

---

**scCAMEL**

***Release 0.26b***

**Yizhou Hu**

**Jul 07, 2023**



## CONTENTS:

<b>1</b>	<b>Installation</b>	<b>3</b>
<b>2</b>	<b>Tutorials_scCAMEL_SWAPLINEv2</b>	<b>5</b>
2.1	Tutorials_scCAMEL-SWAPLINE_SensoryNeurons_Training-SharmaMouse_Predict_ZeiselMouse .	5
2.1.1	Training . . . . .	5
2.1.2	Prefiltering_and_SelectFeatures . . . . .	8
2.1.3	Neural-Network learning . . . . .	10
2.1.4	Accuracy plot, the overall clustering accuracy is ~85% . . . . .	11
2.1.5	Make prediction and visualization in Radar plot . . . . .	11
2.1.6	permutation control . . . . .	14
2.1.7	Cluster consistency and accuracy . . . . .	15
2.1.8	Cell_Type Purity . . . . .	16
2.1.9	association between cell-types . . . . .	18
2.1.10	Save data . . . . .	21
2.1.11	Prediction . . . . .	21
2.1.12	ZeiselMouse_cluster . . . . .	21
2.1.13	Save data . . . . .	26
2.2	Tutorials_scCAMEL-SWAPLINE_mouseDentateGyrus_humanGlioblastoma . . . . .	26
2.2.1	Training . . . . .	27
2.2.2	Prefiltering_and_SelectFeatures . . . . .	28
2.2.3	Neural-Network learning . . . . .	34
2.2.4	Accuracy plot, the overall clustering accuracy is ~95% . . . . .	34
2.2.5	Make prediction and visualization in Radar plot . . . . .	35
2.2.6	permutation control . . . . .	38
2.2.7	Cell_Type Purity . . . . .	39
2.2.8	association between cell-types . . . . .	41
2.2.9	Save data . . . . .	44
2.2.10	Prediction . . . . .	45
2.2.11	Couturier2020_humanGlioblastoma . . . . .	45
2.2.12	Save data . . . . .	48
2.3	Tutorials_scCAMEL-SWAPLINEv1_LiverMacrophage . . . . .	49
2.4	Accuracy plot, the overall clustering accuracy is ~85% . . . . .	53
<b>3</b>	<b>Tutorials_scCAMEL_EVO</b>	<b>57</b>
<b>4</b>	<b>Tutorials_scCAMEL_VICUNA</b>	<b>59</b>
<b>5</b>	<b>Reference</b>	<b>61</b>





Pip installation via: <https://pypi.org/project/scCAMEL/>

Author: Yizhou Hu, YZstudio, Patrik Ernfors lab, Department of Medical Biochemistry and Biophysics, Karolinska Institutet

Reference of scCAMEL-SWAPLINE package: Hu Y.#, Jiang Y.#, Behnan J., Ribeiro MM., Kalantzi C., Zhang M., Lou D., Häring M., Sharma N., Okawa S., Del Sol A., Adameyko I., Svensson M., Persson O., Ernfors P., “Neural-network learning defines glioblastoma features to be of neural crest perivascular or radial glia lineages”, Science Advances, 2022 Jun 10;8(23) <https://www.science.org/doi/10.1126/sciadv.abm6340>

---

Content:



## INSTALLATION

Pip installation via: <https://pypi.org/project/scCAMEL/>

pip install scCAMEL

The original installation will be introduced here. Neural-network based projection and single cell assignment “SWAPLINE.beta” will be released in [https://github.com/ernforslab/Hu-et-al.\\_GBMlineage2022](https://github.com/ernforslab/Hu-et-al._GBMlineage2022)

this part includes 1 section





## TUTORIALS\_SCCAMEL\_SWAPLINEV2

### 2.1 Tutorials\_scCAMEL-SWAPLINE\_SensoryNeurons\_Training-SharmaMouse\_Predict\_ZeiselMouse

**Original Article:** Single cell transcriptomics of primate sensory neurons identifies cell types associated with chronic pain,"Nat Commun", 2021

**Original Article:** Single-Soma Deep RNA sequencing of Human DRG Neurons Reveals Novel Molecular and Cellular Mechanisms Underlying Somatosensation

**Package:** scCAMEL-SWAPLINE.v1

**Research Group:** Ernfors lab

**Link of the datasets:** Mouse sensory neurons, **Dataset references:** Sharma, et al., Zeisel, et al.

#### 2.1.1 Training

```
[1]: import datetime
      today=f"{datetime.datetime.now():%Y-%m-%d}"
      today
```

```
[1]: '2023-05-23'
```

```
[2]: import torch
      import torch.nn as nn
      from torch.autograd import Variable
      import torch.utils.data as Data
      import torchvision
      import matplotlib.pyplot as plt
      from mpl_toolkits.mplot3d import Axes3D
      import torch.utils.data as data_utils
      from matplotlib import cm
      import numpy as np
      import pandas as pd
      import pickle as pickle
      from scipy.spatial.distance import cdist, pdist, squareform
      import pandas as pd
      from sklearn.linear_model import LogisticRegression, LogisticRegressionCV
      from sklearn.model_selection import StratifiedShuffleSplit
      from collections import defaultdict
```

(continues on next page)

(continued from previous page)

```

from sklearn import preprocessing
import matplotlib.patches as mpatches
import torch.nn.functional as F
import math
#import gpytorch

```

```

import urllib.request
import os.path
from scipy.io import loadmat
from math import floor
import anndata
# Make plots inline
%pylab inline

```

%pylab is deprecated, use %matplotlib inline and import the required libraries.  
Populating the interactive namespace from numpy and matplotlib

```

/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/IPython/core/magics/pylab.
↳py:162: UserWarning: pylab import has clobbered these variables: ['floor']
`%matplotlib` prevents importing * from pylab and numpy
warn("pylab import has clobbered these variables: %s" % clobbered +

```

```
[3]: torch.manual_seed(1)    # reproducible
```

```
[3]: <torch._C.Generator at 0x7f7df12cde90>
```

```

[4]: import scCAMEL as scm
from scCAMEL import CamelPrefiltering
from scCAMEL import CamelSwapline
from scCAMEL import CamelEvo

```

```

/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelSwapline.py:
↳637: FutureWarning: The default dtype for empty Series will be 'object' instead of
↳'float64' in a future version. Specify a dtype explicitly to silence this warning.
def addcolor(datax, clustername="Cluster", colorcode="color", predef=pd.Series()):

```

Read data

```
[5]: cd .
```

```
/mnt/f/Dropbox/data/proj/PE_HYZ/IpythonRecord/Monkey_DRG
```

```

[6]: cd /mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets
/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets

```

```

[7]: screfall=anndata.read("SensoryNeurons_Zeisel-Sharma.h5ad")
screfall

```

```

[7]: AnnData object with n_obs × n_vars = 12635 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'
      uns: 'refcolor_dict'

```

```
[8]: #if the matrix is sparse matrix
      #screfall.X=screfall.X.todense()
```

```
[9]: set(screfall.obs["Cluster"])
```

```
[9]: {'NF1',
      'NF2_3',
      'NF4',
      'NP1',
      'NP2',
      'NP3',
      'PEP1_PSPEP2',
      'PEP1_PSPEP3',
      'PEP1_PSPEP4',
      'PEP1_PSPEP5',
      'PEP2',
      'Sharma_Abeta-Field',
      'Sharma_Abeta-RA-LTMR',
      'Sharma_Adelta-LTMR',
      'Sharma_C-LTMR',
      'Sharma_CGRP-Alpha',
      'Sharma_CGRP-Beta',
      'Sharma_CGRP-Epsilon',
      'Sharma_CGRP-Eta',
      'Sharma_CGRP-Gamma',
      'Sharma_CGRP-Theta',
      'Sharma_CGRP-Zeta',
      'Sharma_Nonpeptidergic nociceptors',
      'Sharma_Proprioceptors',
      'Sharma_Sst',
      'Sharma_TrpM8',
      'TRPM8',
      'Th'}
```

```
[10]: scref=screfall[screfall.obs["batch"]=="0"]
```

```
[11]: set(scref.obs["Cluster"])
```

```
[11]: {'Sharma_Abeta-Field',
      'Sharma_Abeta-RA-LTMR',
      'Sharma_Adelta-LTMR',
      'Sharma_C-LTMR',
      'Sharma_CGRP-Alpha',
      'Sharma_CGRP-Beta',
      'Sharma_CGRP-Epsilon',
      'Sharma_CGRP-Eta',
      'Sharma_CGRP-Gamma',
      'Sharma_CGRP-Theta',
      'Sharma_CGRP-Zeta',
      'Sharma_Nonpeptidergic nociceptors',
      'Sharma_Proprioceptors',
      'Sharma_Sst',
      'Sharma_TrpM8'}
```

```
[12]: scref.obs.groupby(["Cluster"]).count()
```

```
[12]:
```

	mtrain_index	sample	species	batch	color
Cluster					
Sharma_Abeta-Field	257	257	257	257	257
Sharma_Abeta-RA-LTMR	273	273	273	273	273
Sharma_Adelta-LTMR	182	182	182	182	182
Sharma_C-LTMR	1554	1554	1554	1554	1554
Sharma_CGRP-Alpha	1440	1440	1440	1440	1440
Sharma_CGRP-Beta	135	135	135	135	135
Sharma_CGRP-Epsilon	850	850	850	850	850
Sharma_CGRP-Eta	270	270	270	270	270
Sharma_CGRP-Gamma	705	705	705	705	705
Sharma_CGRP-Theta	758	758	758	758	758
Sharma_CGRP-Zeta	333	333	333	333	333
Sharma_Nonpeptidergic nociceptors	2817	2817	2817	2817	2817
Sharma_Proprioceptors	234	234	234	234	234
Sharma_Sst	761	761	761	761	761
Sharma_TrpM8	488	488	488	488	488

## 2.1.2 Prefiltering\_and\_SelectFeatures

```
[13]: path='/mnt/f/Dropbox/data/proj/PE_HYZ/PublicDataSet/'
filename='PANTHER_cell_cycle_genes.txt'
#dfpfc2= prefilteringTest.prefilter(df_f=dfpfc,filename=filename, path=path)
scref= scm.CamelPrefiltering.prefilter(datax=scref,filename=filename, path=path)

CamelRunning_Prefilter...

/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
py:70: ImplicitModificationWarning: Trying to modify attribute `.var` of view,
initializing view as actual.
datax.var["Filter1"]=ftlist

CamelRunning_Prefilter...Finished
```

```
[14]: scref=scm.CamelPrefiltering.DataScaling(screef)
```

```
[15]: dfdev2=pd.DataFrame(screef.X,index=screef.obs.index,columns=screef.var.index).T
dfdev2=dfdev2.loc[screef.var['Filter1']==True]
dfdev2.shape
```

```
[15]: (12240, 11057)
```

```
[16]: markerlist=scm.CamelSwapline.enrichmentscoreBETA(dfpcfclus=screef.obs["Cluster"], df_
dev=dfdev2, fc=3, pvalcutoff=0.1, shortcut=True)

Camel...Running: clusteringValue1...
Camel...Running: clusteringValue2...
2448
[Processing15
Camel...Running: CrossChecking...
-0%--6%--13%--20%--26%--33%--40%--46%--53%--60%--66%--73%--80%--86%--93%-Camel...Running:
output genelist...
```

```
[17]: markerlist2=list(set(markerlist))
      len(markerlist2)
```

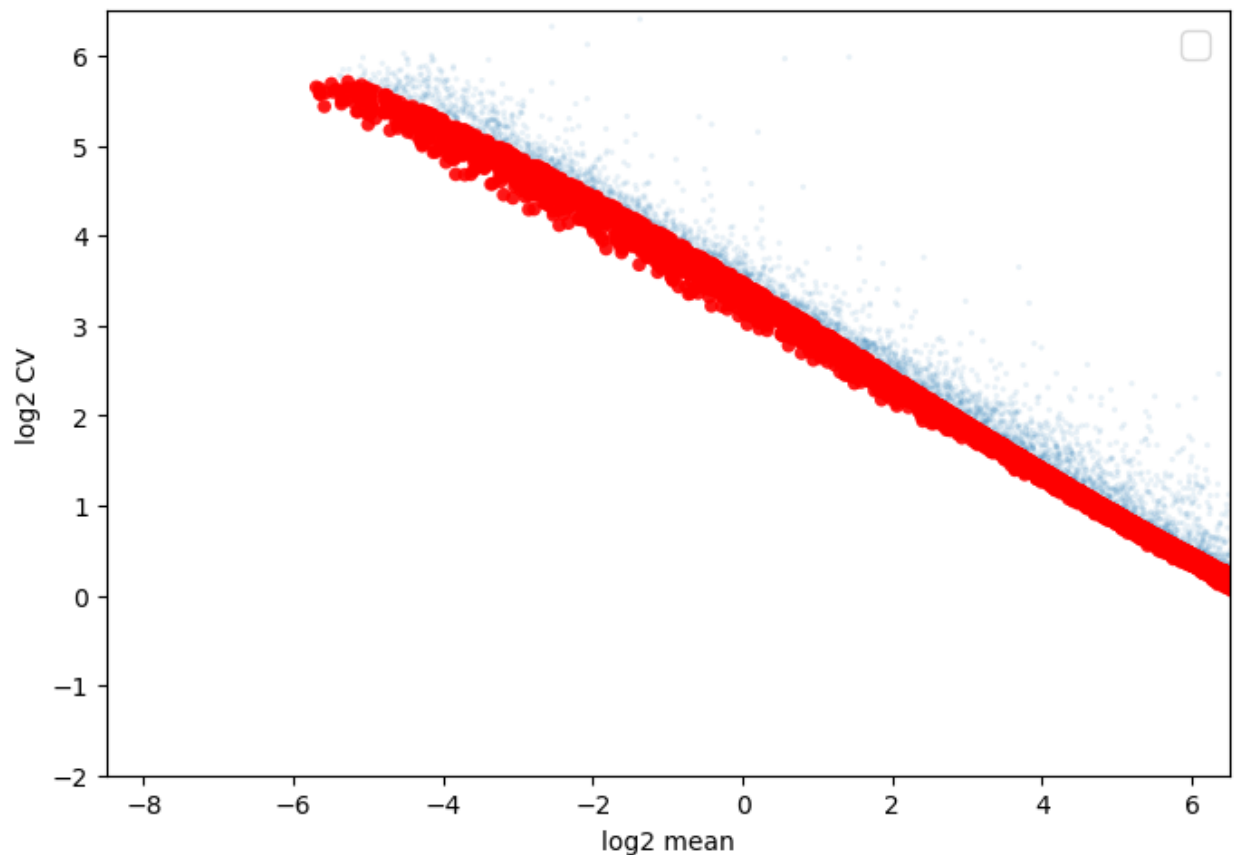
```
[17]: 2738
```

```
[18]: scref, score, thrs, ax=scm.CamelPrefiltering.MVgenes(datax=scref,wt=3,
      ↪markerlist=markerlist2,
      plotfig=True, Xlow=-8.5, Xhigh=6.5,
      ↪Ylow=-2, Yhigh=6.5,alphaValue=0.9, sValue=30,
      fig_args={'figsize': (8, 8), 'facecolor': 'white', 'edgecolor': 'white'})
```

CamelRunning\_MVgenes...

No artists with labels found to put in legend. Note that artists whose label start with  
 ↪an underscore are ignored when legend() is called with no argument.

CamelRunning\_MVgenes...Finished



```
[19]: len(scref.var.index[scref.var["MVgene"]])
```

```
[19]: 5958
```

```
[20]: scref2=scref
```

```
[21]: #####
      #####
```

(continues on next page)

(continued from previous page)

```
#remeber to change the file path in tftable
#####
#####
scref =scm.CamelPrefiltering.MVgene_Scaling(datax=scref2,score=score,
                                           thr=thr,
↳ TPTT=100000,      mprotogruop=scref.obs["Cluster"].values,commongene=None,
↳
↳ sharedMVgenes=None,std_scaling=True,
    tftable="/mnt/f/Dropbox/data/proj/PE_HYZ/PublicDataSet/FantomTF2CLUSTER_human_
↳official.txt", learninggroup="train")
```

CamelRunning---GenesScaling...

```
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:791: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna
↳will be keyword-only.
    scalepfc = dfpfc.div(dfpfc.std(1), axis=0).dropna(0)
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:792: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna
↳will be keyword-only.
    scalepfc = dfpfc.astype(float).dropna(0)
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:794: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna
↳will be keyword-only.
    scalepfc = scalepfc.dropna(0)
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:795: FutureWarning: Passing a set as an indexer is deprecated and will raise in a
↳future version. Use a list instead.
    dfpfc_dev = scalepfc.loc[set(scalepfc.index) & set(sharedMVgenes)].dropna()
```

CamelRunning---TrainingGenesScaling...Finished

[22]: scref

```
[22]: AnnData object with n_obs × n_vars = 11057 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names'
      obsm: 'train_set_values'
```

### 2.1.3 Neural-Network learning

```
[23]: net=scm.CamelPrefiltering.NNclassifier(
      datax=scref,
      epochNum=100,
      learningRate=0.0075,
      verbose=0,
      optimizerMmentum=0.8,
      dropout=0.3,
```

(continues on next page)

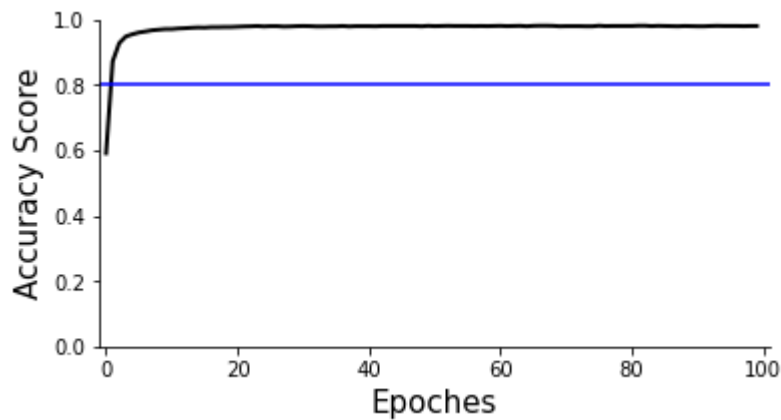
(continued from previous page)

```
#imizer__nesterov=True,
)
```

```
CamelRunning---NNclasffier_in_cpu...
CamelRunning---NNclasffier_in_cpu...Finished
```

### 2.1.4 Accuracy plot, the overall clustering accuracy is ~85%

```
[24]: ax=scm.CamelPrefiltering.AccuracyPlot( nnModel=net, accCutoff=0.8,
      Xlow=-1, Ylow=0.0, Yhigh=1,
      )
```



### 2.1.5 Make prediction and visualization in Radar plot

```
[23]: net=scm.CamelPrefiltering.NNclassifier(
      datax=scref,
      epochNum=30,
      learningRate=0.0075,
      verbose=0,
      optimizerMomentum=0.8,
      dropout=0.3,
      #imizer__nesterov=True,
      )
```

```
CamelRunning---NNclasffier_in_cpu...
CamelRunning---NNclasffier_in_cpu...Finished
```

```
[24]: scref
```

```
[24]: AnnData object with n_obs × n_vars = 11057 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names'
      obsm: 'train_set_values'
```

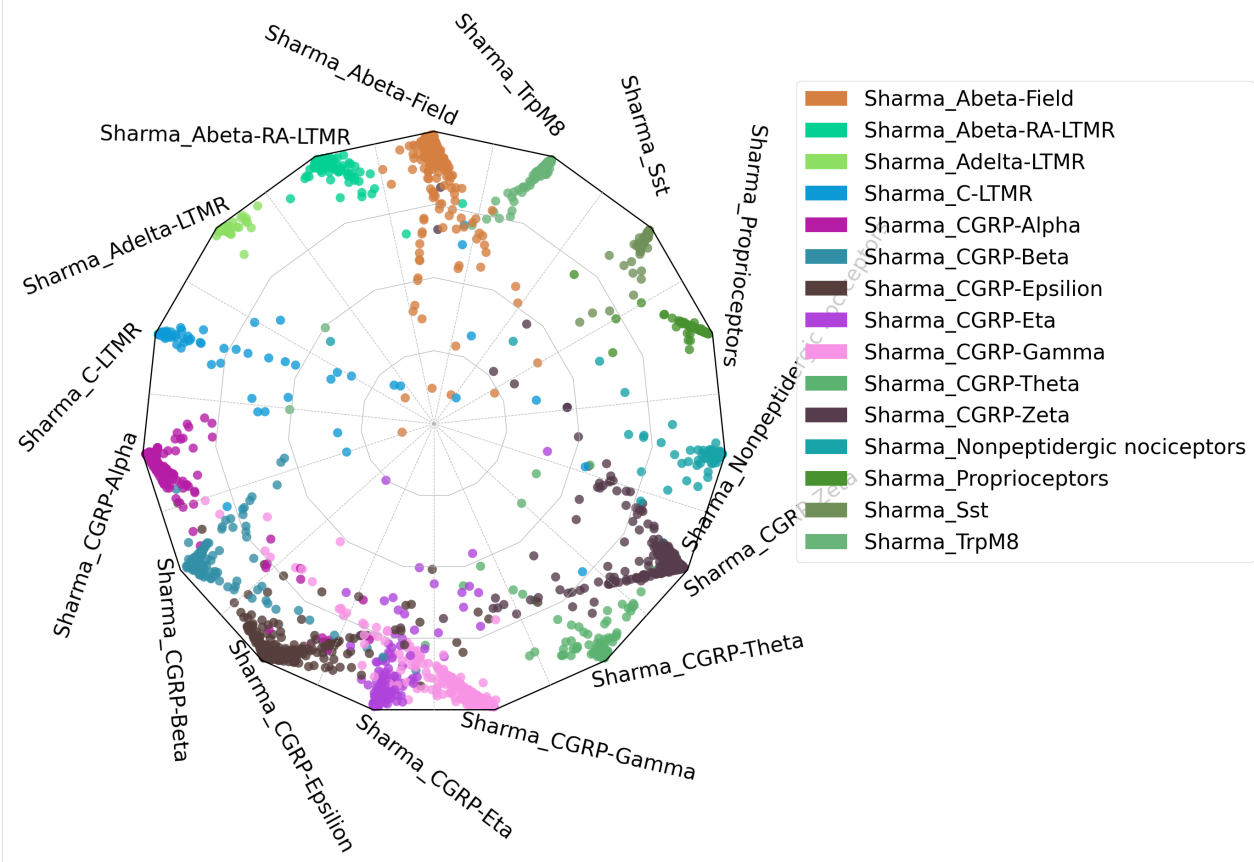
```
[25]: #if color is not defined: scref.obs[ 'color']
      #scref=scm.CamelSwapline.addcolor(datax=scref,clustername="Cluster", colorcode="color")
```

```
[26]: scref.uns["mwanted_order"] =list(sort(list(set(scref.obs["Cluster"]))))
```

```
[27]: cd .
```

```
/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets
```

```
[28]: #radar plot
      scref=scm.CamelSwapline.prediction(datax=scref, mcolor_dict=scref.uns["refcolor_dict"] ,
      ↪ net=net,learninggroup="train", radarplot=True,fontsizeValue=18,
      ncolnm=3, bbValue=(1.2, 1.05) )
      #plt.savefig("upload_%s_RadarPlot_MacaqueRef_Merged_cluster.pdf"%today,bbox_inches='tight'
      ↪')
```



```
[29]: scref
```

```
[29]: AnnData object with n_obs × n_vars = 11057 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_
      ↪ Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'train_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'
```

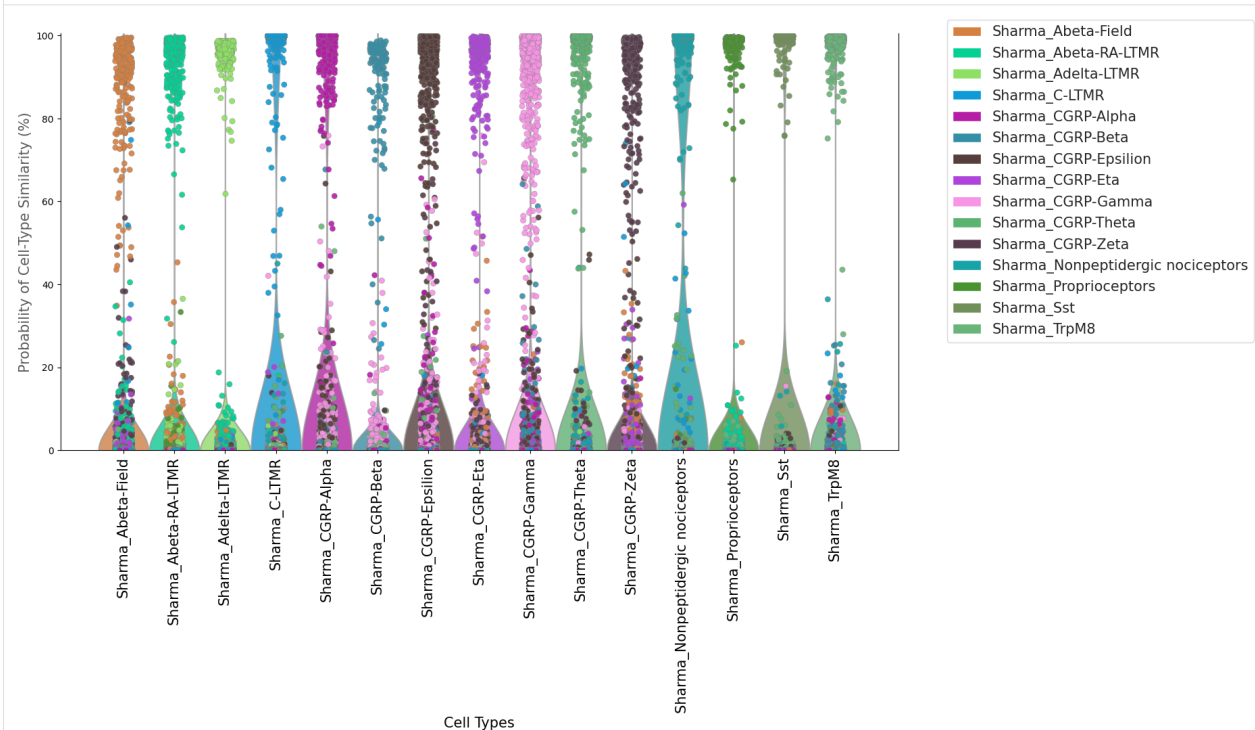
```
dfpfccclus = scref.obs[["mtrain_index","Cluster"]].T dfpfccclus
```



```
[ ]: #####
#####
#Multiplot_Violin
# each column along X axis represents a cell type, and the y-axis represents the cell-
↳ type probability
# each column along X axis contain all cells in the training, shown as dots with
↳ different colors
# in the perfect condition only the cells of the same cell-type column will reach to the
↳ maxium (100%) probability, all other cells will be 0 probability
#however, in each cell-type column, some cells from other cell types may exhibit high
↳ probability, two reasons: 1. they may shared some features with this---
# cell-type, and these features drag them to the current cell-type; 2. they are
↳ improperly classified due to many factors, such as low quality data, different
↳ clustering algorithm.....
#####
#####
```

```
[31]: fig=scm.CamelSwapline.ProbMultiPlot( datax=scref, mcolor_dict=scref.uns["refcolor_dict"],
↳ )
```

<Figure size 2500x1000 with 0 Axes>

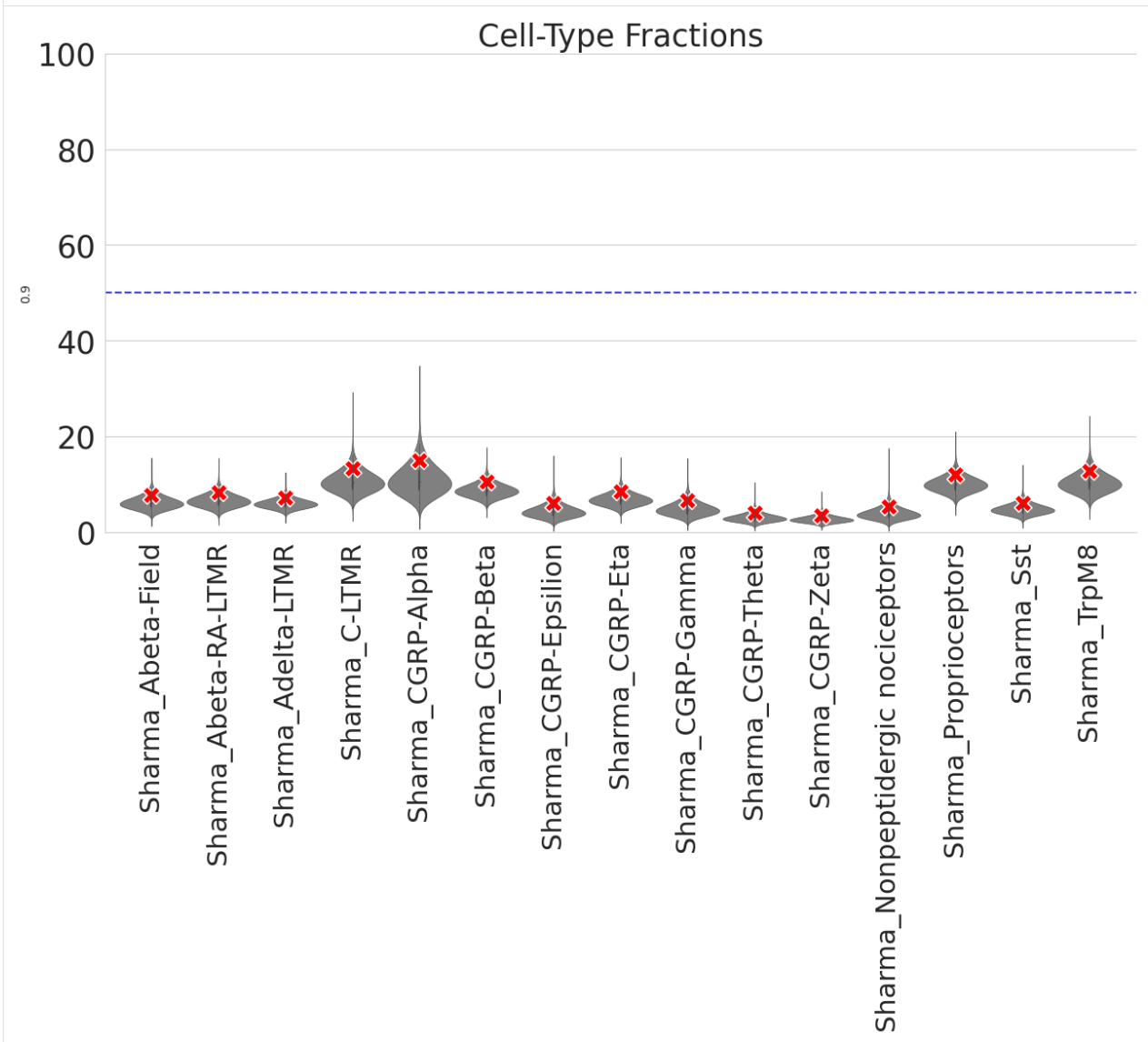


## 2.1.6 permutation control

```
[32]: ## the whole data matrix is randomized, the red X represents 95% confidence of each cell-  
      ↪ type
```

```
[33]: dfest0, ratiodef=scm.CamelSwapline.permutationTest(datax=scref,net=net,num=50,↪  
      ↪plotshow=True)
```

<Figure size 640x480 with 0 Axes>



## 2.1.7 Cluster consistency and accuracy

```
[ ]: #####
# The script uses the original, defined cell type as a reference to compare with the
# predicted cell type of each cell.
# If the predicted cell type matches the original defined cell type, the cell is deemed
# to be 'consistently assigned'.
# Conversely, if the predicted cell type does not match, the cell is considered
# 'inconsistently assigned', indicating possible incorrect clustering.
# Output File: This contains the ID of inconsistent cells, the originally defined cell
# type, and the predicted cell type.
# Output Plot: This displays the ratio (%) of cells that are correctly (consistently)
# and incorrectly (inconsistently) assigned cell types.

#####
```

```
[35]: len(set(scref.obs.index))
```

```
[35]: 11057
```

```
[36]: scref
```

```
[36]: AnnData object with n_obs × n_vars = 11057 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_
      ↪ Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'train_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'
```

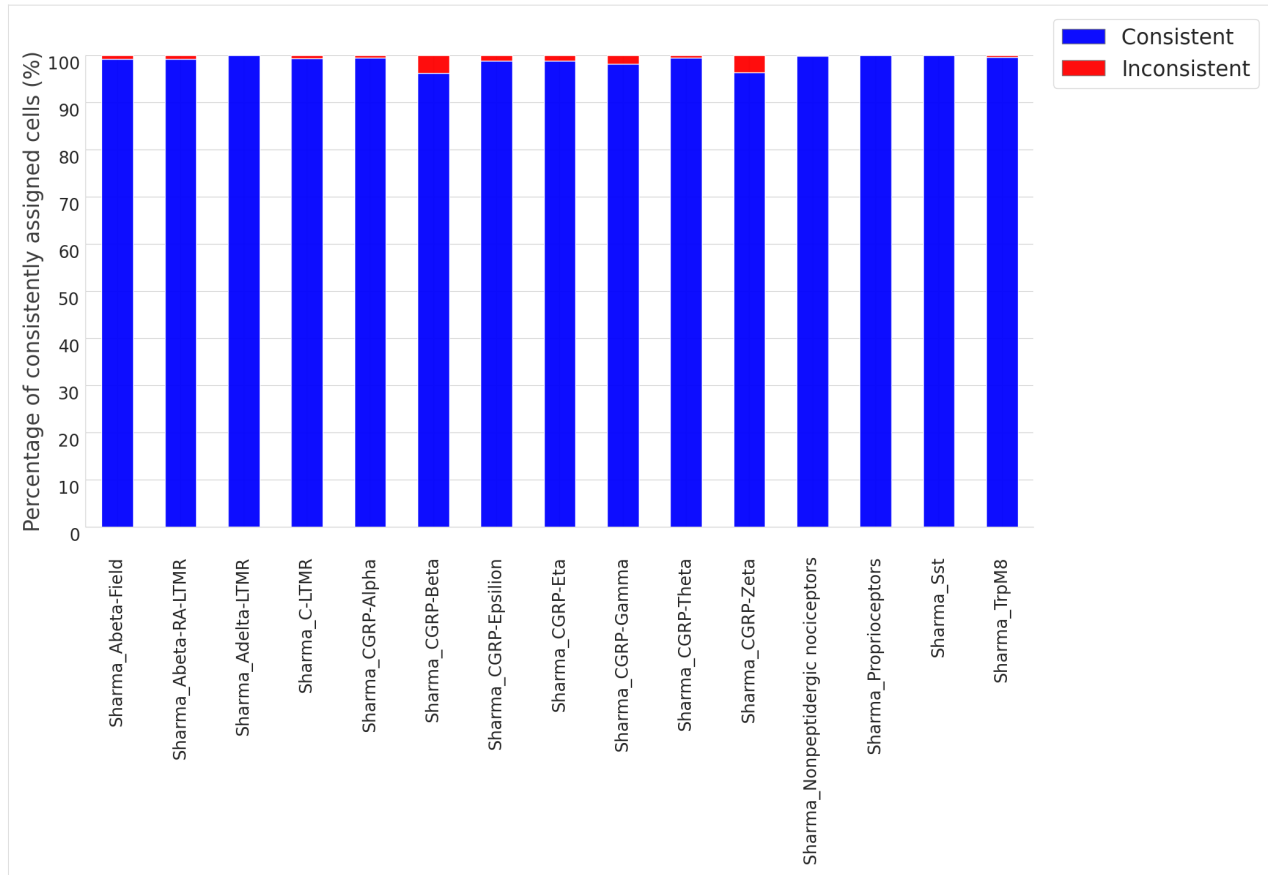
```
[37]: cd .
```

```
/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets
```

```
[38]: scref=scm.CamelSwapline.ConsistantAssign(datax=scref,dfsig=dfsig0,outputfilepath=
      ↪ "upload_%s_Sharma_Ref_InconsistantCells.csv"%today,
      outputPlot=True )
```

```
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelSwapline.py:
      ↪ 466: FutureWarning: The frame.append method is deprecated and will be removed from
      ↪ pandas in a future version. Use pandas.concat instead.
      dfprob1=dfprob1posi.append(dfprob1neg)
```

```
<Figure size 2500x1000 with 0 Axes>
```

