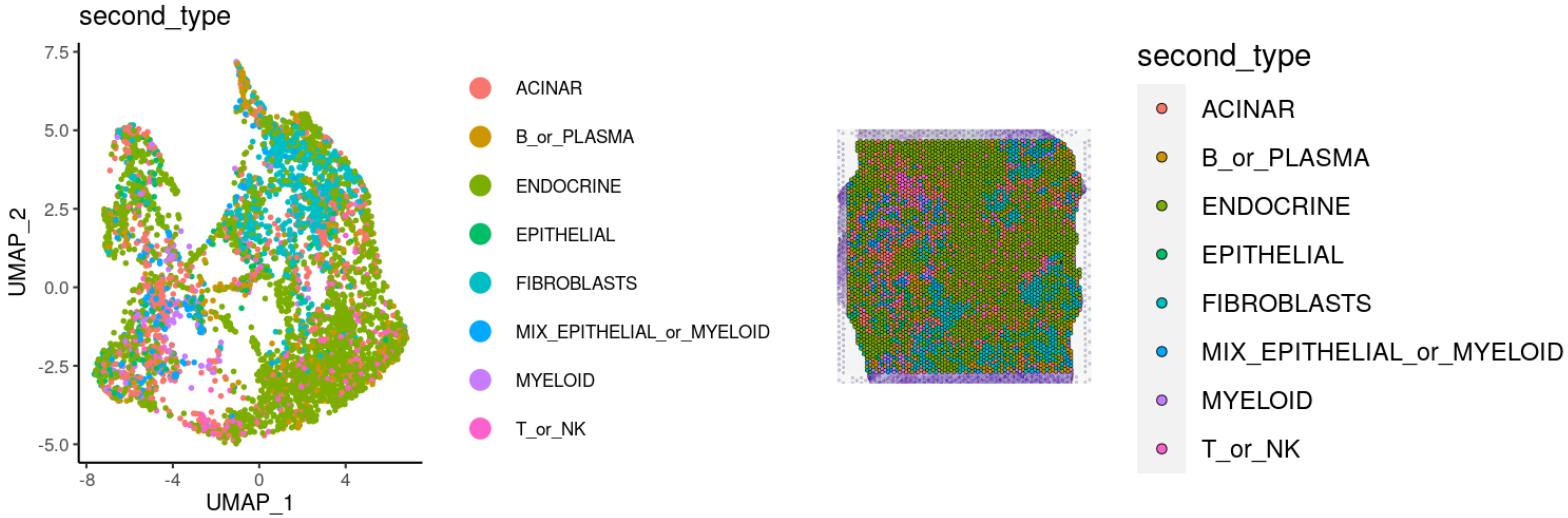


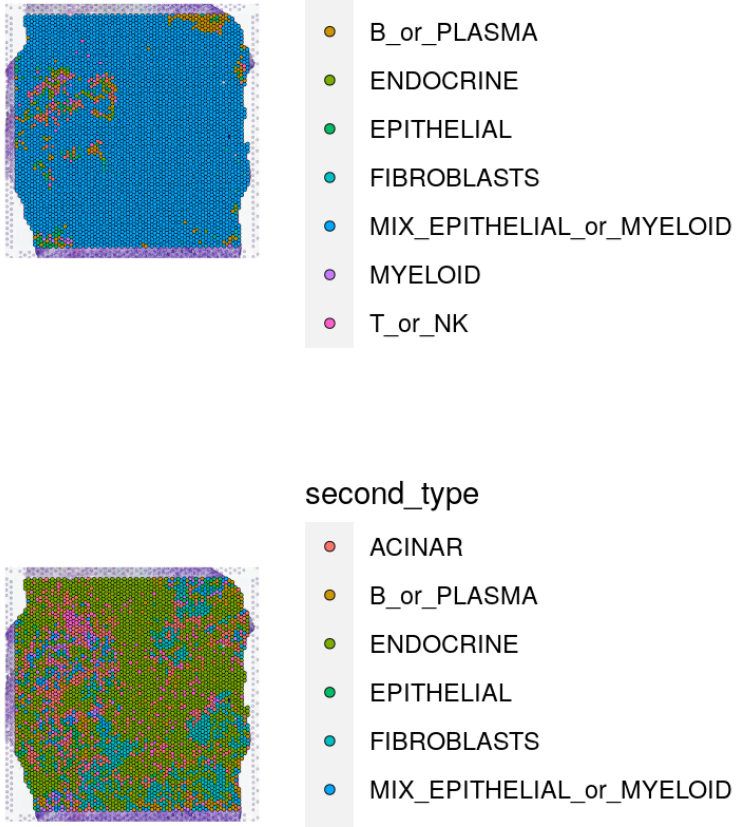
Method: Seurat RCTD

(scRNAseq GPT4 guided annotations)

Spatial data deconvolution of cell types







Method: Seurat RCTD

(scRNAseq GPT4 guided annotations)

Take-homes

- Successfully downloaded and utilized scRNAseq and ST HTAN dataset
- Extracted cell type labels using two methods and applied to ST data for spot deconvolution
- Future directions: proportional spot deconvolution (and finish InferCNV assignments), formalize GPT pipeline infused with manual annotation information/knowledge, correlate immune specific findings to clinical metadata
- **Thank you to all of you and especially the NCI for hosting this!**