```python
import pdb
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.pylab as pl
import matplotlib
matplotlib.rcParams['text.usetex'] = True


def scinot(num):
    '''
    Formats numbers in scientific notation for LaTeX.
    '''
    numstrs = '{:.1E}'.format(num).split('E+')
    return r'${} \times 10^{{{}}}$'.format(numstrs[0], int(numstrs[1]))


# Set up figure
fig, ax = plt.subplots(1, 1)

# Collect data from external files
cluster_data = np.loadtxt('GalWCls19.txt', skiprows=41)
galaxy_data = np.loadtxt('GalWGal.txt', skiprows=9)

# Bin the clusters by mass
n_bins = 5
cluster_masses = cluster_data[:, 18]
cluster_distances = cluster_data[:, 5]

# Only take clusters larger than 10^14 for completeness
cluster_data = cluster_data[np.logical_and(
    cluster_masses >= 1e14, cluster_distances <= 225)]
cluster_masses = cluster_masses[np.logical_and(
    cluster_masses >= 1e14, cluster_distances <= 225)]

# Make mass bins, if necessary
mass_bins = np.logspace(
    np.log10(
        min(cluster_masses)), np.log10(
            max(cluster_masses)), n_bins + 1)
inds = np.digitize(cluster_masses, mass_bins)

colors = pl.cm.jet(np.linspace(0, 1, n_bins))

# For each mass bin in the sample...
for i in range(1, n_bins + 1):

    # Narrow the sample to clusters in this mass range
    clusters = cluster_data[inds == i]
    center_distances = np.array([])
    elliptical_flags = np.array([])

    if len(clusters) == 0:
        continue

    # Loop over all the clusters in this group
    for j in range(len(clusters)):
        # Collect galaxies in this cluster by matching cluster ID
        galaxies = galaxy_data[galaxy_data[:, 10] == clusters[j, 0]]

        # Calculate the distance of each galaxy to the cluster center
        r_gal = galaxies[:, 3]
        r_c = clusters[j, 5]
```

```
        d_los = np.abs(r_gal - r_c)  # Line-of-sight distance

        # Get sky-projected distance
        dRA = (galaxies[:, 0] - clusters[j, 1]) * np.pi / 180.
        dDEC = (galaxies[:, 1] - clusters[j, 2]) * np.pi / 180.
        c = np.arccos(np.cos(dRA) * np.cos(dDEC))  # Distance on sky in radians
        d_sky = r_gal * c  # Distance on sky in same units as r_gal

        # Distance from galaxies in this sample to this cluster's center
        d = np.sqrt(d_los**2 + d_sky**2)
        is_elliptical = galaxies[:, 9] == 2

        # Add data from this cluster to the mass group's data
        center_distances = np.concatenate((center_distances, d))
        elliptical_flags = np.concatenate((elliptical_flags, is_elliptical))

    # Bin by distance to create x and y data
    n_dist_bins = 50
    try:
        #dist_bins = np.logspace(np.log10(min(center_distances)),
np.log10(max(center_distances)), n_dist_bins)
        dist_bins = np.linspace(0, 10, n_dist_bins)
    except BaseException:
        pdb.set_trace()
    dist_inds = np.digitize(center_distances, dist_bins)

    # Find fraction of ellipticals in each distance bin
    all_bincenters = (dist_bins[:-1] + dist_bins[1:]) / 2
    frac_ellip = []
    bincenters = []

    for k in range(1, len(dist_bins)):

        #    rule = dist_inds == k
        rule = dist_inds <= k
        if len(elliptical_flags[rule]) > 10:
            frac_ellip.append(
                sum(elliptical_flags[rule]) / len(elliptical_flags[rule]))
            bincenters.append(all_bincenters[k - 1])

    # Plot results
    ax.plot(bincenters,
            frac_ellip,
            alpha=0.75,
            c=colors[i - 1],
            lw=1,
            marker='o',
            markersize=2,
            label=r'{} $< \rm M_{{200}} < $ {} $\rm M_{{\odot}}
$'.format(scinot(mass_bins[i - 1] * 1e14),

scinot(mass_bins[i] * 1e14)))


if n_bins > 1:
    sm = plt.cm.ScalarMappable(
        cmap=pl.cm.jet,
        norm=plt.Normalize(
            vmin=min(
                cluster_masses *
                1e14),
            vmax=max(
```

```
                cluster_masses *
                1e14)))
    cbar = fig.colorbar(sm)
    cbar.ax.set_ylabel(r'Cluster Mass ($\rm M_{\odot}$)', rotation=90)
ax.set_xlabel(r'$\rm d_{center}\ (\rm Mpc\ h^{-1})$')
ax.set_ylabel(r'$\rm N_e / N$ within $\rm d_{center}$')
ax.set_title('Galaxy Morphology by Distance from Cluster Center')

ax.grid(ls='--', alpha=0.25)
ax.legend(fontsize=8)
plt.show()
```