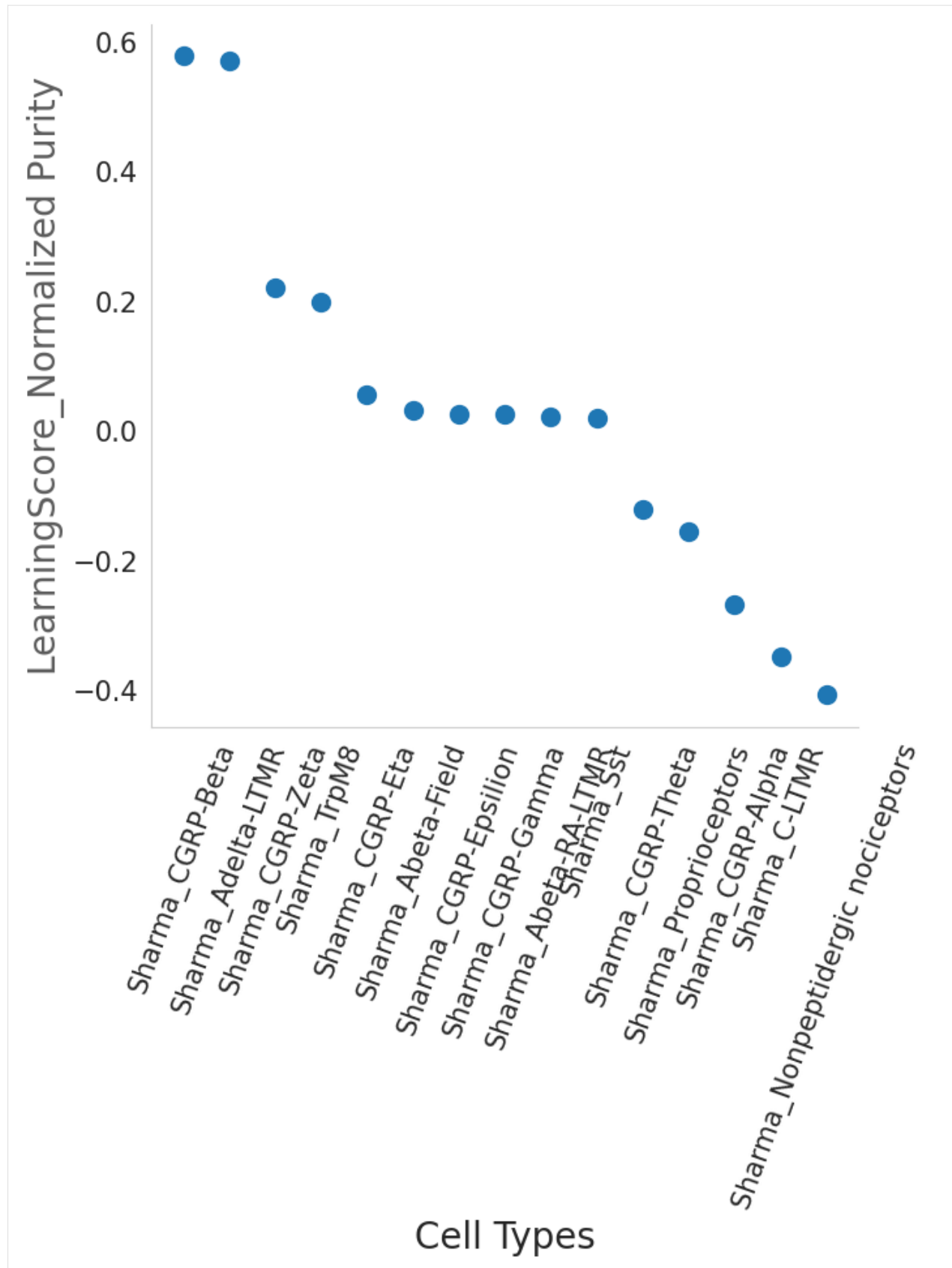


## 2.1.8 Cell\_Type Purity

```
[39]: #The ratio of the purity entropy for each cluster based on their learning scores, is_
      ↪ used as a measure of purity.
      #The function returns a pandas dataframe sorted by the purity score
```

```
[40]: dfpurity1=scm.CamelSwapline.PurityEstimationLearningScore(datax=scref, clusterlist=
      ↪ "Cluster", elbow=False, figureplot=True)
```

<Figure size 640x480 with 0 Axes>



## 2.1.9 association between cell-types

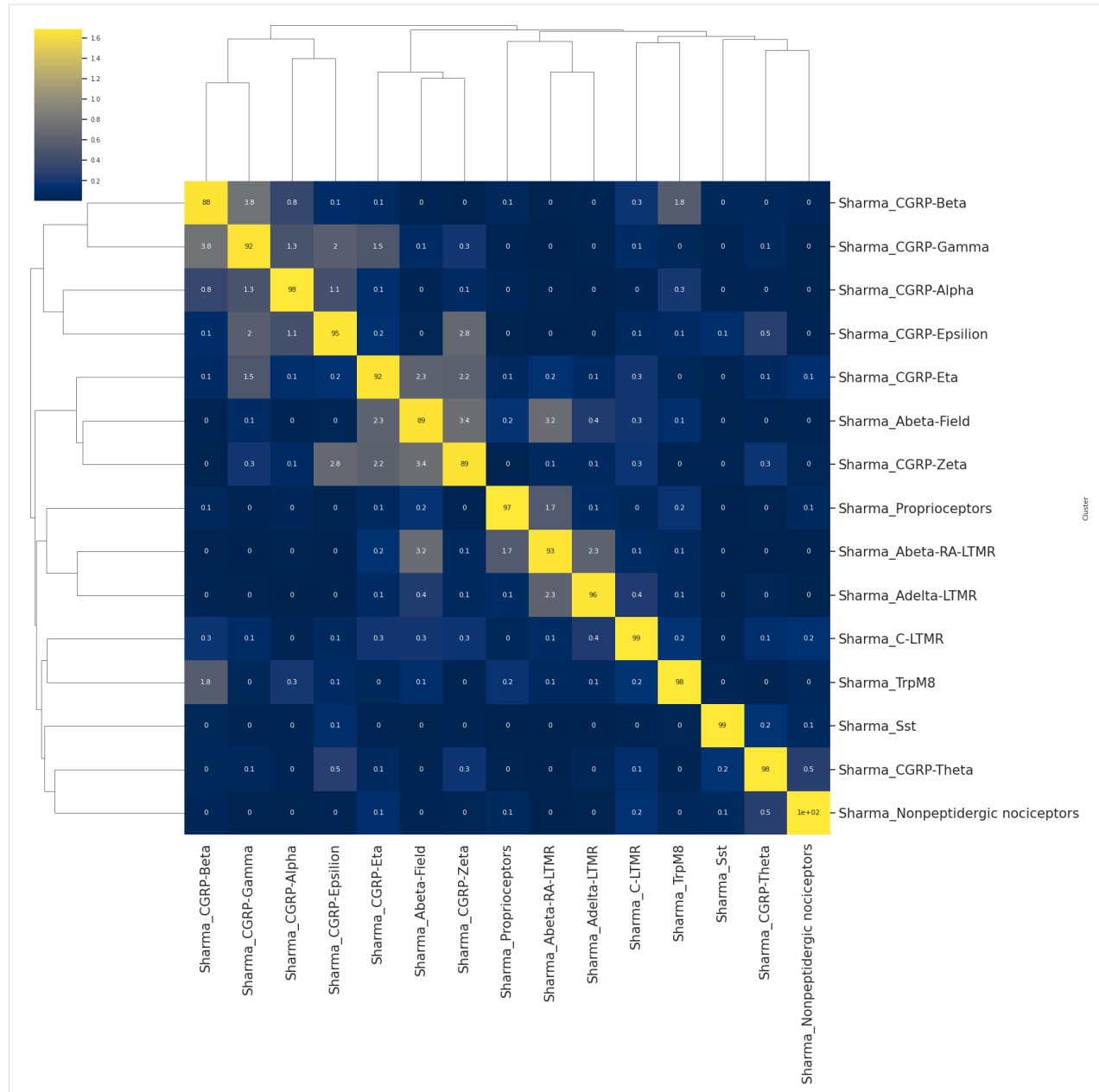
```
[41]: scref
```

```
[41]: AnnData object with n_obs × n_vars = 11057 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color',
      ↪ 'PredictCluster', 'ClusterConsistanceScore'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_
      ↪ Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'train_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'
```

```
[42]: # the heatmap of hierarchical clustering represents the cell-type similarity or
      ↪ association
      #color from dark purple to light yellow represents the association from low to high
      #number inside of eahc square indicating the association value.
```

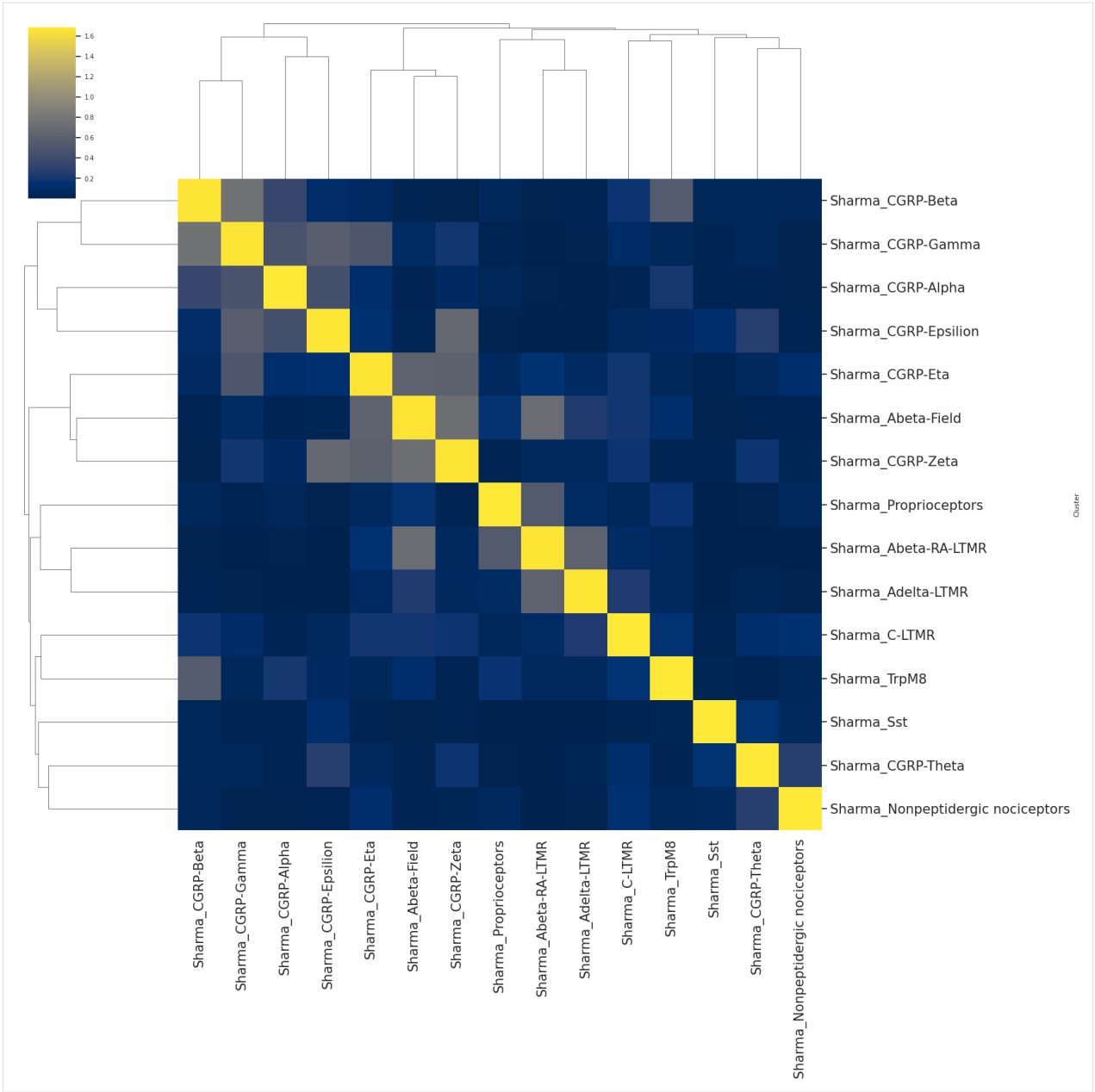
```
[43]: scm.CamelSwapline.CellTypeSimilarity(datax=scref, labelnum=True, metricvalue=
      ↪ 'correlation', methodvalue="complete")
```

```
<Figure size 1500x1500 with 0 Axes>
```



```
[44]: scm.CamelSwapline.CellTypeSimilarity(datax=scref, labelnum=False, metricvalue=
      ↳ 'correlation', methodvalue="complete")
```

<Figure size 1500x1500 with 0 Axes>



[ ]:



## 2.1.10 Save data

```
[45]: scref
[45]: AnnData object with n_obs × n_vars = 11057 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color',
      ↪ 'PredictCluster', 'ClusterConsistanceScore'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_
      ↪ Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'train_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'

[46]: cd .

      /mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets

[47]: work_dir="/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/
      ↪ Datasets"
      QueryName="SharmaMouse"
      TrainingName="SharmaMouse"
      filename="%s_%s_Ref%s_MergeCluster.h5ad"%(QueryName,TrainingName,today)

[48]: os.path.join(work_dir,filename)
[48]: '/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets/
      ↪ SharmaMouse_SharmaMouse_Ref2023-05-23_MergeCluster.h5ad'

[49]: CamelSwapline.writedata(adatax=scref,filename=filename,filepath=work_dir)

[50]: scref.X=scref.X.todense()

[ ]:
```

## 2.1.11 Prediction

## 2.1.12 ZeiselMouse\_cluster

```
[51]: scpdt=screfall[screfall.obs["batch"]=="1"]

[52]: set(scpdt.obs["Cluster"])
[52]: {'NF1',
      'NF2_3',
      'NF4',
      'NP1',
      'NP2',
      'NP3',
      'PEP1_PSPEP2',
      'PEP1_PSPEP3',
      'PEP1_PSPEP4',
      'PEP1_PSPEP5',
```

(continues on next page)

(continued from previous page)

```
'PEP2',
'TRPM8',
'Th'}
```

[53]: scpdt

[53]: View of AnnData object with  $n_{\text{obs}} \times n_{\text{vars}} = 1578 \times 15677$   
 obs: 'Cluster', 'mtrain\_index', 'sample', 'species', 'batch', 'color'  
 uns: 'refcolor\_dict'

[54]: scpdt.var.index

[54]: Index(['HTR1F', 'GPAT2', 'CMTM7', 'SST', 'NTS', 'IL31RA', 'NPPB', 'ADA',  
 'CFAP61', 'CBLN1',  
 ...  
 'KRTAP7-1', 'NKX1-1', 'NKX1-2', 'NKX2-2', 'NKX3-1', 'NKX3-2', 'NKX6-1',  
 'NKX6-2', 'NKX6-3', 'RPL9'],  
 dtype='object', length=15677)

[55]: scpdt2=scpdt.copy()  
 scpdt2.X=scpdt.X\*(scref.X.std()/scpdt2.X.std())

[56]: #####  
 #####  
 #remeber to change the file path in tftable  
 #####  
 #####  
 scpdt =scm.CamelPrefiltering.MVgene\_Scaling(datax=scpdt2,TPTT=1000000,   
 ↪commongene=scref.var.index.tolist(),  
 sharedMVgenes=scref.uns[ 'train\_set\_gene'].  
 ↪tolist(),  
 ↪ std\_scaling=True,score=None, thr=None, mprotogruop=None,  
 tftable="f:/Dropbox/data/proj/PE\_HYZ/PublicDataSet/FantomTF2CLUSTER\_human\_official.  
 ↪txt", learninggroup="test")

CamelRunning---GenesScaling...

```
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.  

↪py:820: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna  

↪will be keyword-only.  

scalegbm = dfpfc.div(dfpfc.std(1), axis=0).dropna(0)
```

CamelRunning---TestGenesScaling...Finished

[57]: scpdt

[57]: AnnData object with  $n_{\text{obs}} \times n_{\text{vars}} = 1578 \times 15677$   
 obs: 'Cluster', 'mtrain\_index', 'sample', 'species', 'batch', 'color'  
 var: 'RefGeneList'  
 uns: 'refcolor\_dict', 'train\_set\_gene', 'mclasses\_names'  
 obsm: 'test\_set\_values'

```
[58]: #del scpdt.obs["color"]
```

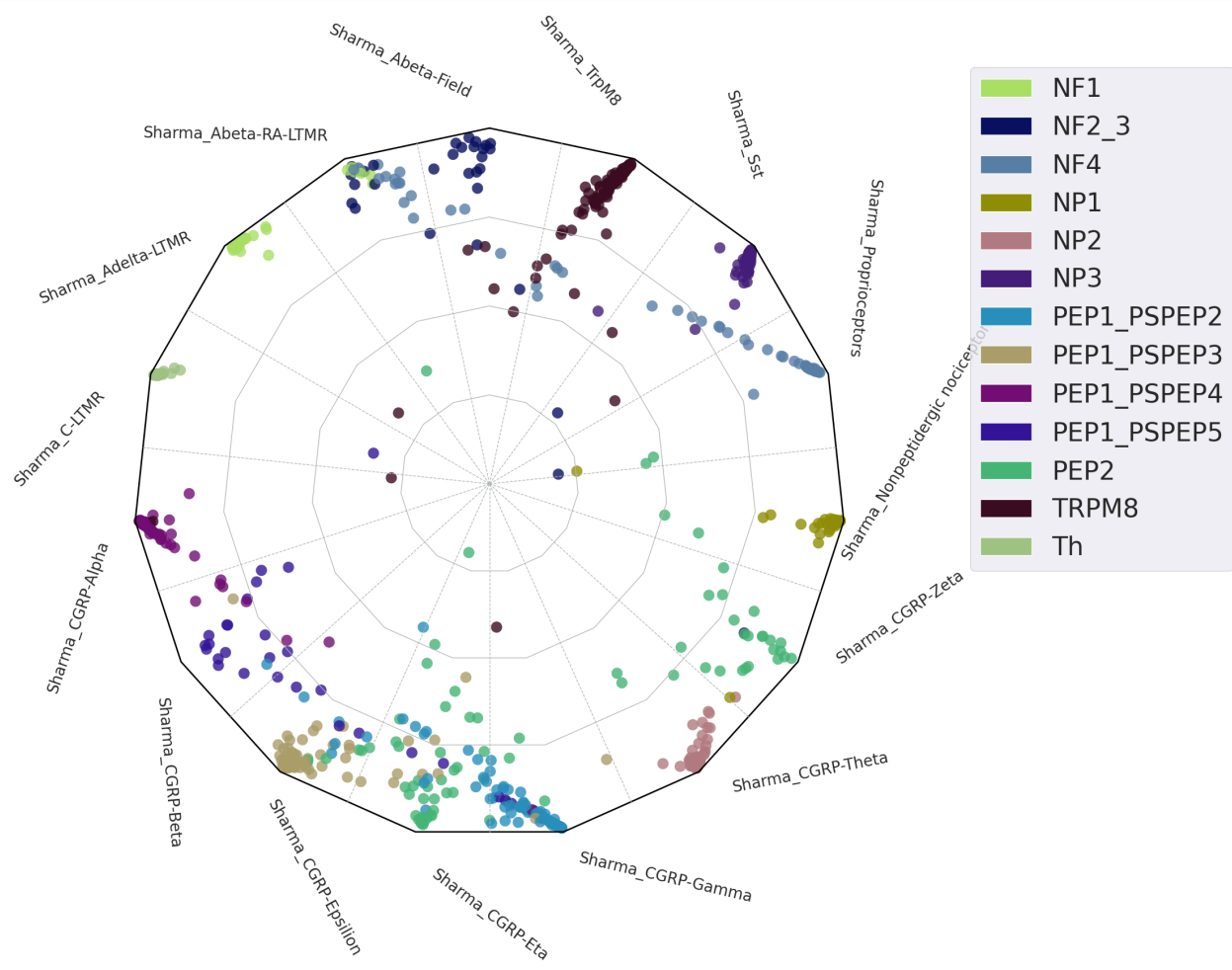
```
[59]: # if color is not defined
#scpdt=scm.CamelSwapline.addcolor(datax=scpdt,clustername="Cluster", colorcode="color")
```

```
[60]: scpdt.uns["mwanted_order"] =list(sort(list(set(scpdt.obs["Cluster"]))))
```

```
[61]: scpdt
```

```
[61]: AnnData object with n_obs x n_vars = 1578 x 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'
      var: 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names', 'mwanted_order'
      obsm: 'test_set_values'
```

```
[62]: test=scm.CamelSwapline.prediction(datax=scpdt, mcolor_dict=scpdt.uns["refcolor_dict"],
      net=net,learninggroup="test", radarplot=True, fontsizeValue=35,
      datarefplot=scref, ncolnm=1, bbValue=(1.1, 1.05))
```

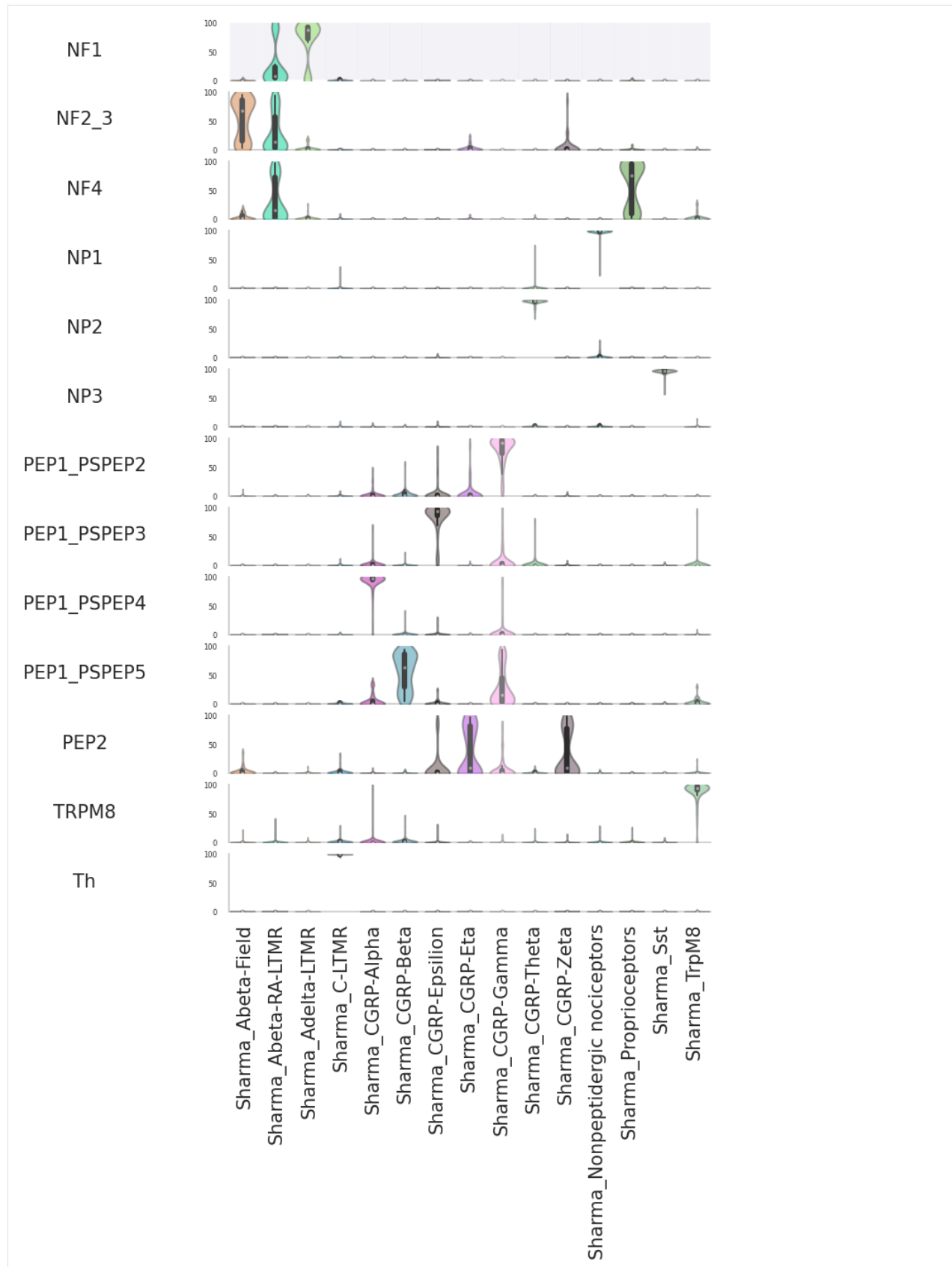


```
[63]: scpdt
```

```
[63]: AnnData object with n_obs × n_vars = 1578 × 15677
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'
      var: 'RefGeneList'
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_
      ↪ Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'test_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'

[64]: genename=sort(list(set(scpdt.obs["Cluster"])))
      name=sort(list(set(scref.obs["Cluster"])))

[65]: dfprob=pd.DataFrame(scpdt.obsm['Celltype_Score'])
      dfprob.columns=scpdt.uns['Celltype_Score_RefCellType']
      dfprob.index=scpdt.obs.index
      dfmk=dfprob.astype(float).join(scpdt.obs["Cluster"],how="inner").T
      dfprob=scm.CamelSwapline.CellTypeSimilarityViolinPlot(datax=scpdt, dataref=scref)
```



### 2.1.13 Save data

```
[66]: cd .  
      /mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets  
  
[67]: scpdt  
[67]: AnnData object with n_obs × n_vars = 1578 × 15677  
      obs: 'Cluster', 'mtrain_index', 'sample', 'species', 'batch', 'color'  
      var: 'RefGeneList'  
      uns: 'refcolor_dict', 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_  
      ↪ Score_RefCellType', 'Celltype_OrderNumber'  
      obsm: 'test_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'  
  
[68]: work_dir="/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/  
      ↪ Datasets"  
      QueryName="ZeiselMouse"  
      TrainingName="SharmaMouse"  
      filename="%s_%s_Ref%s_MergeCluster.h5ad"%(QueryName,TrainingName,today)  
  
[69]: os.path.join(work_dir,filename)  
[69]: '/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/Datasets/  
      ↪ ZeiselMouse_SharmaMouse_Ref2023-05-23_MergeCluster.h5ad'  
  
[70]: CamelSwapline.writedata(adatax=scref,filename=filename,filepath=work_dir)  
  
[ ]:
```

## 2.2 Tutorials\_scCAMEL-SWAPLINE\_mouseDentateGyrus\_humanGlioblastoma

**Original Article:** Neural network learning defines glioblastoma features to be of neural crest perivascular or radial glia lineages,"Sci. Adv.", 2022

**Package:** scCAMEL-SWAPLINE.v1

**Author:** Yizhou Hu, **Research Group:** Ernfors lab

**Link of the datasets:** Mouse Dentate Gyrus, Human glioblastoma, **Dataset references:** Hochgerner and Zeisel, et al., Couturier, et al.

**Resource gene list** cell cycle genes, Homologene-HumanMouse

## 2.2.1 Training

```
[1]: import datetime
today=f"{datetime.datetime.now():%Y-%m-%d}"
today
```

```
[1]: '2023-05-29'
```

```
[2]: import torch
import torch.nn as nn
from torch.autograd import Variable
import torch.utils.data as Data
import torchvision
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import torch.utils.data as data_utils
from matplotlib import cm
import numpy as np
import pandas as pd
import pickle as pickle
from scipy.spatial.distance import cdist, pdist, squareform
import pandas as pd
from sklearn.linear_model import LogisticRegression, LogisticRegressionCV
from sklearn.model_selection import StratifiedShuffleSplit
from collections import defaultdict
from sklearn import preprocessing
import matplotlib.patches as mpatches
import torch.nn.functional as F
import math
#import gpytorch
```

```
import urllib.request
import os.path
from scipy.io import loadmat
from math import floor
import anndata
# Make plots inline
%pylab inline
```

%pylab is deprecated, use %matplotlib inline and import the required libraries.  
Populating the interactive namespace from numpy and matplotlib

```
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/IPython/core/magics/pylab.
↳ py:162: UserWarning: pylab import has clobbered these variables: ['floor']
`%matplotlib` prevents importing * from pylab and numpy
warn("pylab import has clobbered these variables: %s" % clobbered +
```

```
[3]: torch.manual_seed(1)    # reproducible
```

```
[3]: <torch._C.Generator at 0x7fcfe16fbe90>
```

```
[4]: import scCAMEL as scm
from scCAMEL import CamelPrefiltering
```

(continues on next page)

(continued from previous page)

```
from scCAMEL import CamelSwapline
from scCAMEL import CamelEvo
```

```
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelSwapline.py:
↳637: FutureWarning: The default dtype for empty Series will be 'object' instead of
↳'float64' in a future version. Specify a dtype explicitly to silence this warning.
def addcolor(datax, clustername="Cluster", colorcode="color", predef=pd.Series()):
```

```
[5]: cd /mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/
↳AdultBrain

/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/AdultBrain
```

```
[6]: scref=anndata.read("ZeiselDentateGyrus_Ref2023-05-27.h5ad")
scref
```

```
[6]: AnnData object with n_obs × n_vars = 5454 × 14545
      obs: 'Cluster', 'Color'
```

```
[7]: #if the matrix is sparse matrix
      #screfall.X=screfall.X.todense()
```

```
[8]: set(scref.obs["Cluster"])
```

```
[8]: {'Astrocytes',
      'Cajal-Retzius',
      'Cck-Tox',
      'Endo',
      'GABA',
      'Granule',
      'Microglia',
      'Mossy',
      'NFOL',
      'Neuroblast',
      'OLIG',
      'OPC',
      'PVM',
      'Peri/VLMC',
      'nIPC/Rgl'}
```

## 2.2.2 Prefiltering\_and\_SelectFeatures

```
[9]: scref.X=scref.X.todense()
```

```
[10]: dfpdt=pd.DataFrame(scref.X.T, index=scref.var.index, columns=scref.obs.index)
dfpdt.shape
```

```
[10]: (14545, 5454)
```

```
[11]: path="/mnt/f/Dropbox/data/proj/PE_HYZ/PublicDataSet/"
dictfilename1="Homologene_mouse2human_dict2.pickle"
```

(continues on next page)



(continued from previous page)

```
dfpdt= scm.CamelPrefiltering.TransSpeciesGeneName(dfm=dfpdt, dictfilename=dictfilename1,
↳path=path)
samegene=set(dfpdt.index)
len(samegene)
```

[11]: 12515

[12]: dfpdt

```
[12]:      10X46_1_GCCTACACGGGAGT-1  10X46_1_AAGCACTGATGGTC-1  \
A2M                                0.0                        0.0
AAAS                                0.0                        0.0
AACS                                0.0                        0.0
AAED1                               0.0                        0.0
AAGAB                               0.0                        0.0
...                                ...                        ...
ZXDC                                0.0                        0.0
ZYG11B                              0.0                        0.0
ZYG11B                              0.0                        0.0
ZYG11B                              0.0                        0.0
ZZEF1                               0.0                        0.0
ZZZ3                                0.0                        0.0

      10X43_1_GCGCATCTAGCTAC-1  10X46_1_TGGTAGTGAGAACA-1  \
A2M                                0.0                        0.0
AAAS                                0.0                        0.0
AACS                                0.0                        0.0
AAED1                               0.0                        0.0
AAGAB                               0.0                        0.0
...                                ...                        ...
ZXDC                                0.0                        0.0
ZYG11B                              0.0                        0.0
ZYG11B                              0.0                        0.0
ZYG11B                              1.0                        0.0
ZZEF1                               0.0                        0.0
ZZZ3                                0.0                        0.0

      10X46_1_GAGCTCCTGTGCAT-1  10X43_1_TAGGCATGGCCTTC-1  \
A2M                                0.0                        0.0
AAAS                                0.0                        0.0
AACS                                0.0                        0.0
AAED1                               0.0                        0.0
AAGAB                               0.0                        0.0
...                                ...                        ...
ZXDC                                0.0                        0.0
ZYG11B                              0.0                        0.0
ZYG11B                              0.0                        0.0
ZYG11B                              0.0                        0.0
ZZEF1                               0.0                        0.0
ZZZ3                                0.0                        0.0

      10X43_1_TAAGTAACGGACGA-1  10X43_1_GTTACTACTGTCCC-1  \
A2M                                0.0                        0.0
AAAS                                0.0                        0.0
AACS                                1.0                        0.0
AAED1                               0.0                        0.0
```

(continues on next page)

(continued from previous page)

AAGAB	0.0	0.0	
...	...	...	
ZXDC	0.0	0.0	
ZYG11B	0.0	0.0	
ZYX	0.0	0.0	
ZZEF1	1.0	0.0	
ZZZ3	0.0	0.0	
10X46_1_CAGATGACACCCTC-1	10X43_1_GGTTTACTCTTATC-1	...	\
A2M	0.0	0.0	...
AAAS	0.0	0.0	...
AACS	0.0	0.0	...
AAED1	0.0	0.0	...
AAGAB	0.0	0.0	...
...	...	...	...
ZXDC	0.0	0.0	...
ZYG11B	0.0	0.0	...
ZYX	0.0	5.0	...
ZZEF1	0.0	1.0	...
ZZZ3	0.0	0.0	...
10X46_1_TGTAGTCTCGCTAA-1	10X46_1_GATCGTGAGAGCTT-1		\
A2M	0.0	0.0	
AAAS	0.0	0.0	
AACS	0.0	0.0	
AAED1	0.0	0.0	
AAGAB	0.0	0.0	
...	...	...	
ZXDC	0.0	0.0	
ZYG11B	0.0	0.0	
ZYX	0.0	0.0	
ZZEF1	0.0	0.0	
ZZZ3	0.0	0.0	
10X43_1_CAATCGGACCCTAC-1	10X46_1_CGGACTCTCTATTC-1		\
A2M	0.0	0.0	
AAAS	0.0	0.0	
AACS	0.0	0.0	
AAED1	0.0	0.0	
AAGAB	0.0	0.0	
...	...	...	
ZXDC	0.0	0.0	
ZYG11B	0.0	0.0	
ZYX	0.0	0.0	
ZZEF1	0.0	0.0	
ZZZ3	0.0	0.0	
10X46_1_CTAAGGACATCAGC-1	10X43_1_CTAAGTACTGACAC-1		\
A2M	0.0	0.0	
AAAS	0.0	0.0	
AACS	0.0	0.0	
AAED1	0.0	0.0	

(continues on next page)

(continued from previous page)

AAGAB	0.0	0.0
...	...	...
ZXDC	0.0	0.0
ZYG11B	0.0	0.0
ZYX	0.0	0.0
ZZEF1	0.0	1.0
ZZZ3	0.0	0.0
	10X46_1_ATCTGACTAATGCC-1	10X46_1_CCATATACTTGACG-1 \
A2M	0.0	0.0
AAAS	0.0	0.0
AACS	0.0	0.0
AAED1	0.0	0.0
AAGAB	0.0	0.0
...	...	...
ZXDC	0.0	0.0
ZYG11B	0.0	0.0
ZYX	0.0	0.0
ZZEF1	0.0	0.0
ZZZ3	0.0	0.0
	10X43_1_ATGAAGGAATGCCA-1	10X46_1_GGACAGGATAGCGT-1
A2M	0.0	0.0
AAAS	0.0	0.0
AACS	0.0	0.0
AAED1	0.0	0.0
AAGAB	0.0	0.0
...	...	...
ZXDC	0.0	0.0
ZYG11B	0.0	0.0
ZYX	0.0	0.0
ZZEF1	0.0	0.0
ZZZ3	0.0	0.0

[12515 rows x 5454 columns]

```
[13]: scref2= anndata.AnnData(dfpd.T.astype(float))
scref2
```

```
[13]: AnnData object with n_obs x n_vars = 5454 x 12515
```

```
[14]: scref2.obs=scref.obs
```

```
[15]: scref=scref2
```

```
[16]: path='/mnt/f/Dropbox/data/proj/PE_HYZ/PublicDataSet/'
filename='PANTHER_cell_cycle_genes.txt'
scref= scm.CamelPrefiltering.prefilter(datax=scref,filename=filename, path=path)

CamelRunning_Prefilter...
CamelRunning_Prefilter...Finished
```

```
[17]: scref=scm.CamelPrefiltering.DataScaling(scref)
```

```
[18]: dfdev2=pd.DataFrame(scref.X,index=scref.obs.index,columns=scref.var.index).T
dfdev2=dfdev2.loc[scref.var['Filter1']==True]
dfdev2.shape
```

```
[18]: (8351, 5454)
```

```
[19]: markerlist=scm.CamelSwapline.enrichmentscoreBETA(dfpfcclus=scref.obs["Cluster"], df_
↳ dev=dfdev2, fc=3, pvalcutoff=0.1, shortcut=True)

Camel...Running: clusteringValue1...
Camel...Running: clusteringValue2...
1670
[Processing15
Camel...Running: CrossChecking...
-0%--6%--13%--20%--26%--33%--40%--46%--53%--60%--66%--73%--80%--86%--93%-Camel...Running:
↳ output genelist...
```

```
[20]: markerlist2=list(set(markerlist))
len(markerlist2)
```

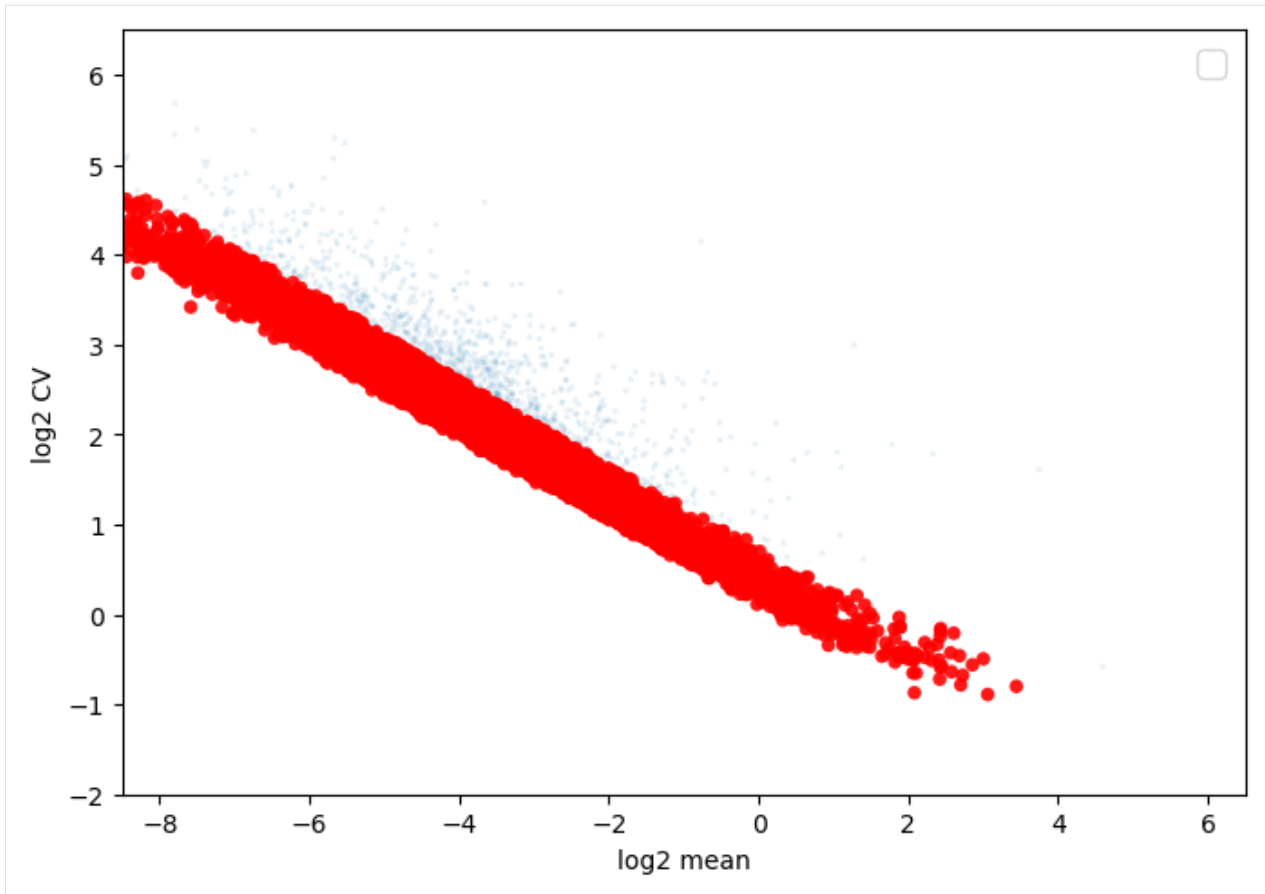
```
[20]: 2182
```

```
[21]: scref, score, thrs, ax=scm.CamelPrefiltering.MVgenes(datax=scref,wt=3,↳
↳ markerlist=markerlist2,
plotfig=True, Xlow=-8.5, Xhigh=6.5,↳
↳ Ylow=-2, Yhigh=6.5,alphaValue=0.9, sValue=30,
fig_args={'figsize': (8, 8), 'facecolor': 'white', 'edgecolor': 'white'})

CamelRunning_MVgenes...

No artists with labels found to put in legend. Note that artists whose label start with↳
↳ an underscore are ignored when legend() is called with no argument.

CamelRunning_MVgenes...Finished
```



```
[22]: len(scref.var.index[scref.var["MVgene"]])
```

```
[22]: 2826
```

```
[23]: scref2=scref
```

```
[24]: #####
#####
#remeber to change the file path in tftable
#####
#####
scref =scm.CamelPrefiltering.MVgene_Scaling(datax=scref2,score=score,
                                           thrs=thrs, TPTT=0,
↳ mprotogruop=scref.obs["Cluster"].values,commongene=None,
↳
↳ sharedMVgenes=None,std_scaling=True,
  tftable="/mnt/f/Dropbox/data/proj/PE_HYZ/PublicDataSet/FantomTF2CLUSTER_human_
↳official.txt", learninggroup="train")
```

```
CamelRunning---GenesScaling...
```

```
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:791: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna
```

(continues on next page)

(continued from previous page)

```

↳will be keyword-only.
    scalepfc = dfpfc.div(dfpfc.std(1), axis=0).dropna(0)
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:792: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna_
↳will be keyword-only.
    scalepfc = dfpfc.astype(float).dropna(0)
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:794: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna_
↳will be keyword-only.
    scalepfc = scalepfc.dropna(0)
/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↳py:795: FutureWarning: Passing a set as an indexer is deprecated and will raise in a_
↳future version. Use a list instead.
    dfpfc_dev = scalepfc.loc[set(scalepfc.index) & set(sharedMVgenes)].dropna()

CamelRunning---TrainingGenesScaling...Finished

```

```

[25]: scref
[25]: AnnData object with n_obs × n_vars = 5454 × 12515
      obs: 'Cluster', 'Color', 'mtrain_index'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'train_set_gene', 'mclasses_names'
      obsm: 'train_set_values'

```

## 2.2.3 Neural-Network learning

```

[26]: net=scm.CamelPrefiltering.NNclassifier(
      datax=scref,
      epochNum=100,
      learningRate=0.05,
      verbose=0,
      optimizerMomentum=0.8,
      dropout=0.3,
      #imizer__nesterov=True,
      )

CamelRunning---NNclasffier_in_cpu...
CamelRunning---NNclasffier_in_cpu...Finished

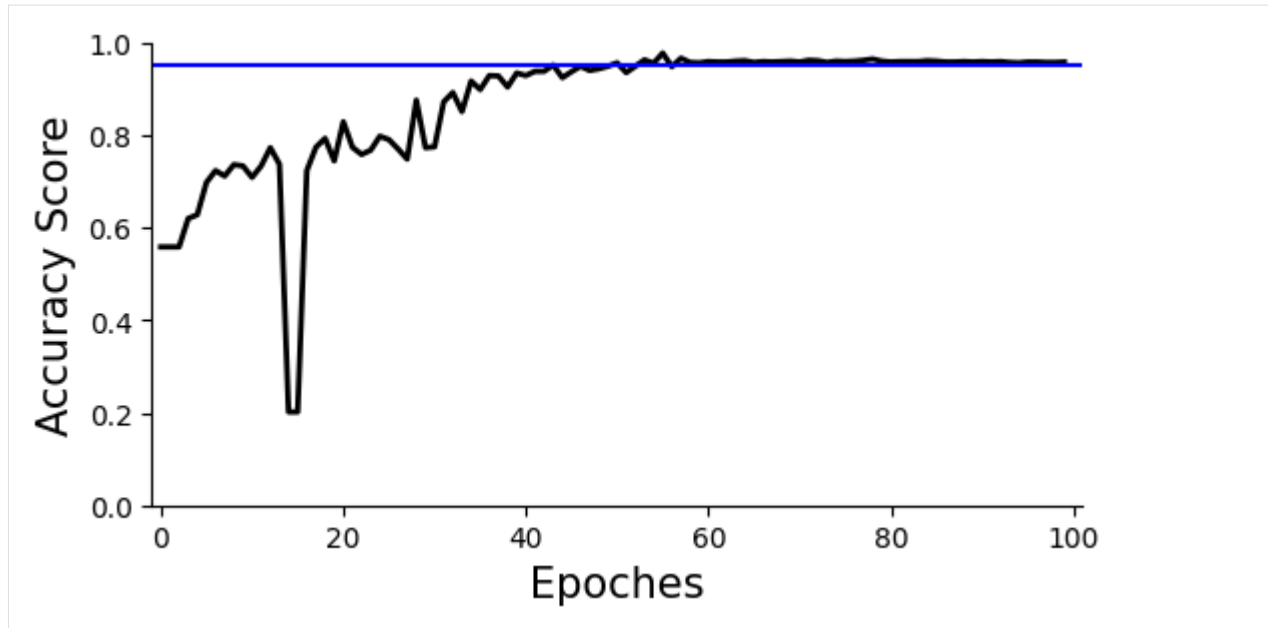
```

## 2.2.4 Accuracy plot, the overall clustering accuracy is ~95%

```

[27]: ax=scm.CamelPrefiltering.AccuracyPlot( nnModel=net, accCutoff=0.95,
      Xlow=-1, Ylow=0.0, Yhigh=1,
      )

```



### 2.2.5 Make prediction and visualization in Radar plot

```
[91]: net=scm.CamelPrefiltering.NNclassifier(
    datax=scref,
    epochNum=60,
    learningRate=0.05,
    verbose=0,
    optimizerMomentum=0.8,
    dropout=0.3,
    #imizer__nesterov=True,
)
```

```
CamelRunning---NNclasffier_in_cpu...
CamelRunning---NNclasffier_in_cpu...Finished
```

```
[92]: scref
```

```
[92]: AnnData object with n_obs × n_vars = 5454 × 12515
      obs: 'Cluster', 'Color', 'mtrain_index'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_Score_RefCellType
      ↪', 'Celltype_OrderNumber', 'refcolor_dict'
      obsm: 'train_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'
```

```
[93]: #if color is not defined: scref.obs['color']
      #scref=scm.CamelSwapline.addcolor(datax=scref,clustername="Cluster", colorcode="color")
```

```
[94]: #scref.uns["mwanted_order"] =list(sort(list(set(scref.obs["Cluster"]))))
      scref.uns["mwanted_order"] =[
        'Mossy', 'Cajal-Retzius', 'Cck-Tox', 'GABA', 'Endo',
        'Peri/VLMC', 'PVM', 'Microglia', 'Astrocytes', 'OLIG',
```

(continues on next page)

(continued from previous page)

```
'NFOL', 'OPC', 'nIPC/Rgl', 'Neuroblast', 'Granule'
]
```

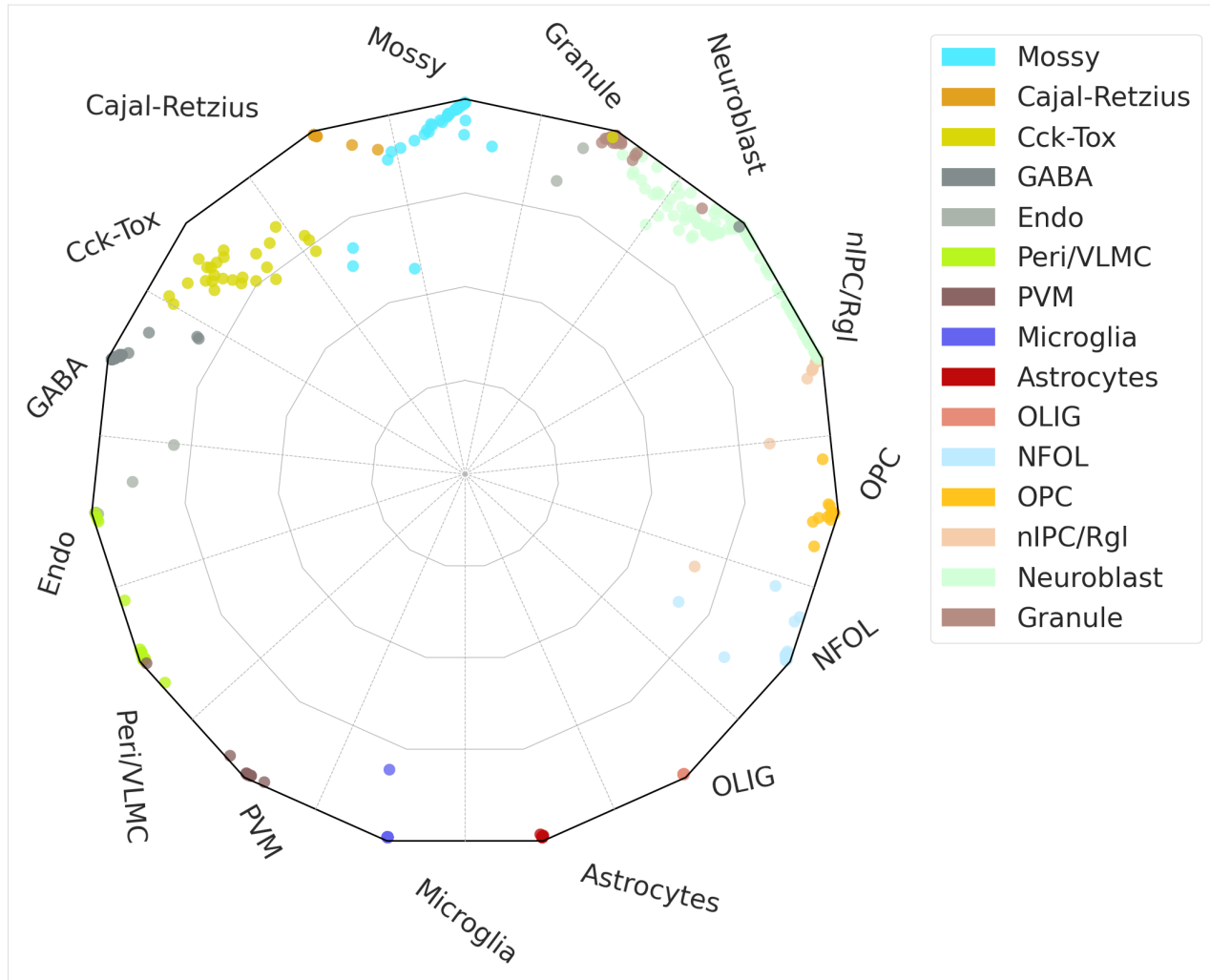
```
[95]: scref.uns["refcolor_dict"]= pd.Series({'Astrocytes': (190, 10, 10), 'Cajal-Retzius': ↵
↵(225, 160, 30), 'Cck-Tox': (217, 215, 7),
↵      'Endo': (170, 180, 170), 'GABA': (130, 140, 140), 'Granule': (180, ↵
↵140, 130),
↵      'Microglia': (100, 100, 240), 'Mossy': ( 80, 235, 255), 'NFOL':(190, 235, ↵
↵255),
↵      'Neuroblast':(210, 255, 215), 'OLIG':(230, 140, 120), 'OPC': (255, 195, ↵
↵28),
↵      'PVM': (139, 101, 100), 'Pericytes': (252, 183, 26), 'Radial Glia-like': ↵
↵(214, 194, 39),
↵      'VLMC': (255, 120, 155), 'nIPC': (250, 145, 45), 'hRgl2a': (250, 125, ↵
↵25),
↵      'hDA0': (190, 200, 190), 'hOPC': (255, 35, 155), 'hRN': (199, 121, ↵
↵41),
↵      'hNbGaba': ( 40, 55, 130), 'hGaba': ( 7, 121, 61), 'hOMTN': ( 95, 186, ↵
↵70),
↵      'hSert': ( 50, 180, 180), 'nIPC/Rgl': (245, 205, 170), 'Peri/VLMC': ↵
↵(185, 245, 30),
↵      'eSCc':(205,205,220)})
```

```
[96]: cd .
```

```
/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/AdultBrain
```

```
[97]: #radar plot
scref=scm.CamelSwapline.prediction(datax=scref, mcolor_dict=scref.uns["refcolor_dict"] ,
↵net=net, learninggroup="train", radarplot=True, fontsizeValue=18,
↵      ncolnm=3, bbValue=(1.2, 1.05) )
#plt.savefig("upload_%s_RadarPlot_cluster.pdf"%today, bbox_inches='tight')
```

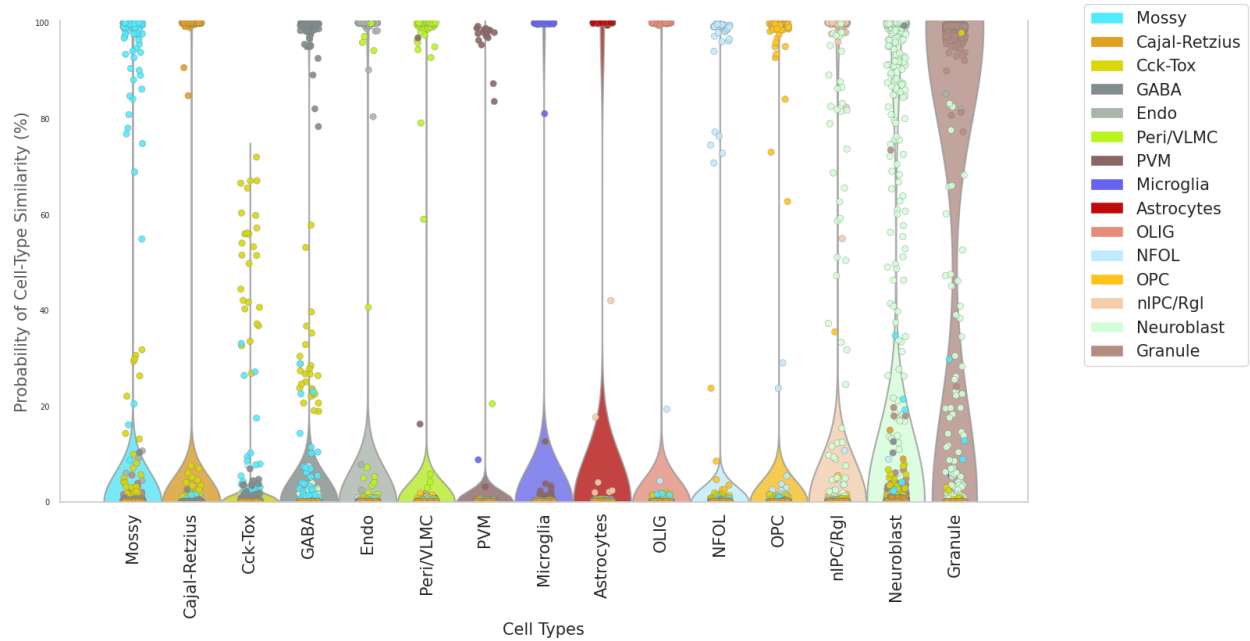




```
[99]: #####
#####
#Multiplot_Violin
# each column along X axis represents a cell type, and the y-axis represents the cell-
# type probability
# each column along X axis contain all cells in the training, shown as dots with
# different colors
# in the perfect condition only the cells of the same cell-type column will reach to the
# maxium (100%) probability, all other cells will be 0 probability
# however, in each cell-type column, some cells from other cell types may exhibit high
# probability, two reasons: 1. they may shared some features with this---
# cell-type, and these features drag them to the current cell-type; 2. they are
# improperly classified due to many factors, such as low quality data, different
# clustering algorithm....
#####
#####
```

```
[100]: fig=scm.CamelSwapline.ProbMultiPlot( datax=scref, mcolor_dict=scref.uns["refcolor_dict"]
# )
```

&lt;Figure size 2500x1000 with 0 Axes&gt;

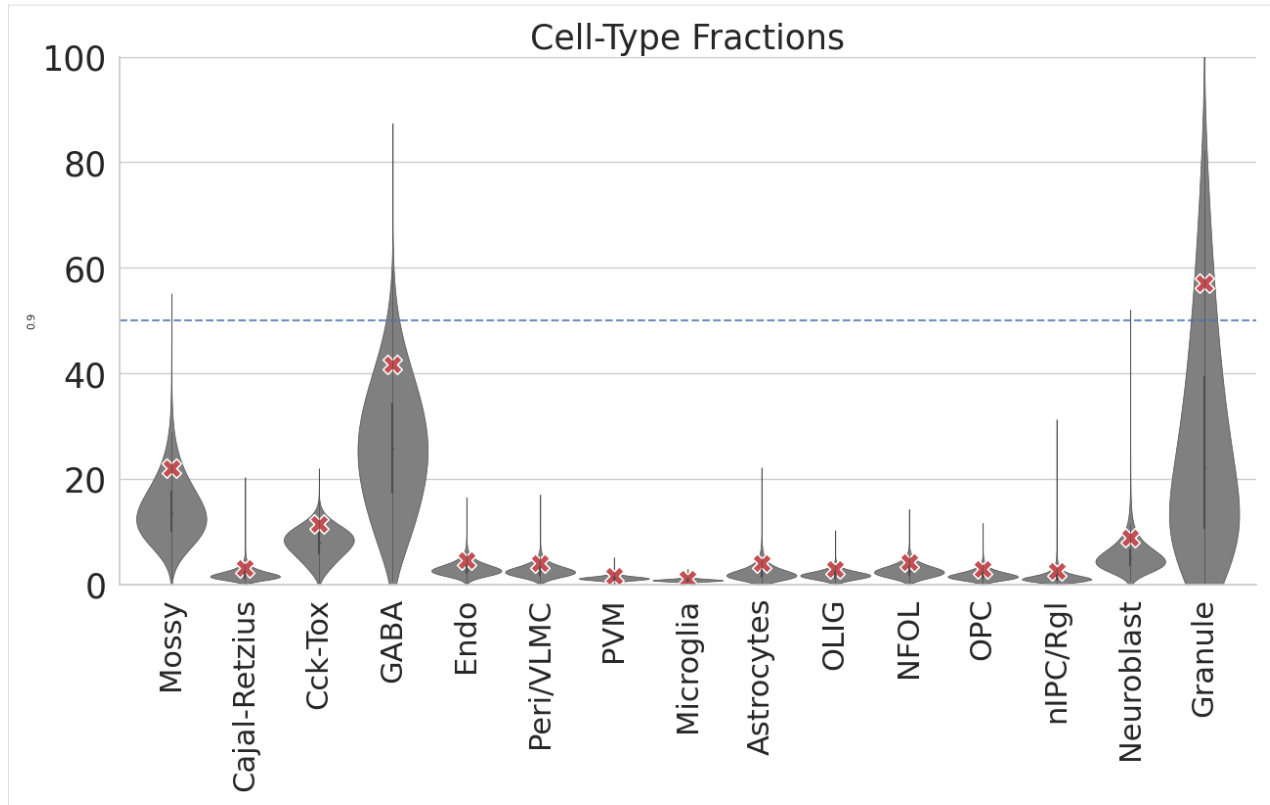


## 2.2.6 permutation control

```
[101]: ## the whole data matrix is randomized, the red X represents 95% confidence of each cell-
        ↪ type
```

```
[102]: dfest0, ratiodef=scm.CamelSwapline.permutationTest(datax=scref,net=net,num=50,
        ↪ plotshow=True)
```

&lt;Figure size 640x480 with 0 Axes&gt;

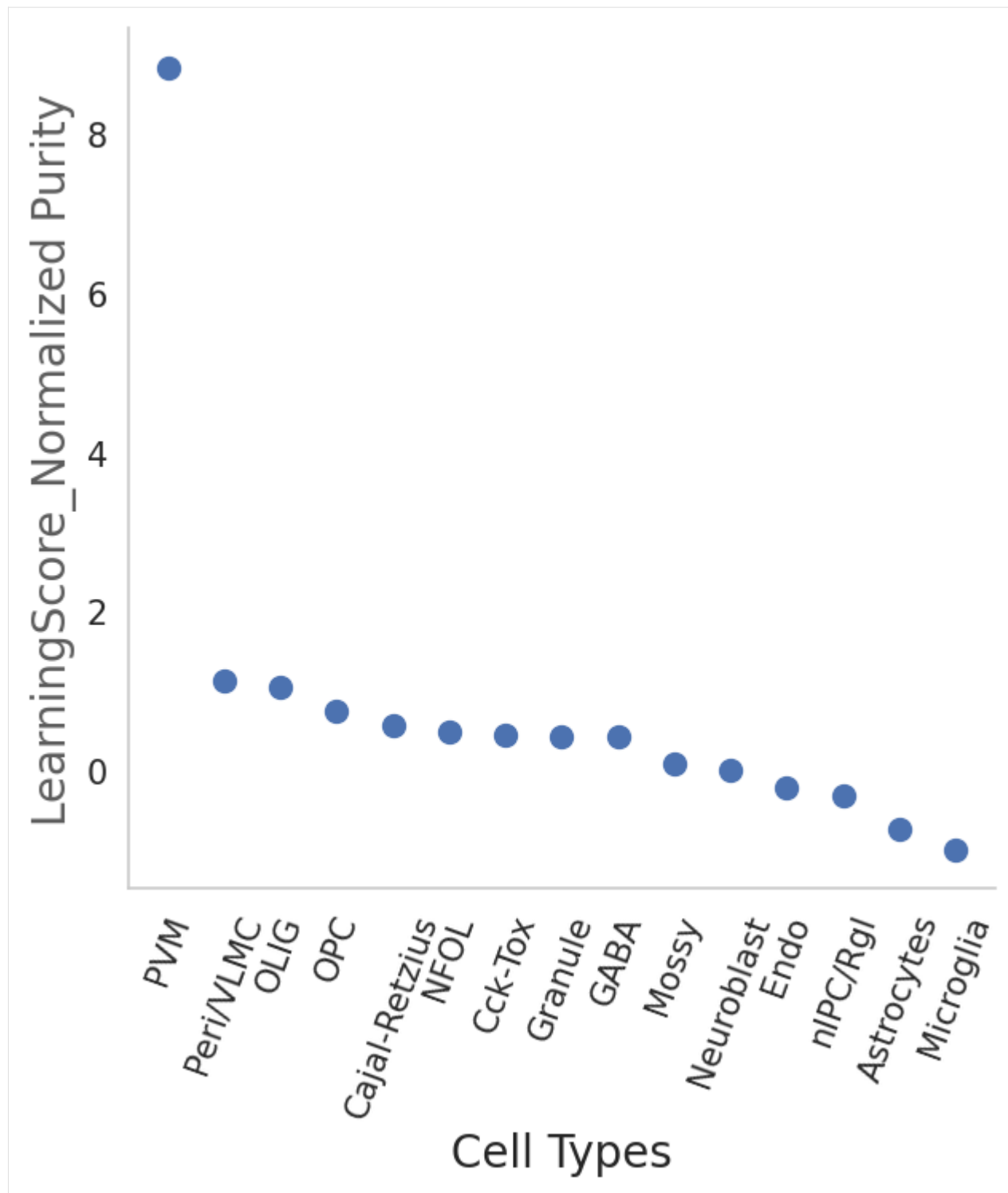


## 2.2.7 Cell\_Type Purity

```
[103]: #The ratio of the purity entropy for each cluster based on their learning scores, is_
        ↳ used as a measure of purity.
        #The function returns a pandas dataframe sorted by the purity score
```

```
[104]: dfpurity1=scm.CamelSwapline.PurityEstimationLearningScore(datax=scref, clusterlist=
        ↳ "Cluster",    elbow=False, figureplot=True)
```

<Figure size 640x480 with 0 Axes>



## 2.2.8 association between cell-types

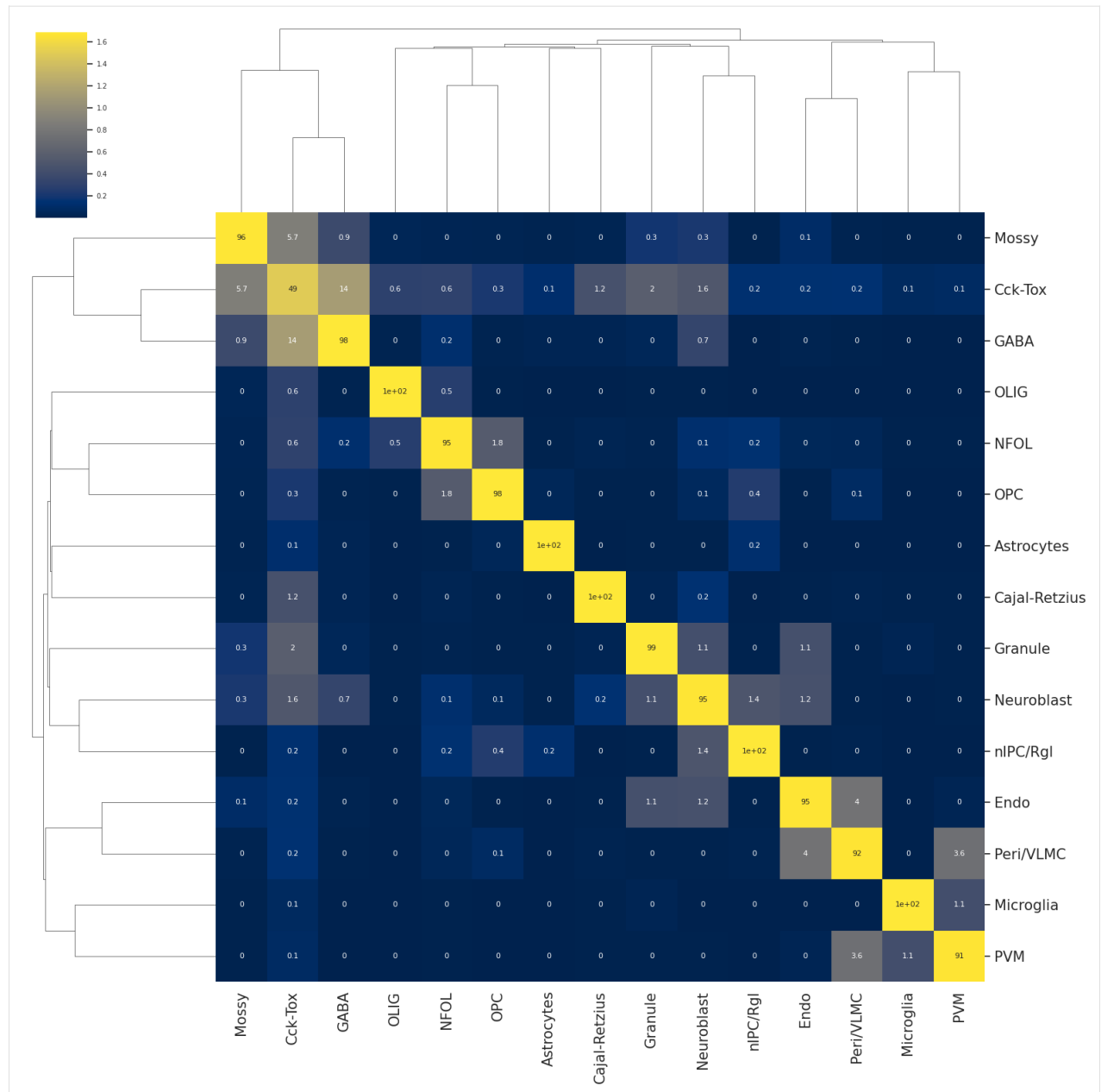
[105]: scref

[105]: AnnData object with n\_obs × n\_vars = 5454 × 12515  
 obs: 'Cluster', 'Color', 'mtrain\_index'  
 var: 'Filter1', 'MVgene', 'RefGeneList'  
 uns: 'train\_set\_gene', 'mclasses\_names', 'mwanted\_order', 'Celltype\_Score\_RefCellType'  
 ↪, 'Celltype\_OrderNumber', 'refcolor\_dict'  
 obsm: 'train\_set\_values', 'Celltype\_Score', 'CelltypeScoreCoordinates'

[106]: *# the heatmap of hierarchical clustering represents the cell-type similarity or ↪ association*  
*#color from dark purple to light yellow represents the association from low to high*  
*#number inside of each square indicating the association value.*

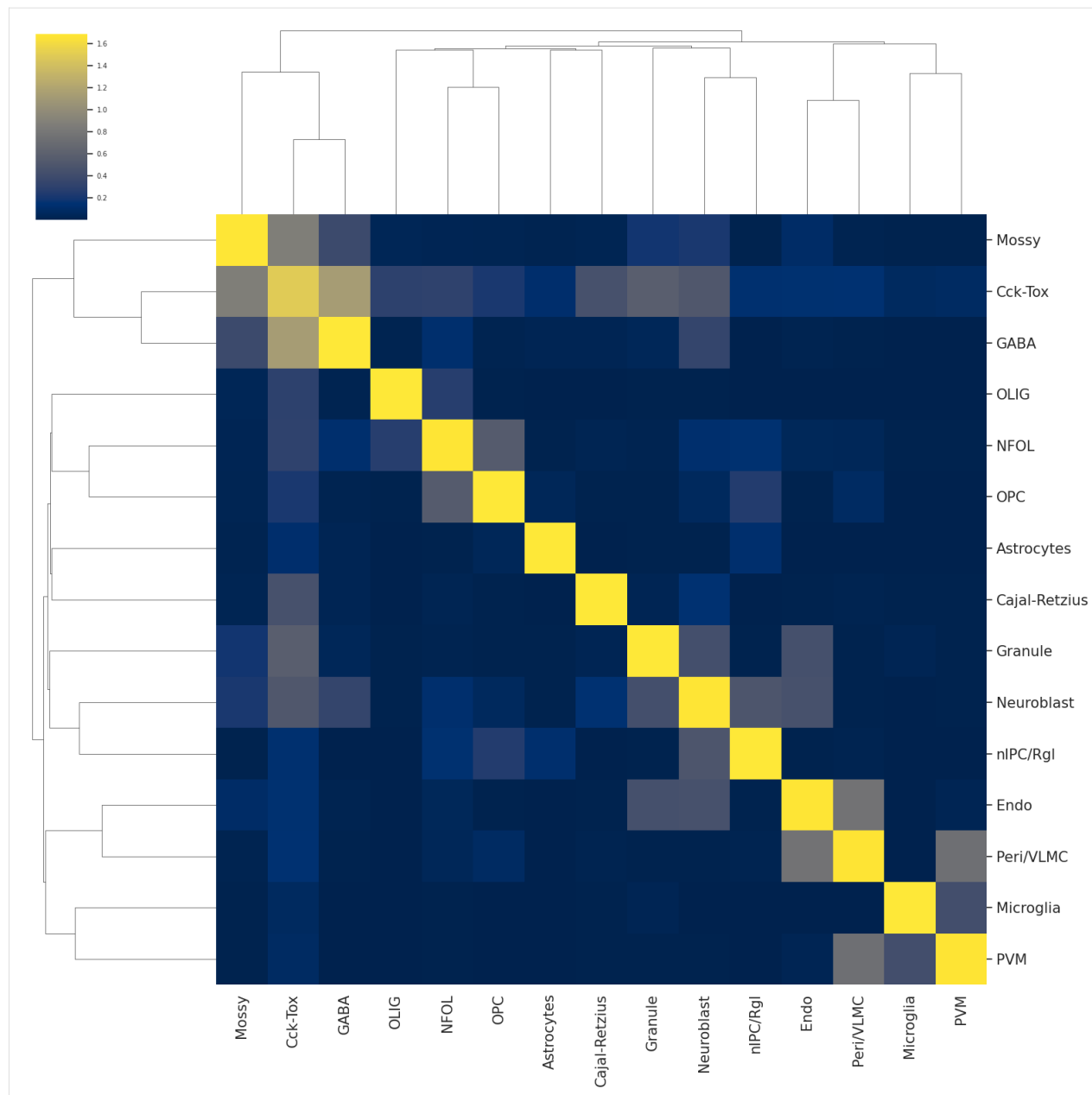
[107]: scm.CamelSwapline.CellTypeSimilarity(datax=scref, labelnum=True, metricvalue=↪'correlation',methodvalue="complete")

<Figure size 1500x1500 with 0 Axes>



```
[108]: scm.CamelSwapline.CellTypeSimilarity(datax=scref, labelnum=False, metricvalue=
        ↪ 'correlation', methodvalue="complete")
```

<Figure size 1500x1500 with 0 Axes>



[ ]:

## 2.2.9 Save data

```
[109]: scref
[109]: AnnData object with n_obs × n_vars = 5454 × 12515
      obs: 'Cluster', 'Color', 'mtrain_index'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'train_set_gene', 'mclasses_names', 'mwanted_order', 'Celltype_Score_RefCellType'
      ↪ ', 'Celltype_OrderNumber', 'refcolor_dict'
      obsm: 'train_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'

[110]: cd .

/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/AdultBrain

[111]: work_dir="/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/
      ↪ AdultBrain"
      QueryName="ZeiselMouseDG"
      TrainingName="ZeiselMouseDG"
      filename="%s_%s_Ref%s_MergeCluster.h5ad"%(QueryName,TrainingName,today)

[112]: os.path.join(work_dir,filename)
[112]: '/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/AdultBrain/
      ↪ ZeiselMouseDG_ZeiselMouseDG_Ref2023-05-29_MergeCluster.h5ad'

[113]: del scref.uns['refcolor_dict']

[114]: CamelSwapline.writedata(adatax=scref,filename=filename,filepath=work_dir)

[115]: scref.X=scref.X.todense()

[ ]:

[116]: scref.uns["refcolor_dict"]= pd.Series({'Astrocytes': (190, 10, 10),'Cajal-Retzius':
      ↪ (225, 160, 30),'Cck-Tox': (217, 215, 7),
      'Endo': (170, 180, 170), 'GABA': (130, 140, 140),'Granule': (180,
      ↪ 140, 130),
      'Microglia': (100, 100, 240),'Mossy': ( 80, 235, 255),'NFOL':(190, 235,
      ↪ 255),
      'Neuroblast':(210, 255, 215),'OLIG':(230, 140, 120), 'OPC': (255, 195,
      ↪ 28),
      'PVM': (139, 101, 100),'Pericytes': (252, 183, 26),'Radial Glia-like':
      ↪ (214, 194, 39),
      'VLMC': (255, 120, 155),'nIPC': (250, 145, 45),'hRgl2a': (250, 125,
      ↪ 25),
      'hDA0': (190, 200, 190),'hOPC': (255, 35, 155),'hRN': (199, 121,
      ↪ 41),
      'hNbGaba': ( 40, 55, 130),'hGaba': ( 7, 121, 61),'hOMTN': ( 95, 186,
      ↪ 70),
      'hSert': ( 50, 180, 180),'nIPC/Rgl': (245, 205, 170), 'Peri/VLMC':
      ↪ (185, 245, 30),
      'eSCc':(205,205,220)})
```



## 2.2.10 Prediction

### 2.2.11 Couturier2020\_humanGlioblastoma

```
[118]: scpdt=anndata.read("Couturier2020_DevGBM_Ref2023-05-27.h5ad")

[119]: set(scpdt.obs["Cluster"])

[119]: {'Astro', 'Mesenchymal', 'Neuronal', 'Oligo', 'Progenitor', 'Unassigned'}

[120]: scpdt

[120]: AnnData object with n_obs × n_vars = 18475 × 33660
      obs: 'Patient', 'Cluster', 'Color'

[121]: scpdt.var.index

[121]: Index(['A1BG', 'A1BG-AS1', 'A1CF', 'A2M', 'A2M-AS1', 'A2ML1', 'A2ML1-AS1',
      'A2ML1-AS2', 'A3GALT2', 'A4GALT',
      ...,
      'ZXDC', 'ZYG11A', 'ZYG11B', 'ZYG', 'ZZEF1', 'ZZZ3', 'bP-21264C1.2',
      'bP-2171C21.3', 'bP-218909.3', 'hsa-mir-1253'],
      dtype='object', length=33660)

[122]: scpdt.X=scpdt.X.todense()

[123]: scpdt2=scpdt.copy()
      scpdt2=scm.CamelPrefiltering.DataScaling(scpdt2)

[126]: #####
      #####
      #remeber to change the file path in tftable
      #####
      #####
      scpdt =scm.CamelPrefiltering.MVgene_Scaling(datax=scpdt2,TPTT=0,   commongene=scref.var.
      ↪index.tolist(),
      ↪sharedMVgenes=scref.uns[ 'train_set_gene'].
      ↪tolist(),
      ↪std_scaling=True,score=None, thr=None, mprotogruop=None,
      tftable="/mnt/f/Dropbox/data/proj/PE_HYZ/PublicDataSet/FantomTF2CLUSTER_human_
      ↪official.txt", learninggroup="test")

CamelRunning---GenesScaling...

/home/huyiz/anaconda3/envs/newpy38/lib/python3.8/site-packages/scCAMEL/CamelPrefiltering.
↪py:820: FutureWarning: In a future version of pandas all arguments of DataFrame.dropna
↪will be keyword-only.
      scalegbm = dfpfc.div(dfpfc.std(1), axis=0).dropna(0)

CamelRunning---TestGenesScaling...Finished
```

```
[127]: scpdt
[127]: AnnData object with n_obs × n_vars = 18475 × 33660
      obs: 'Patient', 'Cluster', 'Color', 'mtrain_index'
      var: 'RefGeneList'
      uns: 'train_set_gene', 'mclasses_names'
      obsm: 'test_set_values'

[128]: #del scpdt.obs["color"]

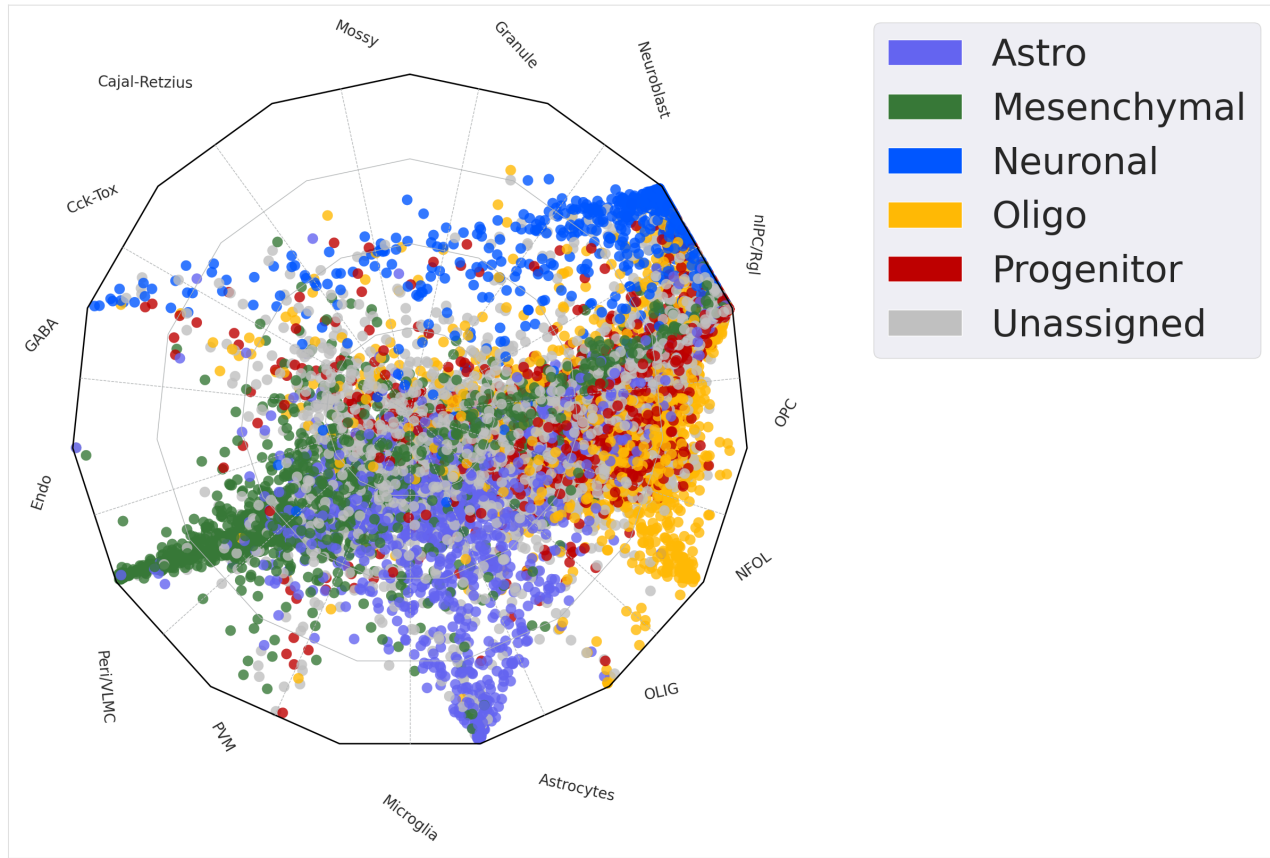
[129]: # if color is not definedi
      #scpdt=scm.CamelSwapline.addcolor(datax=scpdt,clustername="Cluster", colorcode="color")

[130]: scpdt.uns["mwanted_order"] =list(sort(list(set(scpdt.obs["Cluster"]))))

[131]: scpdt.uns["refcolor_dict"] = pd.Series({'Astro': (100, 100, 240), 'Neuronal': ( 0, 86,
      ↪ 255),
      'Mesenchymal': (55, 120, 55), 'Oligo': ( 255,185, 5), 'Unassigned': (192,
      ↪192,192),
      'Progenitor': (190, 0, 0)})

[133]: scpdt
[133]: AnnData object with n_obs × n_vars = 18475 × 33660
      obs: 'Patient', 'Cluster', 'Color', 'mtrain_index'
      var: 'RefGeneList'
      uns: 'train_set_gene', 'mclasses_names', 'mwanted_order', 'refcolor_dict'
      obsm: 'test_set_values'

[134]: test=scm.CamelSwapline.prediction(datax=scpdt, mcolor_dict=scpdt.uns["refcolor_dict"],
      ↪net=net,learninggroup="test", radarplot=True, fontsizeValue=35,
      datarefplot=scref, ncolnm=1, bbValue=(1.1, 1.05))
```

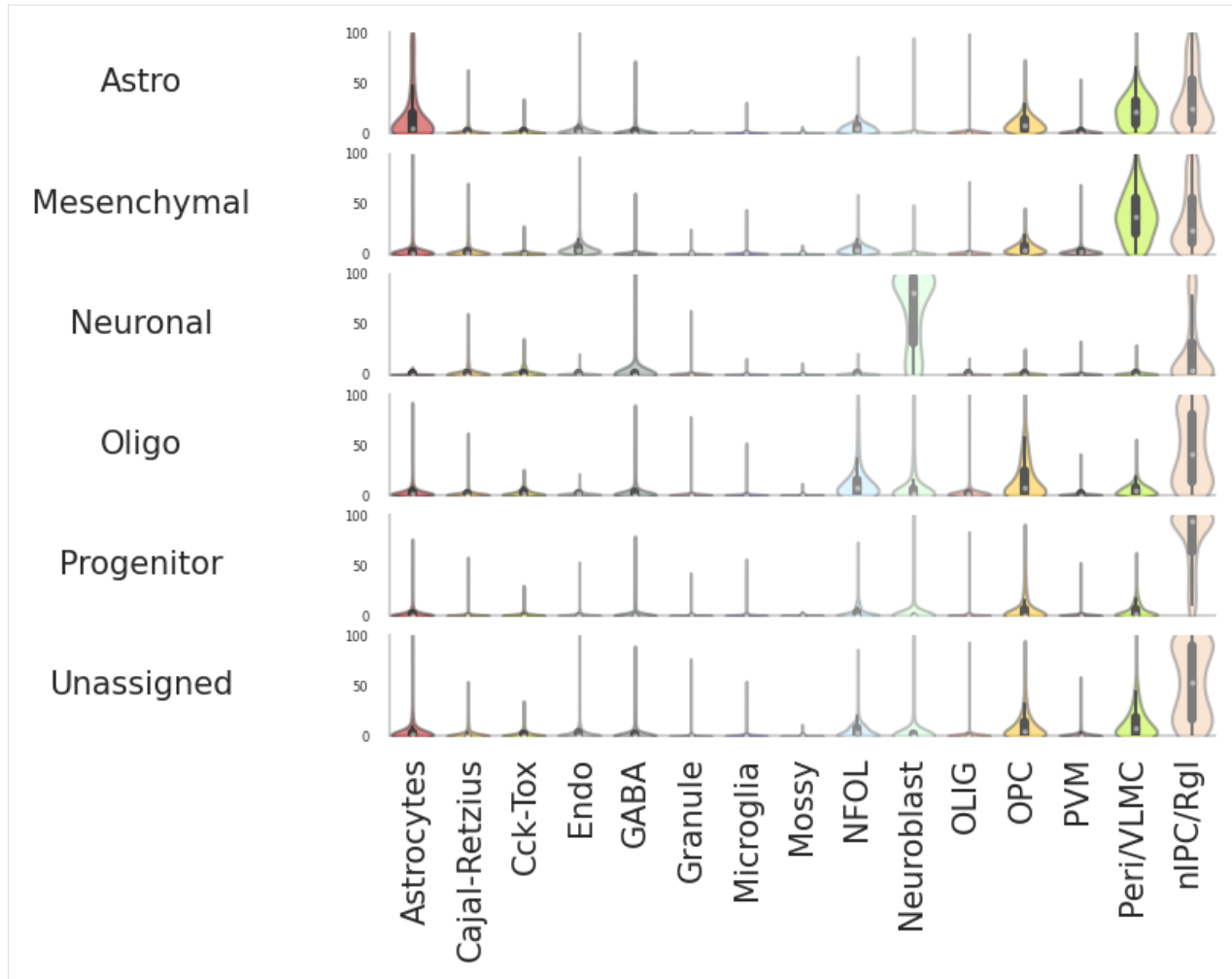


```
[139]: scpdt
```

```
[139]: AnnData object with n_obs × n_vars = 18475 × 33660
      obs: 'Patient', 'Cluster', 'Color', 'mtrain_index'
      var: 'RefGeneList'
      uns: 'train_set_gene', 'mclasses_names', 'mwanted_order', 'refcolor_dict', 'Celltype_
      ↪ Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'test_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'
```

```
[140]: genename=sort(list(set(scpdt.obs["Cluster"])))
      name=sort(list(set(scref.obs["Cluster"])))
```

```
[141]: dfprob=pd.DataFrame(scpdt.obsm['Celltype_Score'])
      dfprob.columns=scpdt.uns['Celltype_Score_RefCellType']
      dfprob.index=scpdt.obs.index
      dfmk=dfprob.astype(float).join(scpdt.obs["Cluster"],how="inner").T
      dfprob=scm.CamelSwapline.CellTypeSimilarityViolinPlot(datax=scpdt, dataref=scref)
```



## 2.2.12 Save data

```
[142]: cd .

/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/AdultBrain

[143]: scpdt
[143]: AnnData object with n_obs × n_vars = 18475 × 33660
      obs: 'Patient', 'Cluster', 'Color', 'mtrain_index'
      var: 'RefGeneList'
      uns: 'train_set_gene', 'mclasses_names', 'mwanted_order', 'refcolor_dict', 'Celltype_
      ↪ Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'test_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'

[144]: work_dir="/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/
      ↪ AdultBrain"
      QueryName="Couturier2020"
      TrainingName="ZeiselMouseDG"
      filename="%s_%s_Ref%s_MergeCluster.h5ad"%(QueryName,TrainingName,today)
```

```
[145]: os.path.join(work_dir, filename)
[145]: '/mnt/f/Yizhou_KI_OneCloud/OneDrive - Karolinska Institutet/Datasets_Template/AdultBrain/
↳ Couturier2020_ZeiselMouseDG_Ref2023-05-29_MergeCluster.h5ad'
[147]: del scpdt.uns["refcolor_dict"]
[149]: CamelSwapline.writedata(adatax=scpdt, filename=filename, filepath=work_dir)
[ ]:
```

## 2.3 Tutorials\_scCAMEL-SWAPLINEv1\_LiverMacrophage

**Original Article:** Human resident liver myeloid cells protect against metabolic stress in obesity,"Nature Metabolism.", 2023

**Package:** scCAMEL-SWAPLINE.v1

```
[11]: import datetime
today=f"{datetime.datetime.now():%Y-%m-%d}"
today
[11]: '2023-07-07'

[2]: import torch
import torch.nn as nn
from torch.autograd import Variable
import torch.utils.data as Data
import torchvision
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import torch.utils.data as data_utils
from matplotlib import cm
import numpy as np
import pandas as pd
import pickle as pickle
from scipy.spatial.distance import cdist, pdist, squareform
import pandas as pd
from sklearn.linear_model import LogisticRegression, LogisticRegressionCV
from sklearn.model_selection import StratifiedShuffleSplit
from collections import defaultdict
from sklearn import preprocessing
import matplotlib.patches as mpatches
import torch.nn.functional as F
import math
#import gpytorch

import urllib.request
import os.path
from scipy.io import loadmat
from math import floor
```

(continues on next page)

(continued from previous page)

```
import anndata
# Make plots inline
%pylab inline
```

%pylab is deprecated, use %matplotlib inline and import the required libraries.  
Populating the interactive namespace from numpy and matplotlib

```
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\IPython\
↳core\magics\pylab.py:162: UserWarning: pylab import has clobbered these variables: [
↳'floor']
`%matplotlib` prevents importing * from pylab and numpy
warn("pylab import has clobbered these variables: %s" % clobbered +
```

```
[3]: torch.manual_seed(1)    # reproducible
```

```
[3]: <torch._C.Generator at 0x24082a98eb0>
```

```
[4]: import scCAMEL as scm
from scCAMEL import CamelPrefiltering
from scCAMEL import CamelSwapline
from scCAMEL import CamelEvo
```

```
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scCAMEL\
↳CamelSwapline.py:534: FutureWarning: The default dtype for empty Series will be 'object
↳' instead of 'float64' in a future version. Specify a dtype explicitly to silence this.
↳warning.
def addcolor(datax, clustername="Cluster", colorcode="color", predef=pd.Series()):
```

```
[26]: screfall=anndata.read("LiverMacrophage_474cells_Ref2023-01-16_MergeCluster_35epch.h5ad")
screfall
```

```
[26]: AnnData object with n_obs × n_vars = 474 × 21397
      obs: 'cellID', 'Cluster'
```

```
[28]: set(screfall.obs["Cluster"])
```

```
[28]: {'LM1', 'LM2-C1', 'LM2-C2', 'LM3', 'LM4'}
```

```
[29]: scref=screfall
```

```
[30]: set(scref.obs["Cluster"])
```

```
[30]: {'LM1', 'LM2-C1', 'LM2-C2', 'LM3', 'LM4'}
```

```
[31]: scref.obs.groupby(["Cluster"]).count()
```

```
[31]:
```

	cellID
Cluster	
LM1	133
LM2-C1	39
LM2-C2	73
LM3	96
LM4	133

```
[35]: scref.X=scref.X.todense()
```

```
[36]: path='/Dropbox/data/proj/PE_HYZ/PublicDataSet/'
      filename='PANTHER_cell_cycle_genes.txt'
      #dfpfc2= prefilteringTest.prefilter(df_f=dfpfc,filename=filename, path=path)
      #scref= scm.CamelPrefiltering.prefilter(datax=scref,filename=filename, path=path)
```

```
[37]: scref=scm.CamelPrefiltering.DataScaling(scref)
```

```
[38]: scref.var['Filter1']=[True]*scref.var.shape[0]
      scref
```

```
[38]: AnnData object with n_obs × n_vars = 474 × 21397
      obs: 'cellID', 'Cluster'
      var: 'Filter1'
```

```
[43]: scref=CamelPrefiltering.SelectFeatures(datax=scref, clustername='Cluster',methodname=
      ↳ 'wilcoxon', numbergenes=1000, folderchange=1.5)
```

```
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:417: RuntimeWarning: overflow encountered in expm1
      foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:418: RuntimeWarning: overflow encountered in expm1
      self.expm1_func(mean_rest) + 1e-9
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:417: RuntimeWarning: invalid value encountered in true_
↳ divide
      foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:420: RuntimeWarning: divide by zero encountered in log2
      self.stats[group_name, 'logfoldchanges'] = np.log2(
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:417: RuntimeWarning: overflow encountered in expm1
      foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:418: RuntimeWarning: overflow encountered in expm1
      self.expm1_func(mean_rest) + 1e-9
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:417: RuntimeWarning: invalid value encountered in true_
↳ divide
      foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:420: RuntimeWarning: divide by zero encountered in log2
      self.stats[group_name, 'logfoldchanges'] = np.log2(
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:417: RuntimeWarning: overflow encountered in expm1
      foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:418: RuntimeWarning: overflow encountered in expm1
      self.expm1_func(mean_rest) + 1e-9
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
↳ tools\_rank_genes_groups.py:417: RuntimeWarning: invalid value encountered in true_
```

(continues on next page)

(continued from previous page)

```

→divide
    foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:420: RuntimeWarning: divide by zero encountered in log2
    self.stats[group_name, 'logfoldchanges'] = np.log2(
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:417: RuntimeWarning: overflow encountered in expm1
    foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:418: RuntimeWarning: overflow encountered in expm1
    self.expm1_func(mean_rest) + 1e-9
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:417: RuntimeWarning: invalid value encountered in true_
→divide
    foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:420: RuntimeWarning: divide by zero encountered in log2
    self.stats[group_name, 'logfoldchanges'] = np.log2(
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:417: RuntimeWarning: overflow encountered in expm1
    foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:418: RuntimeWarning: overflow encountered in expm1
    self.expm1_func(mean_rest) + 1e-9
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:417: RuntimeWarning: invalid value encountered in true_
→divide
    foldchanges = (self.expm1_func(mean_group) + 1e-9) / (
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scanpy\
→tools\_rank_genes_groups.py:420: RuntimeWarning: divide by zero encountered in log2
    self.stats[group_name, 'logfoldchanges'] = np.log2(

```

```

[45]: #####
#####
#remeber to change the file path in tftable
#####
#####
scref =scm.CamelPrefiltering.LabelGene_Scaling(datax=scref,
                                                    TPTT=100000,
→mprotogruop=scref.obs["Cluster"].values,commongene=None,
→
→    sharedMVgenes=None,std_scaling=True,
    tftable="(file path to tftable)/FantomTF2CLUSTER_human_official.txt", learninggroup=
→"train")

```

CamelRunning---GenesScaling...

```

C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scCAMEL\
→CamelPrefiltering.py:576: FutureWarning: In a future version of pandas all arguments_
→of DataFrame.dropna will be keyword-only.
    scalepfc = dfpfc.div(dfpfc.std(1), axis=0).dropna(0)

```

(continues on next page)



(continued from previous page)

```

C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scCAMEL\
→ CamelPrefiltering.py:577: FutureWarning: In a future version of pandas all arguments
→ of DataFrame.dropna will be keyword-only.
    scalepfc = dfpfc.astype(float).dropna(0)
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scCAMEL\
→ CamelPrefiltering.py:579: FutureWarning: In a future version of pandas all arguments
→ of DataFrame.dropna will be keyword-only.
    scalepfc = scalepfc.dropna(0)
C:\Users\huyiz\AppData\Local\Continuum\anaconda3\envs\py39\lib\site-packages\scCAMEL\
→ CamelPrefiltering.py:580: FutureWarning: Passing a set as an indexer is deprecated and
→ will raise in a future version. Use a list instead.
    dfpfc_dev = scalepfc.loc[set(scalepfc.index) & set(sharedMVgenes)].dropna()

CamelRunning---TrainingGenesScaling...Finished

```

```
[46]: scref
```

```

[46]: AnnData object with n_obs × n_vars = 474 × 21397
      obs: 'cellID', 'Cluster', 'mtrain_index'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'rank_genes_groups', 'train_set_gene', 'mclasses_names'
      obsm: 'train_set_values'

```

```
[47]: len(scref.var.index[scref.var["MVgene"]])
```

```
[47]: 3987
```

```

[48]: net=scm.CamelPrefiltering.NNclassifier(
      datax=scref,
      epochNum=200,
      learningRate=0.03,
      verbose=0,
      optimizerMomentum=0.8,
      dropout=0.3,
      #imizer__nesterov=True,
      )

```

```

CamelRunning---NNclasffier_in_cuda...
CamelRunning---NNclasffier_in_cuda...Finished

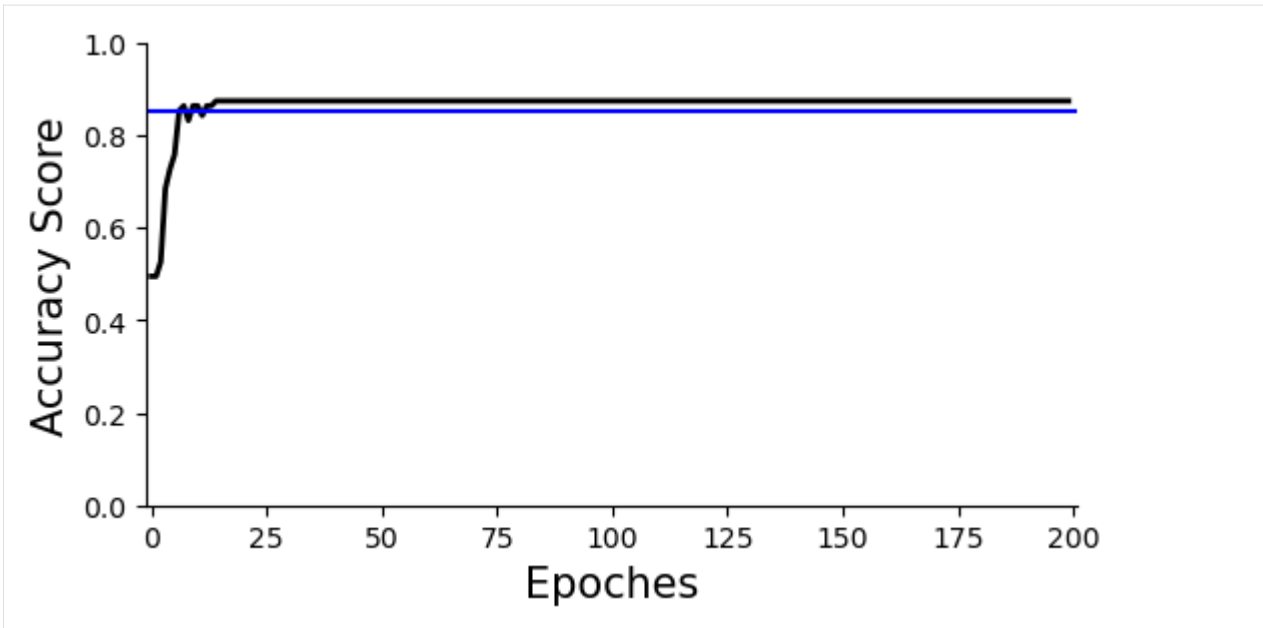
```

## 2.4 Accuracy plot, the overall clustering accuracy is ~85%

```

[52]: ax=scm.CamelPrefiltering.AccuracyPlot( nnModel=net, accCutoff=0.85,
      Xlow=-1, Ylow=0.0, Yhigh=1,
      )
plt.savefig("upload_%s_CurvePlot_learningAccuracy.pdf"%today, bbox_inches='tight')

```



```
[53]: net=scm.CamelPrefiltering.NNclassifier(
      datax=scref,
      epochNum=35,
      learningRate=0.03,
      verbose=0,
      optimizerMomentum=0.8,
      dropout=0.3,
      #imizer__nesterov=True,
      )
```

```
CamelRunning---NNclasffier_in_cuda...
CamelRunning---NNclasffier_in_cuda...Finished
```

```
[54]: scref
```

```
[54]: AnnData object with n_obs × n_vars = 474 × 21397
      obs: 'cellID', 'Cluster', 'mtrain_index'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'rank_genes_groups', 'train_set_gene', 'mclasses_names'
      obsm: 'train_set_values'
```

```
[56]: #if color is not defined
      scref=scm.CamelSwapline.addcolor(datax=scref,clustername="Cluster", colorcode="color")
```

```
[57]: scref
```

```
[57]: AnnData object with n_obs × n_vars = 474 × 21397
      obs: 'cellID', 'Cluster', 'mtrain_index', 'color'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'rank_genes_groups', 'train_set_gene', 'mclasses_names', 'refcolor_dict'
      obsm: 'train_set_values'
```

```
[58]: set(scref.obs["Cluster"])
```

```
[58]: {'LM1', 'LM2-C1', 'LM2-C2', 'LM3', 'LM4'}
```

```
[ ]: LM1: #ABD9E9
      LM2-C1 (use name LM2): #2C7BB6
      LM2-C2 (use name CDC2): purple
      LM3: #FDAE61
      LM4: #D7191C
```

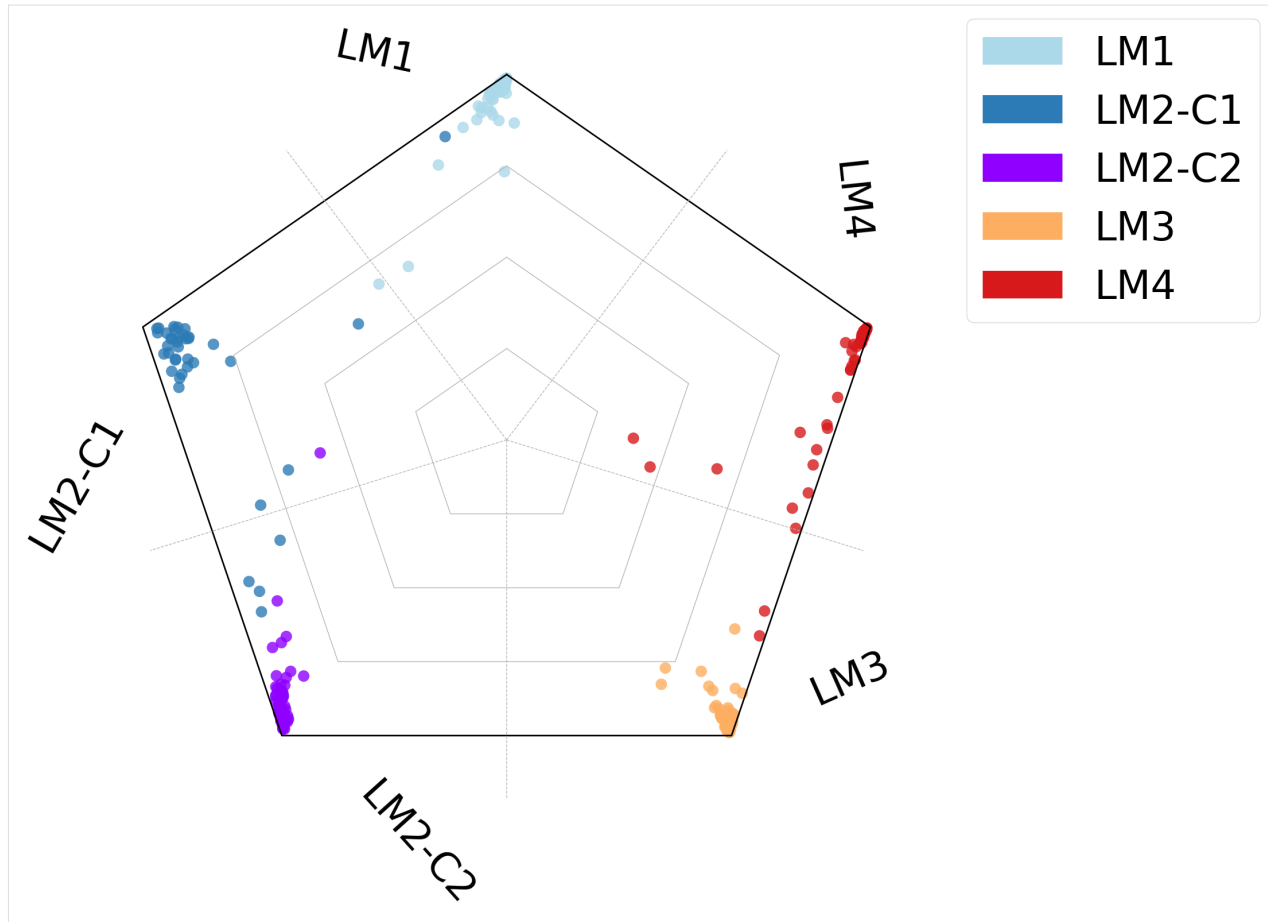
```
[59]: clist=[]
      for item in scref.obs["Cluster"]:
          if item=="LM1":
              clist.append("#ABD9E9")
          elif item=="LM2-C2":
              clist.append("#2C7BB6")
          elif item=="LM2-C1":
              clist.append("purple")
          elif item=="LM3":
              clist.append("#FDAE61")
          elif item=="LM4":
              clist.append("#D7191C")
```

```
[60]: scref.obs["color"]=clist
```

```
[61]: scref.uns['refcolor_dict']={'LM2-C1': [44, 123, 182],
      'LM2-C2': [143, 0, 255],
      "LM3": [253,174,97],
      "LM4": [215,25,28],
      'LM1': [171, 217, 233],
      }
```

```
[62]: scref.uns["mwanted_order"] =list(sort(list(set(scref.obs["Cluster"]))))
```

```
[64]: scref=scm.CamelSwapline.prediction(datax=scref, mcolor_dict=scref.uns["refcolor_dict"] ,
      ↪net=net,learninggroup="train", radarplot=True,fontSizeValue=18,
      ncolnm=3, bbValue=(1.2, 1.05) )
      plt.savefig("upload_%s_RadarPlot_Merged_cluster.pdf"%today,bbox_inches='tight')
```



```
[65]: scref
```

```
[65]: AnnData object with n_obs × n_vars = 474 × 21397
      obs: 'cellID', 'Cluster', 'mtrain_index', 'color'
      var: 'Filter1', 'MVgene', 'RefGeneList'
      uns: 'rank_genes_groups', 'train_set_gene', 'mclasses_names', 'refcolor_dict',
      ↪ 'mwanted_order', 'Celltype_Score_RefCellType', 'Celltype_OrderNumber'
      obsm: 'train_set_values', 'Celltype_Score', 'CelltypeScoreCoordinates'
```

```
[70]: #work_dir=""
      #filename="%s_Ref%s_MergeCluster_35epch.h5ad"%("LiverMacrophageNNlearned", today)
```

```
[71]: os.path.join(work_dir, filename)
```

```
[71]: '/Dropbox/data/proj/PE_HYZ/PublicDataSet/Liver_Macrophage/LiverMacrophageNNlearned_
      ↪Ref2022-12-21_MergeCluster_35epch.h5ad'
```

```
[72]: scm.CamelSwapline.writedata(adatax=scref, filename=filename, filepath=work_dir)
```

```
[196]: scref.obs.to_csv("Metatable.csv", sep=",")
```

## TUTORIALS\_SCCAMEL\_EVO

CAMELevo package for single-cell data integration across species is coming soon!

Part of the analysis has been applied for the research project below:

bioRxiv. 2023 Mar 18;2023.03.17.533207. doi: 10.1101/2023.03.17.533207. Preprint. Single-Soma Deep RNA sequencing of Human DRG Neurons Reveals Novel Molecular and Cellular Mechanisms Underlying Somatosensation. Huasheng Yu, Dmitry Usoskin, Saad S Nagi, Yizhou Hu, Jussi Kupari, Otmame Bouchatta, Suna Li Cranfill, Yijing Su, You Lv, Hongjun Song, Guo-Li Ming, Stephen Prouty, John Seykora, Hao Wu, Minghong Ma, Håkan Olausson, Patrik Ernfors, Wenqin Luo



## **TUTORIALS\_SCCAMEL\_VICUNA**

VIVUNA package for intercellular communication and Spatial Transcriptomics analysis is coming soon.

The intercellular communication analysis has been applied for the research project below:

Gao, Y., Yao, X., Zhai, Y., Li, L., Li, H., Sun, X., Yu, P., Xue, T., Li, Y.#\*, and Hu, Y.#\* (2021) “Single cell transcriptional zonation of human psoriasis skin identifies an alternative immuno- regulatory axis conducted by skin resident cells”. *Cell Death Dis.*, 2021 May 6;12(5):450.,





## REFERENCE

SWAPLINEv1.0 pipeline:

Hu Y.#, Jiang Y.#, Behnan J., Ribeiro MM., Kalantzi C., Zhang M., Lou D., Häring M., Sharma N., Okawa S., Del Sol A., Adameyko I., Svensson M., Persson O., Ernfors P., “Neural-network learning defines glioblastoma features to be of neural crest perivascular or radial glia lineages”, *Science Advances*, 2022 Jun 10;8(23) <https://www.science.org/doi/10.1126/sciadv.abm6340>

(1). Kupari, J., Usoskin, D., Parisien, M., Lou, D., Hu, Y., Fatt, M., Lonnerberg, P., Spangberg, M., Eriksson, B., Barkas, N., Kharchenko, PV., Loré, K., Khoury, S., Diatchenko, L., Ernfors, P. (2021) “Single cell transcriptomics of primate sensory neurons identifies cell types associated with chronic pain”, *Nat Commun.* 2021 Mar 8;12(1):1510.

(2). Gao, Y., Yao, X., Zhai, Y., Li, L., Li, H., Sun, X., Yu, P., Xue, T., Li, Y.#\*, and Hu, Y.#\* (2021) “Single cell transcriptional zonation of human psoriasis skin identifies an alternative immuno- regulatory axis conducted by skin resident cells”. *Cell Death Dis.*, 2021 May 6;12(5):450.,

(3). He B., Chen P., Zambrano S., Dabaghie D., Hu Y., Moller-Hackbarth K., Unnersjo-Jess D., Korkut G. G., Charrin E., Jeansson M., Bintanel-Morcillo M., Witasp A., Wennberg L., Wernerson A., Schermer B., Benzing T., Ernfors P., Betsholtz C., Lal M., Sandberg R., Patrakka J. (2021). Single-cell RNA sequencing reveals the mesangial identity and species diversity of glomerular cell transcriptomes. *Nat Commun* 12, 2141.,

(4). bioRxiv. 2023 Mar 18;2023.03.17.533207. doi: 10.1101/2023.03.17.533207. Preprint. Single-Soma Deep RNA sequencing of Human DRG Neurons Reveals Novel Molecular and Cellular Mechanisms Underlying Somatosensation. Huasheng Yu, Dmitry Usoskin, Saad S Nagi, Yizhou Hu, Jussi Kupari, Otmane Bouchatta, Suna Li Cranfill, Yijing Su, You Lv, Hongjun Song, Guo-Li Ming, Stephen Prouty, John Seykora, Hao Wu, Minghong Ma, Håkan Olausson, Patrik Ernfors, Wenqin Luo