In [1]:
```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import pandas as pd
import rpy2.robjects as robjects
from rpy2.robjects import pandas2ri

# install the R packages
if os.path.exists('Rinstall.success'):
    print('R packages already installed')
else:
    robjects.r('''
        install.packages("pheatmap")
        install.packages("ggplot2")
        if (!requireNamespace("BiocManager", quietly = TRUE))
            install.packages("BiocManager")
        BiocManager::install("DESeq2")
        BiocManager::install("limma")
        BiocManager::install("clusterProfiler")

    ''')
    open('Rinstall.success', 'w').close()
```

```
R packages already installed
```

Bulk-RNA Sequencing Data from Different Body Parts from Tiphane (Kronauer Lab)

Goal: To verify that candidate genes are expressed specifically in the head of Ooceraea biroi.

In [2]:
```
# get gene lit review data
gene_lit_review = pd.read_excel('../../../Genes_LitReview.xlsx')
gene_lit_review.head()
```

Out[2]:

	Cell Class	Status	Gene Identifier	Reference	Drosphila Notes	Dmelanogaster ID	OrthoDB Search Found?	Obiroi Ortholog ID	Obiroi Fasta	Genome Investigation with Kip	BlastP with Drosophila	BlastP with Formicidae	Tissue Specifcity with Tiphane	Anindita for Neuron Specificity	D E
0	Dopaminergic Neurons	In verification	ple (pale)	Mao, Zhengmei, and Ronald L. Davis. "Eight dif...	Also known as DTH or TH. More in PPLs	FBgn0005626	yes	LOC105279021	>2015173_0:001f11 {"organism":"Ooceraea biroi"...	looks good	NaN	NaN	NaN	NaN	
1	Dopaminergic Neurons	In verification	Vmat (vesicular monoamine transporter)	Li, Qiye, et al. "A single-cell transcriptomic...	NaN	FBgn0260964	yes, multiple	LOC105288016	>2015173_0:0018c3 {"organism":"Ooceraea biroi"...	looks good	NaN	NaN	NaN	NaN	
2	Dopaminergic Neurons	In verification	Vmat (vesicular monoamine transporter)	Li, Qiye, et al. "A single-cell transcriptomic...	NaN	FBgn0260964	yes, multiple	LOC105274983	>2015173_0:0018f9 {"organism":"Ooceraea biroi"...	looks good	NaN	NaN	NaN	NaN	
3	Dopaminergic Neurons	In verification	DAT (dopamine transporter)	Li, Qiye, et al. "A single-cell transcriptomic...	Also known as R58E02, so more in PAMs	FBgn0034136	yes	LOC105277848	>2015173_0:0004c3 {"organism":"Ooceraea biroi"...	looks good	NaN	NaN	NaN	NaN	
4	Dopaminergic Neurons	In verification	ddc (dopa-decaboxylase)	Aso, Yoshinori, et al. "Three dopamine pathway...	More in PAMs?	FBgn0000422	yes, multiple	LOC105285288	>2015173_0:002674 {"organism":"Ooceraea biroi"...	looks good	NaN	NaN	NaN	NaN	

In [3]:
```
# get ortholog IDs
genes_of_interest = gene_lit_review[['Gene Identifier','Obiroi Ortholog ID']].dropna().astype(str)

IDs = genes_of_interest['Gene Identifier'].apply(lambda x: x.split(' ')[0]).values
LOCs = genes_of_interest['Obiroi Ortholog ID'].apply(lambda x: x.strip()).values

# for duplicate IDs, enumerate them to avoid overwriting
ID_counts = pd.Series(IDs).value_counts()
ID_counts = ID_counts[ID_counts>1]
for ID in ID_counts.index:
    idxs = np.where(IDs == ID)[0]
    for i, idx in enumerate(idxs):
        IDs[idx] = ID + '-' + str(i+1)
```

In [4]:
```
# get data folder
data_folder = "../../data/" + os.path.basename(os.getcwd()) + "/"

# find the .RData file
salmon_file = list(filter(lambda x: x.endswith('.RData'), os.listdir(data_folder)))[0]

# find the .xlsx file
norm_counts_file = list(filter(lambda x: x.endswith('.xlsx'), os.listdir(data_folder)))[0]
```

Analysis of Normalized Counts

In [5]:
```
# load the xlsx file
norm_counts = pd.read_excel(data_folder + norm_counts_file)
# make all column names uppercase
norm_counts.columns = norm_counts.columns.str.upper()
norm_counts.head()
```

Out[5]:

	GENE_ID	DG1	DG2	DG3	DG4	LEGS5	LEGS6	LEGS7	LEGS8	MG1	...	RESTREC3	RESTREC4	THO5	
0	Csp1	323.264945	515.233704	340.021650	1025.545657	743.020390	943.999424	757.880819	871.956102	2283.726887	...	384.181805	338.376772	593.441900	523.4
1	Csp10	7798.355782	35862.039270	14775.591250	17562.545430	8911.835161	9571.659496	9198.747413	11056.031550	10203.526880	...	34841.790140	41365.187650	1980.625890	4898.
2	Csp11	3438.641086	4206.924652	1775.984998	3417.255565	382.210529	381.703942	421.318384	384.845946	1029.651940	...	394.548177	529.430059	133.734036	227.6
3	Csp12	79.802984	337.898950	141.686018	856.828728	46680.219810	43899.001070	49296.564120	50721.552220	1924.140955	...	2543.396497	1679.045621	332.546821	524.5
4	Csp13	39.632916	96.235215	10.302762	20.213897	6902.037238	6404.716910	7532.434007	7000.742136	12393.536430	...	2121.593613	2424.529907	2571.419214	3363.8

5 rows × 49 columns

In [6]:
```
# get genes that dont start with 'LOC'
gene_ids = norm_counts['GENE_ID'].values

# make sure all genes of interest (from lit review) are in the data
missing_genes = []
for LOC in LOCs:
    if LOC not in gene_ids:
        missing_genes.append(LOC)

if len(missing_genes) > 0:
    print('Missing genes:', missing_genes)
else:
    print('All genes found in data')
```

All genes found in data

In [7]:
```
# set the index to the first column
norm_counts.set_index('GENE_ID', inplace=True)
```

In [8]:
```
head_columns = list(filter(lambda x: 'RESTMG' in x or 'RESTPPG' in x, norm_counts.columns))
other_columns = list(filter(lambda x: 'RESTMG' not in x and 'RESTPPG' not in x, norm_counts.columns))
# remove the numbers from the column names and make them uppercase
tissue_columns = [''.join([i for i in s if not i.isdigit()]) for s in norm_counts.columns]
# create a sample_info dataframe
sample = list(norm_counts.columns)
print(sample)
group = ['head' if column in head_columns else 'other' for column in norm_counts.columns]
print(group)
sample_info = pd.DataFrame({'sample': sample, 'group': group})
```

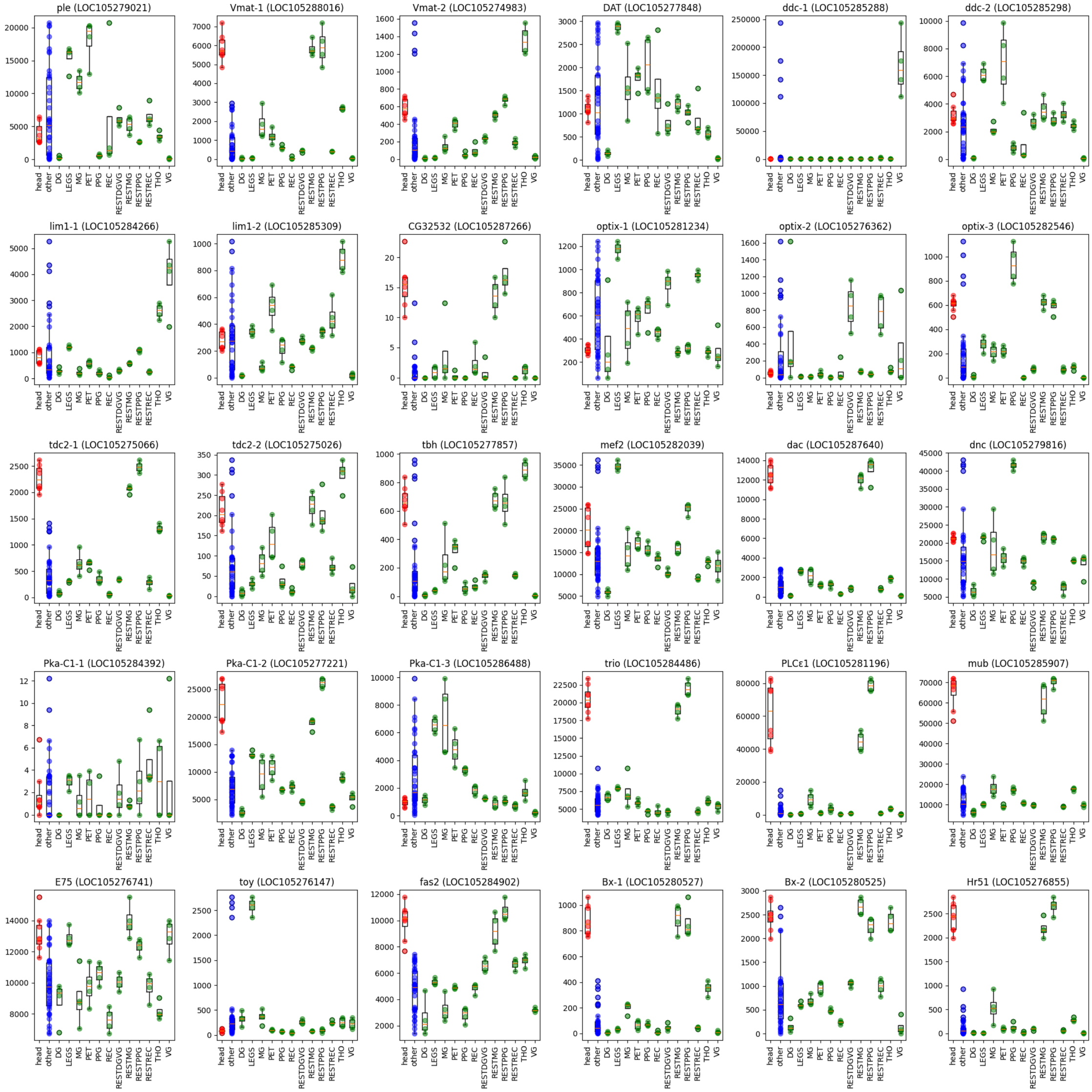
```
['DG1', 'DG2', 'DG3', 'DG4', 'LEGS5', 'LEGS6', 'LEGS7', 'LEGS8', 'MG1', 'MG2', 'MG3', 'MG4', 'PET5', 'PET6', 'PET7', 'PET8', 'PPG1', 'PPG2', 'PPG3', 'PPG4', 'REC1', 'REC2', 'R
EC3', 'REC4', 'RESTDGVG1', 'RESTDGVG2', 'RESTDGVG3', 'RESTDGVG4', 'RESTMG1', 'RESTMG2', 'RESTMG3', 'RESTMG4', 'RESTPPG1', 'RESTPPG2', 'RESTPPG3', 'RESTPPG4', 'RESTREC1', 'REST
REC2', 'RESTREC3', 'RESTREC4', 'THO5', 'THO6', 'THO7', 'THO8', 'VG1', 'VG2', 'VG3', 'VG4']
['other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'ot
her', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'head', 'head', 'head', 'head', 'head', 'head', 'head', 'head', 'other', 'other', 'other', 'othe
r', 'other', 'other', 'other', 'other', 'other', 'other', 'other', 'other']
```

Visualizing the Normalized Counts for the Candidate Genes

In [9]:
```
print("Number of Candidates: ", len(LOCs))
```

```
Number of Candidates:  30
```

In [10]:
```
# plot the normalized counts for the genes of interest between the head and other tissues
fig, ax = plt.subplots(5,6, figsize=(20,20))
for i,(ID, LOC) in enumerate(zip(IDs, LOCs)):
    if LOC in norm_counts.index:
        current_ax = ax.flatten()[i]
        # get norm counts for each tissue
        tissue_counts = []
        for tissue in np.unique(tissue_columns):
            tissue_counts.append(norm_counts.loc[LOC, [col for col in norm_counts.columns if tissue == "".join([i for i in col if not i.isdigit()])]])
        # plot the boxplot
        current_ax.boxplot([norm_counts.loc[LOC, head_columns], norm_counts.loc[LOC, ~norm_counts.columns.isin(head_columns)]] + tissue_counts)
        # set the xticks
        current_ax.set_xticklabels(['head', 'other'] + np.unique(tissue_columns).tolist(), rotation=90)
        #
        # plot the points
        current_ax.plot([1]*len(head_columns), norm_counts.loc[LOC, head_columns], 'ro', alpha=0.5)
        current_ax.plot([2]*len(other_columns), norm_counts.loc[LOC, ~norm_counts.columns.isin(head_columns)], 'bo', alpha=0.5)
        for i, tissue in enumerate(np.unique(tissue_columns)):
            current_ax.plot([3+i]*len(tissue_counts[i]), tissue_counts[i], 'go', alpha=0.5)
        current_ax.set_title(f"{ID} ({LOC})")
    else:
        print(f"{ID} ({LOC}) not found in normalized counts")
plt.tight_layout()
plt.show()
```



In [11]:

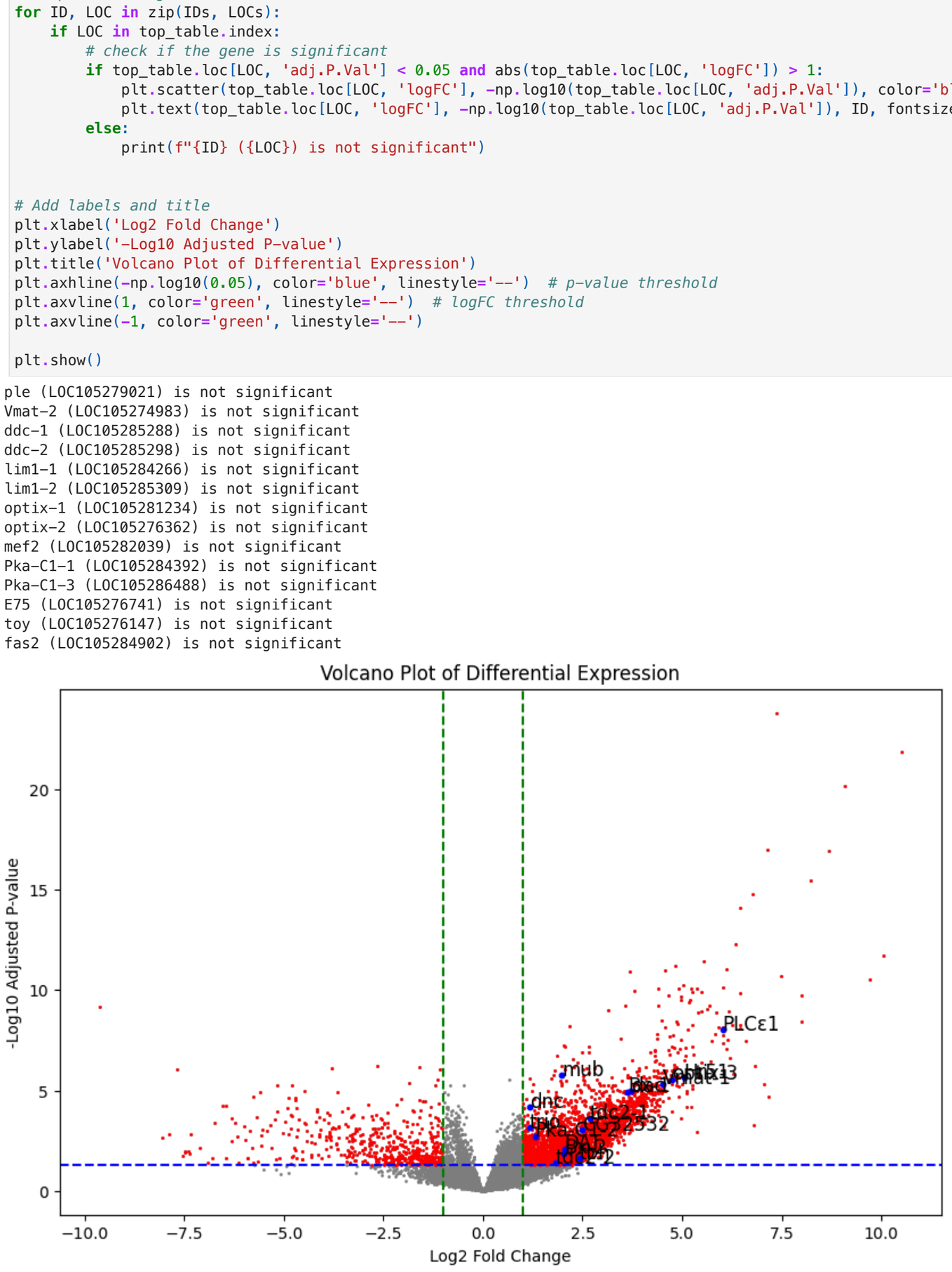
```python
# Activate the pandas to R DataFrame conversion
pandas2ri.activate()

# Pass the data to R
robjects.globalenv['data'] = pandas2ri.py2rpy(norm_counts)
robjects.globalenv['sample_info'] = pandas2ri.py2rpy(sample_info)

# R code to run limma-voom
robjects.r('''
library(limma)

# Convert the 'group' column to a factor
sample_info$group <- factor(sample_info$group)

# Create the design matrix
design <- model.matrix(~ 0 + group, data=sample_info)

# Voom transformation of normalized counts
v <- voom(data, design, plot=FALSE)

# Fit the linear model
fit <- lmFit(v, design)

# Contrast between head and other tissues
contrast.matrix <- makeContrasts(head_vs_other = grouphead - groupother, levels=design)

# Fit the contrast
fit2 <- contrasts.fit(fit, contrast.matrix)

# Apply empirical Bayes moderation
fit2 <- eBayes(fit2)

# Extract top differentially expressed genes
top_table <- topTable(fit2, adjust="fdr", number=Inf)

# Return the results to Python
top_table
''')

# Retrieve the results as a pandas DataFrame
top_table = pandas2ri.rpy2py(robjects.r['top_table'])
```

In [12]:
```
significant_genes = top_table[(top_table['adj.P.Val'] < 0.05) & (abs(top_table['logFC']) > 1)]
# Display the number of significant genes
print(f"Number of significant genes: {len(significant_genes)}")
print(significant_genes.head())
```

```
Number of significant genes: 2763
                  logFC   AveExpr          t       P.Value     adj.P.Val  \
Or5-L8         7.382776 -3.841739  23.857564  1.147524e-28  1.561780e-24
LOC105283508  10.502109  4.778391  21.273269  1.964363e-26  1.336749e-22
LOC105281007   9.085531  5.450897  19.259685  1.553300e-24  7.046802e-21
LOC105288220   8.693549  3.758919  15.994886  4.139127e-21  1.126670e-17
Or5-L11        7.147466 -3.344248  16.129348  2.926551e-21  9.957590e-18

                     B
Or5-L8       51.852059
LOC105283508 49.222853
LOC105281007 45.349469
LOC105288220 37.647313
Or5-L11      37.170116
```

In [13]:
```
import matplotlib.pyplot as plt
import numpy as np

# Create a volcano plot
plt.figure(figsize=(10, 6))

# Scatter plot of logFC vs -log10(p-value)
plt.scatter(top_table['logFC'], -np.log10(top_table['adj.P.Val']), color='gray', s=1)

# Highlight significant genes
sig_genes = top_table[(top_table['adj.P.Val'] < 0.05) & (abs(top_table['logFC']) > 1)]
plt.scatter(sig_genes['logFC'], -np.log10(sig_genes['adj.P.Val']), color='red', s=1)

# Loop over the genes of interest
for ID, LOC in zip(IDs, LOCs):
    if LOC in top_table.index:
        # check if the gene is significant
        if top_table.loc[LOC, 'adj.P.Val'] < 0.05 and abs(top_table.loc[LOC, 'logFC']) > 1:
            plt.scatter(top_table.loc[LOC, 'logFC'], -np.log10(top_table.loc[LOC, 'adj.P.Val']), color='blue', s=10)
            plt.text(top_table.loc[LOC, 'logFC'], -np.log10(top_table.loc[LOC, 'adj.P.Val']), ID, fontsize=12)
        else:
            print(f"{ID} ({LOC}) is not significant")


# Add labels and title
plt.xlabel('Log2 Fold Change')
plt.ylabel('-Log10 Adjusted P-value')
plt.title('Volcano Plot of Differential Expression')
plt.axhline(-np.log10(0.05), color='blue', linestyle='--')  # p-value threshold
plt.axvline(1, color='green', linestyle='--')  # logFC threshold
plt.axvline(-1, color='green', linestyle='--')

plt.show()
```

```
ple (LOC105279021) is not significant
Vmat-2 (LOC105274983) is not significant
ddc-1 (LOC105285288) is not significant
ddc-2 (LOC105285298) is not significant
lim1-1 (LOC105284266) is not significant
lim1-2 (LOC105285309) is not significant
optix-1 (LOC105281234) is not significant
optix-2 (LOC105276362) is not significant
mef2 (LOC105282039) is not significant
Pka-C1-1 (LOC105284392) is not significant
Pka-C1-3 (LOC105286488) is not significant
E75 (LOC105276741) is not significant
toy (LOC105276147) is not significant
fas2 (LOC105284902) is not significant
```



Analysis of Differential Expression using Salmon and DESeq2

In [14]:
```
# get absolute path of the salmon file
salmon_file = os.path.abspath(data_folder + salmon_file)
print(salmon_file)
```

```
/Users/neurorishika/Projects/Rockefeller/Kronauer/Transgenic Ants/Candidate Genes/rna-seq-analysis/data/Tiphane_Brain-Body_RNA_Data/dds.RData
```

In [15]:
```
# Load the DESeq2 library and load the dds object
import rpy2.robjects as robjects
from rpy2.robjects import pandas2ri
import pandas as pd

# Activate automatic conversion between pandas DataFrames and R data frames
pandas2ri.activate()

robjects.r('''
    library(DESeq2)
    load(''' + f'"{salmon_file}"' + ''')

    # Create tissue group (head vs other)
    sample_names <- colnames(dds)
    tissue_group <- ifelse(grepl("restMG|restPPG", sample_names), "head", "other")
    colData(dds)$tissue <- factor(tissue_group)
```

```
    # Update design formula and run DESeq2
    design(dds) <- ~ tissue
    dds <- DESeq(dds)

    res <- results(dds, contrast = c("tissue", "head", "other"))
    resOrdered <- res[order(res$padj), ]
''')

# Convert the R results to a pandas DataFrame
res_df = pandas2ri.rpy2py(robjects.r('as.data.frame(resOrdered)'))
print(res_df.head())
```

```
R[write to console]: Loading required package: S4Vectors

R[write to console]: Loading required package: stats4

R[write to console]: Loading required package: BiocGenerics

R[write to console]:
Attaching package: ‘BiocGenerics’


R[write to console]: The following object is masked from ‘package:limma’:

    plotMA


R[write to console]: The following objects are masked from ‘package:stats’:

    IQR, mad, sd, var, xtabs


R[write to console]: The following objects are masked from ‘package:base’:

    Filter, Find, Map, Position, Reduce, anyDuplicated, aperm, append,
    as.data.frame, basename, cbind, colnames, dirname, do.call,
    duplicated, eval, evalq, get, grep, grepl, intersect, is.unsorted,
    lapply, mapply, match, mget, order, paste, pmax, pmax.int, pmin,
    pmin.int, rank, rbind, rownames, sapply, setdiff, table, tapply,
    union, unique, unsplit, which.max, which.min


R[write to console]:
Attaching package: ‘S4Vectors’


R[write to console]: The following object is masked from ‘package:utils’:

    findMatches


R[write to console]: The following objects are masked from ‘package:base’:

    I, expand.grid, unname


R[write to console]: Loading required package: IRanges

R[write to console]: Loading required package: GenomicRanges

R[write to console]: Loading required package: GenomeInfoDb

R[write to console]: Loading required package: SummarizedExperiment

R[write to console]: Loading required package: MatrixGenerics

R[write to console]: Loading required package: matrixStats

R[write to console]:
Attaching package: ‘MatrixGenerics’


R[write to console]: The following objects are masked from ‘package:matrixStats’:

    colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
    colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
    colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
    colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
    colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
    colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
    colWeightedMeans, colWeightedMedians, colWeightedSds,
    colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
    rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
    rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
    rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
    rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
    rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
    rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
    rowWeightedSds, rowWeightedVars


R[write to console]: Loading required package: Biobase

R[write to console]: Welcome to Bioconductor

    Vignettes contain introductory material; view with
    'browseVignettes()'. To cite Bioconductor, see
    'citation("Biobase")', and for packages 'citation("pkgname")'.


R[write to console]:
Attaching package: ‘Biobase’


R[write to console]: The following object is masked from ‘package:MatrixGenerics’:

    rowMedians


R[write to console]: The following objects are masked from ‘package:matrixStats’:

    anyMissing, rowMedians


R[write to console]: using pre-existing normalization factors

R[write to console]: estimating dispersions

R[write to console]: found already estimated dispersions, replacing these

R[write to console]: gene-wise dispersion estimates

R[write to console]: mean-dispersion relationship

R[write to console]: final dispersion estimates

R[write to console]: fitting model and testing

R[write to console]: -- replacing outliers and refitting for 421 genes
-- DESeq argument 'minReplicatesForReplace' = 7
-- original counts are preserved in counts(dds)

R[write to console]: estimating dispersions

R[write to console]: fitting model and testing
```

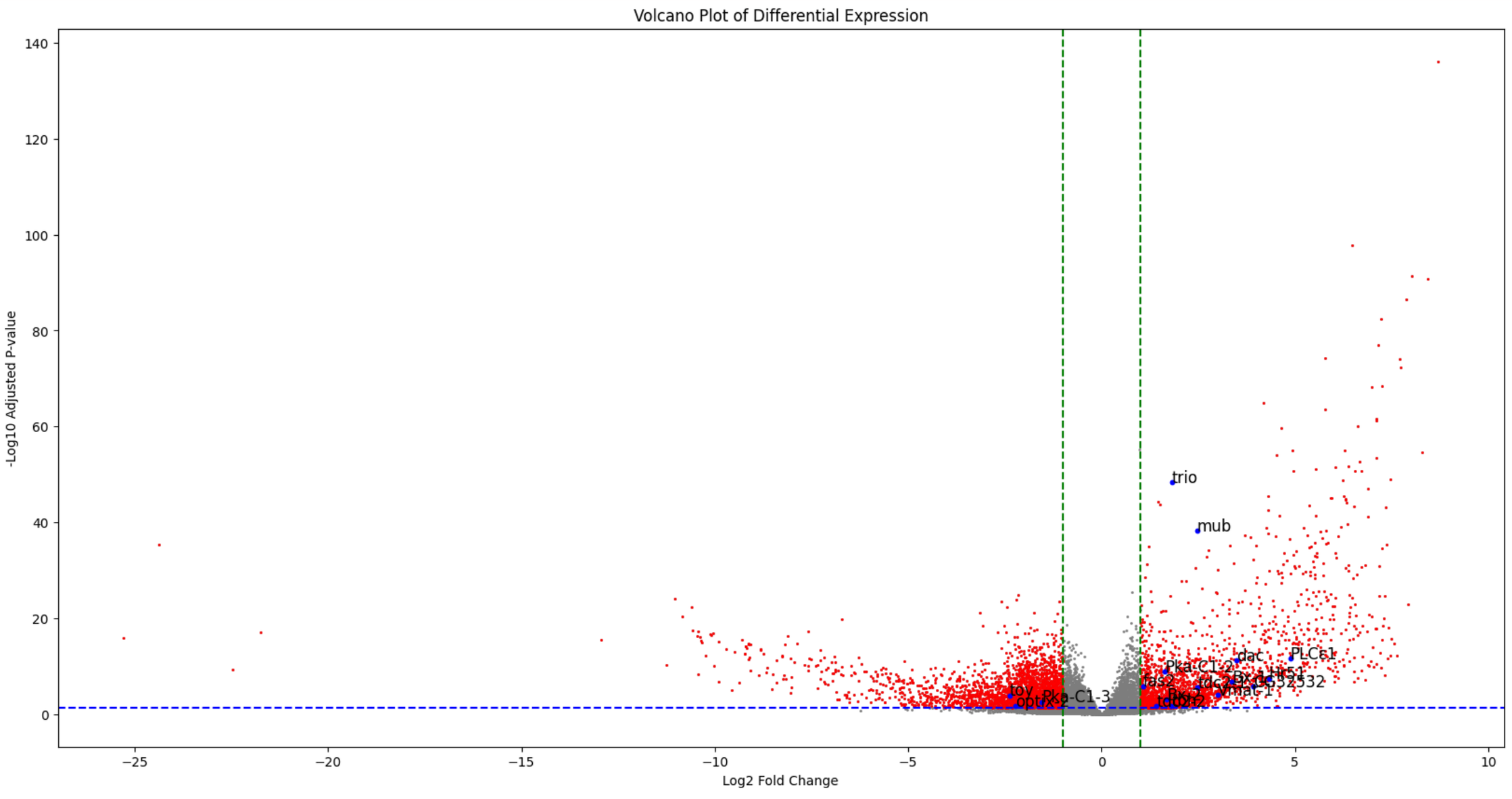
```
            baseMean  log2FoldChange     lfcSE       stat         pvalue  \
Or5-H2     24.038396        8.703844  0.344267  25.282222  5.011165e-141
Or5-9E218  26.754190        6.479692  0.301791  21.470761  2.921922e-102
Or5-9E196  12.896244        8.023818  0.386391  20.766077   8.773909e-96
Or5-V13    14.623048        8.430686  0.407381  20.694859   3.853530e-95
Or5-V21    14.296295        7.876253  0.390183  20.186043   1.298574e-90

                    padj
Or5-H2     6.739516e-137
Or5-9E218   1.964846e-98
Or5-9E196   3.933344e-92
Or5-V13     1.295653e-91
Or5-V21     3.492905e-87
```

In [16]:
```python
# Filter for significant genes (adjusted p-value < 0.05 and absolute log2 fold change > 1)
significant_genes = res_df[(res_df['padj'] < 0.05) & (abs(res_df['log2FoldChange']) > 1)]
print(f"Number of significant genes: {len(significant_genes)}")
print(significant_genes.head())
```

```
Number of significant genes: 4790
            baseMean  log2FoldChange     lfcSE       stat         pvalue  \
Or5-H2     24.038396        8.703844  0.344267  25.282222  5.011165e-141
Or5-9E218  26.754190        6.479692  0.301791  21.470761  2.921922e-102
Or5-9E196  12.896244        8.023818  0.386391  20.766077   8.773909e-96
Or5-V13    14.623048        8.430686  0.407381  20.694859   3.853530e-95
Or5-V21    14.296295        7.876253  0.390183  20.186043   1.298574e-90

                    padj
Or5-H2     6.739516e-137
Or5-9E218   1.964846e-98
Or5-9E196   3.933344e-92
Or5-V13     1.295653e-91
Or5-V21     3.492905e-87
```

In [17]:
```python
import matplotlib.pyplot as plt
import numpy as np

# Volcano plot
plt.figure(figsize=(20, 10))

# Scatter plot of log2 fold change vs -log10(p-value)
plt.scatter(res_df['log2FoldChange'], -np.log10(res_df['padj']), color='gray', s=1)

# Highlight significant genes
sig_genes = res_df[(res_df['padj'] < 0.05) & (abs(res_df['log2FoldChange']) > 1)]
plt.scatter(sig_genes['log2FoldChange'], -np.log10(sig_genes['padj']), color='red', s=1)

# Loop over the genes of interest
for ID, LOC in zip(IDs, LOCs):
    if LOC in res_df.index:
        # check if the gene is significant
        if res_df.loc[LOC, 'padj'] < 0.05 and abs(res_df.loc[LOC, 'log2FoldChange']) > 1:
            plt.scatter(res_df.loc[LOC, 'log2FoldChange'], -np.log10(res_df.loc[LOC, 'padj']), color='blue', s=10)
            plt.text(res_df.loc[LOC, 'log2FoldChange'], -np.log10(res_df.loc[LOC, 'padj']), ID, fontsize=12)
        else:
            print(f"{ID} ({LOC}) is not significant")

# Add labels and title
plt.xlabel('Log2 Fold Change')
plt.ylabel('-Log10 Adjusted P-value')
plt.title('Volcano Plot of Differential Expression')
plt.axhline(-np.log10(0.05), color='blue', linestyle='--')  # p-value threshold
plt.axvline(1, color='green', linestyle='--')  # logFC threshold
plt.axvline(-1, color='green', linestyle='--')
plt.show()
```

```
ple (LOC105279021) is not significant
Vmat-2 (LOC105274983) is not significant
DAT (LOC105277848) is not significant
ddc-1 (LOC105285288) is not significant
ddc-2 (LOC105285298) is not significant
lim1-1 (LOC105284266) is not significant
lim1-2 (LOC105285309) is not significant
optix-1 (LOC105281234) is not significant
optix-3 (LOC105282546) is not significant
mef2 (LOC105282039) is not significant
dnc (LOC105279816) is not significant
Pka-C1-1 (LOC105284392) is not significant
E75 (LOC105276741) is not significant
/Users/neurorishika/Projects/Rockefeller/Kronauer/Transgenic Ants/Candidate Genes/rna-seq-analysis/.venv/lib/python3.10/site-packages/pandas/core/arraylike.py:399: RuntimeWarning: invalid value encountered in log10
  result = getattr(ufunc, method)(*inputs, **kwargs)
```



In [19]:
```
# get the log2 fold change and p-value for the genes of interest
for ID, LOC in zip(IDs, LOCs):
    if LOC in res_df.index:
        logFC = res_df.loc[LOC, 'log2FoldChange']
        pvalue = res_df.loc[LOC, 'padj']
        print(f"{ID} ({LOC}): logFC = {logFC:.2f}, p-value = {pvalue:.2e} ({'significant' if pvalue < 0.05 else 'ns'})")
    else:
        print(f"{ID} ({LOC}) not found in DESeq2 results")
```

```
ple (LOC105279021): logFC = -0.80, p-value = 2.94e-01 (ns)
Vmat-1 (LOC105288016): logFC = 3.01, p-value = 1.07e-04 (significant)
Vmat-2 (LOC105274983): logFC = 1.22, p-value = 1.18e-01 (ns)
DAT (LOC105277848): logFC = -0.13, p-value = 8.34e-01 (ns)
ddc-1 (LOC105285288): logFC = -1.39, p-value = 8.39e-02 (ns)
ddc-2 (LOC105285298): logFC = 0.32, p-value = 6.71e-01 (ns)
lim1-1 (LOC105284266): logFC = -0.21, p-value = 7.78e-01 (ns)
lim1-2 (LOC105285309): logFC = -0.05, p-value = 9.43e-01 (ns)
CG32532 (LOC105287266): logFC = 3.93, p-value = 1.58e-06 (significant)
optix-1 (LOC105281234): logFC = -1.00, p-value = 3.24e-03 (significant)
optix-2 (LOC105276362): logFC = -2.22, p-value = 2.19e-02 (significant)
optix-3 (LOC105282546): logFC = 1.69, p-value = 6.56e-02 (ns)
tdc2-1 (LOC105275066): logFC = 2.48, p-value = 2.70e-06 (significant)
tdc2-2 (LOC105275026): logFC = 1.43, p-value = 2.06e-02 (significant)
tbh (LOC105277857): logFC = 1.80, p-value = 1.66e-02 (significant)
mef2 (LOC105282039): logFC = 0.48, p-value = 8.25e-02 (ns)
dac (LOC105287640): logFC = 3.48, p-value = 6.30e-12 (significant)
dnc (LOC105279816): logFC = 0.38, p-value = 2.51e-01 (ns)
Pka-C1-1 (LOC105284392): logFC = -0.12, p-value = 9.13e-01 (ns)
Pka-C1-2 (LOC105277221): logFC = 1.64, p-value = 1.59e-09 (significant)
Pka-C1-3 (LOC105286488): logFC = -1.54, p-value = 3.03e-03 (significant)
trio (LOC105284486): logFC = 1.82, p-value = 4.60e-49 (significant)
PLCε1 (LOC105281196): logFC = 4.88, p-value = 2.46e-12 (significant)
mub (LOC105285907): logFC = 2.47, p-value = 6.07e-39 (significant)
E75 (LOC105276741): logFC = 0.40, p-value = 4.64e-04 (significant)
toy (LOC105276147): logFC = -2.38, p-value = 1.19e-04 (significant)
fas2 (LOC105284902): logFC = 1.08, p-value = 1.53e-06 (significant)
Bx-1 (LOC105280527): logFC = 3.37, p-value = 2.04e-07 (significant)
Bx-2 (LOC105280525): logFC = 1.68, p-value = 9.31e-04 (significant)
Hr51 (LOC105276855): logFC = 4.32, p-value = 3.88e-08 (significant)
```