notebook

July 6, 2021

[1]:
```
# import deps
import PIL.Image
import scipy.cluster
import scipy.spatial
import numpy as np
```

[2]:
```
# open image and save into `points`
with PIL.Image.open("turquoise.png") as image:
    points = np.array(image)/255  # force float
    w, h = points.shape[:2]
    points = points.reshape((w*h, 3)) # [x][y][rgb] -> [i][rgb]
    # remove colors made identical with above dimension drop
    points = np.unique(points, axis=0)
```

1 Yotam's proposed algorithm

Given a set of N points p_i and an outlier percentage λ, run k-means with $k = \frac{N}{100-\lambda}$, and then compute the convex hull k times, leaving out one cluster each time. Return the smallest convex hull.

[3]:
```
  = 0.5
N = points.shape[0] # not w*h since duplicates were removed
K = int(N / (100 -  ))
print(f"K = {K}")
# K=25
```

```
K = 622
```

[4]:
```
# create clusters using k-means
clusters = scipy.cluster.vq.kmeans2(points, K)
centroids, labels = clusters
```

```
/Users/Jason/Desktop/ASSIP/test/venv/lib/python3.8/site-
packages/scipy/cluster/vq.py:607: UserWarning: One of the clusters is empty. Re-
run kmeans with a different initialization.
  warnings.warn("One of the clusters is empty. "
```

[12]:
```
# create every hull that excludes exactly one cluster
hulls = list()
for label in range(K):
    # exclude the current label (a cluster)
    mask = np.where(labels != label)
    included = points[mask]
    # create convex hull
    hull = scipy.spatial.ConvexHull(included)
    hulls.append((hull, mask))
```

[17]:
```
# find the hull that has the smallest volume
smallest_hull, mask = min(hulls, key=lambda hull: hull[0].volume)
starting_hull = scipy.spatial.ConvexHull(points)
```

1.1 Results

Below is a comparison of the output convex hull made using just `points` and the output convex hull made by finding one of k clusters whose removal minimizes the volume of the hull.

[14]:
```
import matplotlib.pyplot as plot
```

[40]:
```
np.array(labels/(K-1))[mask].T.shape
```

[40]:
```
(61633,)
```

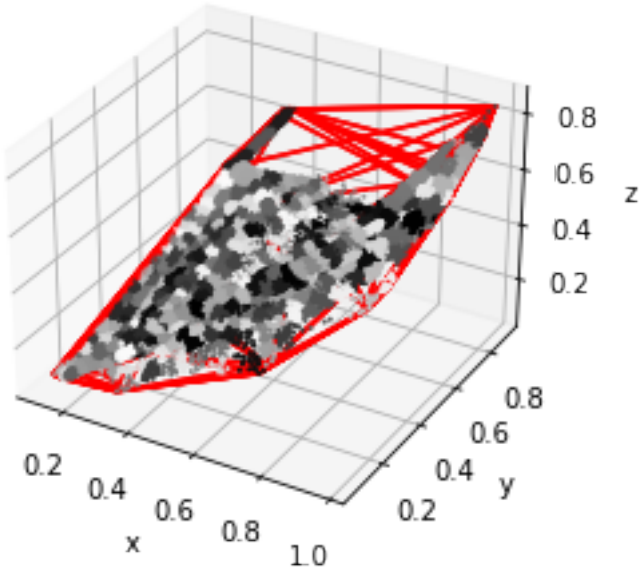
[41]:
```
def plot_points(points, hull, mask=False):
    figure = plot.figure()
    l = labels[mask] if mask else labels
    axes = figure.add_subplot(111, projection="3d")
    axes.scatter(
        points.T[0],
        points.T[1],
        points.T[2],
        c=np.array(labels/(K-1))[mask].T,
        s=1)

    for simplex in hull.simplices:
        simplex = np.append(simplex, simplex[0])
        axes.plot(
            points[simplex, 0],
            points[simplex, 1],
            points[simplex, 2],
            "r-")

    axes.set_xlabel("x")
    axes.set_ylabel("y")
    axes.set_zlabel("z")
```

```
    plot.show()
```

1.1.1 Results: No Manipulation

[30]:
```
%matplotlib inline
plot_points(points, starting_hull)
```

```
(61931,)
```



1.1.2 Results: With Manipulation

[47]:
```
# @hidden
%matplotlib inline
# plot_points(points[mask], smallest_hull, True)
figure = plot.figure()
axes = figure.add_subplot(111, projection="3d")
axes.scatter(
    points[mask].T[0],
    points[mask].T[1],
    points[mask].T[2],
    c=np.array([labels[mask]/(K-1)]*3).T,
    s=1)

for simplex in smallest_hull.simplices:
    simplex = np.append(simplex, simplex[0])
```

```
    axes.plot(
        points[mask][simplex, 0],
        points[mask][simplex, 1],
        points[mask][simplex, 2],
        "r-")

axes.set_xlabel("x")
axes.set_ylabel("y")
axes.set_zlabel("z")

plot.show()
```