In [1]:
```
def install_reqs():
#This defines a function named install_reqs.

  !pip install pandas
#Installs the pandas library, used for reading and manipulating tabular data (li

  !pip install "matplotlib>=3.4"
#Installs matplotlib version 3.4 or higher, used for plotting graphs and visuali

  !pip install numpy
#Installs numpy, a fundamental package for numerical computations and arrays.

  !pip install statsmodels
#Installs statsmodels, a library used for statistical tests (e.g., t-tests, Wilc

  !pip install scipy
#Installs scipy, which contains scientific computing tools, including statistica

install_reqs()
# to install all the above dependencies uncomment the above line.
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in c:\programdata\anaconda3\lib\site-packag
es (2.3.0)
Requirement already satisfied: numpy>=1.22.4 in c:\users\hp\appdata\roaming\pytho
n\python310\site-packages (from pandas) (1.26.4)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3
\lib\site-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-
packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\sit
e-packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-pack
ages (from python-dateutil>=2.8.2->pandas) (1.17.0)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: matplotlib>=3.4 in c:\programdata\anaconda3\lib\si
te-packages (3.10.0)
Requirement already satisfied: contourpy>=1.0.1 in c:\programdata\anaconda3\lib\s
ite-packages (from matplotlib>=3.4) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\programdata\anaconda3\lib\site-
packages (from matplotlib>=3.4) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in c:\programdata\anaconda3\lib
\site-packages (from matplotlib>=3.4) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\programdata\anaconda3\lib
\site-packages (from matplotlib>=3.4) (1.4.8)
Requirement already satisfied: numpy>=1.23 in c:\users\hp\appdata\roaming\python
\python310\site-packages (from matplotlib>=3.4) (1.26.4)
Requirement already satisfied: packaging>=20.0 in c:\programdata\anaconda3\lib\si
te-packages (from matplotlib>=3.4) (25.0)
Requirement already satisfied: pillow>=8 in c:\programdata\anaconda3\lib\site-pac
kages (from matplotlib>=3.4) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in c:\programdata\anaconda3\lib\s
ite-packages (from matplotlib>=3.4) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in c:\programdata\anaconda3\l
ib\site-packages (from matplotlib>=3.4) (2.9.0.post0)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-pack
ages (from python-dateutil>=2.7->matplotlib>=3.4) (1.17.0)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in c:\users\hp\appdata\roaming\python\python
310\site-packages (1.26.4)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: statsmodels in c:\programdata\anaconda3\lib\site-p
ackages (0.14.4)
Requirement already satisfied: numpy<3,>=1.22.3 in c:\users\hp\appdata\roaming\py
thon\python310\site-packages (from statsmodels) (1.26.4)
Requirement already satisfied: scipy!=1.9.2,>=1.8 in c:\programdata\anaconda3\lib
\site-packages (from statsmodels) (1.15.3)
Requirement already satisfied: pandas!=2.1.0,>=1.4 in c:\programdata\anaconda3\li
b\site-packages (from statsmodels) (2.3.0)
Requirement already satisfied: patsy>=0.5.6 in c:\programdata\anaconda3\lib\site-
packages (from statsmodels) (1.0.1)
Requirement already satisfied: packaging>=21.3 in c:\programdata\anaconda3\lib\si
te-packages (from statsmodels) (25.0)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\programdata\anaconda3
\lib\site-packages (from pandas!=2.1.0,>=1.4->statsmodels) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in c:\programdata\anaconda3\lib\site-
packages (from pandas!=2.1.0,>=1.4->statsmodels) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in c:\programdata\anaconda3\lib\sit
e-packages (from pandas!=2.1.0,>=1.4->statsmodels) (2025.2)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-pack
ages (from python-dateutil>=2.8.2->pandas!=2.1.0,>=1.4->statsmodels) (1.17.0)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-package
s (1.15.3)
Requirement already satisfied: numpy<2.5,>=1.23.5 in c:\users\hp\appdata\roaming
\python\python310\site-packages (from scipy) (1.26.4)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
WARNING: Ignoring invalid distribution -atplotlib (c:\programdata\anaconda3\lib\s
ite-packages)
WARNING: Ignoring invalid distribution -orch (c:\programdata\anaconda3\lib\site-p
ackages)
```

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:
```
from compass_analysis import cohens_d, wilcoxon_test, get_reaction_consistencies
```

In [4]:
```
from matplotlib import __version__ as matplotlibversion
if matplotlibversion < "3.4":
    print("Matplotlib versions older than 3.4 may not be able to generate figure
```

```
Matplotlib versions older than 3.4 may not be able to generate figure 2E, as they
do not support alpha arrays
```

In [5]:
```
import os
os.system("compass --data expression.tsv \
          --model RECON2_mat --species mus_musculus --media default-media --lamb
```

```
          --and-function mean --output-dir extdata/Th17 --penalty-diffusion knn
          --isoform-summing legacy --num-processes 50")
```

Out[5]: 0

In [6]:
```
import os
print(os.getcwd())
```

```
C:\Users\HP\Tcon vs Treg\extdata
```

In [7]:
```
reaction_penalties = pd.read_csv("reactions.tsv", sep="\t", index_col=0)
```

In [8]:
```
import pandas as pd

# Load reactions.tsv
reaction_penalties = pd.read_csv("reactions.tsv", sep="\t", index_col=0)

# Extract cell IDs from the columns
cell_ids = reaction_penalties.columns

# Infer cell type from prefix
def infer_cell_type(cell_id):
    if cell_id.startswith("Treg"):
        return "Treg"
    elif cell_id.startswith("Tcon"):
        return "Tcon"
    elif cell_id.startswith("exTreg"):
        return "exTreg"
    else:
        return "Unknown"

cell_types = [infer_cell_type(cell) for cell in cell_ids]

# Create metadata DataFrame
cell_metadata = pd.DataFrame({
    "cell_id": cell_ids,
    "cell_type": cell_types
}).set_index("cell_id")

# Save to CSV
cell_metadata.to_csv("cell_metadata.csv")

print("✅ Created cell_metadata.csv with the following cell types:")
print(cell_metadata["cell_type"].value_counts())
```

```
✅ Created cell_metadata.csv with the following cell types:
cell_type
exTreg    20
Tcon      20
Treg      16
Name: count, dtype: int64
```

In [9]:
```
cell_metadata = pd.read_csv("cell_metadata.csv", sep=",")
cell_metadata.set_index("cell_id", inplace=True)
```

In [10]:
```
print(cell_metadata.columns.tolist())
print(cell_metadata.head())
```

```
['cell_type']
                          cell_type
cell_id
exTreg_Lymph_exTreg2_L1     exTreg
Tcon_Lymph_Tcon9_L2           Tcon
exTreg_Spleen_exTreg8_L1    exTreg
Tcon_Spleen_Tcon5_L2          Tcon
Treg_Spleen_Treg5_L1          Treg
```

In [11]:
```
Treg_cells = cell_metadata.index[cell_metadata["cell_type"] == "Treg"]
Tcon_cells = cell_metadata.index[cell_metadata["cell_type"] == "Tcon"]
exTreg_cells = cell_metadata.index[cell_metadata["cell_type"] == "exTreg"]
```

In [12]:
```
print(cell_metadata.columns.tolist())
```

```
['cell_type']
```

In [13]:
```
print(cell_metadata.head())
print(cell_metadata.columns.tolist())
```

```
                          cell_type
cell_id
exTreg_Lymph_exTreg2_L1     exTreg
Tcon_Lymph_Tcon9_L2           Tcon
exTreg_Spleen_exTreg8_L1    exTreg
Tcon_Spleen_Tcon5_L2          Tcon
Treg_Spleen_Treg5_L1          Treg
['cell_type']
```

In [14]:
```
reaction_metadata = pd.read_csv("reaction_metadata.csv", index_col = 0)
```

In [15]:
```
reaction_metadata.loc[['r0281']]
```

Out[15]:

reaction_no_direction	reaction_name	formula	associated_genes	subsystem	EC_numl
r0281	Putrescine:oxygen oxidoreductase (deaminating)...	1.00 * Water [e] + 1.00 * O2 [e] + 1.00 * Putr...	AOC1	Methionine and cysteine metabolism	1.4

In [16]:
```
#This function is repeated here for clarity
def get_reaction_consistencies(compass_reaction_penalties, min_range=1e-3):
    """
        Converts the raw penalties outputs of compass into scores per reactions
    """
    df = -np.log(compass_reaction_penalties + 1)
    df = df[df.max(axis=1) - df.min(axis=1) >= min_range]
    df = df - df.min().min()
    return df
```

In [17]:
```
reaction_consistencies = get_reaction_consistencies(reaction_penalties)
```

In [18]:
```
common_cells = list(set(reaction_consistencies.columns).intersection(set(cell_me
reaction_consistencies = reaction_consistencies[common_cells]
```

In [19]:
```
wilcox_results = wilcoxon_test(reaction_consistencies, Tcon_cells, Treg_cells)
wilcox_results['metadata_r_id'] = ""
for r in wilcox_results.index:
    if r in reaction_metadata.index:
        wilcox_results.loc[r, 'metadata_r_id'] = r
    elif r[:-4] in reaction_metadata.index:
        wilcox_results.loc[r, 'metadata_r_id'] = r[:-4]
    else:
        print("Should not occur")
```

In [20]:
```
print(reaction_consistencies.columns.tolist()[:10])
```

```
['exTreg_Spleen_exTreg6_L2', 'Treg_Lymph_Treg8_L1', 'Treg_Spleen_Treg4_L1', 'Tcon
_Lymph_Tcon8_L1', 'Tcon_Lymph_Tcon10_L1', 'Tcon_Lymph_Tcon9_L1', 'exTreg_Spleen_e
xTreg6_L1', 'Tcon_Spleen_Tcon3_L1', 'exTreg_Spleen_exTreg9_L1', 'exTreg_Lymph_exT
reg5_L2']
```

In [21]:
```
print(Treg_cells[:10])
```

```
Index(['Treg_Spleen_Treg5_L1', 'Treg_Lymph_Treg7_L2', 'Treg_Spleen_Treg1_L1',
       'Treg_Spleen_Treg4_L2', 'Treg_Lymph_Treg8_L1', 'Treg_Spleen_Treg5_L2',
       'Treg_Lymph_Treg7_L1', 'Treg_Spleen_Treg1_L2', 'Treg_Spleen_Treg4_L1',
       'Treg_Lymph_Treg8_L2'],
      dtype='object', name='cell_id')
```

In [22]:
```
# Run the test
wilcox_results = wilcoxon_test(reaction_consistencies, Treg_cells, exTreg_cells)

# Annotate metadata keys
wilcox_results['metadata_r_id'] = ""

for r in wilcox_results.index:
    if r in reaction_metadata.index:
        wilcox_results.loc[r, 'metadata_r_id'] = r
    elif r[:-4] in reaction_metadata.index:
        wilcox_results.loc[r, 'metadata_r_id'] = r[:-4]
    else:
        print(f"Should not occur → {r}")
```

In [23]:
```
# Run the test
wilcox_results = wilcoxon_test(reaction_consistencies, Tcon_cells, Treg_cells)

# Annotate metadata keys
wilcox_results['metadata_r_id'] = ""

for r in wilcox_results.index:
    if r in reaction_metadata.index:
        wilcox_results.loc[r, 'metadata_r_id'] = r
    elif r[:-4] in reaction_metadata.index:
        wilcox_results.loc[r, 'metadata_r_id'] = r[:-4]
    else:
        print(f"Should not occur → {r}")
```

In [24]:
```
W = wilcox_results.merge(reaction_metadata, how='left',
                         left_on='metadata_r_id', right_index=True, validate='m:
W = W[W['confidence'].isin([0,4])]
```

```
W = W[~W['EC_number'].isna()]
W.loc[(W['formula'].map(lambda x: '[m]' not in x)) & (W['subsystem'] == "Citric
```

In [25]:
```
wilcox_results.loc[['r0281_pos']]
```

Out[25]:

	wilcox_stat	wilcox_pval	cohens_d	adjusted_pval	metadata_r_id
r0281_pos	27.0	0.000025	-1.836818	0.000044	r0281

In [26]:
```
reaction_metadata.loc['r0281']['formula']
```

Out[26]:
```
'1.00 * Water [e] + 1.00 * O2 [e] + 1.00 * Putrescine [e] --> 1.00 * Ammonium
[e] + 1.00 * Hydrogen peroxide [e] + 1.00 * 4-Aminobutanal [e]\nAOC1'
```

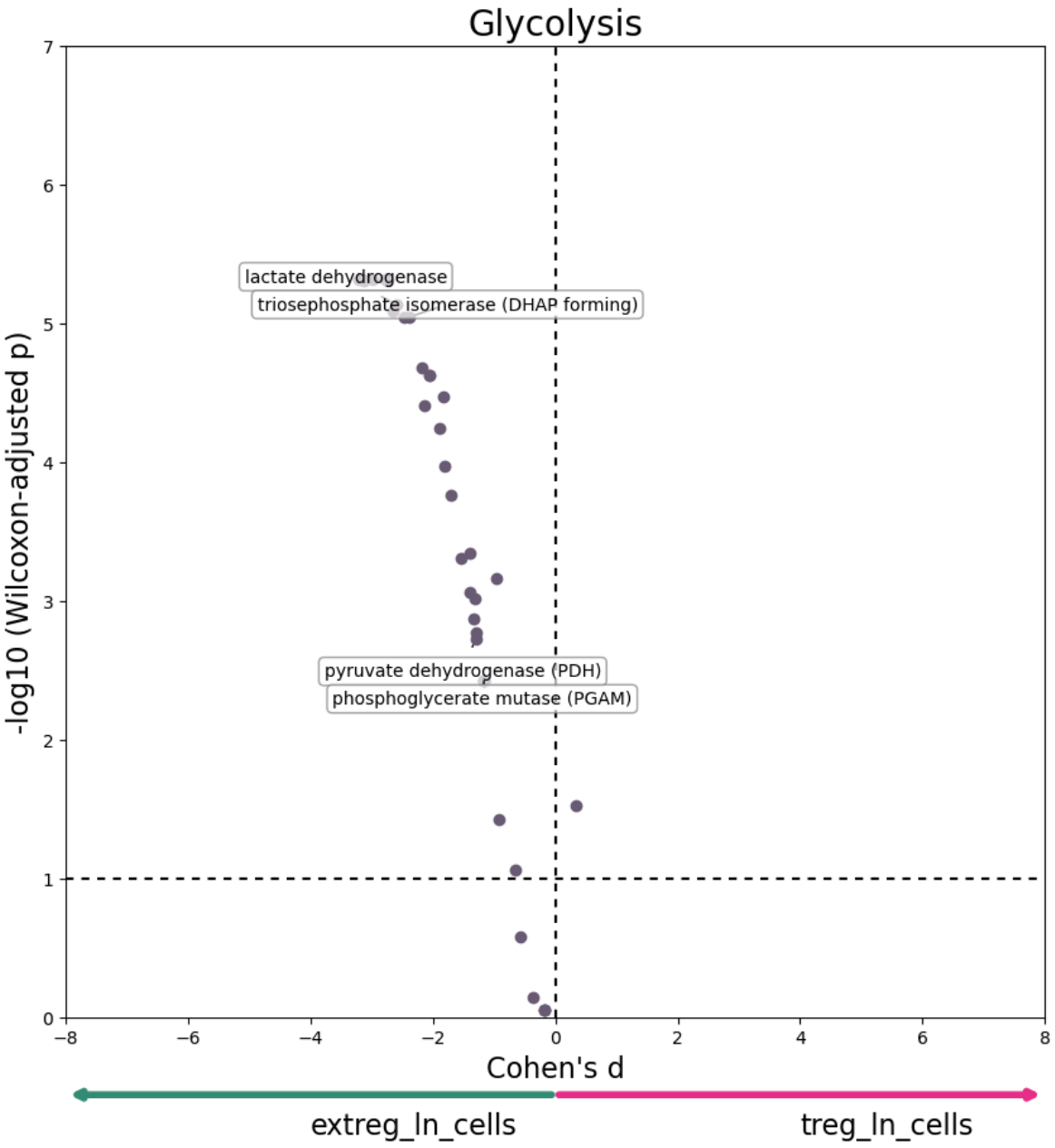
In [152…
```
def plot_differential_scores(data, title, c):
    plt.figure(figsize=(10,10))
    axs = plt.gca()
    axs.scatter(data['cohens_d'], -np.log10(data['adjusted_pval']), c=c)
    axs.set_xlabel("Cohen's d", fontsize=16)
    axs.set_ylabel("-log10 (Wilcoxon-adjusted p)", fontsize=16)

    # Visual markers and title
    axs.set_xlim(-8, 8)
    axs.set_ylim(0, 7)
    axs.axvline(0, dashes=(3,3), c='black')
    axs.axhline(1, dashes=(3,3), c='black')
    axs.set_title(title, fontdict={'fontsize':20})

    # Arrows for sample identity
    axs.annotate('', xy=(0.5, -0.08), xycoords='axes fraction', xytext=(0, -0.08
                 arrowprops=dict(arrowstyle="<-", color='#348C73', linewidth=4))
    axs.annotate('treg_ln_cells', xy=(0.75, -0.12), xycoords='axes fraction', fc
    axs.annotate('', xy=(0.5, -0.08), xycoords='axes fraction', xytext=(1, -0.08
                 arrowprops=dict(arrowstyle="<-", color='#E92E87', linewidth=4))
    axs.annotate('extreg_ln_cells', xy=(0.25, -0.12), xycoords='axes fraction',

    # Dynamic annotation loop with variable offsets and labels
    for i, r in enumerate(data.index):
        if r in labeled_reactions:
            x = data.loc[r, 'cohens_d']
            y = -np.log10(data.loc[r, 'adjusted_pval'])

            dx = 30 if x >= 0 else -120  # horizontal offset
            dy = (i % 6) * 12 - 30       # vertical offset varies to reduce stac

            axs.annotate(
                labeled_reactions[r],
                (x, y),
                xytext=(dx, dy),
                textcoords='offset pixels',
                arrowprops=dict(arrowstyle="-", shrinkA=6, shrinkB=4),
                fontsize=10,
                zorder=5,  # make sure text is on top
                bbox=dict(boxstyle="round,pad=0.3", fc="white", ec="gray", alpha
            )
```

In [154…
```
filtered_data = pd.concat([W[W['subsystem'] == "Glycolysis/gluconeogenesis"],
             W[W['subsystem'] == "Citric acid cycle"],
```

```
            W[W['subsystem'].isin(amino_acid_metab)],
            W[W['subsystem'] == "Fatty acid oxidation"]])
```

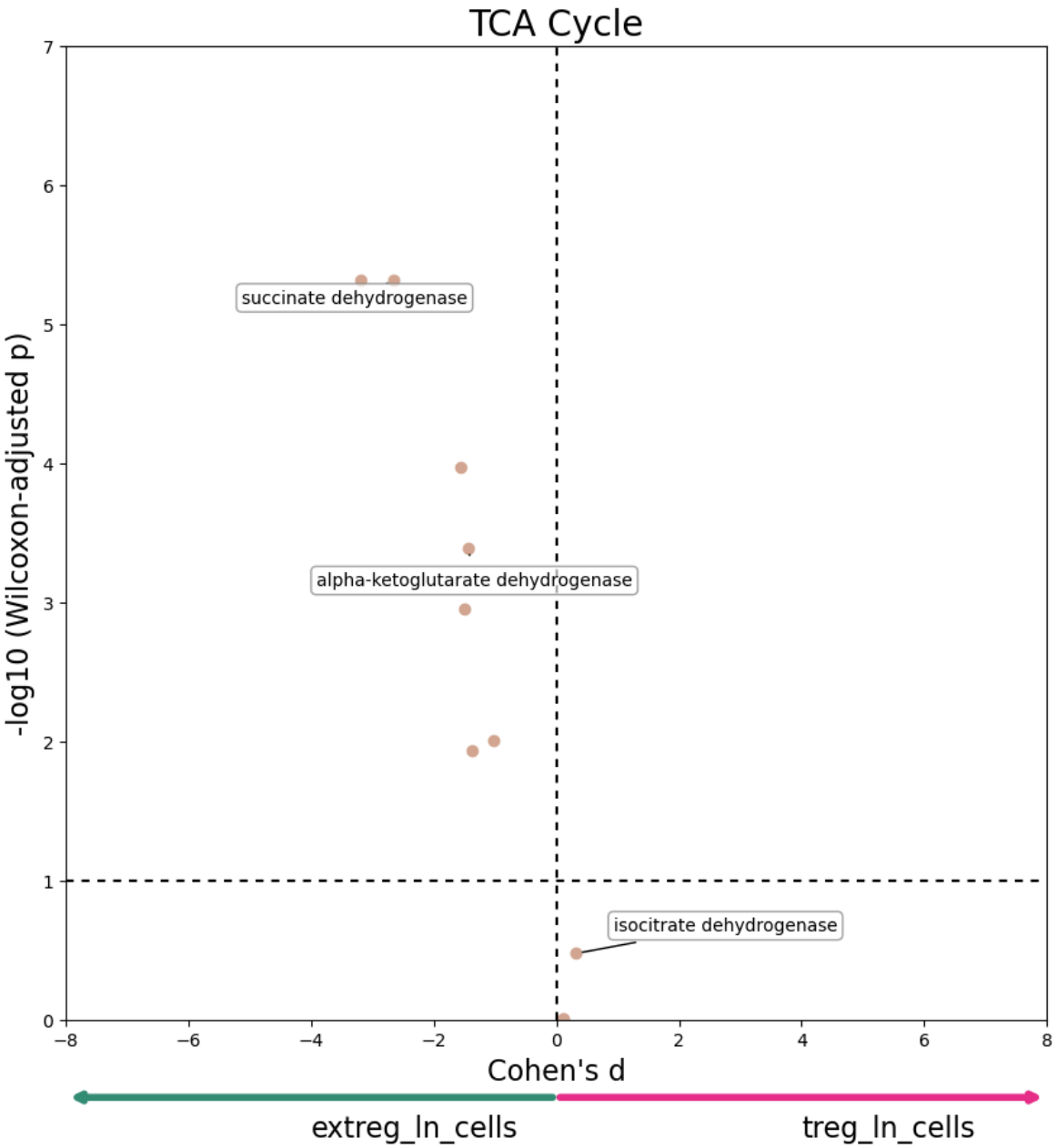
In [156…

```
data = W[W['subsystem'] == "Glycolysis/gluconeogenesis"]
plot_differential_scores(data, title='Glycolysis', c="#695D73")
```



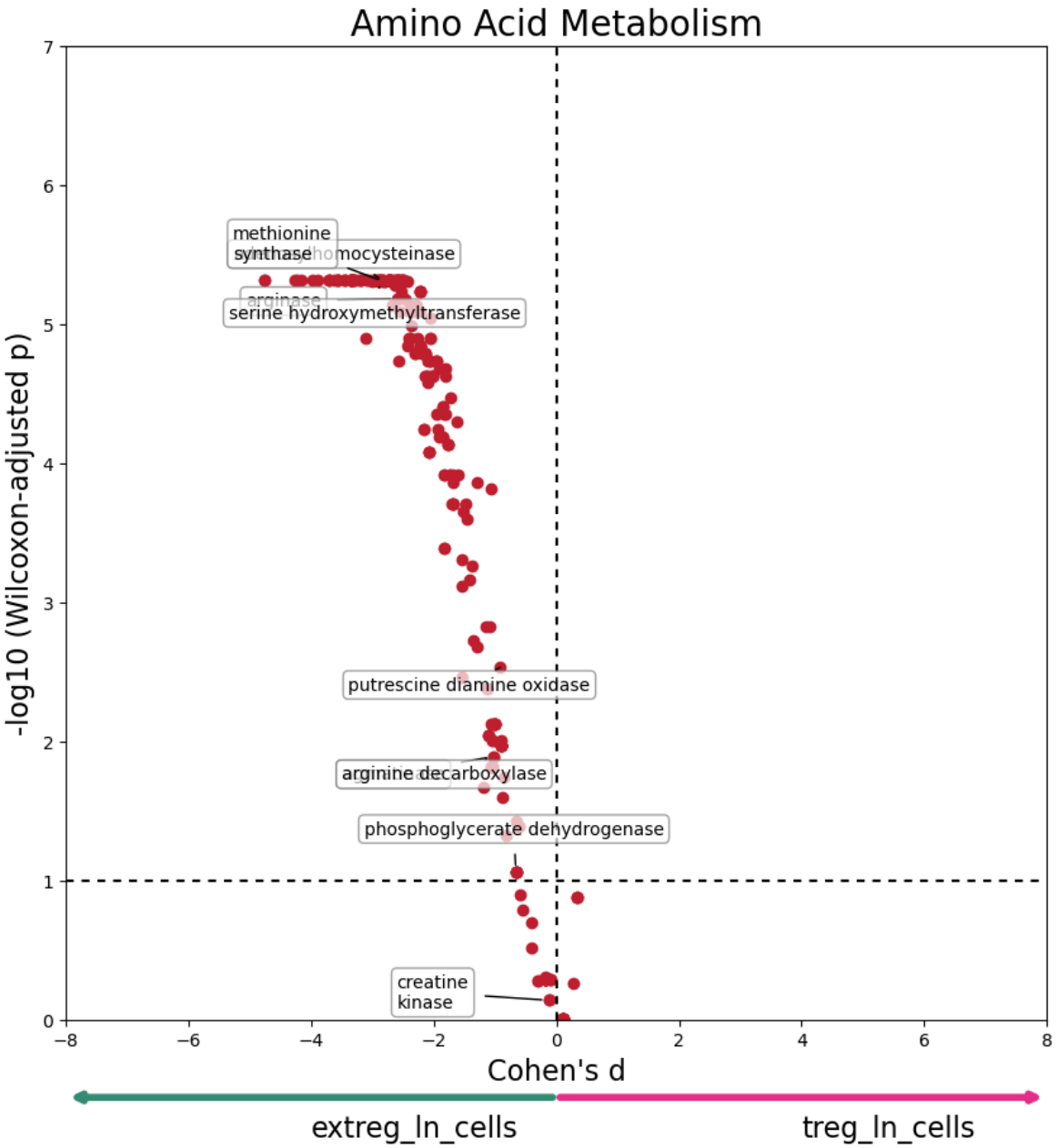
In [158…

```
data = W[W['subsystem'] == "Citric acid cycle"]
plot_differential_scores(data, title="TCA Cycle", c="#D3A991")
```



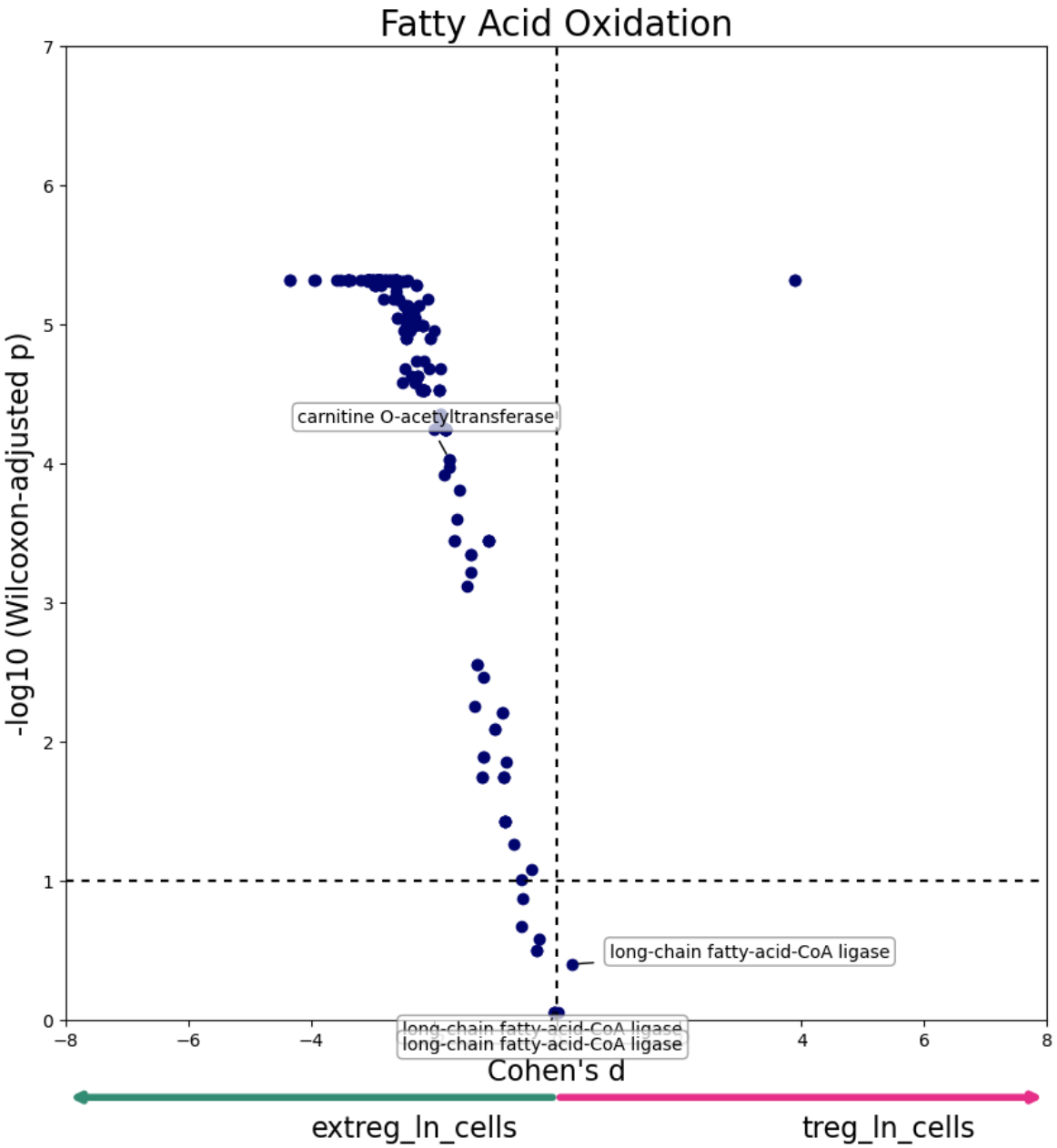
In [160…

```
data = W[W['subsystem'].isin(amino_acid_metab)].copy()
data['adjusted_pval'] = data['adjusted_pval'].clip(1e-12)
plot_differential_scores(data, "Amino Acid Metabolism", c="#BF1E2E")
```



In [161…

```
data = W[W['subsystem'] == "Fatty acid oxidation"]
plot_differential_scores(data, "Fatty Acid Oxidation", c="#040772")
```



In [33]:
```
data = W[~W['subsystem'].isin(["Miscellaneous", "Unassigned"])]
data = data[~data['subsystem'].map(lambda x: "Transport" in x or "Exchange" in x
items, counts = np.unique(data['subsystem'], return_counts=True)
items = [items[i] for i in range(len(items)) if counts[i] > 5] #filter(n() > 5)
data = data[data['subsystem'].isin(items)]
```

In [34]:
```
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 12))
axs = plt.gca()

d = data[data['adjusted_pval'] < 0.1].groupby('subsystem')['cohens_d'].median().

d_sorted = d.sort_values()

axs.scatter(d_sorted, d_sorted.index, alpha=0)

color = data['cohens_d'].map(lambda x: 'r' if x >= 0 else 'b')
alpha = data['adjusted_pval'].map(lambda x: 1.0 if x < 0.1 else 0.25)

axs.scatter(data['cohens_d'], data['subsystem'], c=color, alpha=alpha)
```

```
axs.set_xlabel("Cohen's d")
axs.set_ylabel("Pathway (subsystem)")
axs.set_title("Pathway-Level Differential Activity")
```

Out[34]: Text(0.5, 1.0, 'Pathway-Level Differential Activity')

In [35]:
```
reaction_penalties = pd.read_csv("reactions.tsv", sep="\t", index_col = 0)
reaction_penalties[reaction_penalties <= 1e-4] = 0
reaction_penalties = reaction_penalties[np.all(reaction_penalties != 0, axis=1)]
```

In [36]:
```
reaction_penalties = reaction_penalties[reaction_penalties.max(axis=1) - reactio
```

In [37]:
```
meta_rxns_map = get_metareactions(reaction_penalties)
meta_rxns = reaction_penalties.join(pd.DataFrame(meta_rxns_map, columns=["meta_r
```

In [38]:
```
meta_rxn_consistencies = get_reaction_consistencies(meta_rxns)
```

In [39]:
```
treg_cells = cell_metadata.index[cell_metadata["cell_type"] == "Tcon"]
tcon_cells = cell_metadata.index[cell_metadata["cell_type"] == "Treg"]

wilcox_meta_rxn_results = wilcoxon_test(meta_rxn_consistencies, tcon_cells, treg
```

In [40]:
```
wilcox_meta_rxn_results.iloc[0:1]
```

Out[40]:

meta_rxn_id	wilcox_stat	wilcox_pval	cohens_d	adjusted_pval
1	320.0	3.818334e-07	2.680434	0.000004

In [41]:
```
wilcox_meta_rxn_expanded = pd.DataFrame(index=reaction_penalties.index, columns=
for i in range(len(wilcox_meta_rxn_expanded.index)):
    if (meta_rxns_map[i] in wilcox_meta_rxn_results.index):
        wilcox_meta_rxn_expanded.loc[wilcox_meta_rxn_expanded.index[i]] = wilcox
wilcox_meta_rxn_expanded = wilcox_meta_rxn_expanded.dropna().astype('float64')
```

In [42]:
```
wilcox_meta_rxn_expanded['metadata_r_id'] = ""
for r in wilcox_meta_rxn_expanded.index:
    if r in reaction_metadata.index:
        wilcox_meta_rxn_expanded.loc[r, 'metadata_r_id'] = r
    elif r[:-4] in reaction_metadata.index:
        wilcox_meta_rxn_expanded.loc[r, 'metadata_r_id'] = r[:-4]
    else:
        print("Should not occur")
```

In [43]:
```
wilcox_meta_rxn_expanded.iloc[0:1]
```

Out[43]:

	wilcox_stat	wilcox_pval	cohens_d	adjusted_pval	metadata_r_id
10FTHF5GLUtl_pos	302.0	0.000007	2.21927	0.000016	10FTHF5GLUtl

In [44]:
```
outputs = {
    "wilcox_results.csv": wilcox_results,
    "reaction_consistencies.csv": reaction_consistencies,
    "reaction_metadata.csv": reaction_metadata,
    "wilcox_meta_rxn_results.csv": wilcox_meta_rxn_results,
    "wilcox_meta_rxn_expanded.csv": wilcox_meta_rxn_expanded,
    "final_stats_with_metadata.csv": W,
}

for name, df in outputs.items():
    df.to_csv(name)
```

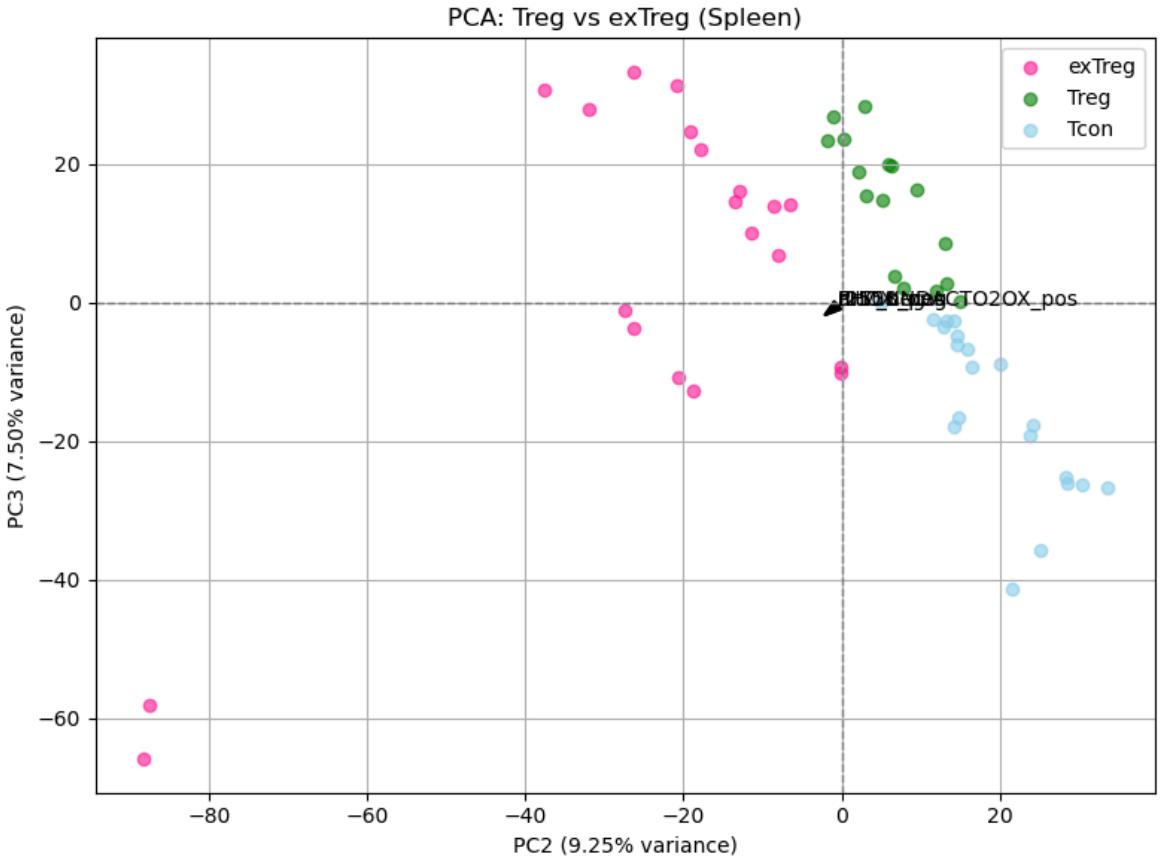
In [45]:
```
import zipfile

with zipfile.ZipFile("Tcon vs Treg.zip", "w") as zipf:
    for filename in outputs:
        zipf.write(filename)
```

In [46]:
```
from IPython.display import FileLink
FileLink("Tcon vs Treg.zip")
```

Out[46]: Tcon vs Treg.zip

In [47]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
X = reaction_consistencies.T

X_scaled = StandardScaler().fit_transform(X)

pca = PCA(n_components=3)
principalComponents = pca.fit_transform(X_scaled)
pca_df = pd.DataFrame(data=principalComponents, columns=['PC1', 'PC2', 'PC3'])
pca_df['group'] = cell_metadata.loc[X.index, 'cell_type'].values  # Adjust to yo
```

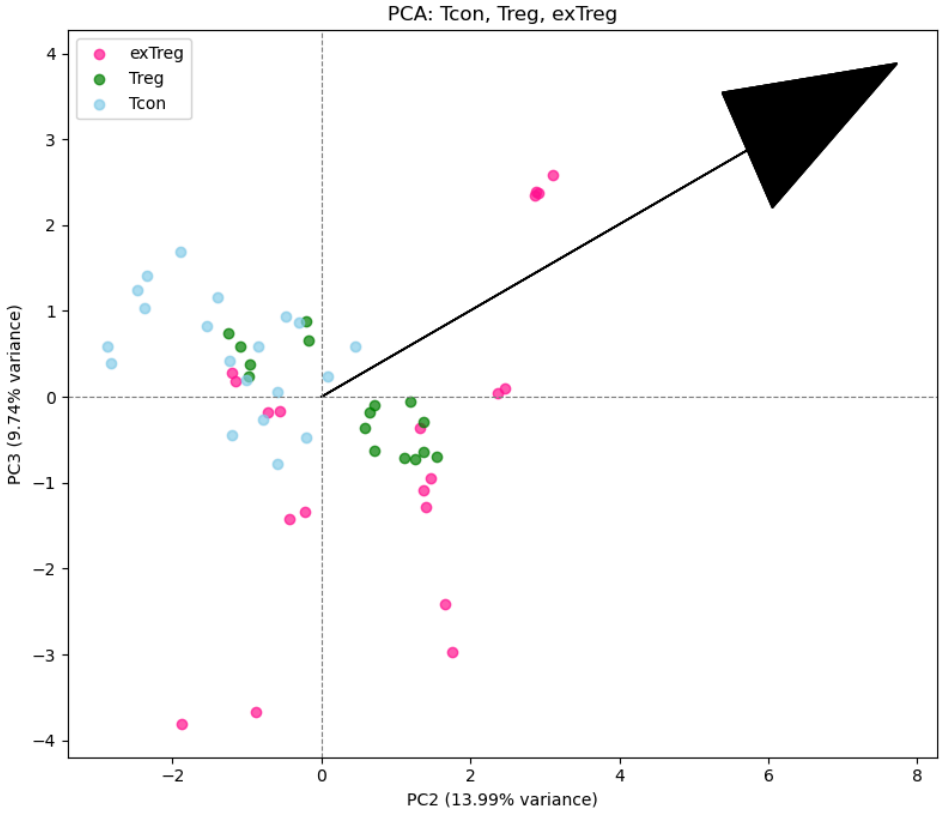
```
plt.figure(figsize=(8,6))
colors = {'Treg': 'green', 'exTreg': 'deeppink', 'Tcon': 'skyblue'}
for group in pca_df['group'].unique():
    subset = pca_df[pca_df['group'] == group]
    plt.scatter(subset['PC2'], subset['PC3'], c=colors[group], label=group, alph

loadings = pca.components_.T[:, 1:3]  # PC2 and PC3
top_indices = np.argsort(np.linalg.norm(loadings, axis=1))[-4:]  # Top 4 feature
for i in top_indices:
    plt.arrow(0, 0, loadings[i,0]*15, loadings[i,1]*15, color='black', alpha=0.7
    plt.text(loadings[i,0]*17, loadings[i,1]*17, X.columns[i], fontsize=10)

plt.xlabel(f"PC2 ({pca.explained_variance_ratio_[1]*100:.2f}% variance)")
plt.ylabel(f"PC3 ({pca.explained_variance_ratio_[2]*100:.2f}% variance)")
plt.title("PCA: Treg vs exTreg (Spleen)")
plt.axhline(0, color='gray', linestyle='--', lw=1)
plt.axvline(0, color='gray', linestyle='--', lw=1)
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



In [48]:

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import numpy as np

X = reaction_consistencies.T
labels = cell_metadata.loc[X.index, 'cell_type']

pca = PCA(n_components=3)
X_pca = pca.fit_transform(X)
```

```
color_map = {'Tcon': 'skyblue', 'Treg': 'green', 'exTreg': 'deeppink'}

plt.figure(figsize=(10, 7))
for group in labels.unique():
    idx = labels == group
    plt.scatter(X_pca[idx, 1], X_pca[idx, 2], label=group, color=color_map[group

plt.xlabel(f"PC2 ({pca.explained_variance_ratio_[1]*100:.2f}% variance)")
plt.ylabel(f"PC3 ({pca.explained_variance_ratio_[2]*100:.2f}% variance)")
plt.title("PCA: Tcon, Treg, exTreg")
plt.axhline(0, color='gray', linestyle='--', lw=0.8)
plt.axvline(0, color='gray', linestyle='--', lw=0.8)

loadings = pca.components_.T[:, 1:3]
magnitudes = np.linalg.norm(loadings, axis=1)
top_indices = np.argsort(magnitudes)[-5:]  # Change 5 to more/less if needed

for i in top_indices:
    x, y = loadings[i, 0]*30, loadings[i, 1]*30  # scale for visibility
    plt.arrow(0, 0, x, y, color='black', alpha=0.7, head_width=1.5)
    label_offset = 4 if x >= 0 else -4
    plt.text(x + label_offset, y, X.columns[i], fontsize=10, ha='left' if x >= 0

plt.legend()
plt.tight_layout()
plt.show()
```



In [ ]: