1. Introduction

MOCHIS is a software that allows the user to perform flexible non-parametric tests of differential gene expression. Such tests include the popular Mann-Whitney (Wilcoxon rank sum) test, which was recently promoted by Li et al. (2022) as an approach to perform differential analysis on RNA-seq data without incurring an inflated false positive rate. In this markdown document, we explore how MOCHIS can detect multiple kinds of differential gene expression signatures, including mean shifts or dispersion shifts. Dispersion shifts have recently been shown to characterize age-related changes in gene expression (see Schaum et al., 2020 and Yamamoto and Chung et al., 2022+). In particular, we:

- perform multiple kinds of two-sample tests on all single-cell tissue data provided in Tabula muris senis
- report and compare findings across the different kinds of tests For Section 3 (Analysis), all our analyses are followed by a summary of key findings, to help the reader quickly grasp the main points.

In [1]:
```
# Setup

import scanpy
import numpy as np
import anndata
import pandas as pd
import matplotlib.pyplot as plt
from main_draft0 import *
import scipy
import statistics
import csv
import os
import seaborn as sns
from matplotlib_venn import venn2
import math

np.random.seed(2022)
scipy.__version__
```

Out[1]:
```
'1.8.0'
```

2. Data

Publicly available *mus musculus* (house mice) single-cell RNA-seq data from the Chan-Zuckerberg Initiative (also known as *Tabula Muris Senis*) is used. We download senescence datasets from here (https://cellxgene.cziscience.com/collections/0b9d8a04-bb9d-44da-aa27-705bb65b54eb). These datasets are made up of single cell gene transcript levels measured using Smart-Seq2, across 22 distinct mice tissues. For each tissue, the cells originate from mice that are either 3 months, 18 months or 24 months old (with the exception of the mammary gland tissue, which has 3 months, 18 months and 21 months). There are also other cell labels like tissue location (identified with guidance from biologists) and mice sex.

Below, we perform the Mann-Whitney test to identify genes that are differentially expressed, also known as differentially expressed genes (DEGs), across age groups. We compare each pair of age group, so that for each gene $\binom{3}{2} = 3$ tests are performed.

We restrict our analysis to those regions where the zero counts are the fewest, using an 80% cut-off. This avoids running tests on genes that have pronounced zero inflation, which hinders the detection of differential expression.

We additionally compute a "ratio of variances" index, which heuristic measures of the difference in dispersion across the pair of age groups. The larger the ratio of variances, the more differentially dispersed the gene expression between the pair of age groups.

In [ ]:

In [2]:
```
# Perform analysis for each tissue
# There are 22 tissues
all_tissues = sorted(["bone-marrow",
                      "brain-myeloid",
                      "heart",
                      "large-intestine",
                      "lung",
                      "skin-of-body",
                      "thymus",
                      "limb-muscle",
                      "spleen",
                      "subcutaneous-adipose-tissue",
                      "tongue",
                      "gonadal-fat-pad",
                      "pancreas",
                      "mammary-gland",
                      "trachea",
                      "mesenteric-fat-pad",
                      "liver",
                      "bladder-lumen",
                      "brown-adipose-tissue",
                      "diaphragm",
                      "kidney",
                      "aorta"])


for tissue in all_tissues:
    #os.mkdir(os.path.join("tissues/", tissue))


    tissue_smartseq2_data = scanpy.read_h5ad('tissues/' + tissue + '.h5ad')
    transcripts = tissue_smartseq2_data.var.n_cells.index
    ages = np.array(tissue_smartseq2_data.obs['age'].index)
    smartseq2_raw_counts = tissue_smartseq2_data.raw.X.toarray()
    #print(smartseq2_raw_counts.shape)  # 14517 mice cells x 21069 regions

    # Get cutoff and restrict to only those genes
    cutoff = round(0.8*smartseq2_raw_counts.shape[0])

    cell_count_sums_by_region = np.count_nonzero(smartseq2_raw_counts, axis

    highly_expressed_genes_indices = [i for i,v in enumerate(cell_count_sum

    smartseq2_high_exp_sparse_mat = []
    for i in highly_expressed_genes_indices:
        smartseq2_high_exp_sparse_mat.append(smartseq2_raw_counts[:, i])

    print("Found ", len(highly_expressed_genes_indices), " genes out of ",

    highly_expressed_transcripts = [transcripts[i] for i in highly_expresse

    # Grab age labels
    #smartseq2_df = anndata.AnnData(np.transpose(smartseq2_high_exp_sparse_
    smartseq2_df = pd.DataFrame(np.append(np.transpose(smartseq2_high_exp_s

    # Run Mann-Whitney test for genes
```

```
    gene_names = smartseq2_df.columns.values[:-1]
    results_df = pd.DataFrame(columns=['TRANSCRIPT', 'MANN_WHITNEY_3_18', '

    print("How many cells of each age group?")
    print(smartseq2_df['ages'].value_counts())


    # Run test for each gene
    for i in range(len(gene_names)):
        to_run_test = smartseq2_df[[gene_names[i], 'ages']]

        if tissue == "mammary-gland":
            print("Reminder that mammary-gland has 3m, 18m and 21m age grou
            age_3m = to_run_test.loc[to_run_test["ages"] == "3m", gene_name
            age_18m = to_run_test.loc[to_run_test["ages"] == "18m", gene_na
            age_24m = to_run_test.loc[to_run_test["ages"] == "21m", gene_na
        else:
            age_3m = to_run_test.loc[to_run_test["ages"] == "3m", gene_name
            age_18m = to_run_test.loc[to_run_test["ages"] == "18m", gene_na
            age_24m = to_run_test.loc[to_run_test["ages"] == "24m", gene_na

        age_3m = [float(i) for i in age_3m]
        age_18m = [float(i) for i in age_18m]
        age_24m = [float(i) for i in age_24m]

        wrs_test_3_18 = scipy.stats.mannwhitneyu(x=age_3m, y=age_18m, alter
        wrs_test_18_24 = scipy.stats.mannwhitneyu(x=age_18m, y=age_24m, alt
        wrs_test_24_3 = scipy.stats.mannwhitneyu(x=age_3m, y=age_24m, alter

        var_3_18 = max(statistics.variance(age_3m)/statistics.variance(age_
        var_18_24 = max(statistics.variance(age_18m)/statistics.variance(ag
        var_24_3 = max(statistics.variance(age_24m)/statistics.variance(age

        results_df = results_df.append({
            'TRANSCRIPT': gene_names[i],
            'MANN_WHITNEY_3_18': wrs_test_3_18.pvalue,
            'MANN_WHITNEY_18_24': wrs_test_18_24.pvalue,
            'MANN_WHITNEY_24_3': wrs_test_24_3.pvalue,
            'VAR_3_18': var_3_18,
            'VAR_18_24': var_18_24,
            'VAR_24_3': var_24_3
        }, ignore_index=True)




    print("Saving results for ", tissue)
    results_df.to_csv("tissues/"+tissue+"/p_val_table.csv")
```

```
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
```

```
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...
Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
24m as 21m...

Reminder that mammary-gland has 3m, 18m and 21m age groups, so interpret
```

2.1 Mann-Whitney DEGs

Given we have the tables of p-values and ratios of variances from the previous step, we now select genes whose p-values, after a Benjamini-Hochberg adjustment procedure, lie below or equal to a 0.05 significance level. These are Mann-Whitney significant genes that would be flagged as potentially carrying biological signal in a typical differential expression analysis procedure.

In [3]:

```python
def p_adjust_bh(p):
    """Benjamini-Hochberg p-value correction for multiple hypothesis testin
    p = np.asfarray(p)
    by_descend = p.argsort()[::-1]
    by_orig = by_descend.argsort()
    steps = float(len(p)) / np.arange(len(p), 0, -1)
    q = np.minimum(1, np.minimum.accumulate(steps * p[by_descend]))
    return q[by_orig]
```

In [4]:
```
ranscript_3_18 = pd.DataFrame(columns=['TRANSCRIPT', 'MANN_WHITNEY', 'VARIAN
ranscript_18_24 = pd.DataFrame(columns=['TRANSCRIPT', 'MANN_WHITNEY', 'VARIA
ranscript_24_3 = pd.DataFrame(columns=['TRANSCRIPT', 'MANN_WHITNEY', 'VARIAN


ue in all_tissues:
t("Reading in summary of p-values and ratios of variances for ", tissue)
ue_mann_whitney_df = pd.read_csv("tissues/"+tissue+"/p_val_table.csv")

ck genes where one of the three pairs (3m, 18m, 24m) has significant p-value
cted_genes_3_18 = tissue_mann_whitney_df[p_adjust_bh(tissue_mann_whitney_df[
cted_genes_3_18 = selected_genes_3_18[["TRANSCRIPT", "MANN_WHITNEY_3_18", "V
cted_genes_3_18= selected_genes_3_18.rename(columns={"MANN_WHITNEY_3_18":"MA
cted_genes_3_18["TISSUE"] = [tissue for i in range(selected_genes_3_18.shape
ue_transcript_3_18 = pd.concat([tissue_transcript_3_18, selected_genes_3_18]

cted_genes_18_24 = tissue_mann_whitney_df[p_adjust_bh(tissue_mann_whitney_df
cted_genes_18_24 = selected_genes_18_24[["TRANSCRIPT", "MANN_WHITNEY_18_24",
cted_genes_18_24 = selected_genes_18_24.rename(columns={"MANN_WHITNEY_18_24"
cted_genes_18_24["TISSUE"] = [tissue for i in range(selected_genes_18_24.sha
ue_transcript_18_24 = pd.concat([tissue_transcript_18_24, selected_genes_18_

cted_genes_24_3 = tissue_mann_whitney_df[p_adjust_bh(tissue_mann_whitney_df[
cted_genes_24_3 = selected_genes_24_3[["TRANSCRIPT", "MANN_WHITNEY_24_3", "V
cted_genes_24_3 = selected_genes_24_3.rename(columns={"MANN_WHITNEY_24_3":"M
cted_genes_24_3["TISSUE"] = [tissue for i in range(selected_genes_24_3.shape
ue_transcript_24_3 = pd.concat([tissue_transcript_24_3, selected_genes_24_3]

ranscript_3_18.to_csv("tissues/mw_sig_3m_18m.csv")
ranscript_18_24.to_csv("tissues/mw_sig_18m_24m.csv")
ranscript_24_3.to_csv("tissues/mw_sig_24m_3m.csv")
```

```
Reading in summary of p-values and ratios of variances for  aorta
Reading in summary of p-values and ratios of variances for  bladder-lumen
Reading in summary of p-values and ratios of variances for  bone-marrow
Reading in summary of p-values and ratios of variances for  brain-myeloid
Reading in summary of p-values and ratios of variances for  brown-adipose
-tissue
Reading in summary of p-values and ratios of variances for  diaphragm
Reading in summary of p-values and ratios of variances for  gonadal-fat-p
ad
Reading in summary of p-values and ratios of variances for  heart
Reading in summary of p-values and ratios of variances for  kidney
Reading in summary of p-values and ratios of variances for  large-intesti
ne
Reading in summary of p-values and ratios of variances for  limb-muscle
Reading in summary of p-values and ratios of variances for  liver
Reading in summary of p-values and ratios of variances for  lung
Reading in summary of p-values and ratios of variances for  mammary-gland
Reading in summary of p-values and ratios of variances for  mesenteric-fa
```

```
t-pad
Reading in summary of p-values and ratios of variances for  pancreas
Reading in summary of p-values and ratios of variances for  skin-of-body
Reading in summary of p-values and ratios of variances for  spleen
Reading in summary of p-values and ratios of variances for  subcutaneous-
adipose-tissue
Reading in summary of p-values and ratios of variances for  thymus
Reading in summary of p-values and ratios of variances for  tongue
Reading in summary of p-values and ratios of variances for  trachea
```

Let us visualize the raw p-values and variance ratios of the Mann-Whitney DEGs fished out from the above procedure.

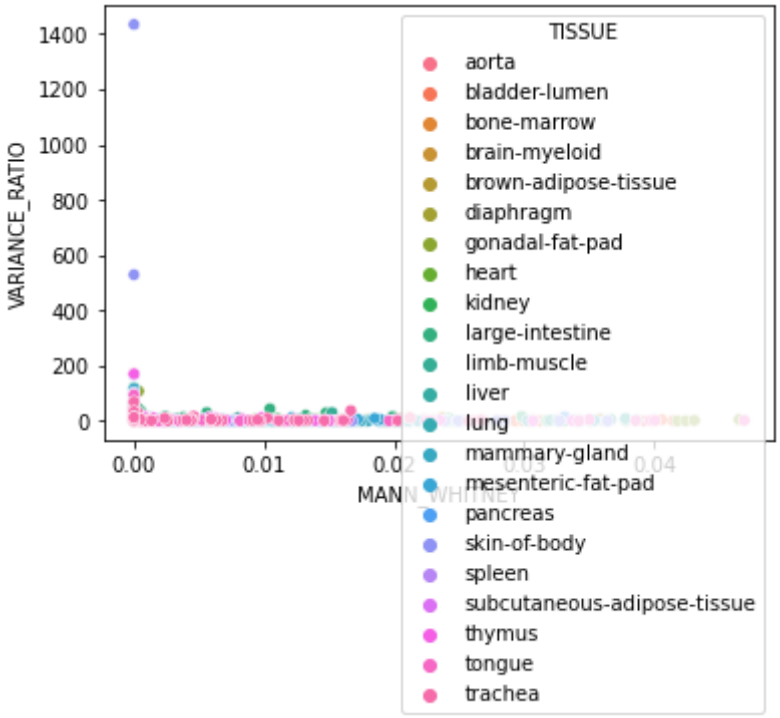
In [5]:
```
df_3_18 = pd.read_csv("tissues/mw_sig_3m_18m.csv")
df_18_24 = pd.read_csv("tissues/mw_sig_18m_24m.csv")
df_24_3 = pd.read_csv("tissues/mw_sig_24m_3m.csv")


df_3_18["PAIR"] = ["3m vs 18m" for i in range(df_3_18.shape[0])]
df_18_24["PAIR"] = ["18m vs 24m" for i in range(df_18_24.shape[0])]
df_24_3["PAIR"] = ["3m vs 24m" for i in range(df_24_3.shape[0])]



'''
groups = df_3_18.groupby("TISSUE")
for name, group in groups:
    plt.plot(group["MANN_WHITNEY"], group["VARIANCE_RATIO"], marker="o", li
plt.legend()

groups = df_18_24.groupby("TISSUE")
for name, group in groups:
    plt.plot(group["MANN_WHITNEY"], group["VARIANCE_RATIO"], marker="o", li
plt.legend()

groups = df_24_3.groupby("TISSUE")
for name, group in groups:
    plt.plot(group["MANN_WHITNEY"], group["VARIANCE_RATIO"], marker="o", li
plt.legend()


grand_df = pd.concat(pd.concat(df_3_18, df_18_24), df_24_3)

g = sns.FacetGrid([grand_df], col="PAIR")
g.map(sns.scatterplot, x='MANN_WHITNEY', y='VARIANCE_RATIO', hue='TISSUE',

#sns.scatterplot(data=grand_df, x='MANN_WHITNEY', y='VARIANCE_RATIO', hue='

sns.scatterplot(data=df_3_18, x='MANN_WHITNEY', y='VARIANCE_RATIO', hue='TI
sns.show()
sns.scatterplot(data=df_3_18, x='MANN_WHITNEY', y='VARIANCE_RATIO', hue='TI
'''
```

Out[5]:
```
'\ngroups = df_3_18.groupby("TISSUE")\nfor name, group in groups:\n    pl
t.plot(group["MANN_WHITNEY"], group["VARIANCE_RATIO"], marker="o", linest
yle="", label=name, alpha=0.5)\nplt.legend()\n\ngroups = df_18_24.groupby
("TISSUE")\nfor name, group in groups:\n    plt.plot(group["MANN_WHITNE
Y"], group["VARIANCE_RATIO"], marker="o", linestyle="", label=name, alpha
=0.5)\nplt.legend()\n\ngroups = df_24_3.groupby("TISSUE")\nfor name, grou
p in groups:\n    plt.plot(group["MANN_WHITNEY"], group["VARIANCE_RATI
O"], marker="o", linestyle="", label=name, alpha=0.5)\nplt.legend()\n\n\n
grand_df = pd.concat(pd.concat(df_3_18, df_18_24), df_24_3)\n\ng = sns.Fa
cetGrid([grand_df], col="PAIR")\ng.map(sns.scatterplot, x=\'MANN_WHITNEY
\', y=\'VARIANCE_RATIO\', hue=\'TISSUE\', alpha=0.5)\n\n#sns.scatterplot
(data=grand_df, x=\'MANN_WHITNEY\', y=\'VARIANCE_RATIO\', hue=\'TISSUE\',
alpha=0.5).FacetGrid\n\nsns.scatterplot(data=df_3_18, x=\'MANN_WHITNEY\',
```

```
y=\'VARIANCE_RATIO\', hue=\'TISSUE\')\nsns.show()\nsns.scatterplot(data=df_3_18, x=\'MANN_WHITNEY\', y=\'VARIANCE_RATIO\', hue=\'TISSUE\')\n'
```

In [6]:
```
sns.scatterplot(data=df_18_24, x='MANN_WHITNEY', y='VARIANCE_RATIO', hue='T
```

Out[6]: <AxesSubplot:xlabel='MANN_WHITNEY', ylabel='VARIANCE_RATIO'>

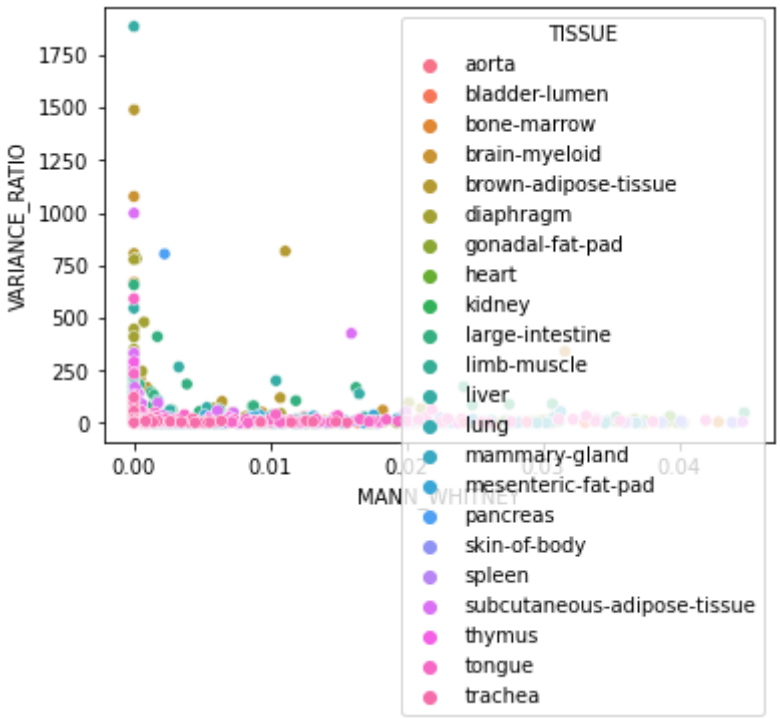


In [7]:
```
sns.scatterplot(data=df_3_18, x='MANN_WHITNEY', y='VARIANCE_RATIO', hue='TI
```

Out[7]: <AxesSubplot:xlabel='MANN_WHITNEY', ylabel='VARIANCE_RATIO'>

In [8]:
```
sns.scatterplot(data=df_24_3, x='MANN_WHITNEY', y='VARIANCE_RATIO', hue='TI
```

Out[8]:
```
<AxesSubplot:xlabel='MANN_WHITNEY', ylabel='VARIANCE_RATIO'>
```



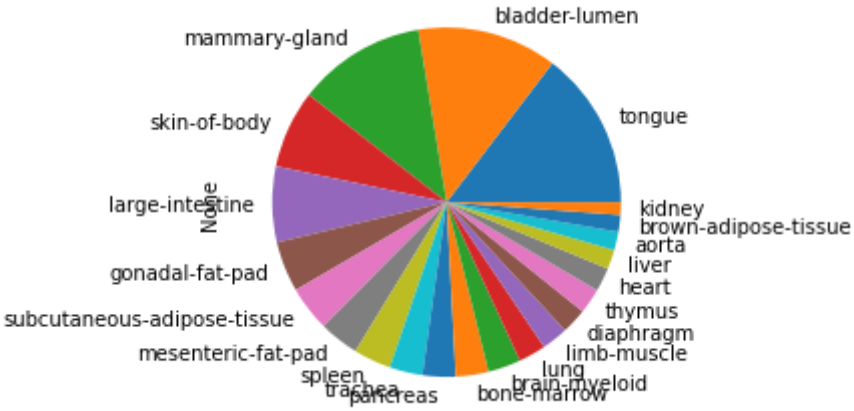
Next we look at the distribution, across tissues, of Mann-Whitney significant genes.

In [9]:
```
print("No. MW significant genes for 3m vs 18m: ", df_3_18.shape[0])
print("No. MW significant genes for 18m vs 24m: ", df_18_24.shape[0])
print("No. MW significant genes for 24m vs 3m: ", df_24_3.shape[0])
```

```
No. MW significant genes for 3m vs 18m:  5571
No. MW significant genes for 18m vs 24m:  5305
No. MW significant genes for 24m vs 3m:  5634
```

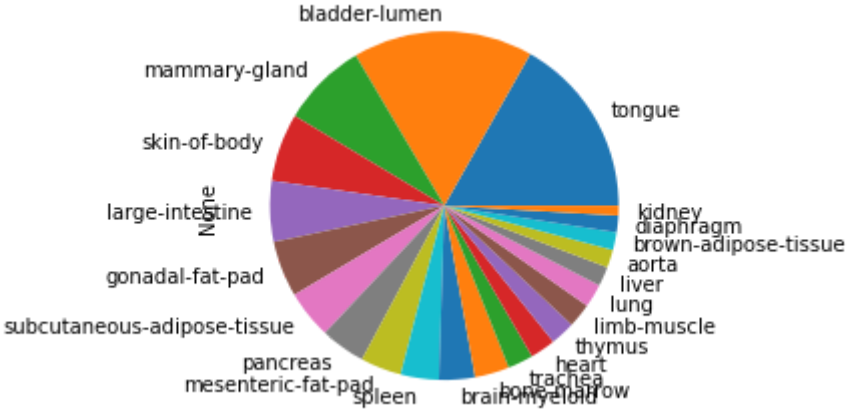
In [10]:
```
df_3_18.value_counts("TISSUE").plot(kind="pie")
```

Out[10]: <AxesSubplot:ylabel='None'>



In [11]:
```
df_18_24.value_counts("TISSUE").plot(kind="pie")
```

Out[11]: <AxesSubplot:ylabel='None'>



In [12]:
```
df_24_3.value_counts("TISSUE").plot(kind="pie")
```

Out[12]: <AxesSubplot:ylabel='None'>

2.2 MOCHIS

We now repeat the DEG identification procedure above, now using our flexible non-parametric testing software MOCHIS. We run MOCHIS with test statistic $||S_{n,k}||^p_{p,\boldsymbol{w}}$. We choose the following parametrization:

- $p = 1$
- $\boldsymbol{w} = \left((\frac{j}{k} - \frac{1}{2})^2 : j = 1, \ldots, k\right)$

This parametrization optimizes detection of dispersion shifts between two samples.

Step 1. Compute p-values.

When computing the p-values, we apply a tie-breaking routine (adding noise ranging from -0.25 to 0.25, which is less than the minimum spacing width of integer counts). To ensure that this routine does not overly contaminate the data, we also compute Mann-Whitney p-values and check that the Mann-Whitney DEGs identified after applying the tie-breaking routine are not markedly different from the original DEGs identified in Section 2.1. We report this latter comparison between post-contamination and original DEGs in Section 2.3. (Heads up: We find little difference.)

In [ ]:

In [13]:
```
for tissue in all_tissues:

    if tissue != 'bone-marrow':
        continue


    #os.mkdir(os.path.join("tissues/", tissue))



    tissue_smartseq2_data = scanpy.read_h5ad('tissues/' + tissue + '.h5ad')
    transcripts = tissue_smartseq2_data.var.n_cells.index
    ages = np.array(tissue_smartseq2_data.obs['age'].index)
    smartseq2_raw_counts = tissue_smartseq2_data.raw.X.toarray()
    print(smartseq2_raw_counts.shape)  # 14517 mice cells x 21069 regions

    # Get cutoff and restrict to only those genes
    cutoff = round(0.8*smartseq2_raw_counts.shape[0])

    cell_count_sums_by_region = np.count_nonzero(smartseq2_raw_counts, axis

    highly_expressed_genes_indices = [i for i,v in enumerate(cell_count_sum

    smartseq2_high_exp_sparse_mat = []
    for i in highly_expressed_genes_indices:
        smartseq2_high_exp_sparse_mat.append(smartseq2_raw_counts[:, i])

    print("Found ", len(highly_expressed_genes_indices), " genes out of ",

    highly_expressed_transcripts = [transcripts[i] for i in highly_expresse

    # Grab age labels
    #smartseq2_df = anndata.AnnData(np.transpose(smartseq2_high_exp_sparse_
    smartseq2_df = pd.DataFrame(np.append(np.transpose(smartseq2_high_exp_s


    # Run Mann-Whitney test for genes
    gene_names = smartseq2_df.columns.values[:-1]


    results_df = pd.DataFrame(columns=['TRANSCRIPT',
                                       'MOCHIS_3_18',
                                       'MW_3_18',
                                       'MOCHIS_18_24',
                                       'MW_18_24',
                                       'MOCHIS_24_3',
                                       'MW_24_3',
                                       'VAR_3_18',
                                       'INV_3_18',
                                       'VAR_18_24',
                                       'INV_18_24',
                                       'VAR_24_3',
                                       'INV_24_3'])
    print("How many cells of each age group?")
    print(smartseq2_df['ages'].value_counts())
```

```
# Run test for each gene
for i in range(len(gene_names)):
    to_run_test = smartseq2_df[[gene_names[i], 'ages']]

    if tissue == "mammary-gland":
        print("Reminder that mammary-gland has 3m, 18m and 21m age grou
        age_3m = to_run_test.loc[to_run_test["ages"] == "3m", gene_name
        age_18m = to_run_test.loc[to_run_test["ages"] == "18m", gene_na
        age_24m = to_run_test.loc[to_run_test["ages"] == "21m", gene_na
    else:
        age_3m = to_run_test.loc[to_run_test["ages"] == "3m", gene_name
        age_18m = to_run_test.loc[to_run_test["ages"] == "18m", gene_na
        age_24m = to_run_test.loc[to_run_test["ages"] == "24m", gene_na

    age_3m = [float(i) for i in age_3m]
    age_18m = [float(i) for i in age_18m]
    age_24m = [float(i) for i in age_24m]


    # Add noise to break ties

    noisy_age_3m = np.sort([value + np.random.uniform(-1/4, 1/4) for va
    noisy_age_18m = np.sort([value + np.random.uniform(-1/4, 1/4) for v
    noisy_age_24m = np.sort([value + np.random.uniform(-1/4, 1/4) for v


    wrs_test_3_18 = scipy.stats.mannwhitneyu(x=noisy_age_3m, y=noisy_ag
    wrs_test_18_24 = scipy.stats.mannwhitneyu(x=noisy_age_18m, y=noisy_
    wrs_test_24_3 = scipy.stats.mannwhitneyu(x=noisy_age_3m, y=noisy_ag




    if len(noisy_age_3m) > len(noisy_age_18m):
        #print("3 > 18")

        k = len(age_18m) + 1
        mochis_weights = [(i/k-0.5)**2 for i in range(1,k+1)]
        mochis_test_3_18 = mochis_py(x = noisy_age_18m,
                                     p = 1,
                                     wList = mochis_weights,
                                     alternative = "two.sided",
                                     approx = "chebyshev",
                                     n_mom = 100,
                                     y = noisy_age_3m)

    else:
        #print(" 18 > 3")

        k = len(age_3m) + 1
        mochis_weights = [(i/k-0.5)**2 for i in range(1,k+1)]
        mochis_test_3_18 = mochis_py(x = noisy_age_3m,
                                     p = 1,
                                     wList = mochis_weights,
                                     alternative = "two.sided",
                                     approx = "chebyshev",
                                     n_mom = 100,
```

```
                                    y = noisy_age_18m)



    if len(noisy_age_18m) > len(noisy_age_24m):
        #print("18 > 24")

        k = len(noisy_age_24m) + 1
        mochis_weights = [(i/k-0.5)**2 for i in range(1,k+1)]
        mochis_test_18_24 = mochis_py(x = noisy_age_24m,
                                      p = 1,
                                      wList = mochis_weights,
                                      alternative = "two.sided",
                                      approx = "chebyshev",
                                      n_mom = 100,
                                      y = noisy_age_18m)

    else:
        #print("24 > 18")

        k = len(noisy_age_18m) + 1
        mochis_weights = [(i/k-0.5)**2 for i in range(1,k+1)]
        mochis_test_18_24 = mochis_py(x = noisy_age_18m,
                                      p = 1,
                                      wList = mochis_weights,
                                      alternative = "two.sided",
                                      approx = "chebyshev",
                                      n_mom = 100,
                                      y = noisy_age_24m)


    if len(noisy_age_3m) > len(noisy_age_24m):
        #print("3 > 24")

        k = len(noisy_age_24m) + 1
        mochis_weights = [(i/k-0.5)**2 for i in range(1,k+1)]
        mochis_test_24_3 = mochis_py(x = noisy_age_24m,
                                      p = 1,
                                      wList = mochis_weights,
                                      alternative = "two.sided",
                                      approx = "chebyshev",
                                      n_mom = 100,
                                      y = noisy_age_3m)

    else:
        #print(" 24 > 3")

        k = len(noisy_age_3m) + 1
        mochis_weights = [(i/k-0.5)**2 for i in range(1,k+1)]
        mochis_test_24_3 = mochis_py(x = noisy_age_3m,
                                      p = 1,
                                      wList = mochis_weights,
                                      alternative = "two.sided",
                                      approx = "chebyshev",
                                      n_mom = 100,
                                      y = noisy_age_24m)
```

```
        var_3_18 = max(statistics.variance(age_3m)/statistics.variance(age_
        var_18_24 = max(statistics.variance(age_18m)/statistics.variance(ag
        var_24_3 = max(statistics.variance(age_24m)/statistics.variance(age

        invert_3_18 = False
        invert_18_24 = False
        invert_24_3 = False
        if var_3_18 == statistics.variance(age_3m)/statistics.variance(age_
            invert_3_18 = True
        if var_18_24 == statistics.variance(age_18m)/statistics.variance(ag
            invert_18_24 = True
        if var_24_3 == statistics.variance(age_3m)/statistics.variance(age_
            invert_24_3 = True

        results_df = pd.concat([results_df, pd.DataFrame([{
            "TRANSCRIPT": gene_names[i],
            "MOCHIS_3_18": mochis_test_3_18,
            "MW_3_18": wrs_test_3_18.pvalue,
            "MOCHIS_18_24": mochis_test_18_24,
            "MW_18_24": wrs_test_18_24.pvalue,
            "MOCHIS_24_3": mochis_test_24_3,
            "MW_24_3": wrs_test_24_3.pvalue,
            "VAR_3_18": var_3_18,
            "INV_3_18": invert_3_18,
            "VAR_18_24": var_18_24,
            "INV_18_24": invert_18_24,
            "VAR_24_3": var_24_3,
            "INV_24_3": invert_24_3
        }])])

    print("Saving results for ", tissue)
    results_df.to_csv("tissues/"+tissue+"/mochis_p_val_table.csv")
```

```
The test statistic for the data is  0.3053510504722908
Sample sizes, n and k, large enough such that k/n > 0; p = 1 or p = 2. Ap
plying Gaussian asymptotics...
Normalizing weight vector...
The test statistic for the data is  0.26422385453581587
Sample sizes, n and k, large enough such that k/n > 0; p = 1 or p = 2. Ap
plying Gaussian asymptotics...
Normalizing weight vector...
The test statistic for the data is  0.3219549623245731
Sample sizes, n and k, large enough such that k/n > 0; p = 1 or p = 2. Ap
plying Gaussian asymptotics...
Normalizing weight vector...
The test statistic for the data is  0.2806508049250802
Sample sizes, n and k, large enough such that k/n > 0; p = 1 or p = 2. Ap
plying Gaussian asymptotics...
Normalizing weight vector...
The test statistic for the data is  0.21997230703276258
Sample sizes, n and k, large enough such that k/n > 0; p = 1 or p = 2. Ap
plying Gaussian asymptotics...
```

```
Saving results for  bone-marrow
```

In [14]:
```
results_df
```

Out[14]:

	TRANSCRIPT	MOCHIS_3_18	MW_3_18	MOCHIS_18_24	MW_18_24	MOCHIS_24_3	I
0	ENSMUSG00000030057	8.890550e-38	5.376231e-07	9.755844e-01	3.326401e-03	1.149667e-37	9.
0	ENSMUSG00000090862	4.544857e-17	1.289933e-01	1.528331e-15	1.393203e-02	5.978204e-72	5.
0	ENSMUSG00000041841	2.603120e-08	4.054963e-34	3.079159e-15	5.656713e-47	1.830482e-06	4.
0	ENSMUSG00000060802	1.949720e-27	2.321067e-211	2.219486e-07	9.283830e-22	2.324125e-136	8.
0	ENSMUSG00000060743	5.174169e-63	4.278586e-05	1.001473e-07	2.362372e-16	2.255800e-104	9.
...	...	...	...	...	...	...	
0	ENSMUSG00000021998	2.468378e-05	4.467454e-06	1.447641e-10	4.349640e-116	7.489741e-02	1.
0	ENSMUSG00000023010	3.868996e-42	1.243954e-66	3.916407e-05	2.588607e-98	3.917135e-72	4.
0	ENSMUSG00000092341	1.656626e-19	9.697577e-64	4.810845e-09	1.191930e-23	2.711812e-79	2.
0	ENSMUSG00000060636	1.362760e-10	1.792408e-11	1.020011e-06	8.210547e-22	3.093533e-30	7.
0	ENSMUSG00000074884	5.827214e-02	1.577553e-75	8.810097e-18	5.357637e-56	2.228374e-78	2.

198 rows × 13 columns

Step 2. Identify MOCHIS significant genes (with FDR control at 0.05)

In [15]:
```
tissue_transcript_3_18 = pd.DataFrame(columns=['TRANSCRIPT', 'MOCHIS', 'VARI
tissue_transcript_18_24 = pd.DataFrame(columns=['TRANSCRIPT', 'MOCHIS', 'VAI
tissue_transcript_24_3 = pd.DataFrame(columns=['TRANSCRIPT', 'MOCHIS', 'VARI


for tissue in all_tissues:
    print("Reading in summary of p-values and ratios of variances for ", tis
    tissue_mochis_df = pd.read_csv("tissues/"+tissue+"/mochis_p_val_table.cs

    # Pick genes where one of the three pairs (3m, 18m, 24m) has significant
    selected_genes_3_18 = tissue_mochis_df[p_adjust_bh(tissue_mochis_df['MOC
    selected_genes_3_18 = selected_genes_3_18[["TRANSCRIPT", "MOCHIS_3_18",
    selected_genes_3_18= selected_genes_3_18.rename(columns={"MOCHIS_3_18":'
    selected_genes_3_18["TISSUE"] = [tissue for i in range(selected_genes_3_
    tissue_transcript_3_18 = pd.concat([tissue_transcript_3_18, selected_ger

    selected_genes_18_24 = tissue_mochis_df[p_adjust_bh(tissue_mochis_df['MO
    selected_genes_18_24 = selected_genes_18_24[["TRANSCRIPT", "MOCHIS_18_24
    selected_genes_18_24 = selected_genes_18_24.rename(columns={"MOCHIS_18_2
    selected_genes_18_24["TISSUE"] = [tissue for i in range(selected_genes_1
    tissue_transcript_18_24 = pd.concat([tissue_transcript_18_24, selected_g

    selected_genes_24_3 = tissue_mochis_df[p_adjust_bh(tissue_mochis_df['MOC
    selected_genes_24_3 = selected_genes_24_3[["TRANSCRIPT", "MOCHIS_24_3",
    selected_genes_24_3 = selected_genes_24_3.rename(columns={"MOCHIS_24_3":
    selected_genes_24_3["TISSUE"] = [tissue for i in range(selected_genes_24
    tissue_transcript_24_3 = pd.concat([tissue_transcript_24_3, selected_ger

tissue_transcript_3_18.to_csv("tissues/mochis_sig_3m_18m.csv")
tissue_transcript_18_24.to_csv("tissues/mochis_sig_18m_24m.csv")
tissue_transcript_24_3.to_csv("tissues/mochis_sig_24m_3m.csv")
```

```
Reading in summary of p-values and ratios of variances for  aorta
Reading in summary of p-values and ratios of variances for  bladder-lumen
Reading in summary of p-values and ratios of variances for  bone-marrow
Reading in summary of p-values and ratios of variances for  brain-myeloid
Reading in summary of p-values and ratios of variances for  brown-adipose
-tissue
Reading in summary of p-values and ratios of variances for  diaphragm
Reading in summary of p-values and ratios of variances for  gonadal-fat-p
ad
Reading in summary of p-values and ratios of variances for  heart
Reading in summary of p-values and ratios of variances for  kidney
Reading in summary of p-values and ratios of variances for  large-intesti
ne
Reading in summary of p-values and ratios of variances for  limb-muscle
Reading in summary of p-values and ratios of variances for  liver
Reading in summary of p-values and ratios of variances for  lung
Reading in summary of p-values and ratios of variances for  mammary-gland
Reading in summary of p-values and ratios of variances for  mesenteric-fa
t-pad
Reading in summary of p-values and ratios of variances for  pancreas
Reading in summary of p-values and ratios of variances for  skin-of-body
Reading in summary of p-values and ratios of variances for  spleen
Reading in summary of p-values and ratios of variances for  subcutaneous-
```

```
adipose-tissue
Reading in summary of p-values and ratios of variances for  thymus
Reading in summary of p-values and ratios of variances for  tongue
Reading in summary of p-values and ratios of variances for  trachea
```

In [ ]:

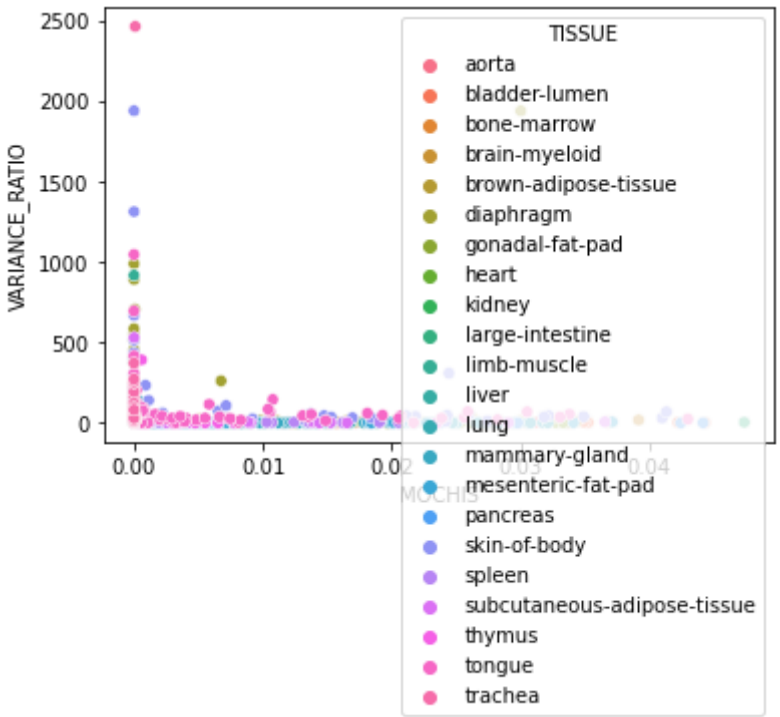
Step 3. Visualization.

First, let us visualize the raw p-values and variance ratios of the MOCHIS DEGs fished out from the above procedure.

In [16]:
```
df_3_18 = pd.read_csv("tissues/mochis_sig_3m_18m.csv")
df_18_24 = pd.read_csv("tissues/mochis_sig_18m_24m.csv")
df_24_3 = pd.read_csv("tissues/mochis_sig_24m_3m.csv")


df_3_18["PAIR"] = ["3m vs 18m" for i in range(df_3_18.shape[0])]
df_18_24["PAIR"] = ["18m vs 24m" for i in range(df_18_24.shape[0])]
df_24_3["PAIR"] = ["3m vs 24m" for i in range(df_24_3.shape[0])]
```

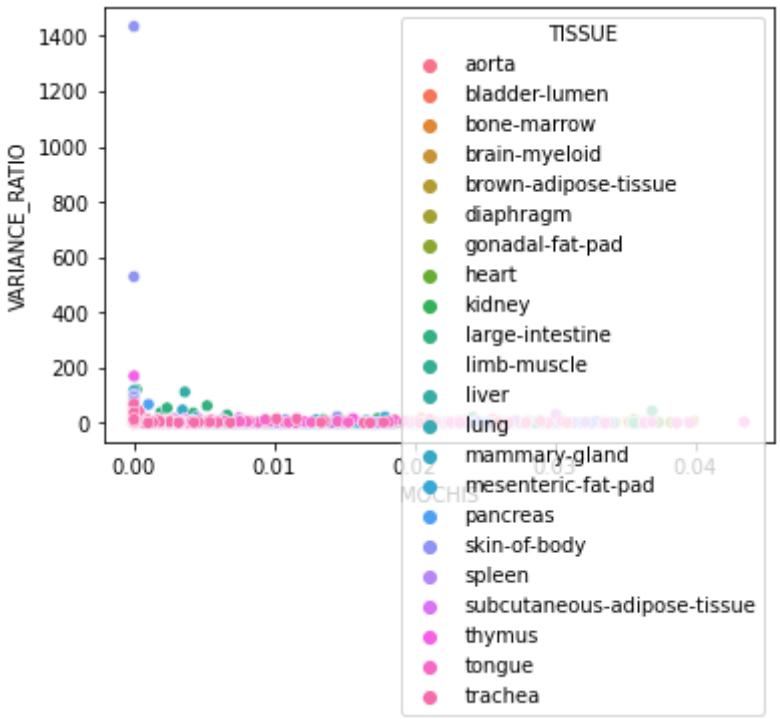
In [17]:
```
sns.scatterplot(data=df_3_18, x='MOCHIS', y='VARIANCE_RATIO', hue='TISSUE')
```

Out[17]: <AxesSubplot:xlabel='MOCHIS', ylabel='VARIANCE_RATIO'>



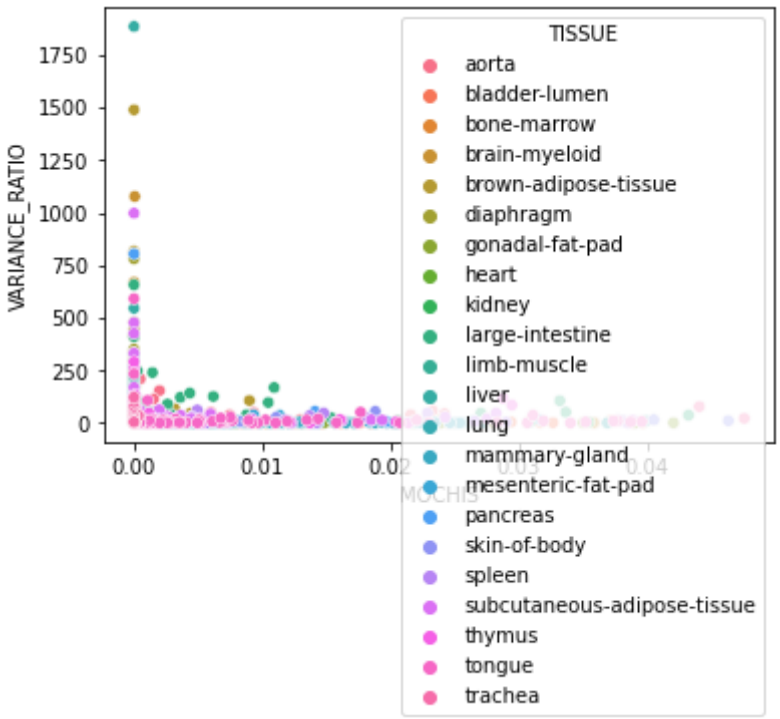
In [18]:
```
sns.scatterplot(data=df_18_24, x='MOCHIS', y='VARIANCE_RATIO', hue='TISSUE'
```

Out[18]:
```
<AxesSubplot:xlabel='MOCHIS', ylabel='VARIANCE_RATIO'>
```



In [19]:
```
sns.scatterplot(data=df_24_3, x='MOCHIS', y='VARIANCE_RATIO', hue='TISSUE')
```

Out[19]:
```
<AxesSubplot:xlabel='MOCHIS', ylabel='VARIANCE_RATIO'>
```



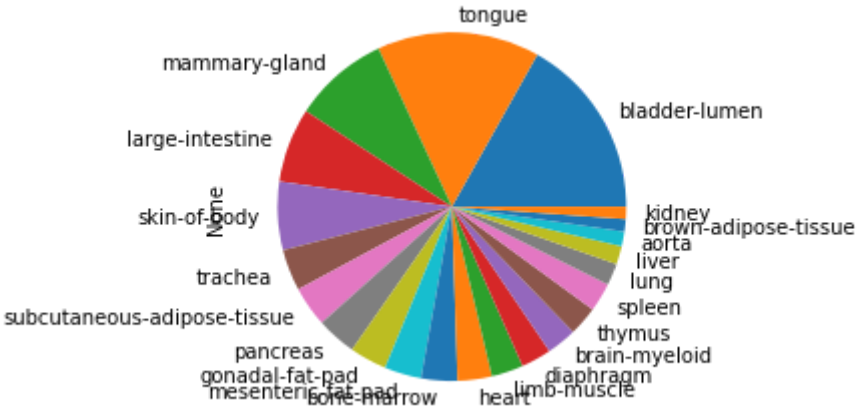
Next we look at the distribution, across tissues, of MOCHIS significant genes.

In [20]:
```
print("No. MOCHIS significant genes for 3m vs 18m: ", df_3_18.shape[0])
print("No. MOCHIS significant genes for 18m vs 24m: ", df_18_24.shape[0])
print("No. MOCHIS significant genes for 24m vs 3m: ", df_24_3.shape[0])
```

```
No. MOCHIS significant genes for 3m vs 18m:  5727
No. MOCHIS significant genes for 18m vs 24m:  4770
No. MOCHIS significant genes for 24m vs 3m:  5478
```

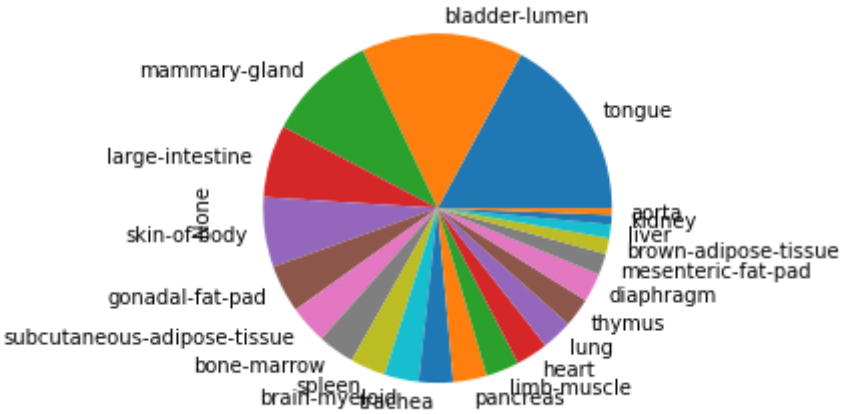
In [21]:
```
df_3_18.value_counts("TISSUE").plot(kind="pie")
```

Out[21]: <AxesSubplot:ylabel='None'>



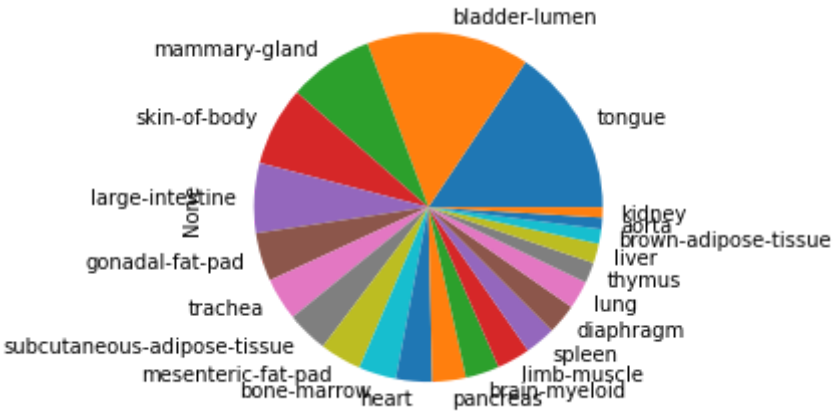
In [22]:
```
df_18_24.value_counts("TISSUE").plot(kind="pie")
```

Out[22]: <AxesSubplot:ylabel='None'>



In [23]:
```
df_24_3.value_counts("TISSUE").plot(kind="pie")
```

Out[23]:
```
<AxesSubplot:ylabel='None'>
```



2.3 Impact of Tie Breaking on Mann-Whitney DEGs

In Section 2.2, we raised the issue of tie breaking potentially affecting the significance of Mann-Whitney DEGs. Here, we check for difference between post-contaminated Mann-Whitney DEGs (this Section) and original DEGs (Section 2.1).

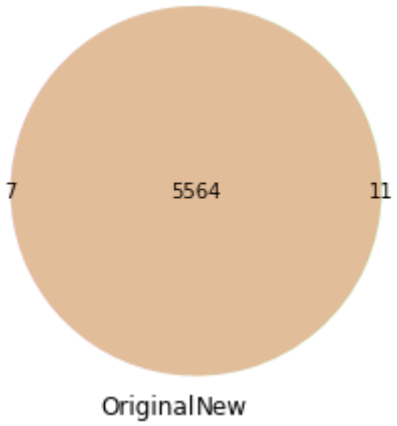
In [24]:
```
tissue_transcript_3_18 = pd.DataFrame(columns=['TRANSCRIPT', 'NEW_MANN_WHIT
tissue_transcript_18_24 = pd.DataFrame(columns=['TRANSCRIPT', 'NEW_MANN_WHI
tissue_transcript_24_3 = pd.DataFrame(columns=['TRANSCRIPT', 'NEW_MANN_WHIT

for tissue in all_tissues:
    print("Reading in summary of p-values and ratios of variances for ", ti

    tissue_mann_whitney_df = pd.read_csv("tissues/"+tissue+"/mochis_p_val_t

    # Pick genes where one of the three pairs (3m, 18m, 24m) has significan
    selected_genes_3_18 = tissue_mann_whitney_df[p_adjust_bh(tissue_mann_wh
    selected_genes_3_18 = selected_genes_3_18[["TRANSCRIPT", "MW_3_18"]]
    selected_genes_3_18= selected_genes_3_18.rename(columns={"MW_3_18":"NEW
    selected_genes_3_18["TISSUE"] = [tissue for i in range(selected_genes_3
    tissue_transcript_3_18 = pd.concat([tissue_transcript_3_18, selected_ge

    selected_genes_18_24 = tissue_mann_whitney_df[p_adjust_bh(tissue_mann_w
    selected_genes_18_24 = selected_genes_18_24[["TRANSCRIPT", "MW_18_24"]]
    selected_genes_18_24= selected_genes_18_24.rename(columns={"MW_18_24":"
    selected_genes_18_24["TISSUE"] = [tissue for i in range(selected_genes_
    tissue_transcript_18_24 = pd.concat([tissue_transcript_18_24, selected_

    selected_genes_24_3 = tissue_mann_whitney_df[p_adjust_bh(tissue_mann_wh
    selected_genes_24_3 = selected_genes_24_3[["TRANSCRIPT", "MW_24_3"]]
    selected_genes_24_3 = selected_genes_24_3.rename(columns={"MW_24_3":"NE
    selected_genes_24_3["TISSUE"] = [tissue for i in range(selected_genes_2
    tissue_transcript_24_3 = pd.concat([tissue_transcript_24_3, selected_ge

# Compare against original MW significant genes
og_transcript_3_18 = pd.read_csv("tissues/mw_sig_3m_18m.csv")
og_transcript_18_24 = pd.read_csv("tissues/mw_sig_18m_24m.csv")
og_transcript_24_3 = pd.read_csv("tissues/mw_sig_24m_3m.csv")
```

```
Reading in summary of p-values and ratios of variances for  aorta
Reading in summary of p-values and ratios of variances for  bladder-lumen
Reading in summary of p-values and ratios of variances for  bone-marrow
Reading in summary of p-values and ratios of variances for  brain-myeloid
Reading in summary of p-values and ratios of variances for  brown-adipose
-tissue
Reading in summary of p-values and ratios of variances for  diaphragm
Reading in summary of p-values and ratios of variances for  gonadal-fat-p
ad
Reading in summary of p-values and ratios of variances for  heart
Reading in summary of p-values and ratios of variances for  kidney
Reading in summary of p-values and ratios of variances for  large-intesti
ne
Reading in summary of p-values and ratios of variances for  limb-muscle
Reading in summary of p-values and ratios of variances for  liver
Reading in summary of p-values and ratios of variances for  lung
Reading in summary of p-values and ratios of variances for  mammary-gland
Reading in summary of p-values and ratios of variances for  mesenteric-fa
t-pad
Reading in summary of p-values and ratios of variances for  pancreas
Reading in summary of p-values and ratios of variances for  skin-of-body
```

```
Reading in summary of p-values and ratios of variances for  spleen
Reading in summary of p-values and ratios of variances for  subcutaneous-
adipose-tissue
Reading in summary of p-values and ratios of variances for  thymus
Reading in summary of p-values and ratios of variances for  tongue
Reading in summary of p-values and ratios of variances for  trachea
```

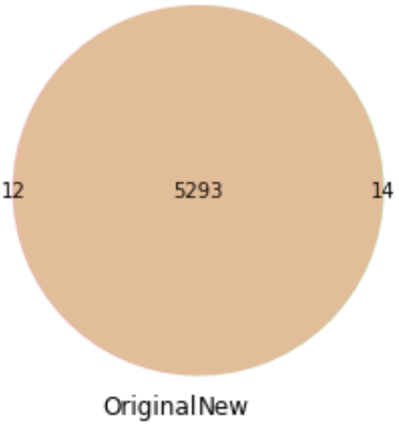
In [25]:
```
set1 = set(og_transcript_3_18['TRANSCRIPT'] + "_" + og_transcript_3_18['TIS
set2 = set(tissue_transcript_3_18['TRANSCRIPT'] + "_" + tissue_transcript_3
venn2([set1, set2], set_labels = ('Original', 'New'))
```

Out[25]: <matplotlib_venn._common.VennDiagram at 0x7f7e00269c10>



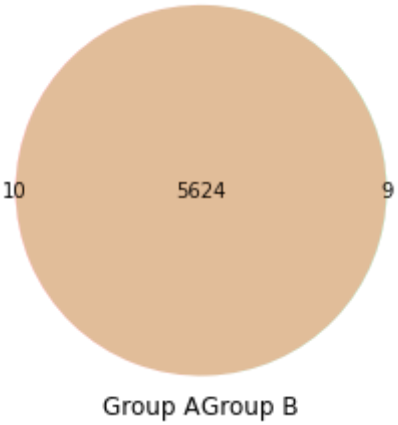
In [26]:
```
# Compare 24m vs 3m
set1 = set(og_transcript_18_24['TRANSCRIPT'] + "_" + og_transcript_18_24['T
set2 = set(tissue_transcript_18_24['TRANSCRIPT'] + "_" + tissue_transcript_
venn2([set1, set2], set_labels = ('Original', 'New'))
```

Out[26]: <matplotlib_venn._common.VennDiagram at 0x7f7e01e4bfd0>



In [27]:
```
# Compare 24m vs 3m
set1 = set(og_transcript_24_3['TRANSCRIPT'] + "_" + og_transcript_24_3['TIS
set2 = set(tissue_transcript_24_3['TRANSCRIPT'] + "_" + tissue_transcript_2
venn2([set1, set2], set_labels = ('Group A', 'Group B'))
```

Out[27]:
```
<matplotlib_venn._common.VennDiagram at 0x7f7e00418bb0>
```



We see that there are very few original Mann-Whitney DEGs that are no longer significant after tie breaking, and conversely there are also very few new Mann-Whitney DEGs that were originally non-significant. This suggests that the tie-breaking procedure hardly affected the gene expression distributions between age groups.

3 Analysis

We examine more closely the differences between Mann-Whitney DEGs and MOCHIS DEGs. Recall that Mann-Whitney DEGs are genes that are typically picked up by standard differential analysis routines, whereas MOCHIS DEGs are genes that are differentially expressed owing to shifts in dispersion. Below, we perform some analyses to answer the following questions.

• How many MOCHIS DEGs were previously not detected by Mann-Whitney?

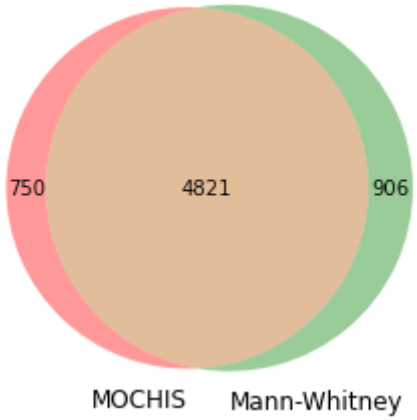
• Does MOCHIS really pick up shifts in dispersion?

• Are there other interesting questions we may answer with our newly detected MOCHIS DEGs?

In [28]:
```
## Compare counts
# Load original DEGs from Section 2.1

og_transcript_3_18 = pd.read_csv("tissues/mw_sig_3m_18m.csv")
og_transcript_18_24 = pd.read_csv("tissues/mw_sig_18m_24m.csv")
og_transcript_24_3 = pd.read_csv("tissues/mw_sig_24m_3m.csv")

# Load MOCHIS DEGs from Section 2.2
tissue_transcript_3_18 = pd.read_csv("tissues/mochis_sig_3m_18m.csv")
tissue_transcript_18_24 = pd.read_csv("tissues/mochis_sig_18m_24m.csv")
tissue_transcript_24_3 = pd.read_csv("tissues/mochis_sig_24m_3m.csv")
```

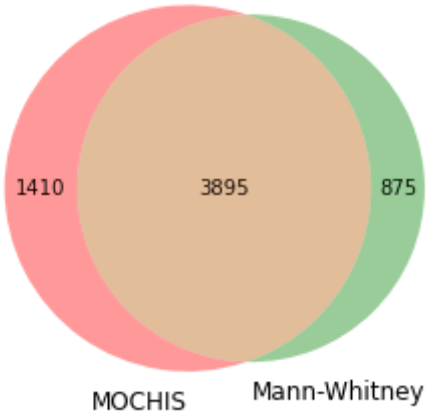
In [29]:
```
# Compare 3m vs 18m
set1 = set(og_transcript_3_18['TRANSCRIPT'] + "_" + og_transcript_3_18['TIS
set2 = set(tissue_transcript_3_18['TRANSCRIPT'] + "_" + tissue_transcript_3
venn2([set1, set2], set_labels = ('MOCHIS', 'Mann-Whitney'))
```

Out[29]:
```
<matplotlib_venn._common.VennDiagram at 0x7f7e0742fbe0>
```



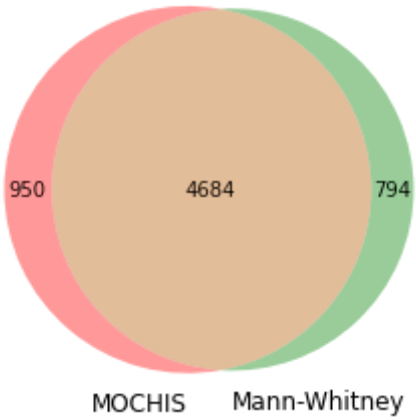
In [30]:
```
# Compare 18m vs 24m
set1 = set(og_transcript_18_24['TRANSCRIPT'] + "_" + og_transcript_18_24['T
set2 = set(tissue_transcript_18_24['TRANSCRIPT'] + "_" + tissue_transcript_
venn2([set1, set2], set_labels = ('MOCHIS', 'Mann-Whitney'))
```

Out[30]:
```
<matplotlib_venn._common.VennDiagram at 0x7f7e000e1a00>
```



In [31]:
```
# Compare 3m vs 24m
set1 = set(og_transcript_24_3['TRANSCRIPT'] + "_" + og_transcript_24_3['TIS
set2 = set(tissue_transcript_24_3['TRANSCRIPT'] + "_" + tissue_transcript_2
venn2([set1, set2], set_labels = ('MOCHIS', 'Mann-Whitney'))
```

Out[31]:
```
<matplotlib_venn._common.VennDiagram at 0x7f7e076a7af0>
```



Summary of Findings

1. In general, there are considerable differences in the genes picked up by Mann-Whitney and MOCHIS. For any pair of age groups, MOCHIS picks up at least 750 DEGs that were not picked up by Mann-Whitney.
2. The number of new genes picked up by MOCHIS is the largest for the pair “3m vs 18m” (= 905), and smallest for the pair “3m vs 21m” (= 794).

3. The number of Mann-Whitney significant genes that are not MOCHIS significant is greatest for the pair “18m vs 24m” (= 1397) and smallest for the pair “3m vs 18m” (= 753).

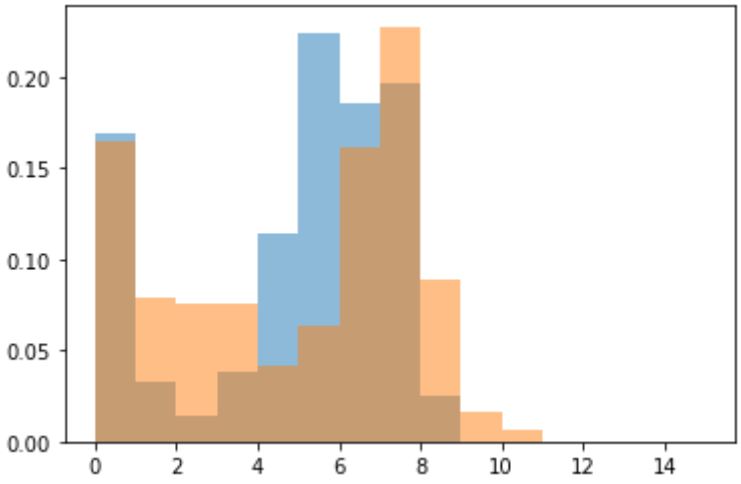
3.2 Visualizing Changes in Dispersion

The skeptical reader may wonder if MOCHIS is really picking up a shift in dispersion between the two age groups. Since we realistically cannot compare gene expression distributions between age groups for each MOCHIS significant gene, here we show some gene expression visualizations of MOCHIS significant genes. We focus on MOCHIS DEGs that were not detected by Mann-Whitney. We show visualizations for each pair of age groups (“3m vs 18m”, “18m vs 24m” and “3m vs 24m”).

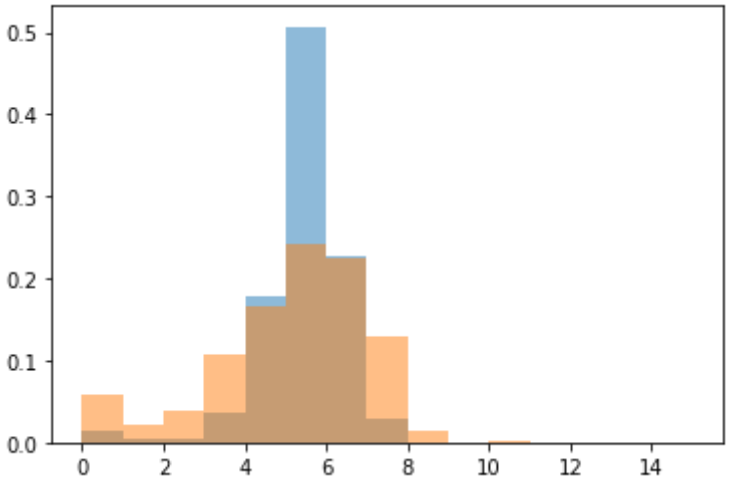
In [32]:
```
set1 = set(og_transcript_3_18['TRANSCRIPT'] + "_" + og_transcript_3_18['TIS
set2 = set(tissue_transcript_3_18['TRANSCRIPT'] + "_" + tissue_transcript_3
mochis_unique = pd.DataFrame()
for elem in set2:
    if elem not in set1:
        tc = elem.split("_")[0]
        ts = elem.split("_")[1]
        mochis_unique = pd.concat([mochis_unique,
                                   tissue_transcript_3_18.loc[(tissue_trans
        mochis_unique.index = [i for i in range(1, len(mochis_unique)+1)]

# Pick genes by hand (I choose the ones with biggest variance ratio in each
curated_degs_df = pd.DataFrame()
mu_aorta = mochis_unique[mochis_unique['TISSUE']=='aorta']
curated_degs_df = pd.concat([curated_degs_df, mu_aorta[mu_aorta['VARIANCE_R
mu_bladder_lumen = mochis_unique[mochis_unique['TISSUE']=='bladder-lumen']
curated_degs_df = pd.concat([curated_degs_df, mu_bladder_lumen[mu_bladder_l
mu_bone_marrow = mochis_unique[mochis_unique['TISSUE']=='bone-marrow']
curated_degs_df = pd.concat([curated_degs_df, mu_bone_marrow[mu_bone_marrow
mu_brain_myeloid = mochis_unique[mochis_unique['TISSUE']=='brain-myeloid']
curated_degs_df = pd.concat([curated_degs_df,mu_brain_myeloid[mu_brain_myel
mu_heart = mochis_unique[mochis_unique['TISSUE']=='heart']
curated_degs_df = pd.concat([curated_degs_df,mu_heart[mu_heart['VARIANCE_RA
mu_pancreas = mochis_unique[mochis_unique['TISSUE']=='pancreas']
mu_pancreas[mu_pancreas['VARIANCE_RATIO'] == max(mu_pancreas['VARIANCE_RATI
curated_degs_df = pd.concat([curated_degs_df,mu_pancreas[mu_pancreas['VARIA

# Generate plots
for i in range(len(curated_degs_df)):

    tissue = curated_degs_df.iloc[i]['TISSUE']
    transcript = curated_degs_df.iloc[i]['TRANSCRIPT']

    tissue_smartseq2_data = scanpy.read_h5ad('tissues/' + tissue + '.h5ad')
    transcripts = tissue_smartseq2_data.var.n_cells.index
    ages = np.array(tissue_smartseq2_data.obs['age'].values)
    smartseq2_raw_counts = tissue_smartseq2_data.raw.X.toarray()

    this_gene_exp_level = pd.DataFrame({
        'TRANSCRIPT': smartseq2_raw_counts[:, np.where(transcripts==transcr
        'AGE': ages
    })

    if tissue == 'bone-marrow':
        print(len(smartseq2_raw_counts))


    this_gene_exp_level_3m = this_gene_exp_level[this_gene_exp_level['AGE']
    this_gene_exp_level_18m = this_gene_exp_level[this_gene_exp_level['AGE'

    # Visualize


    bins = [i for i in range(16)]

    print(transcript + " in " + tissue)
```

```
    plt.hist([math.log(i+1) for i in this_gene_exp_level_3m['TRANSCRIPT'].v
    plt.hist([math.log(i+1) for i in this_gene_exp_level_18m['TRANSCRIPT'].
    plt.show()


    #plt.legend(loc='upper right')
```

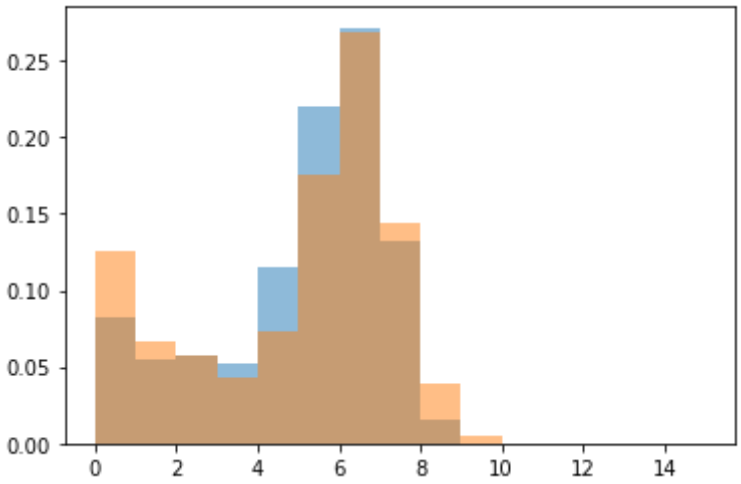
```
ENSMUSG00000032562 in aorta
```



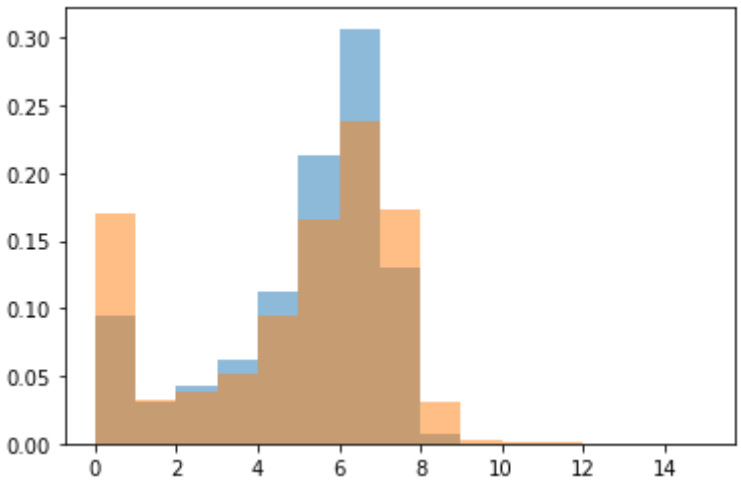

```
ENSMUSG00000020048 in bladder-lumen
```



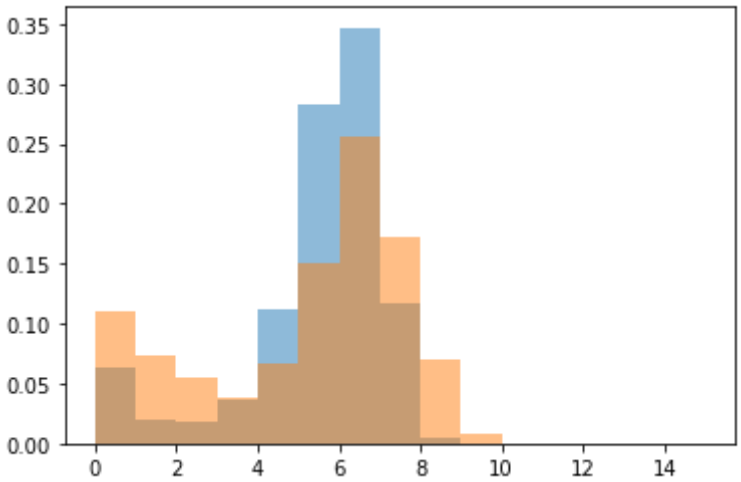

```
14517
ENSMUSG00000036438 in bone-marrow
```



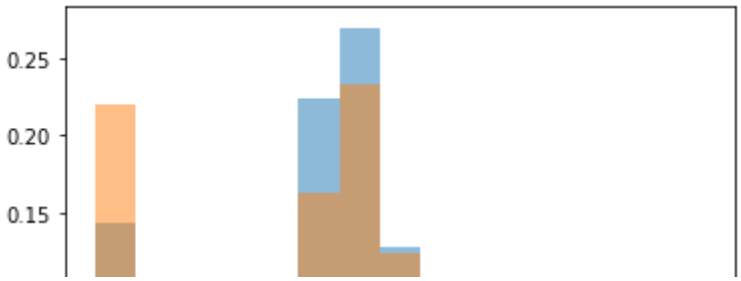
ENSMUSG00000029919 in brain-myeloid



ENSMUSG00000027523 in heart



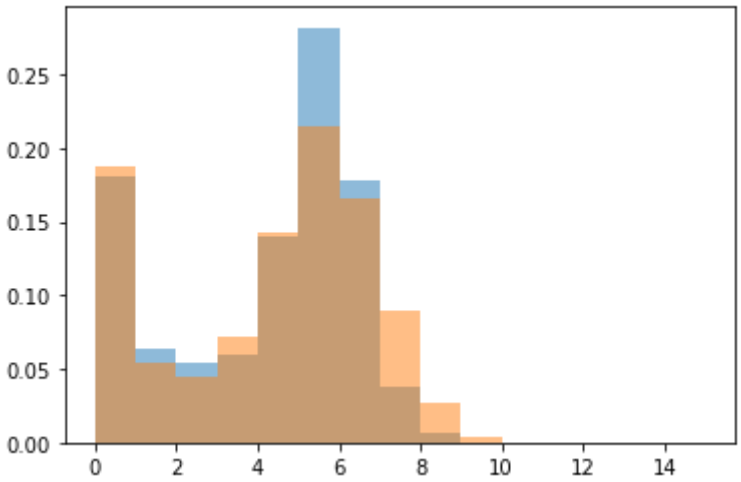
ENSMUSG00000027712 in pancreas


0.25
0.20
0.15

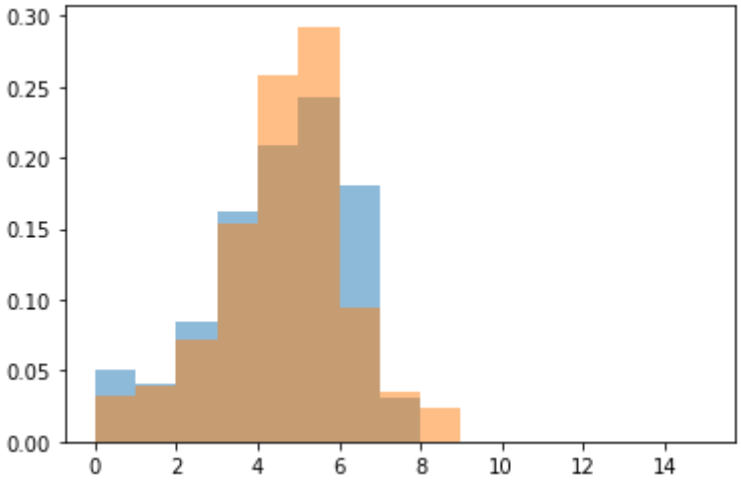
In [33]:
```
'TISSUE'])
ot_18_24['TISSUE'])




ranscript_18_24['TRANSCRIPT'] == tc) & (tissue_transcript_18_24['TISSUE'] ==


ach tissue)


E_RATIO'] == max(mu_aorta['VARIANCE_RATIO'])]])
 ]
r_lumen['VARIANCE_RATIO'] == max(mu_bladder_lumen['VARIANCE_RATIO'])]])

row['VARIANCE_RATIO'] == max(mu_bone_marrow['VARIANCE_RATIO'])]])

VARIANCE_RATIO'] == max(mu_diaphragm['VARIANCE_RATIO'])]])
tine']
_intestine['VARIANCE_RATIO'] == max(mu_large_intestine['VARIANCE_RATIO'])]])

ARIANCE_RATIO'])]
le['VARIANCE_RATIO'] == max(mu_limb_muscle['VARIANCE_RATIO'])]])









l')



script)[0][0]],



GE']=='18m']
GE']=='24m']









].values], bins=bins, density=True, alpha=0.5, label='x')
```

```
].values], bins=bins, density=True, alpha=0.5, label='x')
```

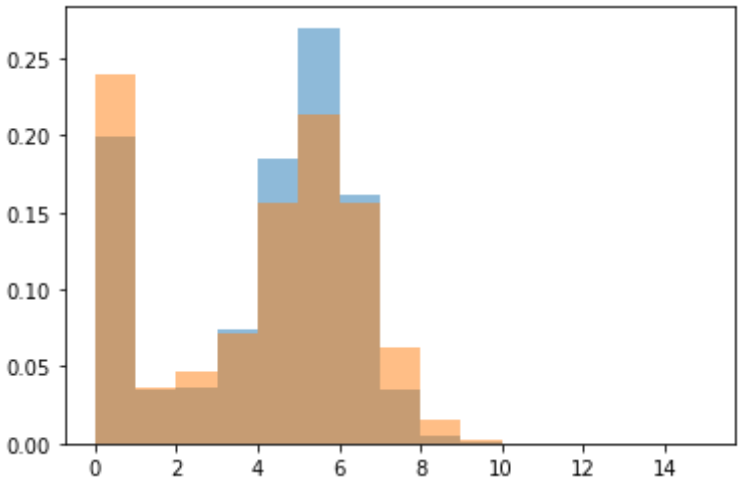
```
ENSMUSG00000058546 in aorta
```



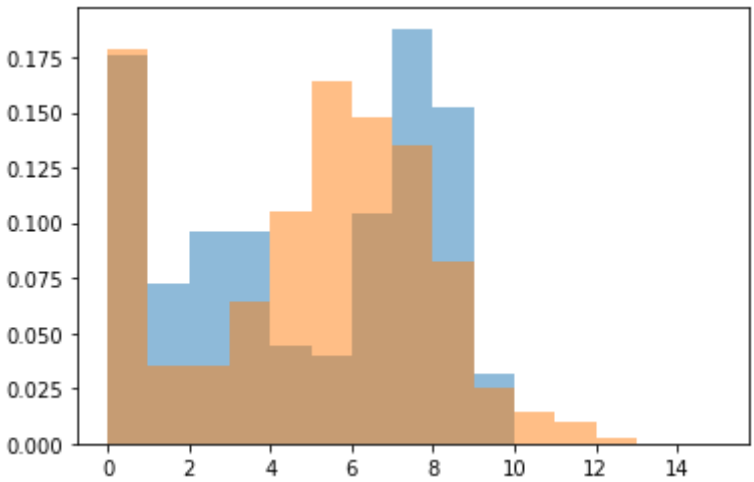

```
ENSMUSG00000018476 in bladder-lumen
```



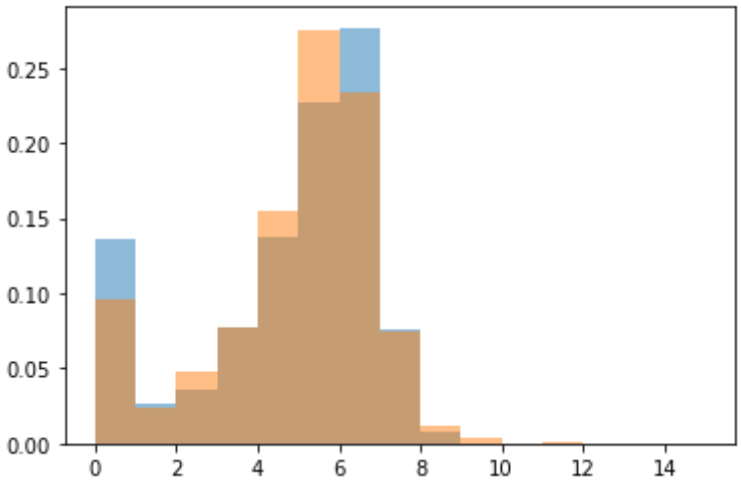

```
ENSMUSG00000022205 in bone-marrow
```



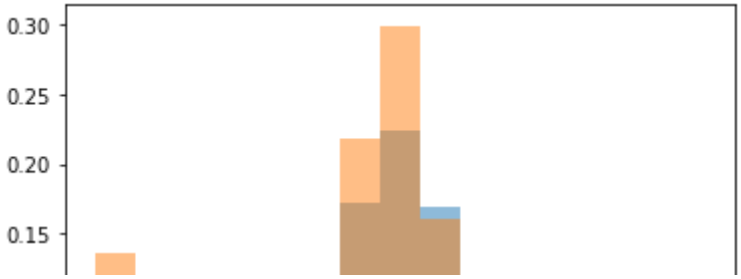
ENSMUSG00000071076 in diaphragm



ENSMUSG00000090862 in large-intestine



ENSMUSG00000025492 in limb-muscle

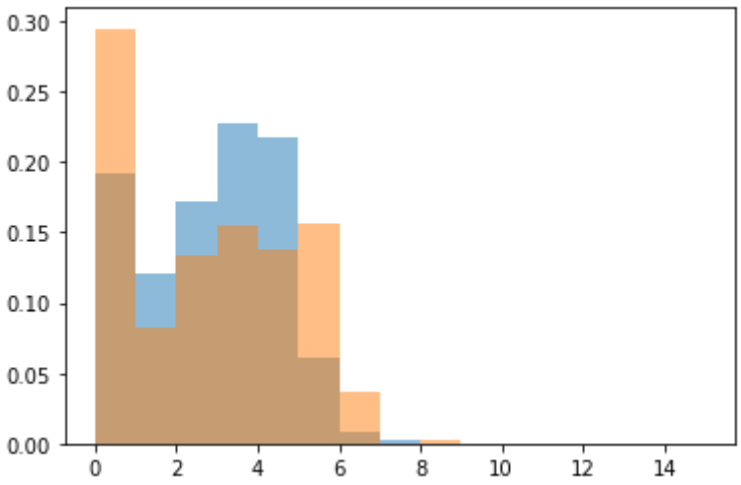

0.30
0.25
0.20
0.15

In [34]:

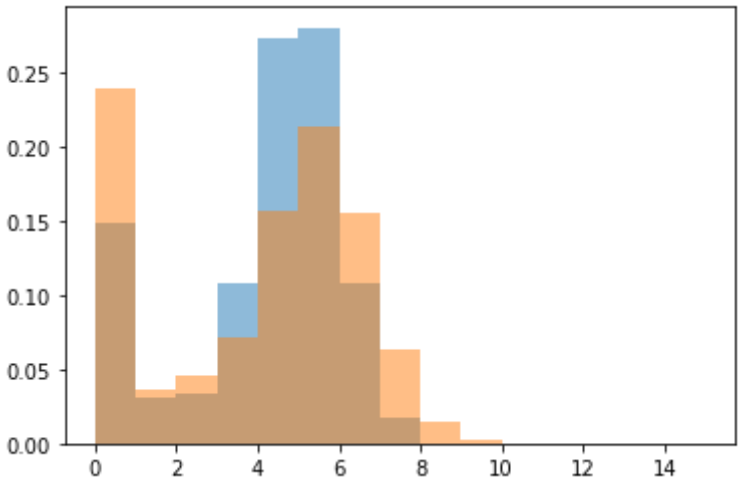
```
SUE'])




_3['TRANSCRIPT'] == tc) & (tissue_transcript_24_3['TISSUE'] == ts)]])




RIANCE_RATIO'] == max(mu_bladder_lumen['VARIANCE_RATIO'])]])

CE_RATIO'] == max(mu_bone_marrow['VARIANCE_RATIO'])]])

RIANCE_RATIO'] == max(mu_brain_myeloid['VARIANCE_RATIO'])]])

sue']
se_tissue['VARIANCE_RATIO'] == max(mu_brown_adipose_tissue['VARIANCE_RATIO'

== max(mu_spleen['VARIANCE_RATIO'])]])

== max(mu_thymus['VARIANCE_RATIO'])]])









0]],



]
```

```
bins=bins, density=True, alpha=0.5, label='x')
 bins=bins, density=True, alpha=0.5, label='x')
```

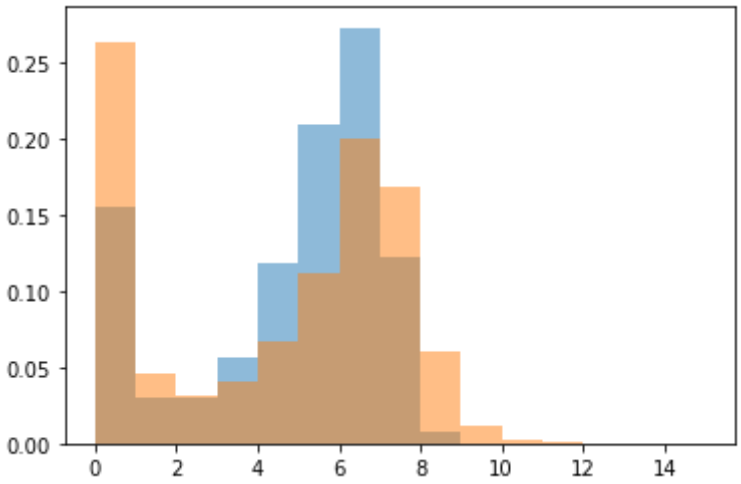
```
ENSMUSG00000020745 in bladder-lumen
```



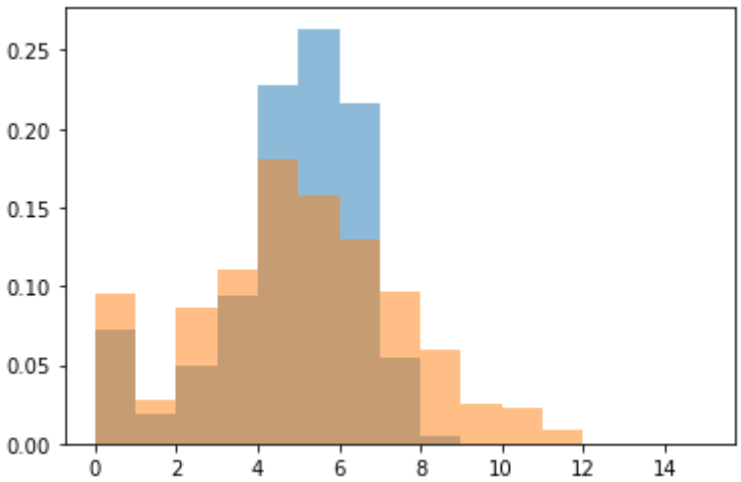

```
ENSMUSG00000022205 in bone-marrow
```



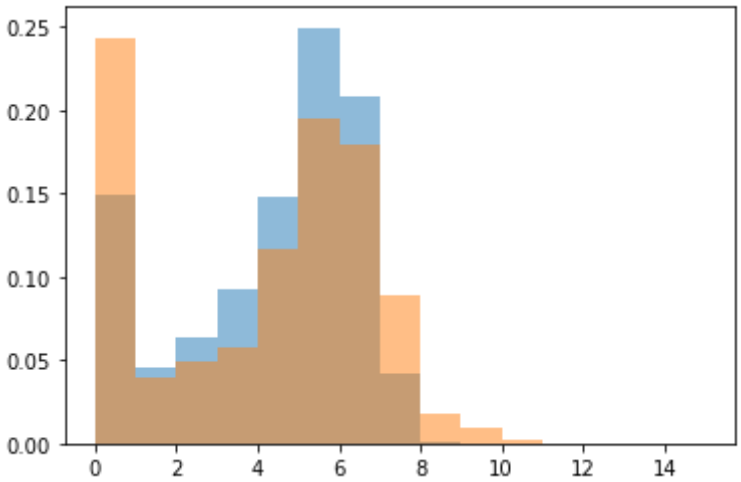

```
ENSMUSG00000000326 in brain-myeloid
```



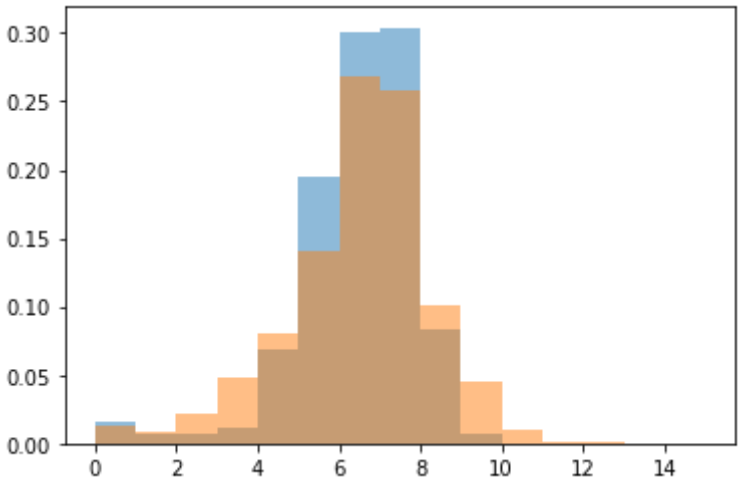
ENSMUSG00000056201 in brown-adipose-tissue



ENSMUSG00000030067 in spleen



ENSMUSG00000050708 in thymus



Summary of Findings

We find that

1. MOCHIS detects shifts in dispersions. These shifts can be in either direction (positive or negative).
2. Some of the shifts can be attributed to more pronounced zero inflation in one age group than another (based on post-analysis visualizations). This raises an important caveat in our analysis, namely, that our first step of filtering out genes that have more than 20% zero-inflation rate effectively removes all contribution by technical noise to the data. If we are skeptical, then we must find other ways to effectively remove contribution by technical noise.

3.3 Other Interesting Results

We show how we can further interpret our results to answer biologically meaningful questions. Gene Up-regulation vs Down-regulation. We have only looked at changes in dispersion, without explicitly tracking the directionality of change. We report, for each tissue and the corresponding pair of age groups, the fraction of positive and negative changes in dispersion, as measured by the ratio of variances.

In [41]:
```
## Report tissue-specific distribution of up-regulated
## and down-regulated genes between age groups
# Define all_tissues again

mochis_degs_3_18 = pd.read_csv("tissues/mochis_sig_3m_18m.csv")
mochis_degs_18_24 = pd.read_csv("tissues/mochis_sig_18m_24m.csv")
mochis_degs_3_24 = pd.read_csv("tissues/mochis_sig_24m_3m.csv")

up_down_reg_df = pd.DataFrame(columns = ['TISSUE',
                                         'DOWN_3_18',
                                         'UP_3_18',
                                         'DOWN_18_24',
                                         'UP_18_24',
                                         'DOWN_3_24',
                                         'UP_3_24'])

for tissue in all_tissues:

    results_df = pd.read_csv("tissues/" + tissue + "/mochis_p_val_table.csv

    # Analysis for 3m vs 18m
    # mochis_degs_3_18 %>% subset(TISSUE == tissue))$TRANSCRIPT
    mochis_degs_3_18_by_tissue = mochis_degs_3_18[mochis_degs_3_18['TISSUE'
    # subset(TRANSCRIPT %in% (mochis_degs_3_18 %>% subset(TISSUE == tissue)
    results_df_by_tissue = results_df[results_df['TRANSCRIPT'].isin(mochis_
    results_df_INV_3_18 = results_df_by_tissue['INV_3_18']
    n_3_18_down = sum(results_df_INV_3_18)
    n_3_18_up = len(results_df_by_tissue) - n_3_18_down



    # Analysis for 18m vs 24m
    # mochis_degs_3_18 %>% subset(TISSUE == tissue))$TRANSCRIPT
    mochis_degs_18_24_by_tissue = mochis_degs_18_24[mochis_degs_18_24['TISS
    # subset(TRANSCRIPT %in% (mochis_degs_3_18 %>% subset(TISSUE == tissue)
    results_df_by_tissue = results_df[results_df['TRANSCRIPT'].isin(mochis_
    results_df_INV_18_24 = results_df_by_tissue['INV_18_24']
    n_18_24_down = sum(results_df_INV_18_24)
    n_18_24_up = len(results_df_by_tissue) - n_18_24_down


    # Analysis for 3m vs 18m
    # mochis_degs_3_18 %>% subset(TISSUE == tissue))$TRANSCRIPT
    mochis_degs_3_24_by_tissue = mochis_degs_3_24[mochis_degs_3_24['TISSUE'
    # subset(TRANSCRIPT %in% (mochis_degs_3_18 %>% subset(TISSUE == tissue)
    results_df_by_tissue = results_df[results_df['TRANSCRIPT'].isin(mochis_
    results_df_INV_3_24 = results_df_by_tissue['INV_24_3']
    n_3_24_down = sum(results_df_INV_3_24)
    n_3_24_up = len(results_df_by_tissue) - n_3_24_down

    fig, (ax1, ax2, ax3) = plt.subplots(1, 3)

    fig.suptitle('Direction of Dispersion Shift for ' + tissue)
    ax1.pie([n_3_18_down,n_3_18_up])
    ax2.pie([n_18_24_down,n_18_24_up])
    ax3.pie([n_24_3_down,n_24_3_up])
```

```
    plt.show()


    up_down_reg_df = pd.concat([up_down_reg_df, pd.DataFrame([{
        'TISSUE': tissue,
        'DOWN_3_18': n_3_18_down,
        'UP_3_18': n_3_18_up,
        'DOWN_18_24': n_18_24_down,
        'UP_18_24': n_18_24_up,
        'DOWN_3_24': n_3_24_down,
        'UP_3_24': n_3_24_up
    }])])
    #print(s)
```

Direction of Dispersion Shift for mammary-gland

Direction of Dispersion Shift for mesenteric-fat-pad

In [42]:
```
print("The following tissues tend to exhibit increased gene regulation from
up_down_reg_df[up_down_reg_df['UP_3_18']/(up_down_reg_df['UP_3_18'] + up_do
```

```
The following tissues tend to exhibit increased gene regulation from 3m t
o 18m:
```

Out[42]:
```
0                          aorta
0                 bladder-lumen
0                   bone-marrow
0                 brain-myeloid
0           brown-adipose-tissue
0                      diaphragm
0                gonadal-fat-pad
0                          heart
0                         kidney
0                large-intestine
0                    limb-muscle
0                          liver
0                           lung
0             mesenteric-fat-pad
0                       pancreas
0                   skin-of-body
0                         spleen
0    subcutaneous-adipose-tissue
0                         thymus
0                         tongue
0                        trachea
Name: TISSUE, dtype: object
```

In [43]:
```
print("The following tissues tend to exhibit increased gene regulation from
up_down_reg_df[up_down_reg_df['UP_18_24']/(up_down_reg_df['UP_18_24'] + up_
```

```
The following tissues tend to exhibit increased gene regulation from 18m
to 24m:
```

Out[43]:
```
0                          aorta
0                 bladder-lumen
0                   bone-marrow
0                 brain-myeloid
0           brown-adipose-tissue
0                          heart
0                         kidney
0                large-intestine
0                          liver
0                           lung
0             mesenteric-fat-pad
0                         spleen
0    subcutaneous-adipose-tissue
0                         thymus
Name: TISSUE, dtype: object
```

In [44]:
```
print("The following tissues tend to exhibit increased gene regulation from
up_down_reg_df[up_down_reg_df['UP_3_24']/(up_down_reg_df['UP_3_24'] + up_do
```

```
The following tissues tend to exhibit increased gene regulation from 3m t
o 24m:
```

Out[44]:
```
0                          aorta
0                 bladder-lumen
0                   bone-marrow
0                 brain-myeloid
0           brown-adipose-tissue
0                      diaphragm
0                gonadal-fat-pad
0                          heart
0                         kidney
0                large-intestine
0                    limb-muscle
0                          liver
0                           lung
0             mesenteric-fat-pad
0                       pancreas
0                  skin-of-body
0                         spleen
0    subcutaneous-adipose-tissue
0                         thymus
0                         tongue
0                        trachea
Name: TISSUE, dtype: object
```

Persistently Differentially Expressed Genes. It is possible that for some tissues, gene regulation is so dynamic over the lifecourse, manifesting in detectable gene expression changes across time. To this end, we consider DEGs that are persistently differentiated; these are genes that are differentially expressed among all pairs of age groups. For brevity, we shall refer to them as persistently DEGs. First, we ask how many such persistently DEGs there are.

In [56]:
```
## Identify persistently differentially expressed genes
# Find all unique tissue-transcript pairs in results
tissue_transcript_combined = pd.concat([
    tissue_transcript_3_18[['TISSUE', 'TRANSCRIPT']],
    tissue_transcript_18_24[['TISSUE', 'TRANSCRIPT']],
    tissue_transcript_24_3[['TISSUE', 'TRANSCRIPT']]
]).drop_duplicates()

# Compute the persistent DEGs
age3m_vs_age18m = []
age18m_vs_age24m = []
age3m_vs_age24m = []

for i in range(len(tissue_transcript_combined)):
    tissue_transcript_24_3_subset = tissue_transcript_24_3[(tissue_transcri
                                                            (tissue_transcrip
    if (len(tissue_transcript_24_3_subset)) > 0:
        age3m_vs_age24m.append(1)
    else:
        age3m_vs_age24m.append(0)

    tissue_transcript_3_18_subset = tissue_transcript_3_18[(tissue_transcri
                                                            (tissue_transcrip
    if (len(tissue_transcript_3_18_subset)) > 0:
        age3m_vs_age18m.append(1)
    else:
        age3m_vs_age18m.append(0)

    tissue_transcript_18_24_subset = tissue_transcript_18_24[(tissue_transc
                                                            (tissue_transcrip
    if (len(tissue_transcript_18_24_subset)) > 0:
        age18m_vs_age24m.append(1)
    else:
        age18m_vs_age24m.append(0)


tissue_transcript_combined['AGE3M_VS_AGE18M'] = age3m_vs_age18m
tissue_transcript_combined['AGE18M_VS_AGE24M'] = age18m_vs_age24m
tissue_transcript_combined['AGE3M_VS_AGE24M'] = age3m_vs_age24m
tissue_transcript_combined['PERSISTENCE'] = np.add(np.add(age3m_vs_age18m,

# How many persistently DEGs are there?
print("There are ",
            np.count_nonzero(tissue_transcript_combined['PERSISTENCE']==3)
            " persistently DEGs.")
```

```
There are  3567  persistently DEGs.
```

In [61]:
```
# Look closely at persistent DEGs
curated_persistent_degs = tissue_transcript_combined[tissue_transcript_comb
curated_persistent_degs
```

Out[61]:

	TISSUE	TRANSCRIPT	AGE3M_VS_AGE18M	AGE18M_VS_AGE24M	AGE3M_VS_AGE2
1	aorta	ENSMUSG00000090862	1	1	
3	aorta	ENSMUSG00000060743	1	1	
19	aorta	ENSMUSG00000004207	1	1	
21	aorta	ENSMUSG00000003970	1	1	
25	aorta	ENSMUSG00000025794	1	1	
...	...	...	...	...	
5721	trachea	ENSMUSG00000015656	1	1	
5723	trachea	ENSMUSG00000023010	1	1	
5724	trachea	ENSMUSG00000092341	1	1	
5725	trachea	ENSMUSG00000060636	1	1	
5726	trachea	ENSMUSG00000074884	1	1	

3567 rows × 6 columns

To underscore the utility of our method at picking up DEGs that would otherwise be overlooked, we look at persistently DEGs (if there are any!) that would not be picked up by Mann-Whitney.

In [67]:
```
## Find and visualize some persistently DEGs
## that are not Mann-Whitney significant
# Identify those in MOCHIS but not Mann-Whitney
set1 = set(og_transcript_3_18['TRANSCRIPT'] + "_" + og_transcript_3_18['TIS
set2 = set(tissue_transcript_3_18['TRANSCRIPT'] + "_" + tissue_transcript_3
mochis_unique_3_18 = pd.DataFrame()
for elem in set2:
    if elem not in set1:
        tc = elem.split("_")[0]
        ts = elem.split("_")[1]
        mochis_unique_3_18 = pd.concat([mochis_unique_3_18,
                                        tissue_transcript_3_18.loc[(tissue_trans



set1 = set(og_transcript_18_24['TRANSCRIPT'] + "_" + og_transcript_18_24['T
set2 = set(tissue_transcript_18_24['TRANSCRIPT'] + "_" + tissue_transcript_
mochis_unique_18_24 = pd.DataFrame()
for elem in set2:
    if elem not in set1:
        tc = elem.split("_")[0]
        ts = elem.split("_")[1]
        mochis_unique_18_24 = pd.concat([mochis_unique_18_24,
                                        tissue_transcript_18_24.loc[(tissue_tran


set1 = set(og_transcript_24_3['TRANSCRIPT'] + "_" + og_transcript_24_3['TIS
set2 = set(tissue_transcript_24_3['TRANSCRIPT'] + "_" + tissue_transcript_2
mochis_unique_24_3 = pd.DataFrame()
for elem in set2:
    if elem not in set1:
        tc = elem.split("_")[0]
        ts = elem.split("_")[1]
        mochis_unique_24_3 = pd.concat([mochis_unique_24_3,
                                        tissue_transcript_24_3.loc[(tissue_trans









## Identify persistently differentially expressed genes
# Find all unique tissue-transcript pairs in results
tissue_transcript_combined = pd.concat([
    mochis_unique_3_18[['TISSUE', 'TRANSCRIPT']],
    mochis_unique_18_24[['TISSUE', 'TRANSCRIPT']],
    mochis_unique_24_3[['TISSUE', 'TRANSCRIPT']]
]).drop_duplicates()

# Compute the persistent DEGs
age3m_vs_age18m = []
age18m_vs_age24m = []
age3m_vs_age24m = []

for i in range(len(tissue_transcript_combined)):
    mochis_unique_24_3_subset = mochis_unique_24_3[(tissue_transcript_24_3[
```

```
                                                                (tissue_transcrip
    if (len(mochis_unique_24_3_subset)) > 0:
        age3m_vs_age24m.append(1)
    else:
        age3m_vs_age24m.append(0)

    mochis_unique_3_18_subset = mochis_unique_3_18[(tissue_transcript_3_18[
                                                                (tissue_transcrip
    if (len(mochis_unique_3_18_subset)) > 0:
        age3m_vs_age18m.append(1)
    else:
        age3m_vs_age18m.append(0)

    mochis_unique_18_24_subset = mochis_unique_18_24[(tissue_transcript_18_
                                                                (tissue_transcrip
    if (len(mochis_unique_18_24_subset)) > 0:
        age18m_vs_age24m.append(1)
    else:
        age18m_vs_age24m.append(0)


tissue_transcript_combined['AGE3M_VS_AGE18M'] = age3m_vs_age18m
tissue_transcript_combined['AGE18M_VS_AGE24M'] = age18m_vs_age24m
tissue_transcript_combined['AGE3M_VS_AGE24M'] = age3m_vs_age24m
tissue_transcript_combined['PERSISTENCE'] = np.add(np.add(age3m_vs_age18m,

# How many persistently DEGs are there?
print("There are ",
             np.count_nonzero(tissue_transcript_combined['PERSISTENCE']==3)
             " persistently DEGs not previously detected by Mann-Whitney.")
```

```
<ipython-input-67-8a77d8d7523f>:56: UserWarning: Boolean Series key will
be reindexed to match DataFrame index.
  mochis_unique_24_3_subset = mochis_unique_24_3[(tissue_transcript_24_3
['TISSUE']==tissue_transcript_combined.iloc[i]['TISSUE']) &
<ipython-input-67-8a77d8d7523f>:63: UserWarning: Boolean Series key will
be reindexed to match DataFrame index.
  mochis_unique_3_18_subset = mochis_unique_3_18[(tissue_transcript_3_18
['TISSUE']==tissue_transcript_combined.iloc[i]['TISSUE']) &
<ipython-input-67-8a77d8d7523f>:70: UserWarning: Boolean Series key will
be reindexed to match DataFrame index.
  mochis_unique_18_24_subset = mochis_unique_18_24[(tissue_transcript_18_
24['TISSUE']==tissue_transcript_combined.iloc[i]['TISSUE']) &

There are  66  persistently DEGs not previously detected by Mann-Whitney.
```

In [69]:
```
# Look closely at persistent DEGs
curated_persistent_degs = tissue_transcript_combined[tissue_transcript_comb
curated_persistent_degs
```

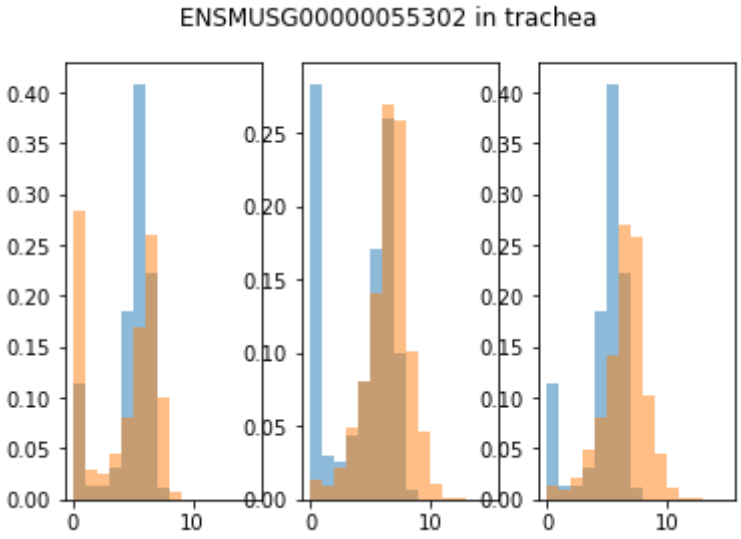
Out[69]:
```
1976                 heart
3472    mesenteric-fat-pad
534          bladder-lumen
1891                 heart
971          bladder-lumen
               ...
917          bladder-lumen
2497           limb-muscle
5627               trachea
5515               trachea
998          bladder-lumen
Name: TISSUE, Length: 66, dtype: object
```

From the chunk above, we find that 66 tissue-specific genes are persistently DEGs. Moreover, most of them come from the bladder lumen and the heart. Let us visualize some of these genes.

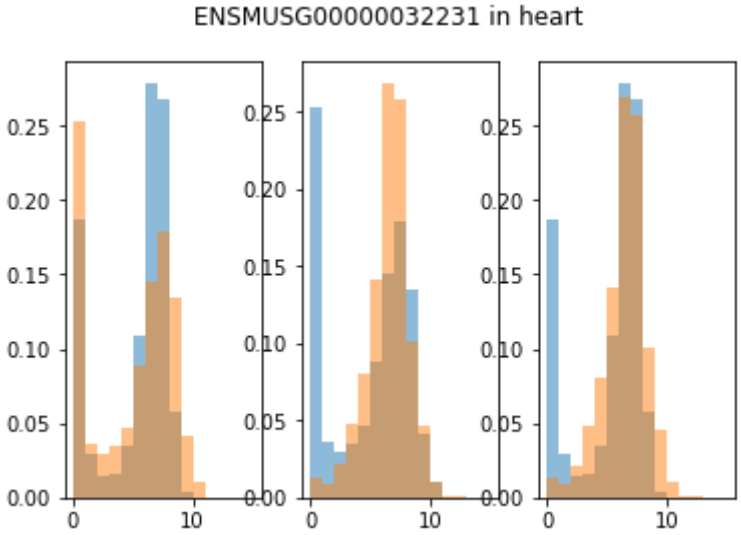
In [ ]:

In [93]:
```
## Visualizing MOCHIS-exclusive persistently DEGs
# Generate plots of gene expressions for persistent DEGs
plot_list = []

for i in [7,17,28]:

    # Get raw read counts data for that tissue and transcript
    tissue = curated_persistent_degs.iloc[i]['TISSUE']
    transcript = curated_persistent_degs.iloc[i]['TRANSCRIPT']

    # Create local list
    this_deg_plot_list = []
    # Open tissue-specific data and select gene
    tissue_smartseq2_data = scanpy.read_h5ad('tissues/' + tissue + '.h5ad')
    transcripts = tissue_smartseq2_data.var.n_cells.index
    ages = np.array(tissue_smartseq2_data.obs['age'].values)
    smartseq2_raw_counts = tissue_smartseq2_data.raw.X.toarray()

    this_gene_exp_level = pd.DataFrame({
        'TRANSCRIPT': smartseq2_raw_counts[:, np.where(transcripts==transcr
        'AGE': ages
    })


    this_gene_exp_level_3m = this_gene_exp_level[this_gene_exp_level['AGE']
    this_gene_exp_level_18m = this_gene_exp_level[this_gene_exp_level['AGE'
    this_gene_exp_level_18m = this_gene_exp_level[this_gene_exp_level['AGE'

    # Visualize


    bins = [i for i in range(16)]

    fig, (ax1, ax2, ax3) = plt.subplots(1, 3)

    fig.suptitle(transcript + " in " + tissue)
    ax1.hist([math.log(i+1) for i in this_gene_exp_level_3m['TRANSCRIPT'].v
    ax1.hist([math.log(i+1) for i in this_gene_exp_level_18m['TRANSCRIPT'].

    ax2.hist([math.log(i+1) for i in this_gene_exp_level_18m['TRANSCRIPT'].
    ax2.hist([math.log(i+1) for i in this_gene_exp_level_24m['TRANSCRIPT'].

    ax3.hist([math.log(i+1) for i in this_gene_exp_level_3m['TRANSCRIPT'].v
    ax3.hist([math.log(i+1) for i in this_gene_exp_level_24m['TRANSCRIPT'].

    plt.show()
```

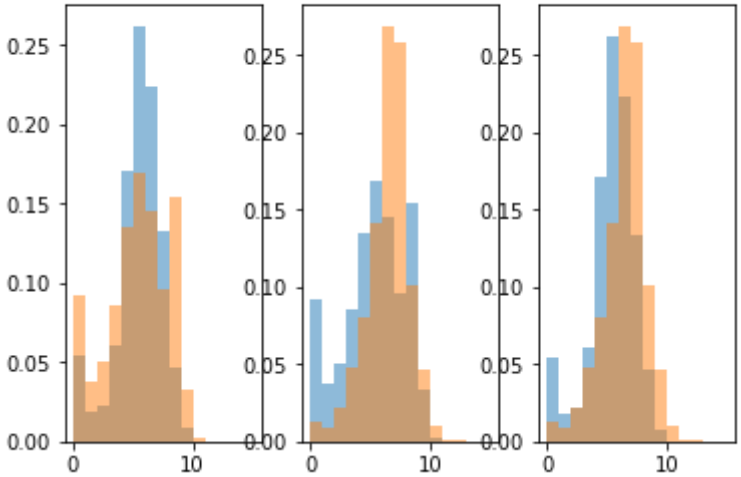
7


ENSMUSG00000055302 in trachea

17


ENSMUSG00000032231 in heart

28


ENSMUSG00000066551 in thymus

In [119]:
```
## Generate dataframe summarizing direction of dispersion changes
persistent_degs_direction_df = pd.DataFrame(columns=['TISSUE', 'TRANSCRIPT'

for i in range(len(curated_persistent_degs)):
    tissue = curated_persistent_degs.iloc[i]['TISSUE']
    transcript = curated_persistent_degs.iloc[i]['TRANSCRIPT']

    results_df = pd.read_csv("tissues/" + tissue + "/mochis_p_val_table.csv

    relevant_row = results_df[results_df['TRANSCRIPT']==transcript]


    inv_3_18 = '+'
    inv_18_24 = '+'
    inv_3_24 = '+'
    if relevant_row['INV_3_18'].values[0]:
        inv_3_18 = "-"
    if relevant_row['INV_18_24'].values[0]:
        inv_18_24 = "-"
    if relevant_row['INV_24_3'].values[0]:
        inv_3_24 = "-"


    persistent_degs_direction_df = pd.concat([persistent_degs_direction_df,
        'TISSUE': tissue,
        'TRANSCRIPT': transcript,
        'AGE3M_VS_AGE18M': inv_3_18,
        'AGE18M_VS_AGE24M': inv_18_24,
        'AGE3M_VS_AGE24M': inv_3_24
    }])]).sort_values('TISSUE')
```

In [120]:
```
persistent_degs_direction_df
```

Out[120]:

	TISSUE	TRANSCRIPT	AGE3M_VS_AGE18M	AGE18M_VS_AGE24M	AGE3M_VS_AGE24M
0	bladder-lumen	ENSMUSG00000037373	+	-	+
0	bladder-lumen	ENSMUSG00000059208	+	-	+
0	bladder-lumen	ENSMUSG00000028757	+	-	+
0	bladder-lumen	ENSMUSG00000030824	+	-	+
0	bladder-lumen	ENSMUSG00000013160	+	+	+
...	...	...	...	...	...
0	trachea	ENSMUSG00000060743	+	-	+
0	trachea	ENSMUSG00000055302	+	-	+
0	trachea	ENSMUSG00000026234	+	-	+
0	trachea	ENSMUSG00000024190	+	-	+
0	trachea	ENSMUSG00000025428	+	-	+

66 rows × 5 columns

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:
```
plt.pie([1,2])
```

In [ ]: