Extracting data with yt: objects, fields, and the clump finder.

Britton Smith

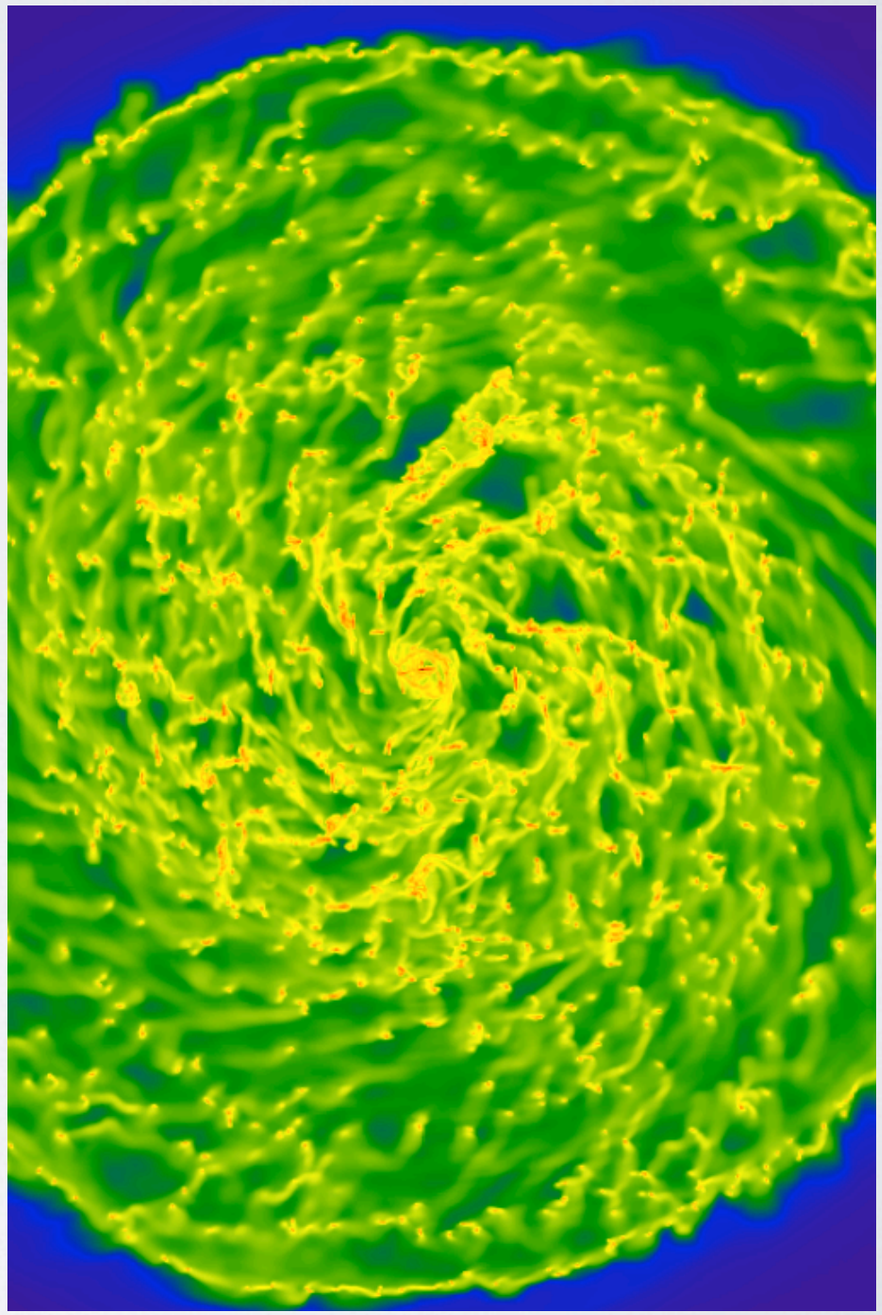
GOALS

1. Work with data containers

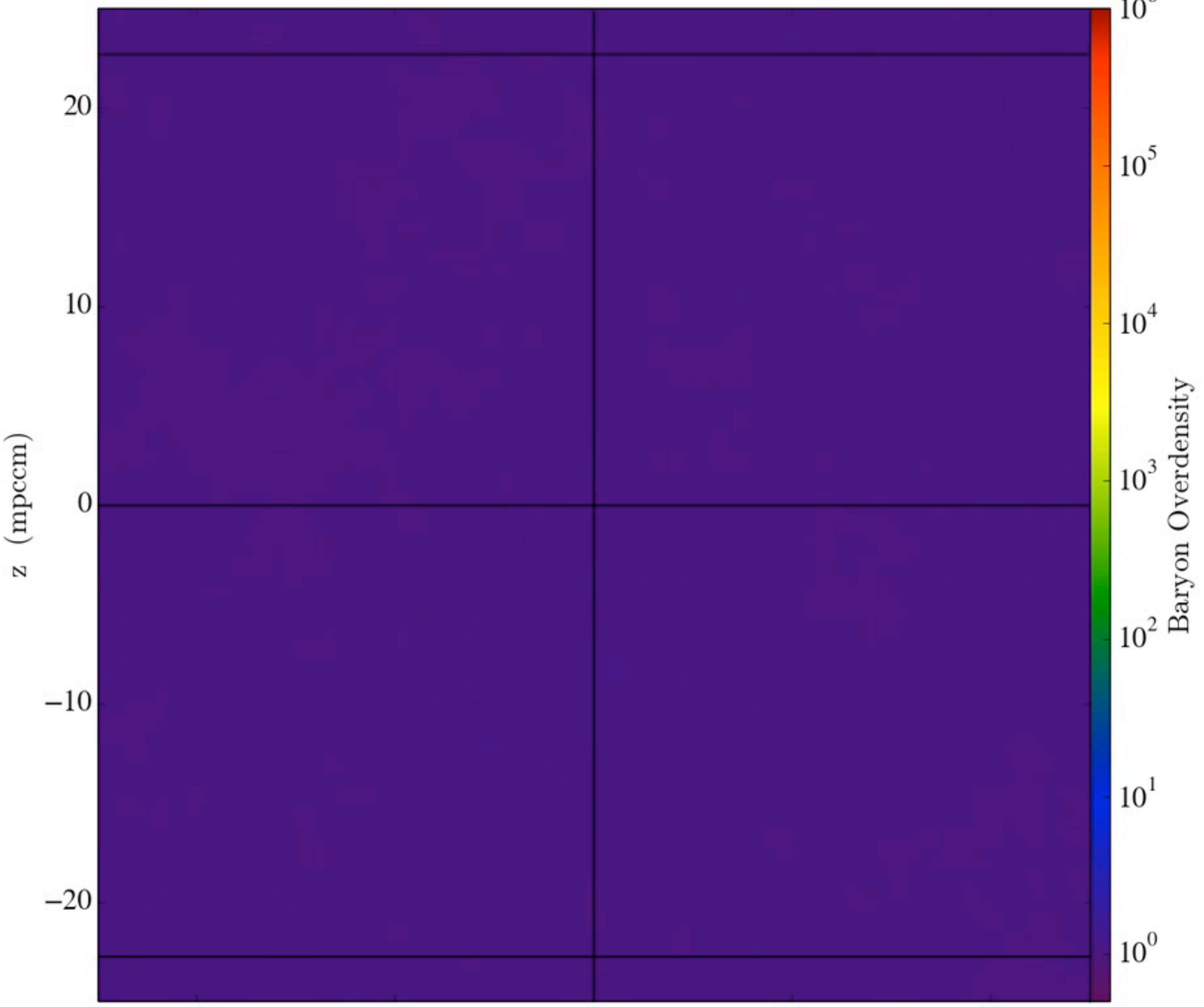
2. Use derived quantities

3. Make new fields

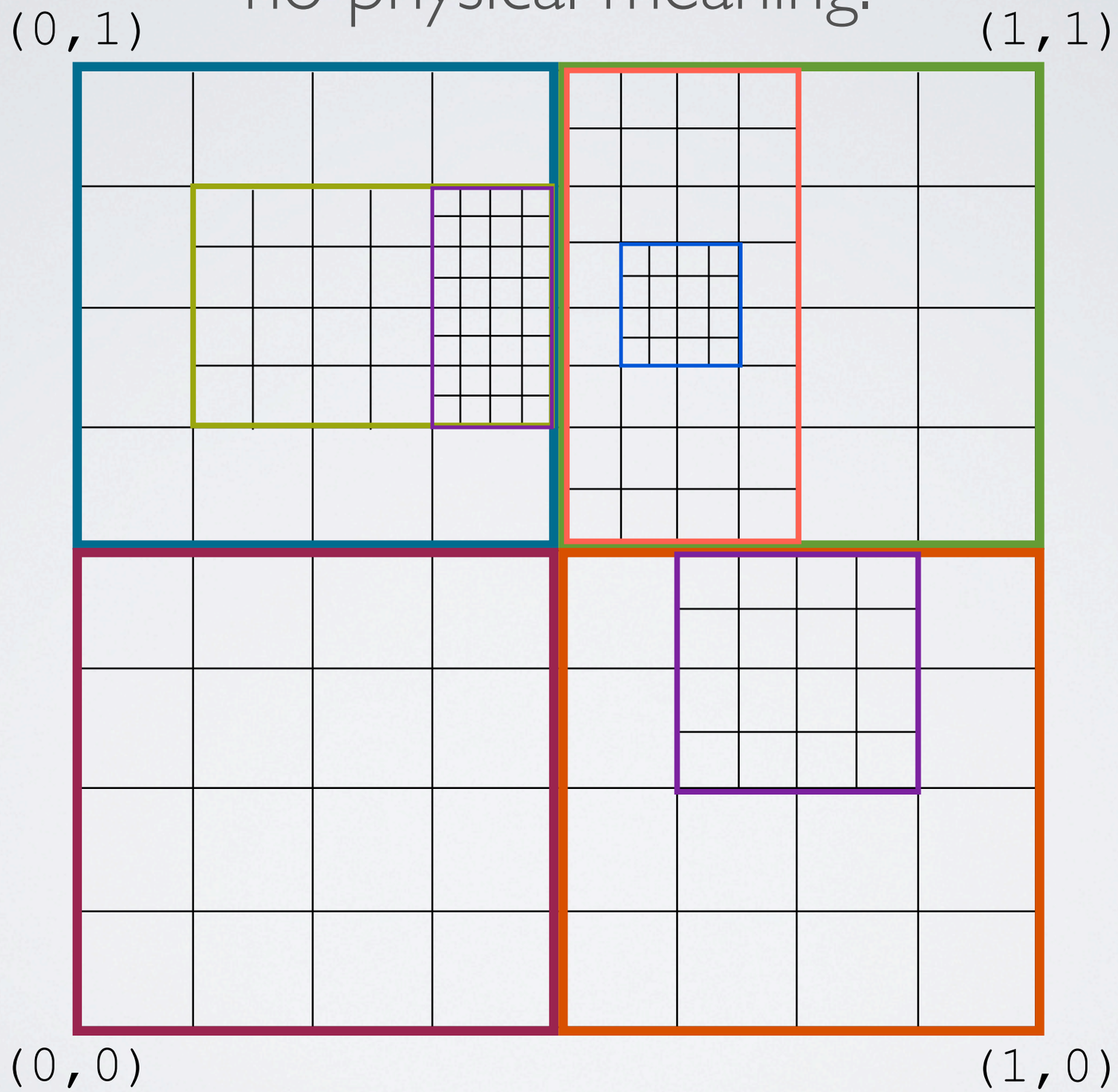
4. Run the clump finder



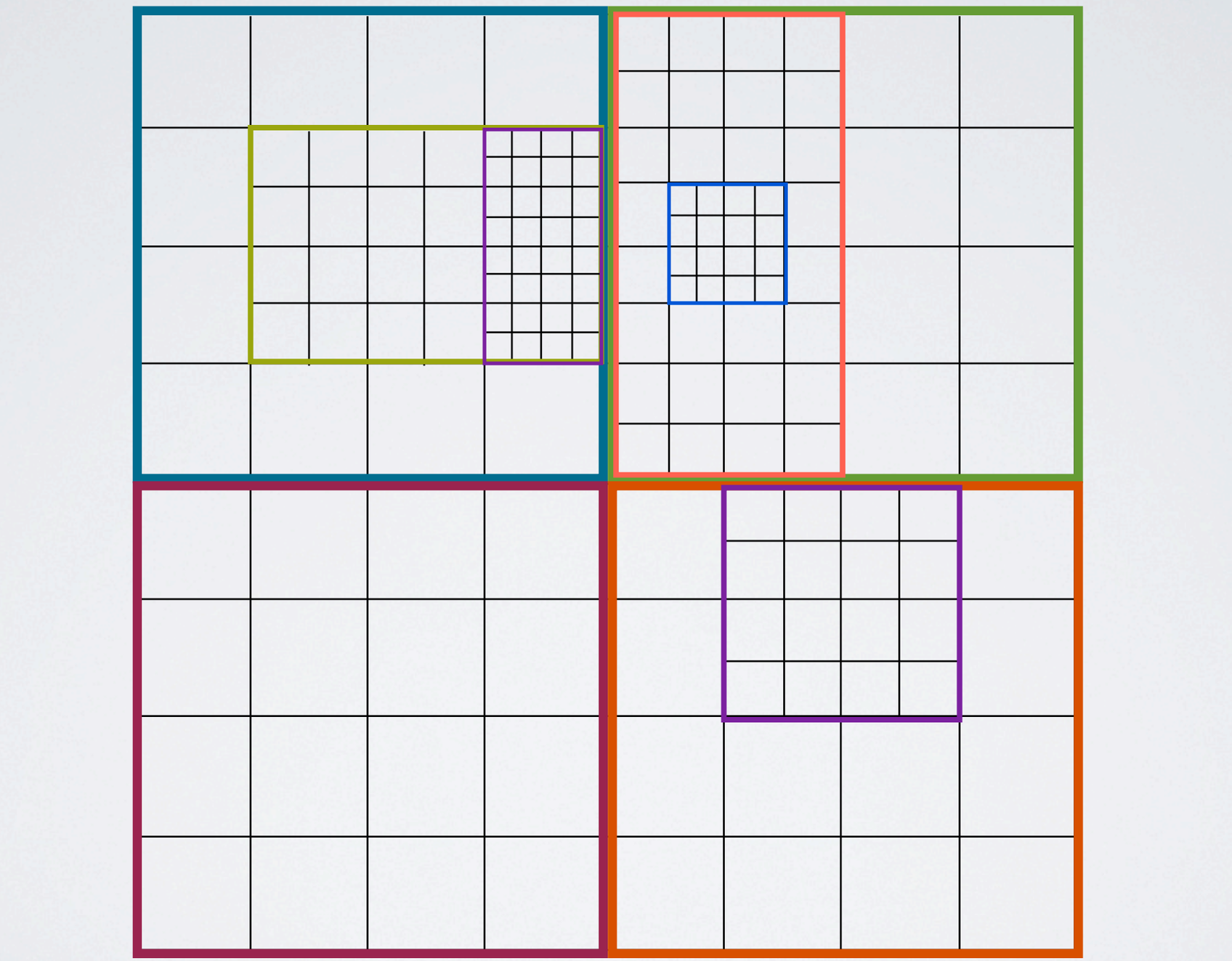
What does grid simulation data actually look like?



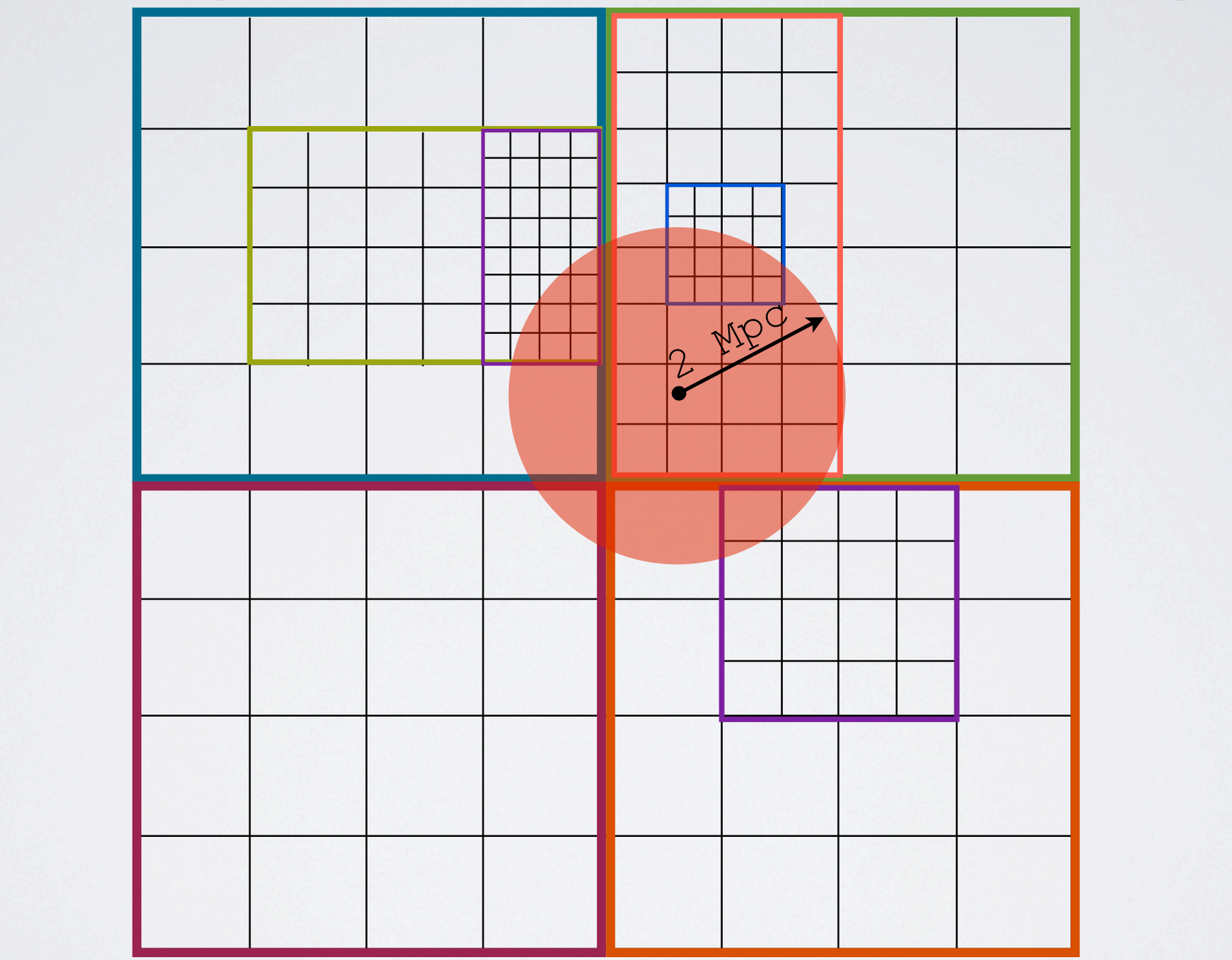
Data on disk has no physical meaning.



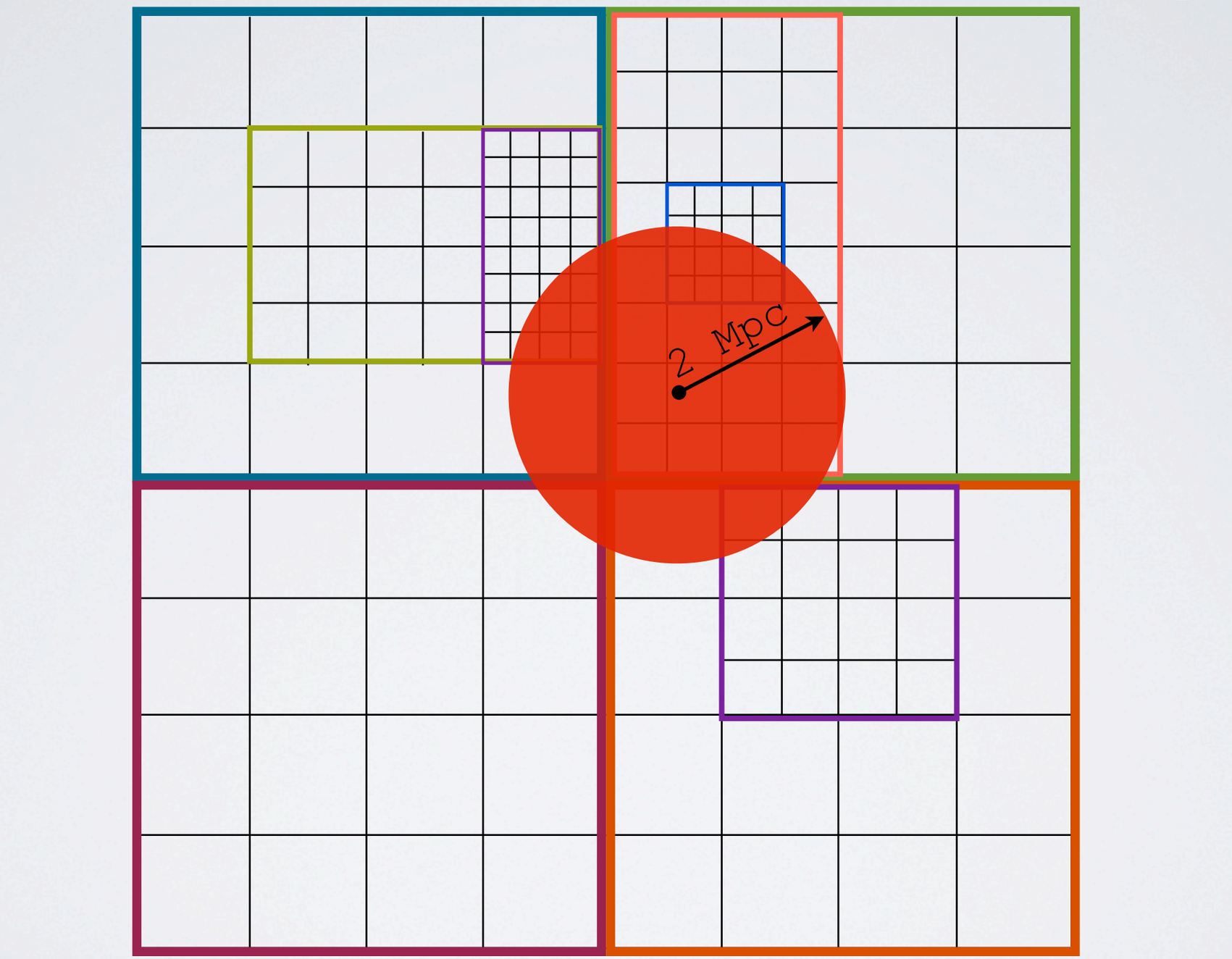
yt lets you think about physical objects



yt lets you think about physical objects

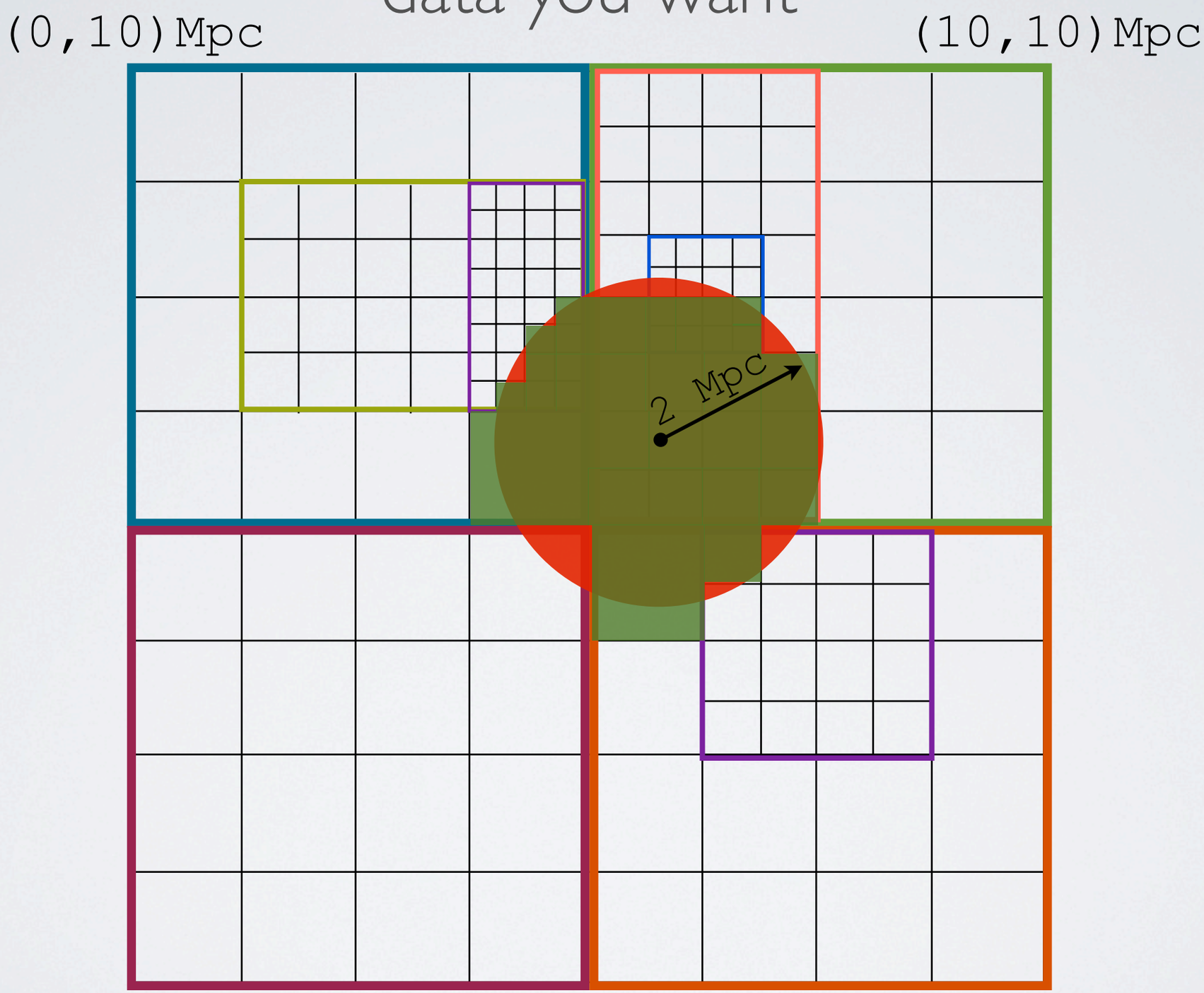


yt lets you think about physical objects

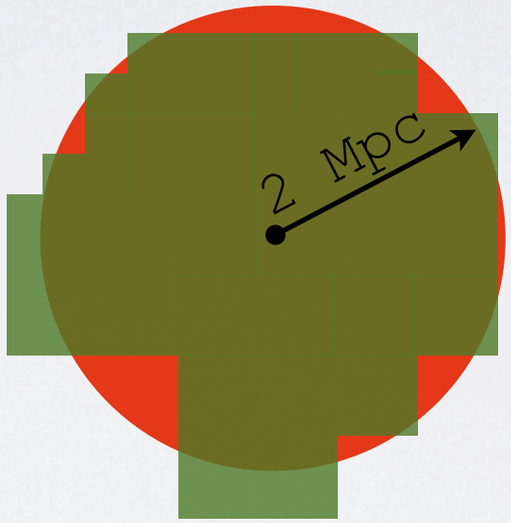


and forget what's underneath.

yt gives you the data you want

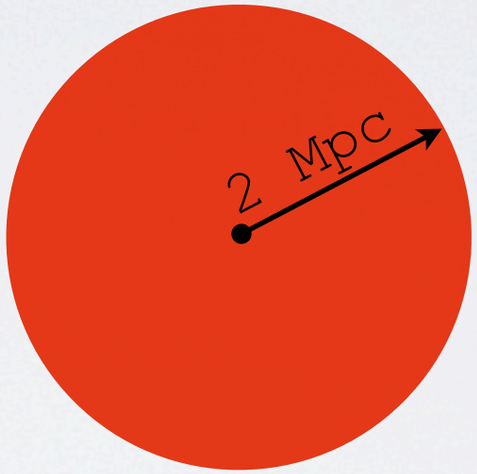


yt gives you the
data you want



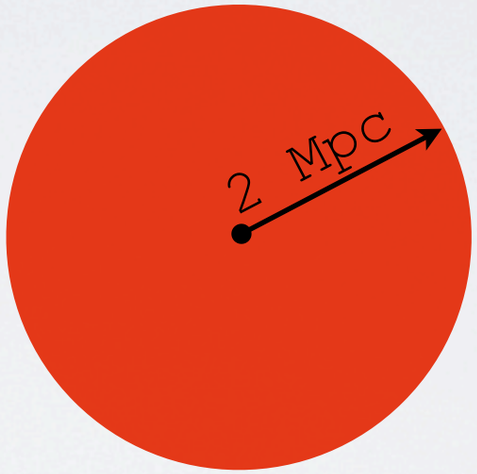
and only the data you want.

You can do whatever
you want with it.



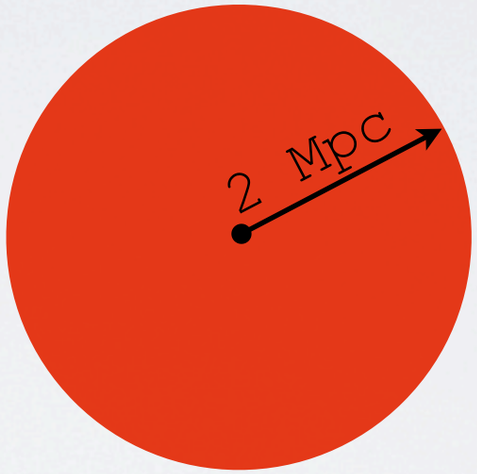
You can do whatever you want with it.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

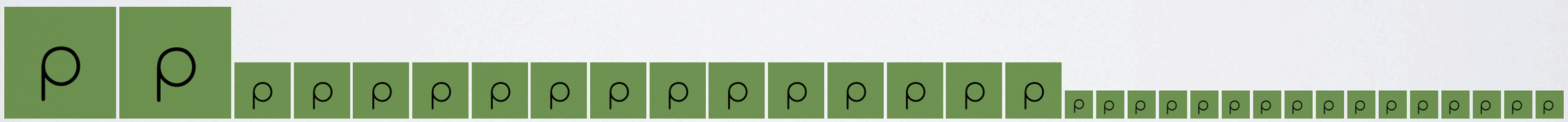


You can do whatever you want with it.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

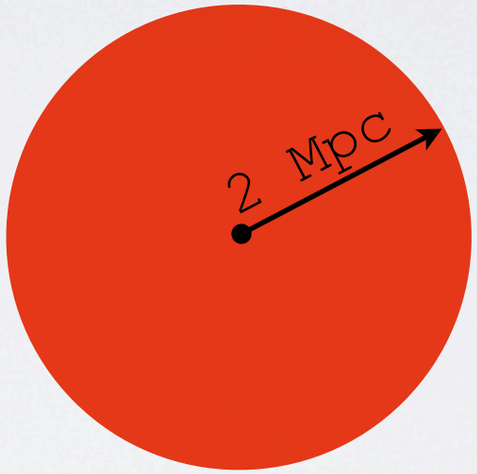



```
sp["density"]
```

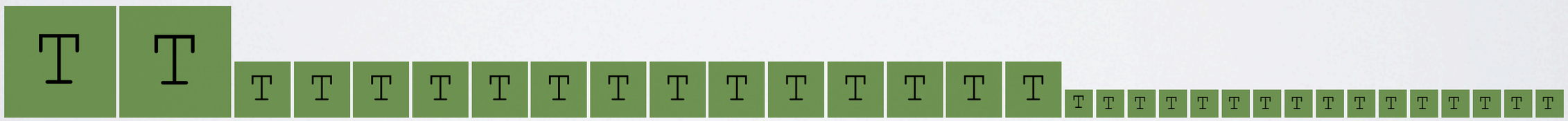


You can do whatever you want with it.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

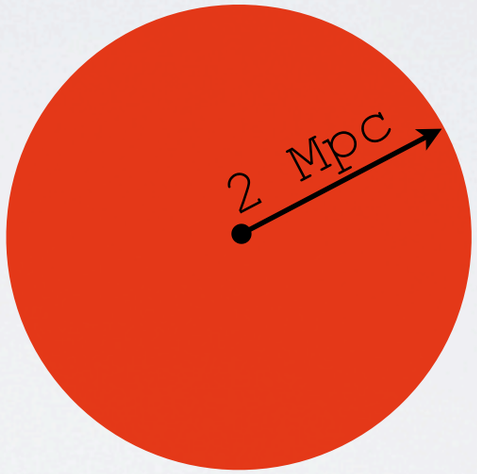



```
sp["temperature"]
```

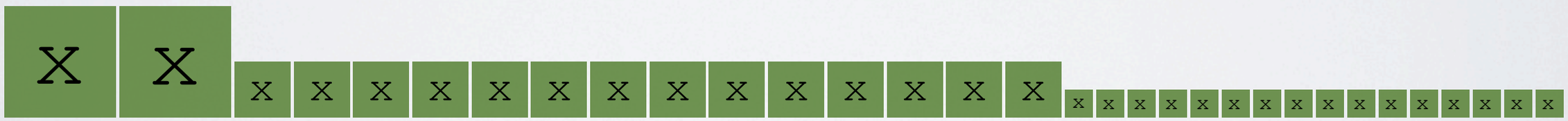


Spatial information is not lost.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

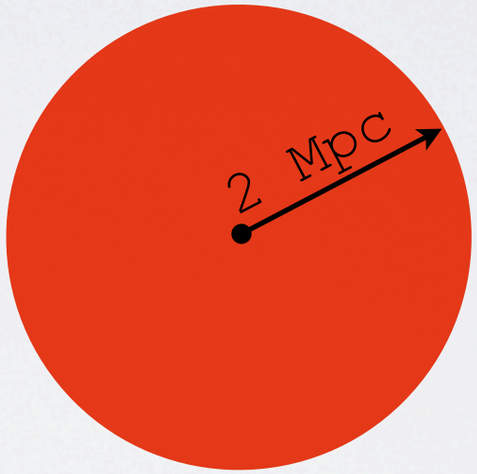



```
sp["x"]
```

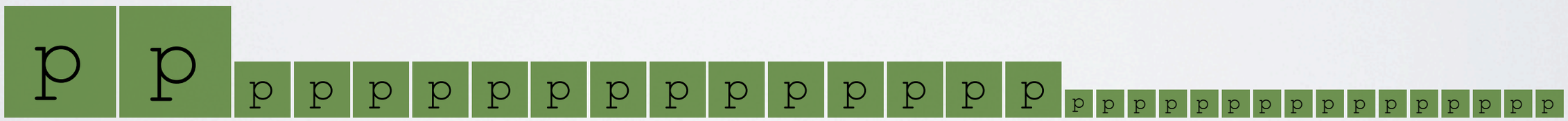


Data containers give fields as Numpy arrays.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

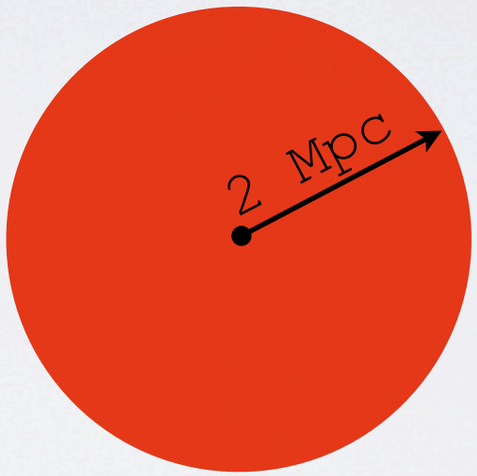



```
sp["density"] * sp["temperature"]
```

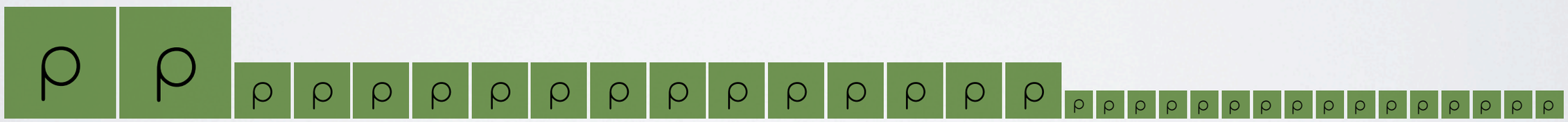


Symbolic units and unit conversion.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

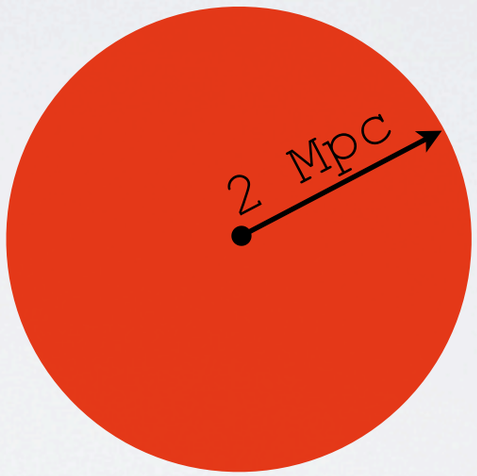



```
sp["density"].in_units("g/cm**3")
```

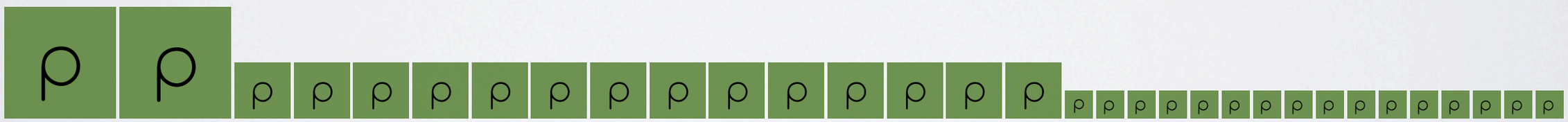


Symbolic units and unit conversion.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

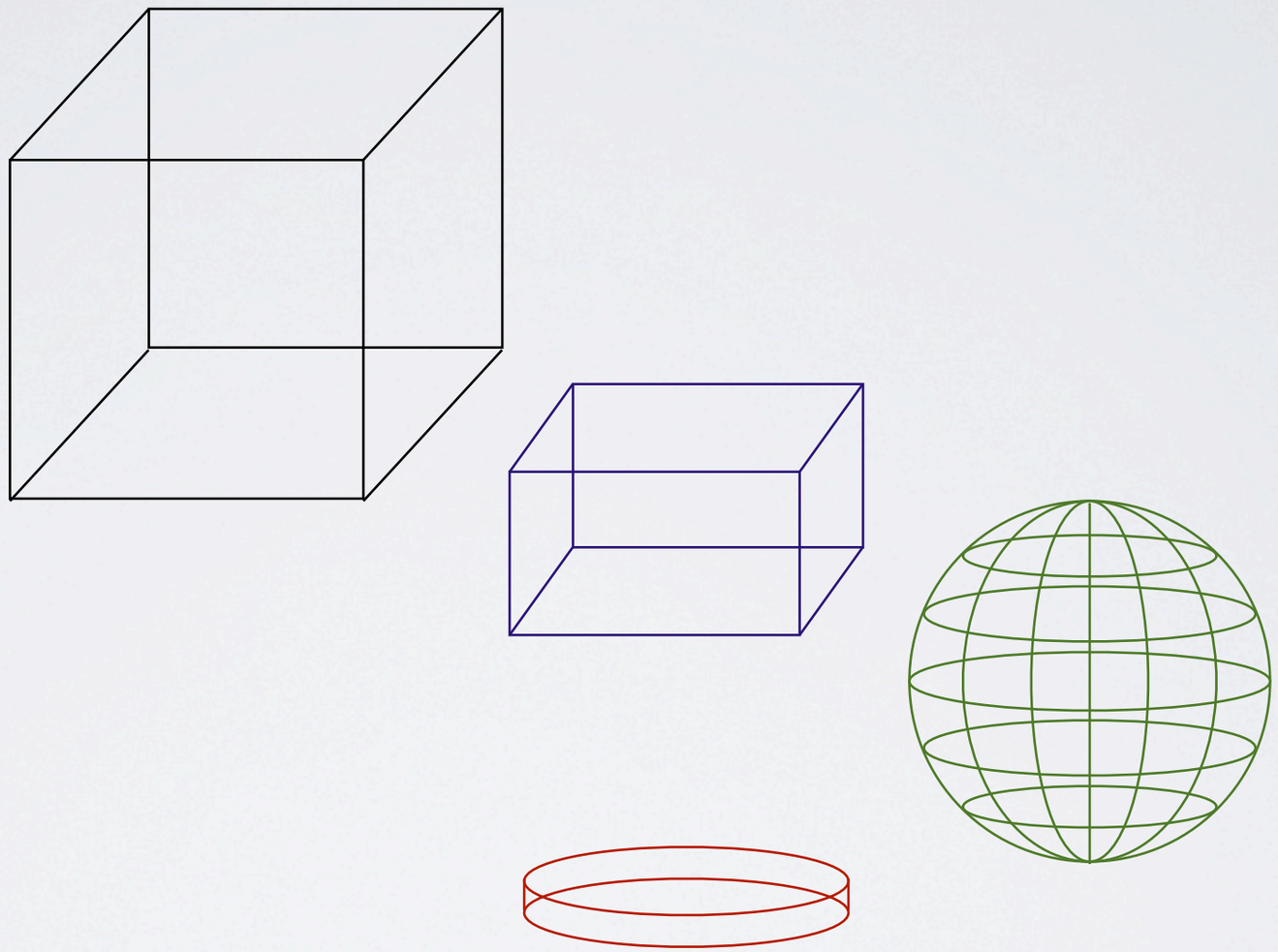



```
sp["density"].in_units("Msun/kpc**3")
```



DATA CONTAINERS

- All Data
- Region
- Sphere
- Disk
- Ray

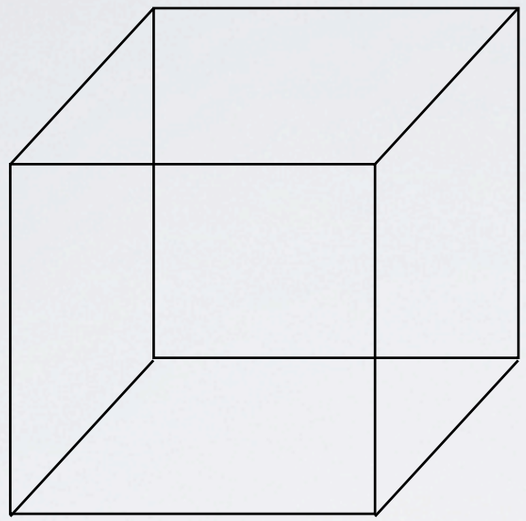


```
import yt
```

```
ds = yt.load("workshop2014/sample_data/
IsolatedGalaxy/galaxy0030/galaxy0030")
```

DATA CONTAINERS

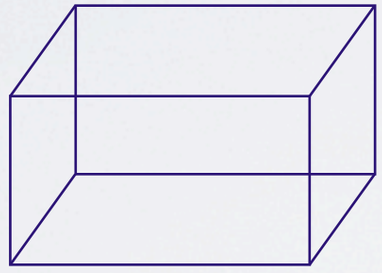
- All Data
-
-
-



```
ds.all_data()
```

DATA CONTAINERS

- Region



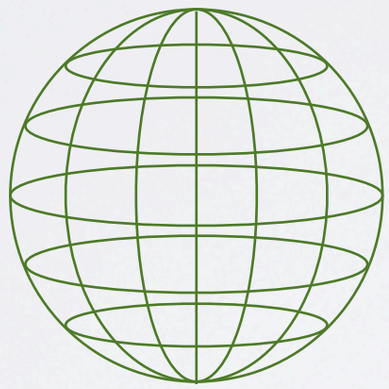
```
ds.region(center,
          left_corner,
          right_corner)
```

or

```
ds.box(left_corner,
       right_corner)
```

DATA CONTAINERS

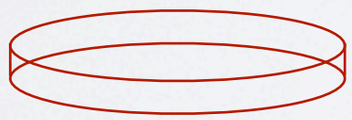
-
-
- Sphere



```
ds.sphere(center, radius)
```

-
-

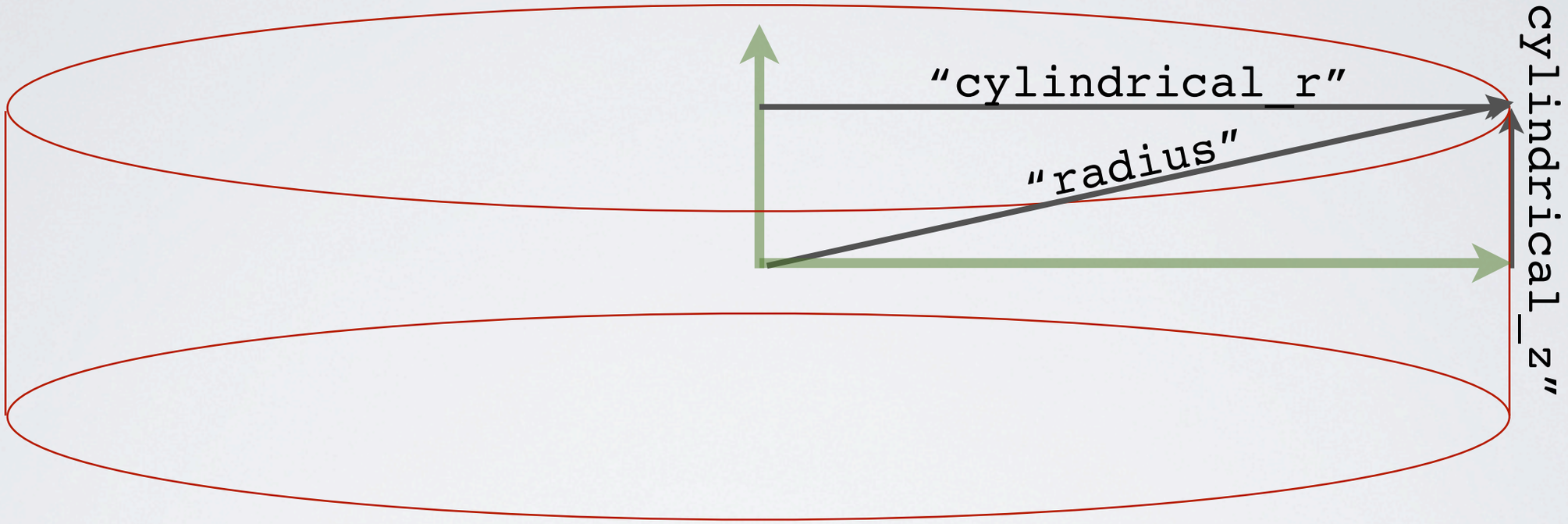
DATA CONTAINERS

-
-
-
- Disk 

```
ds.disk(center, normal_vector,
        radius, height)
```

-

Geometry-specific Fields



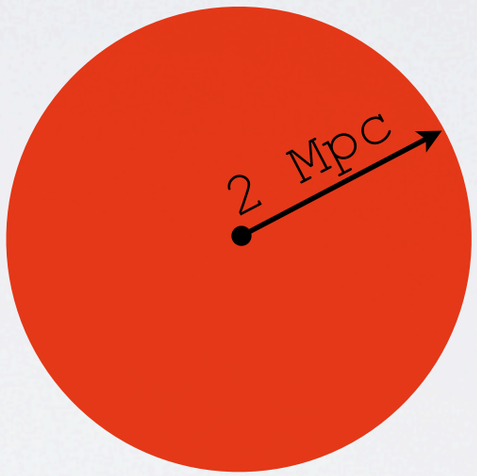
DATA CONTAINERS

-
-
-
-
- Ray

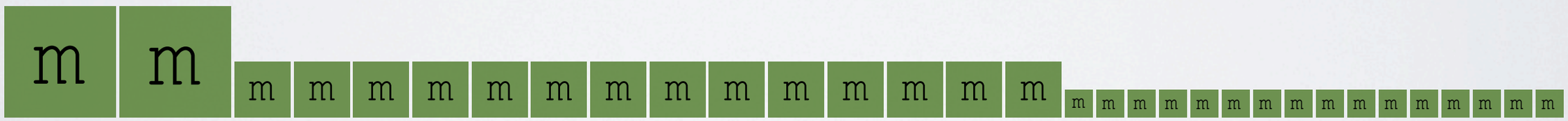
```
ds.ray(start_point, end_point)
```

Derived quantities turn fields into single values.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

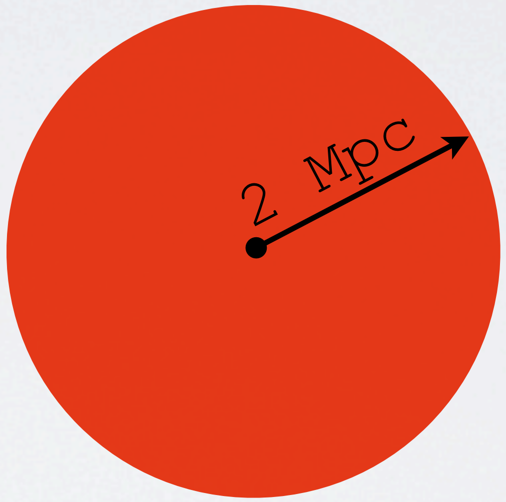



```
sp["cell_mass"]
```



Derived quantities turn fields into single values.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```




```
sp.quantities.total_quantity("cell_mass")
```

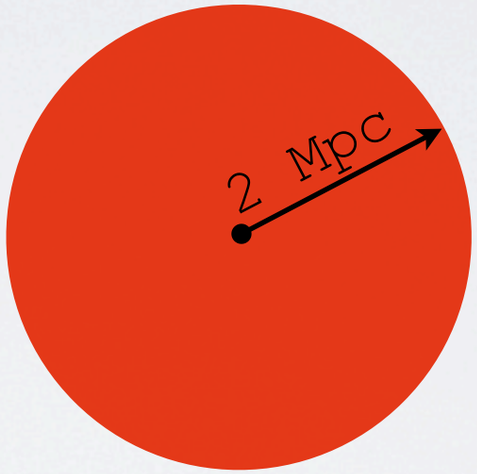
m + m + m + m + m + m + m + m + m + m + m + m + m +

m + m + m + m + m + m + m + m + m + m + m + m + m + m + m + m + m +

m + m = M

Derived quantities turn fields into single values.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```

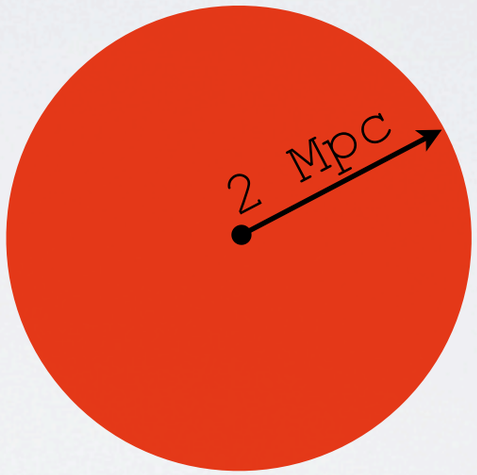



```
sp.quantities.total_quantity("cell_mass")
```

$$M = \sum m_i$$

Derived quantities turn fields into single values.

```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))
```




```
sp.quantities.spin_parameter()
```

$$M = \frac{\sum L_i \left| \sum E_i^{1/2} \right|}{G \sum m_i^{5/2}}$$

CHALLENGE 1

1. Create a sphere centered on the center of mass of the full dataset.
2. Create a disk centered on the sphere's center of mass and aligned with the sphere's angular momentum vector.
3. Make "face-on" and "edge-on" density projections of the disk.
4. Compare the following quantities for the sphere and the disk: spin parameter, mass-weighted average temperature.

BONUS: use HiresIsolatedGalaxy/DD0044/DD0044

PROFILES AND PHASE PLOTS

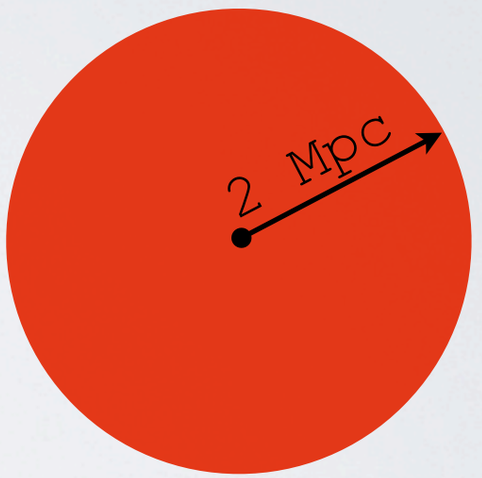
DERIVED FIELDS

Creating new fields is easy.

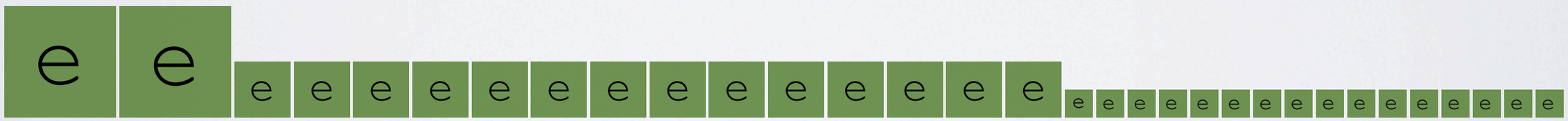
```
ds = load("DD0252/DD0252")
sp = ds.sphere(center, (2, "Mpc"))

def my_field(field, data):
    return k * data["temperature"] * \
      data["number_density"]**(-2./3)

ds.add_field("entropy", function=my_field,
             units="keV*cm**2")
```




```
sp["entropy"]
```



CHALLENGE 2

1. Use the disk object to compare profiles of density vs. radius and height.

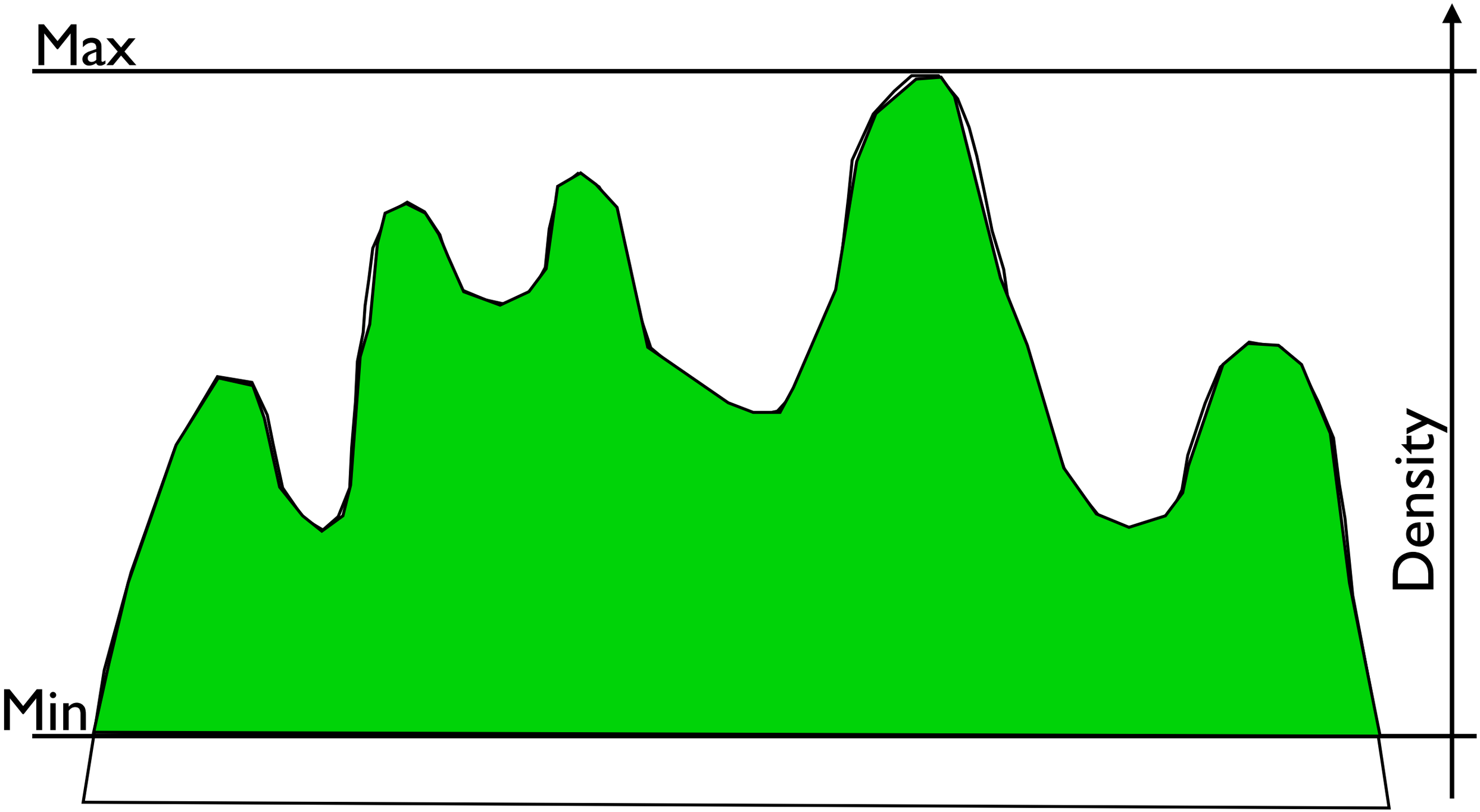
2. Make a phase plot of the mass-weighted average metallicity in bins of radial velocity vs. rotational velocity ("tangential_velocity") in a disk.

3. Make an interesting new field.

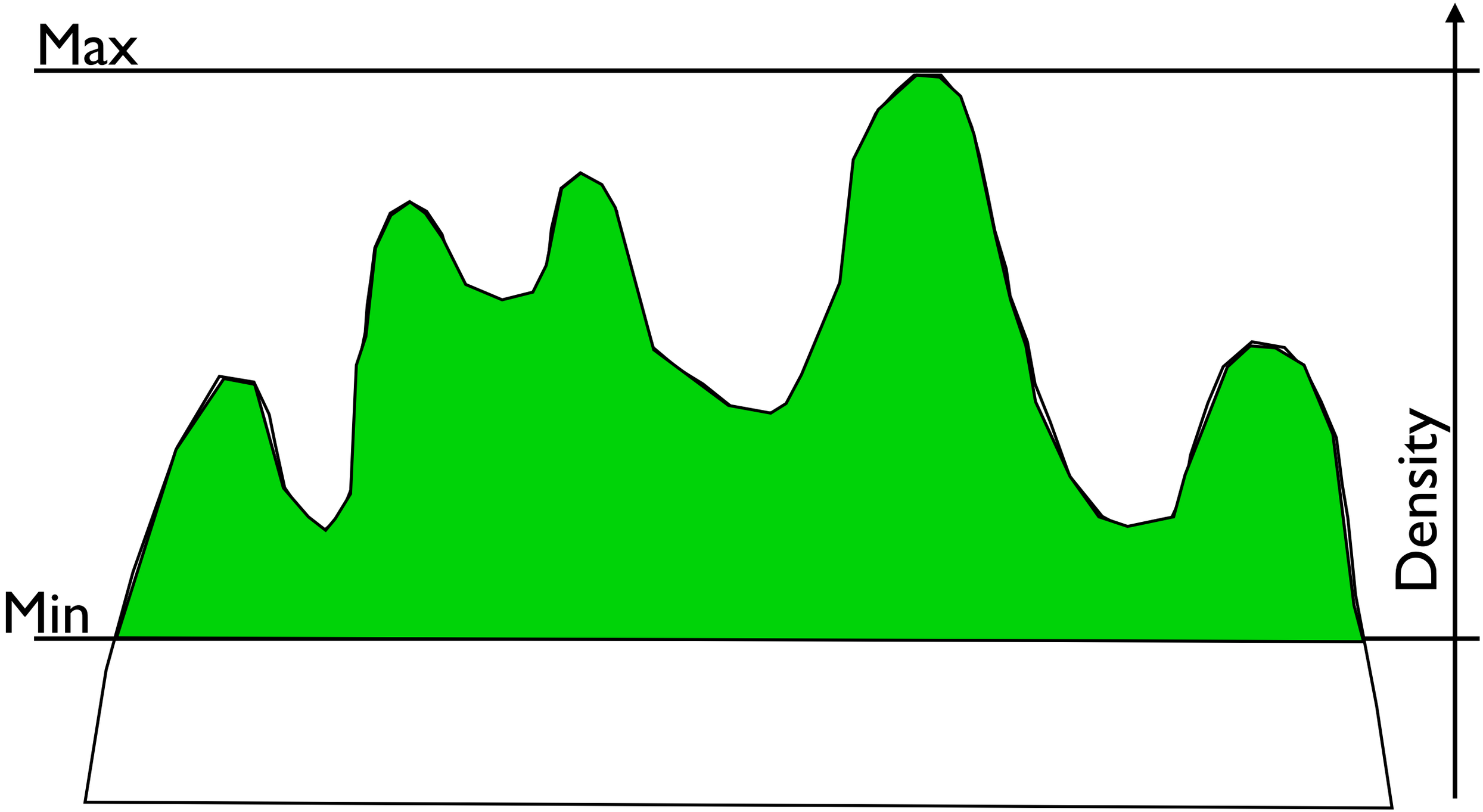
BONUS: use HiresIsolatedGalaxy/DD0044/DD0044

CLUMP FINDING

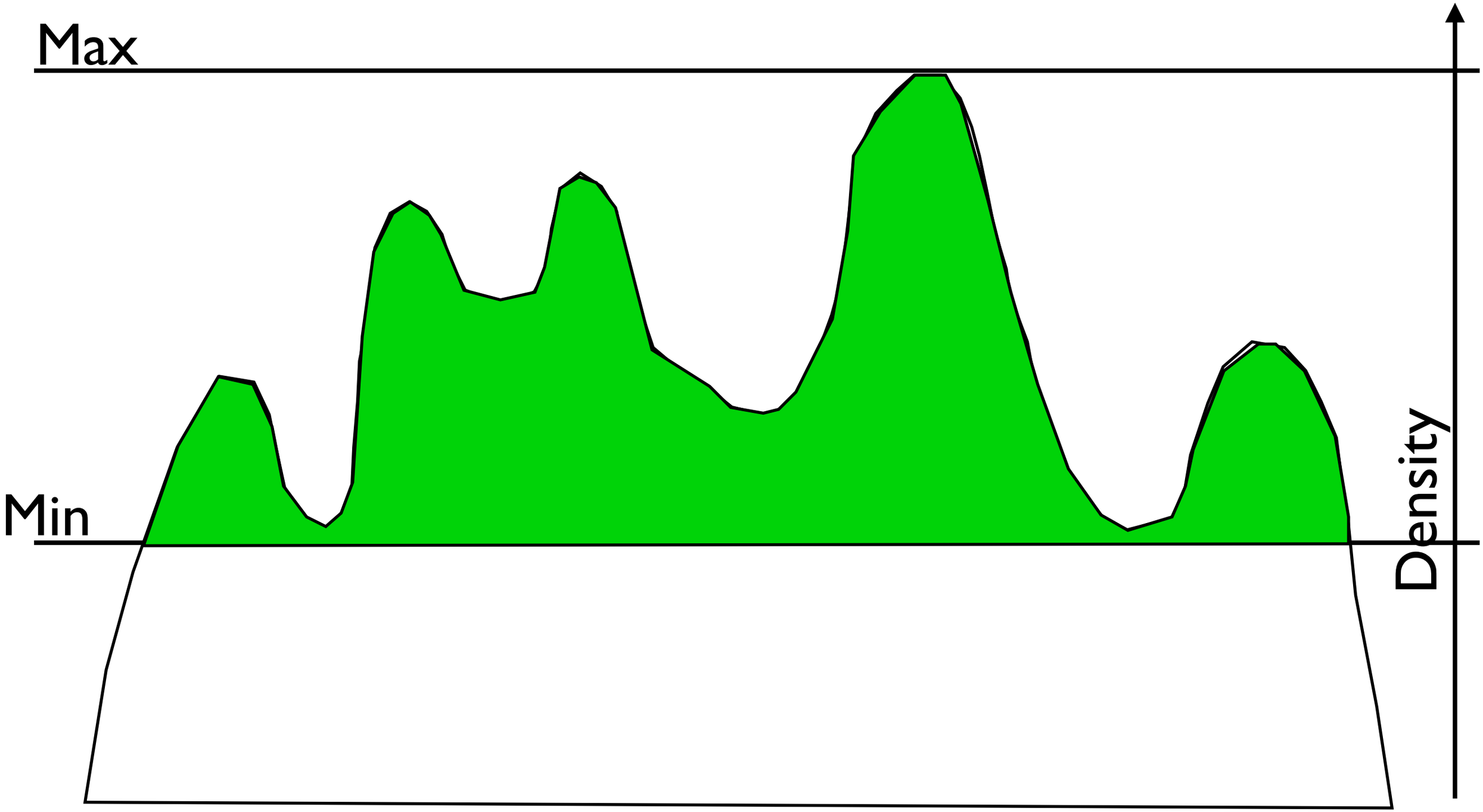
Finding Clumps



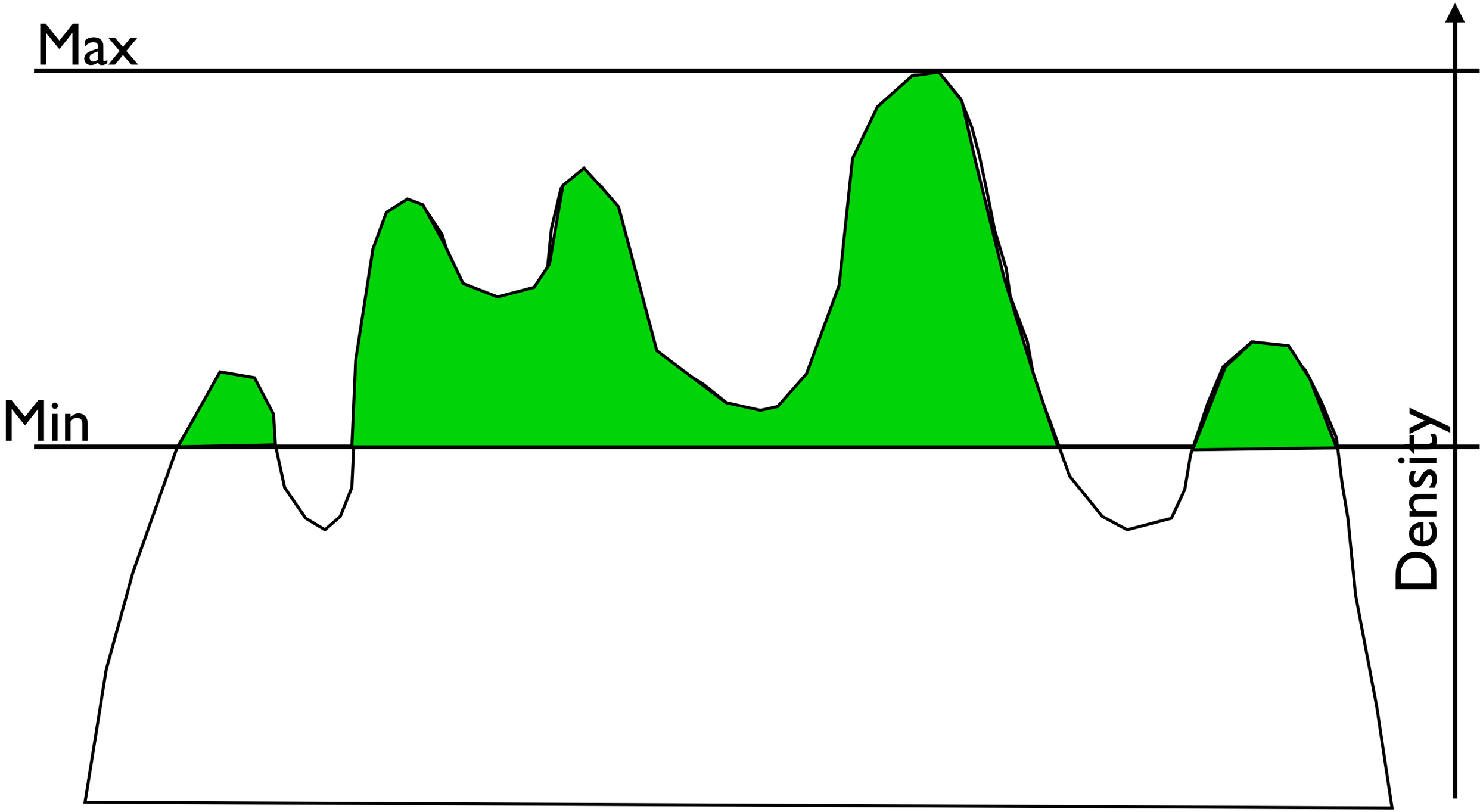
Finding Clumps



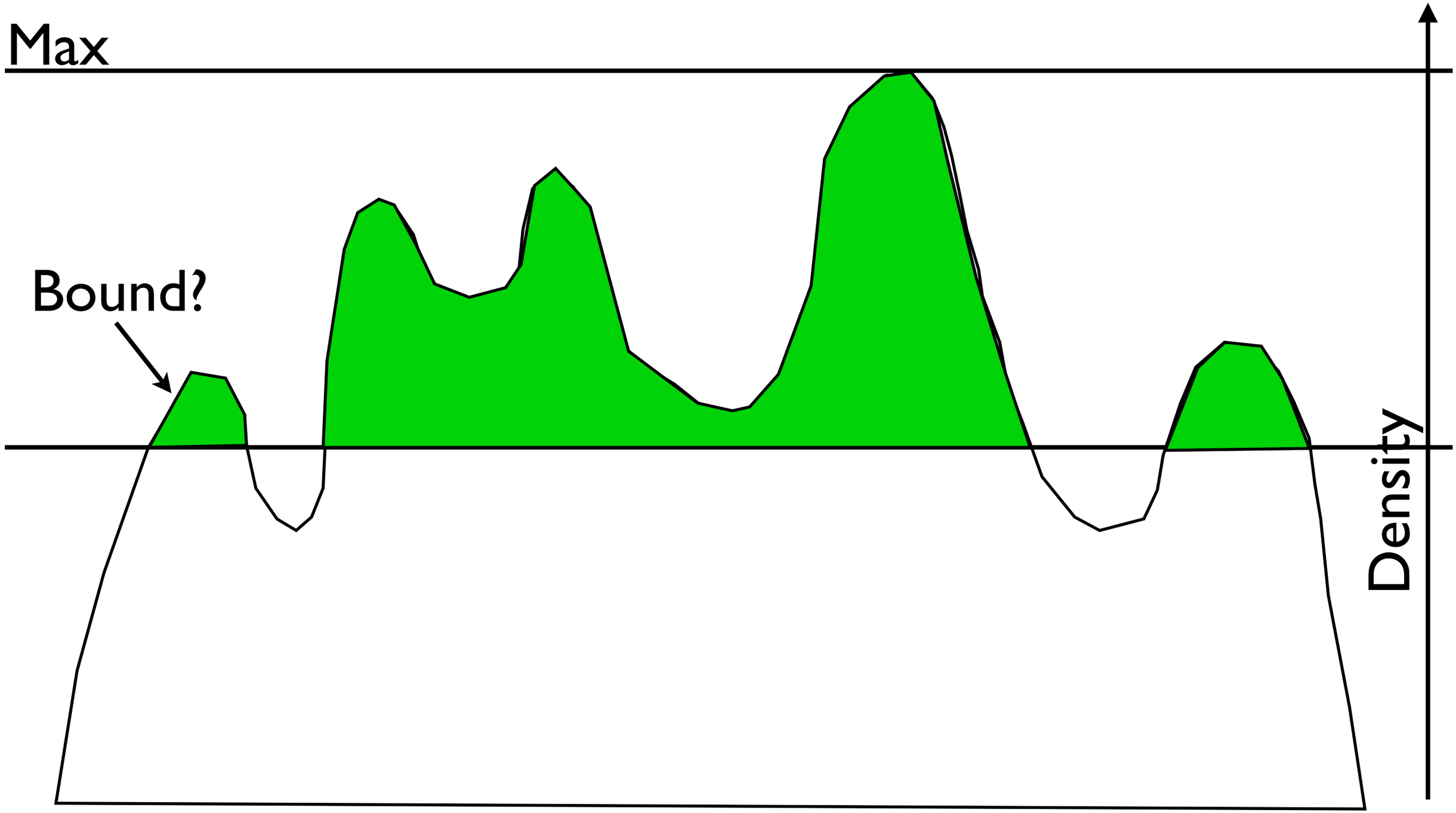
Finding Clumps



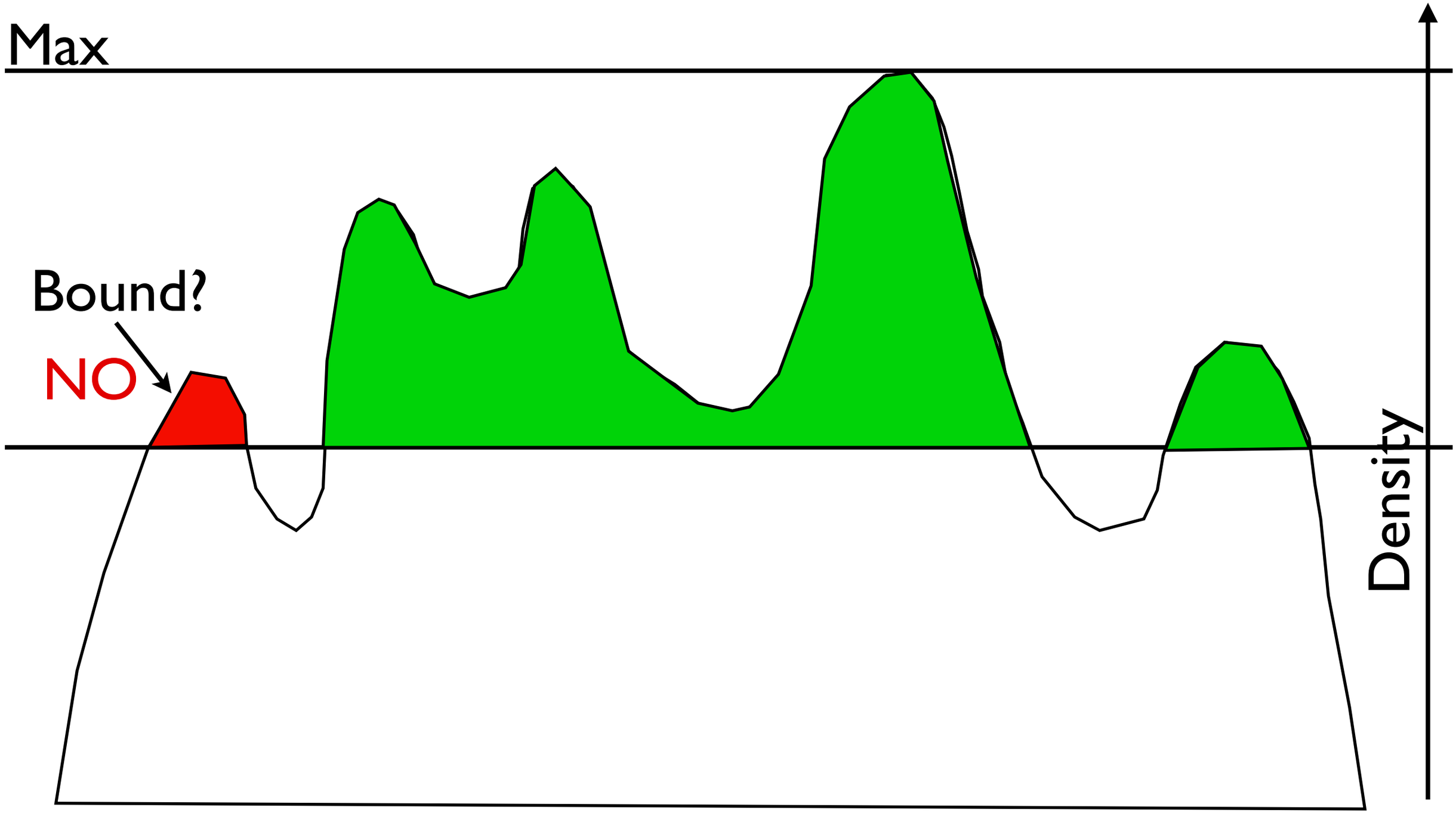
Finding Clumps



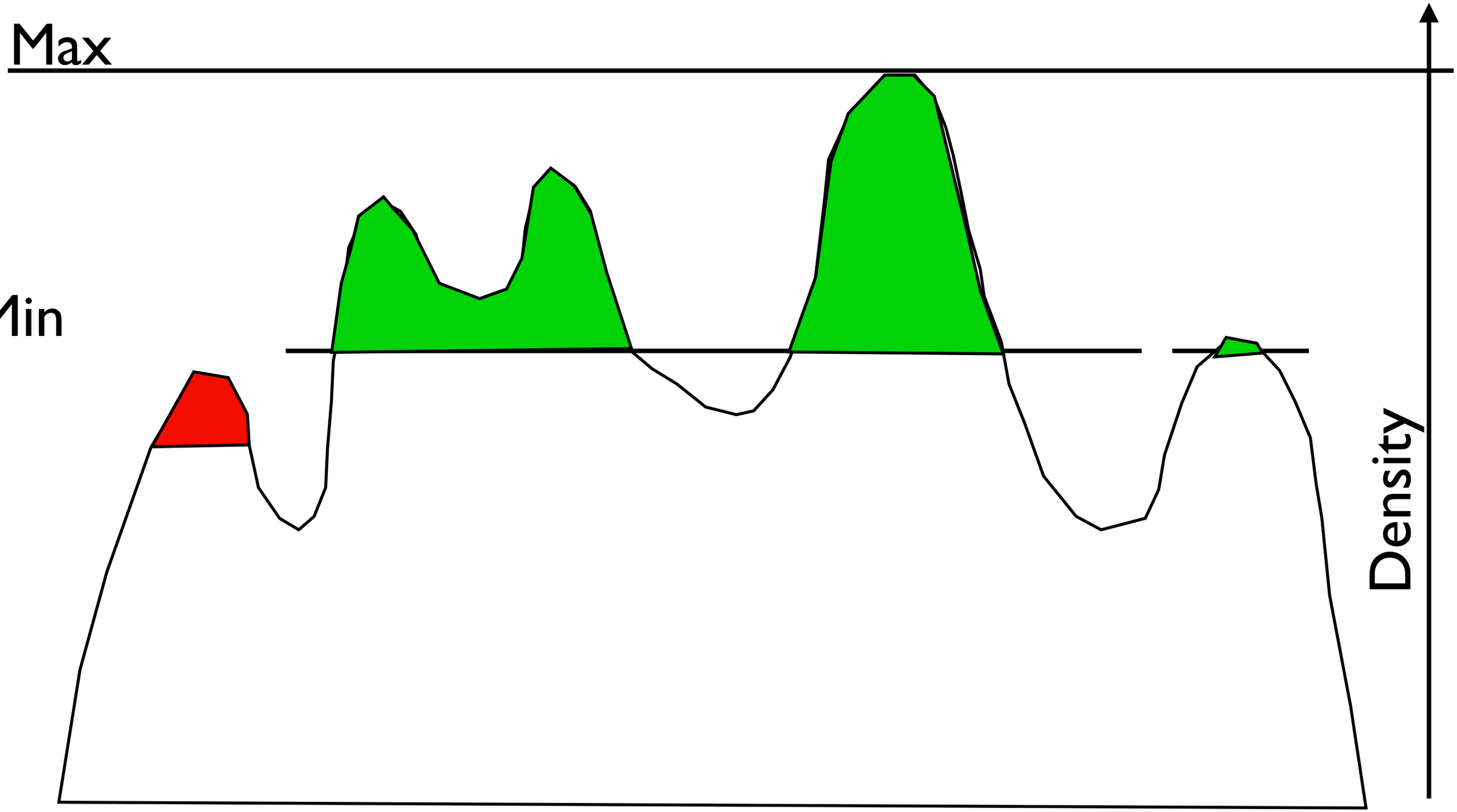
Finding Clumps



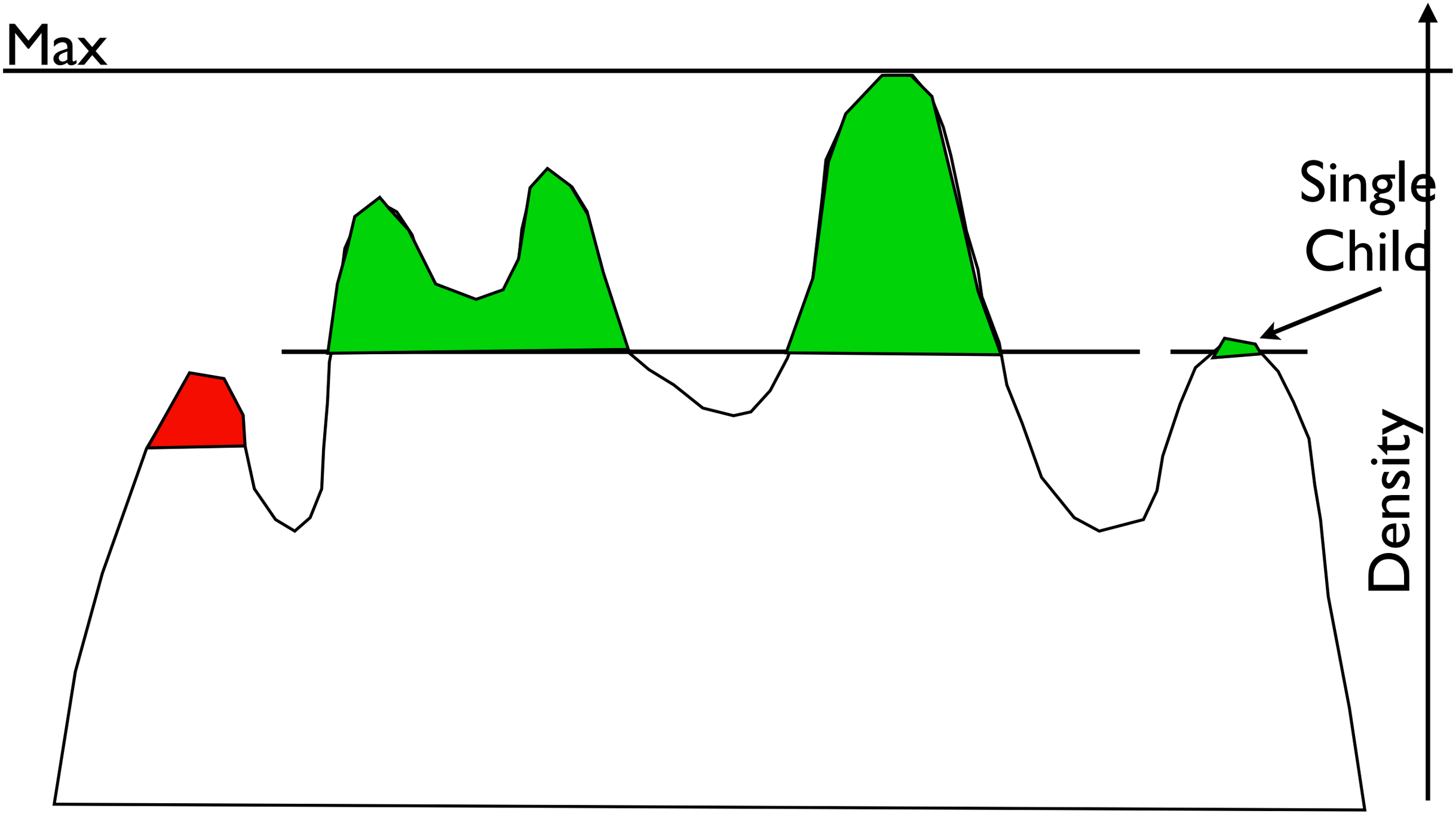
Finding Clumps



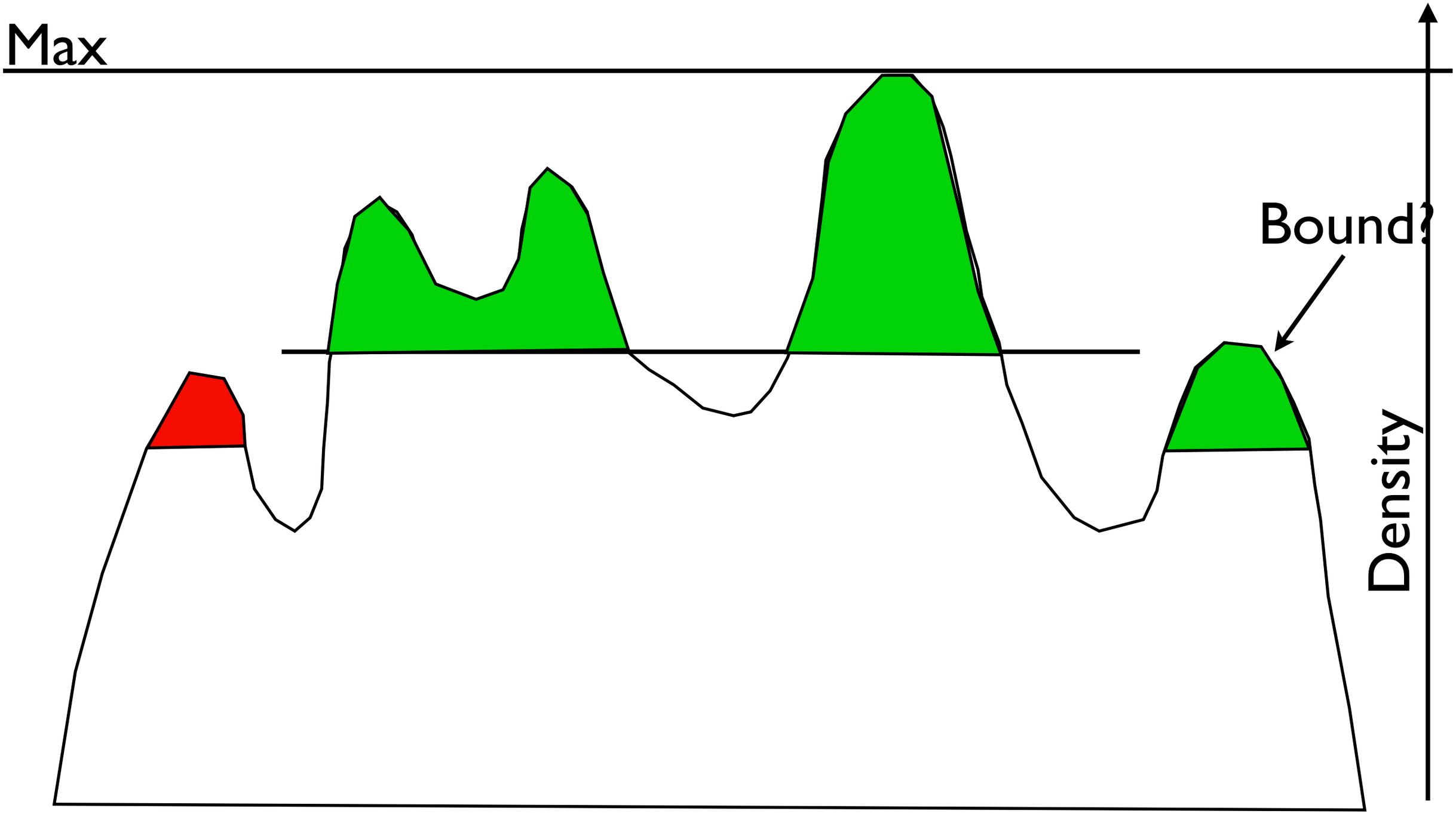
Finding Clumps



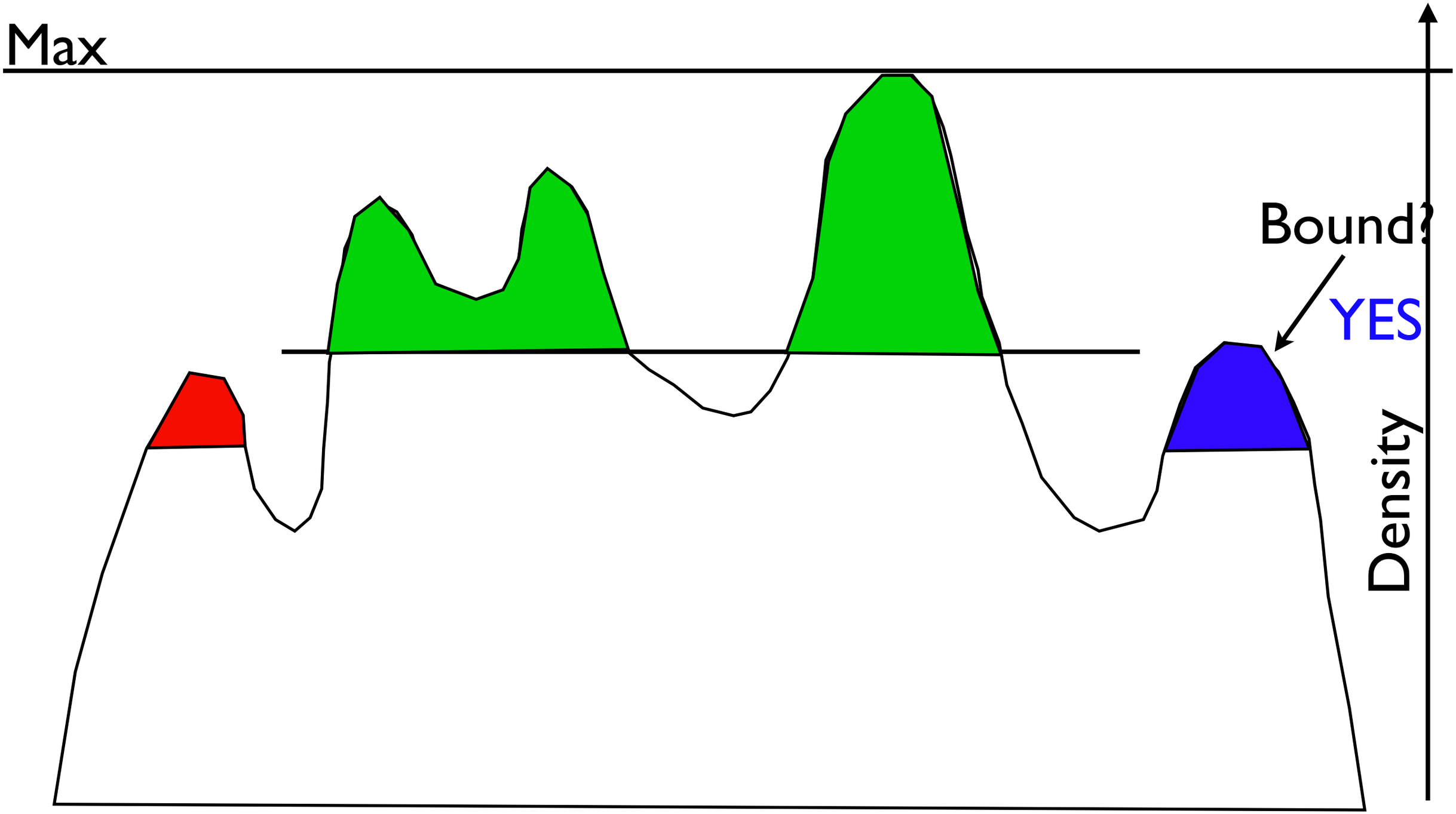
Finding Clumps



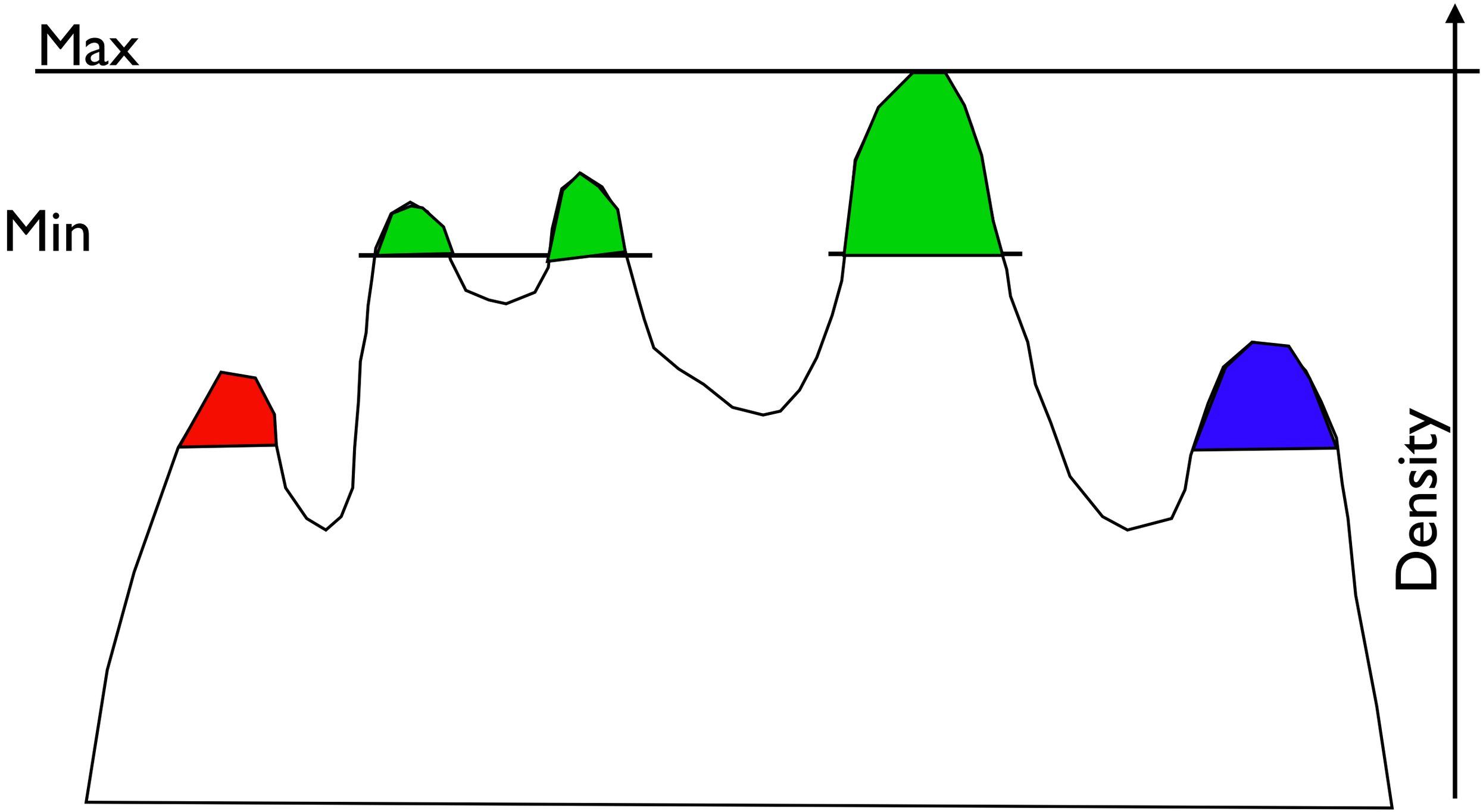
Finding Clumps



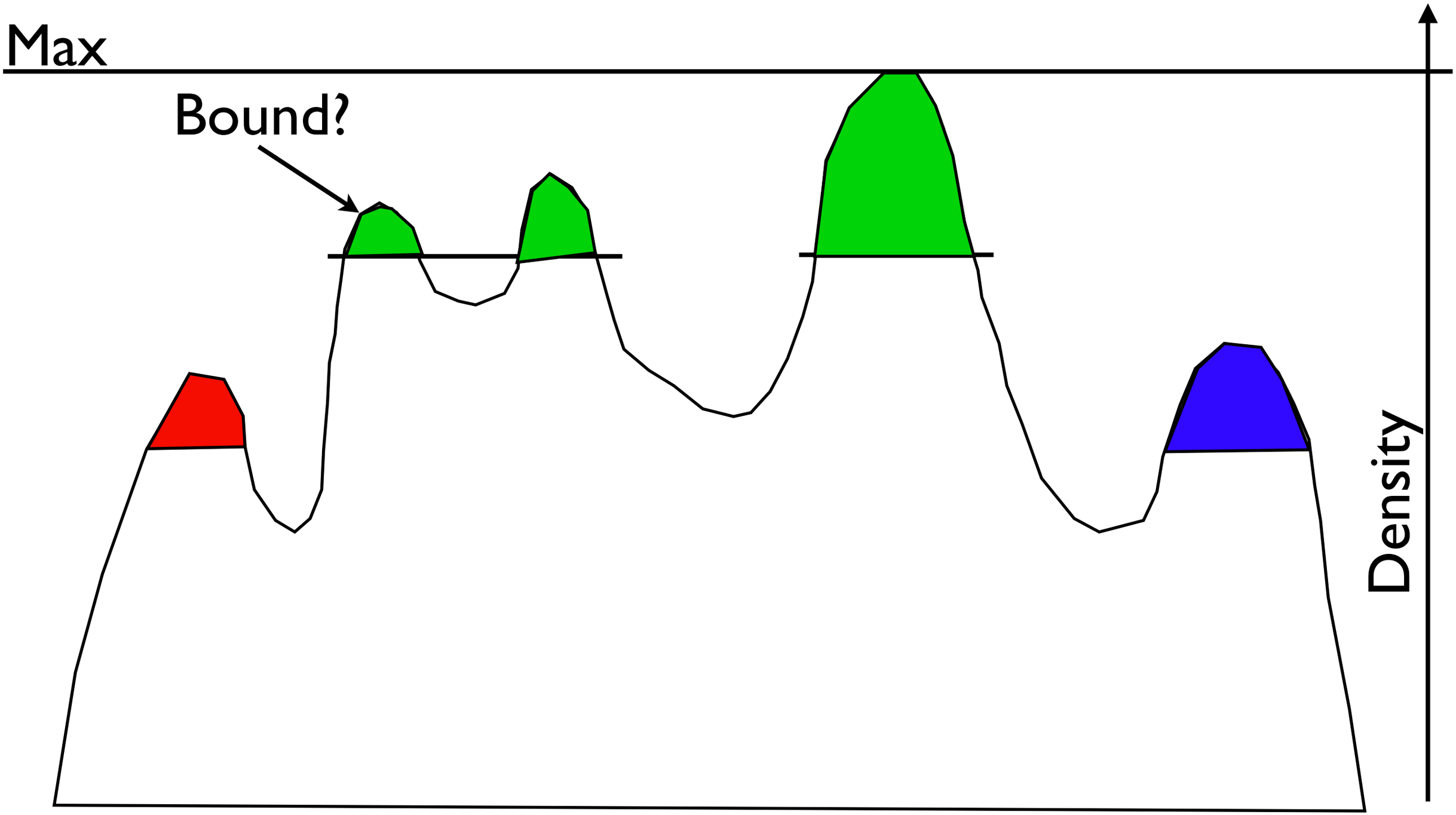
Finding Clumps



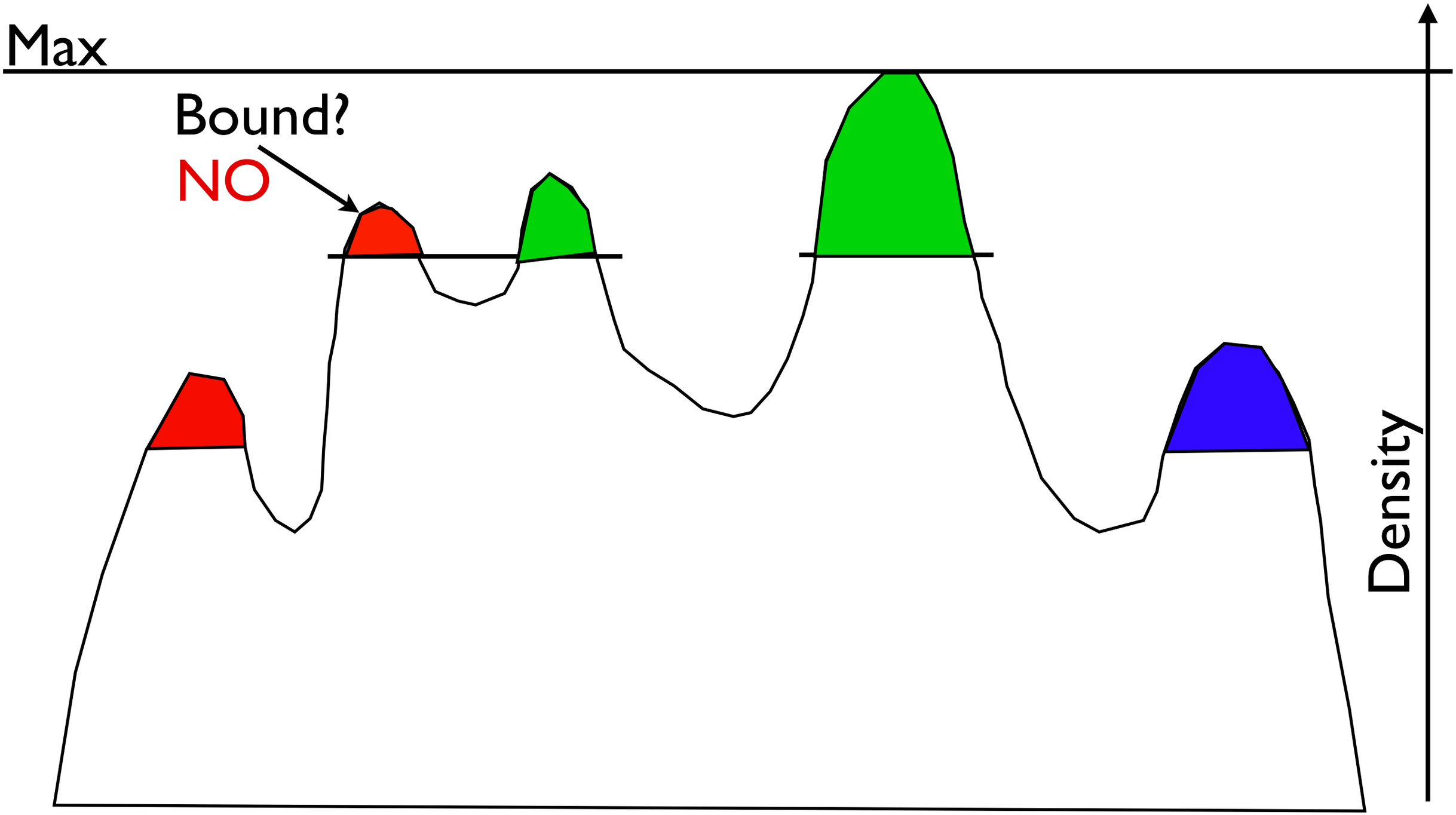
Finding Clumps



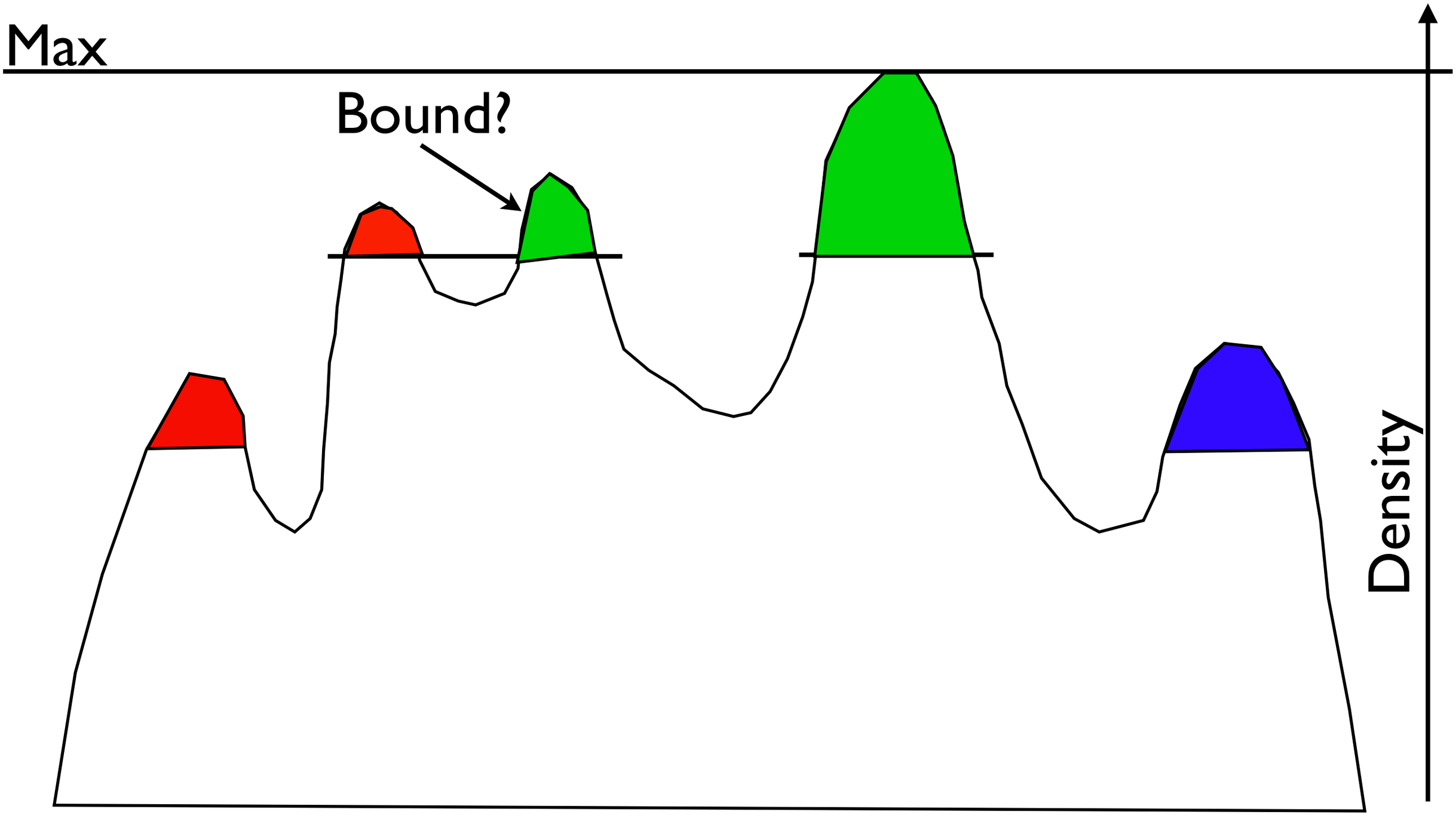
Finding Clumps



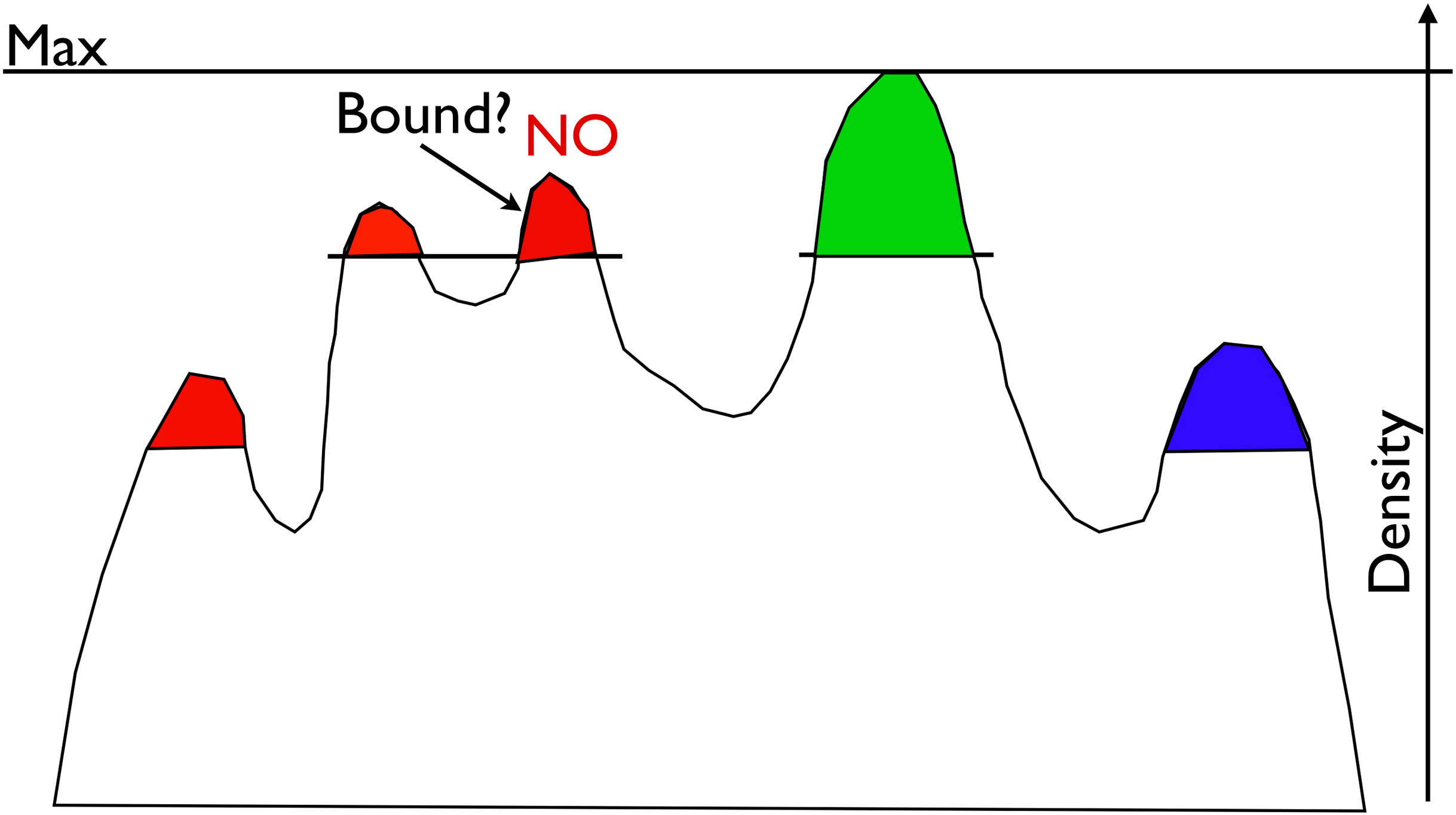
Finding Clumps



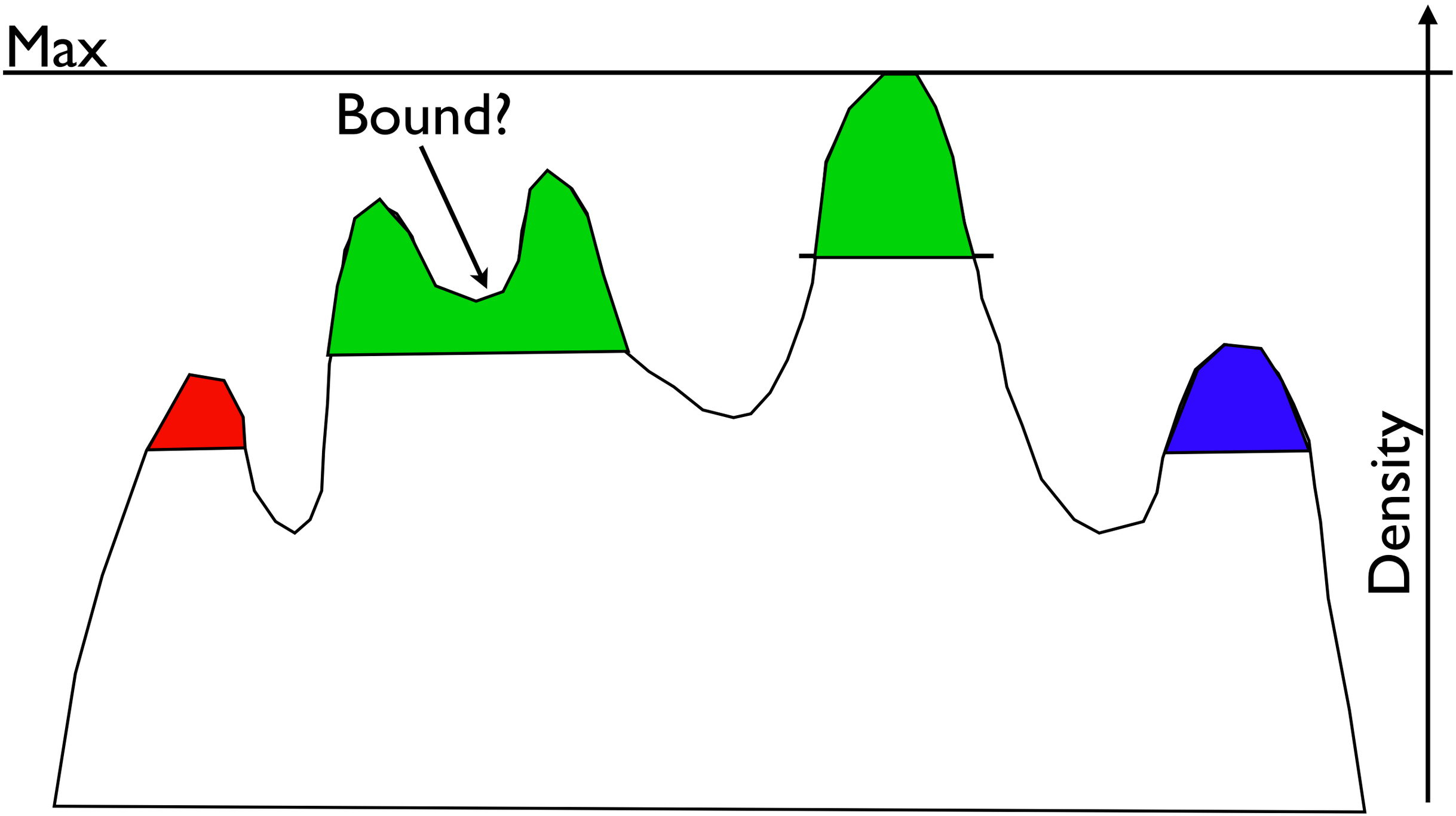
Finding Clumps



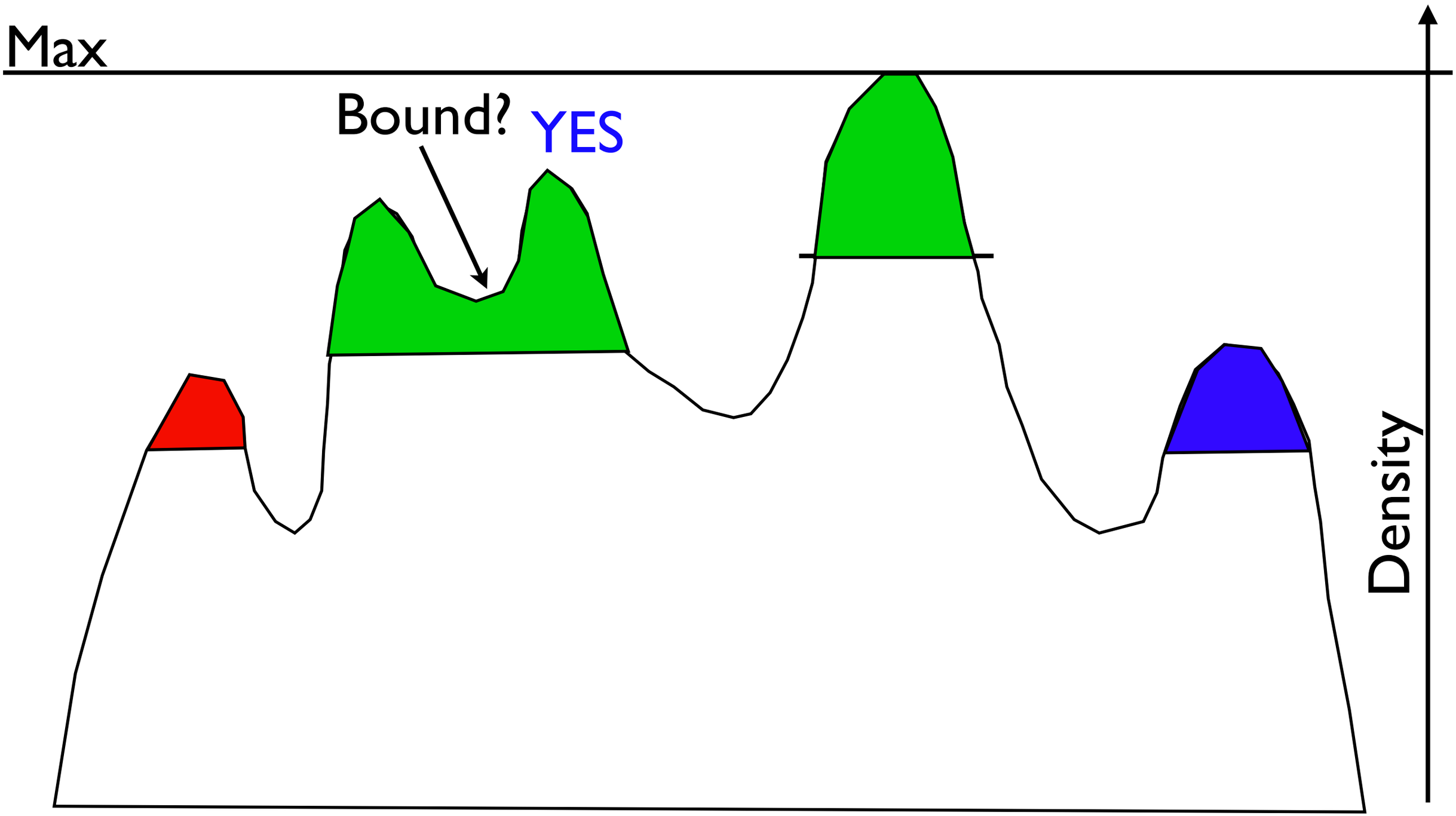
Finding Clumps



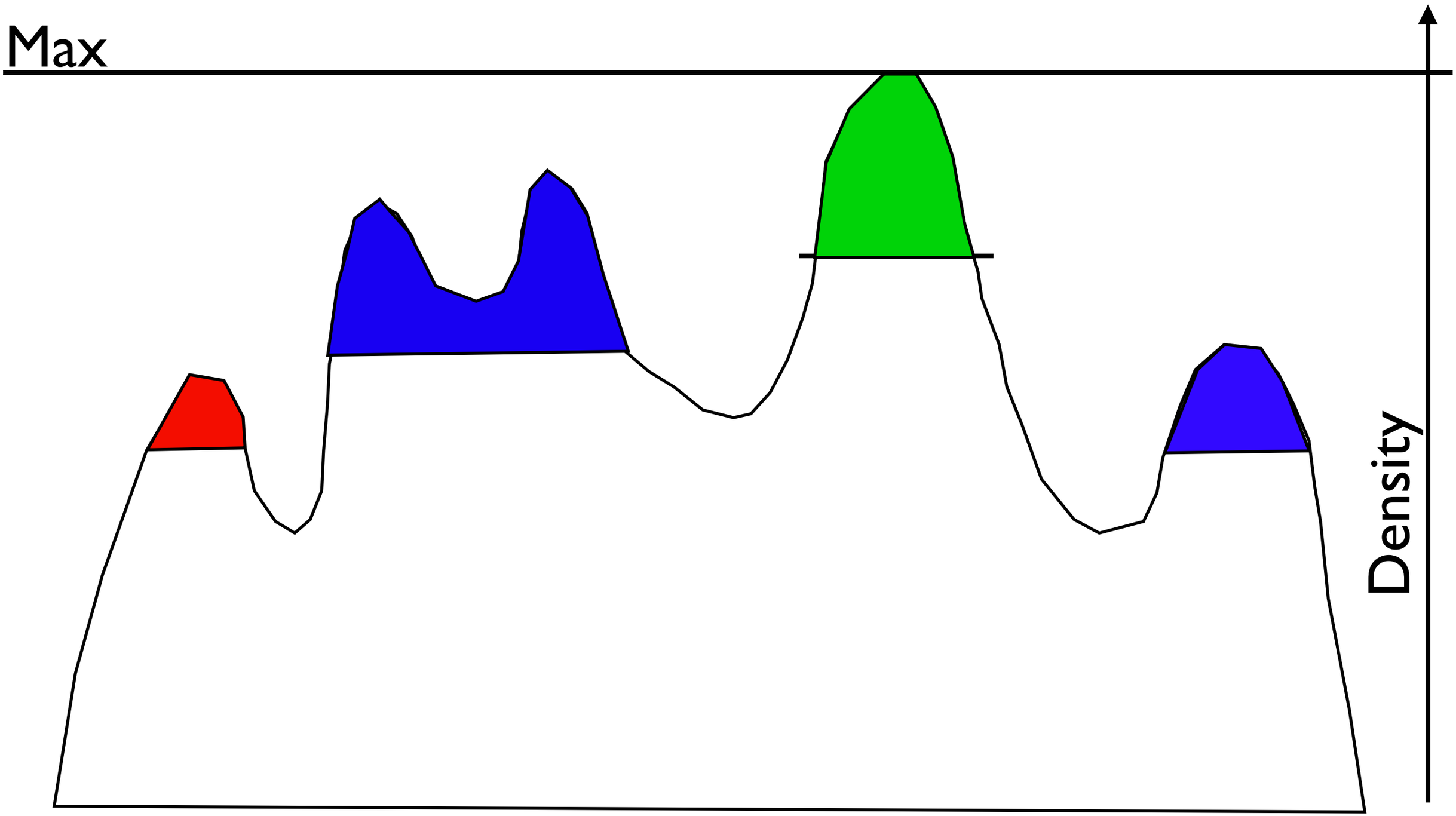
Finding Clumps



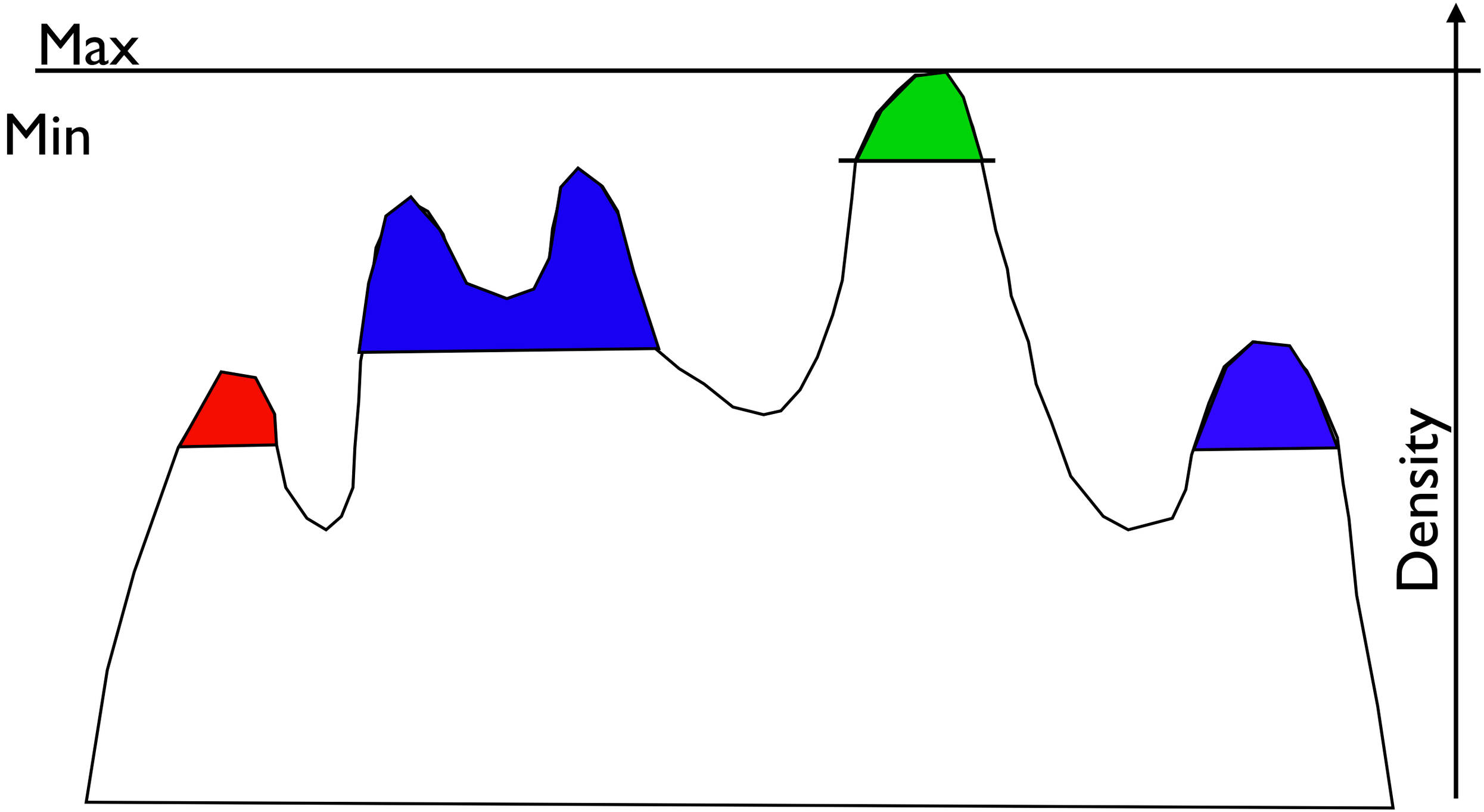
Finding Clumps



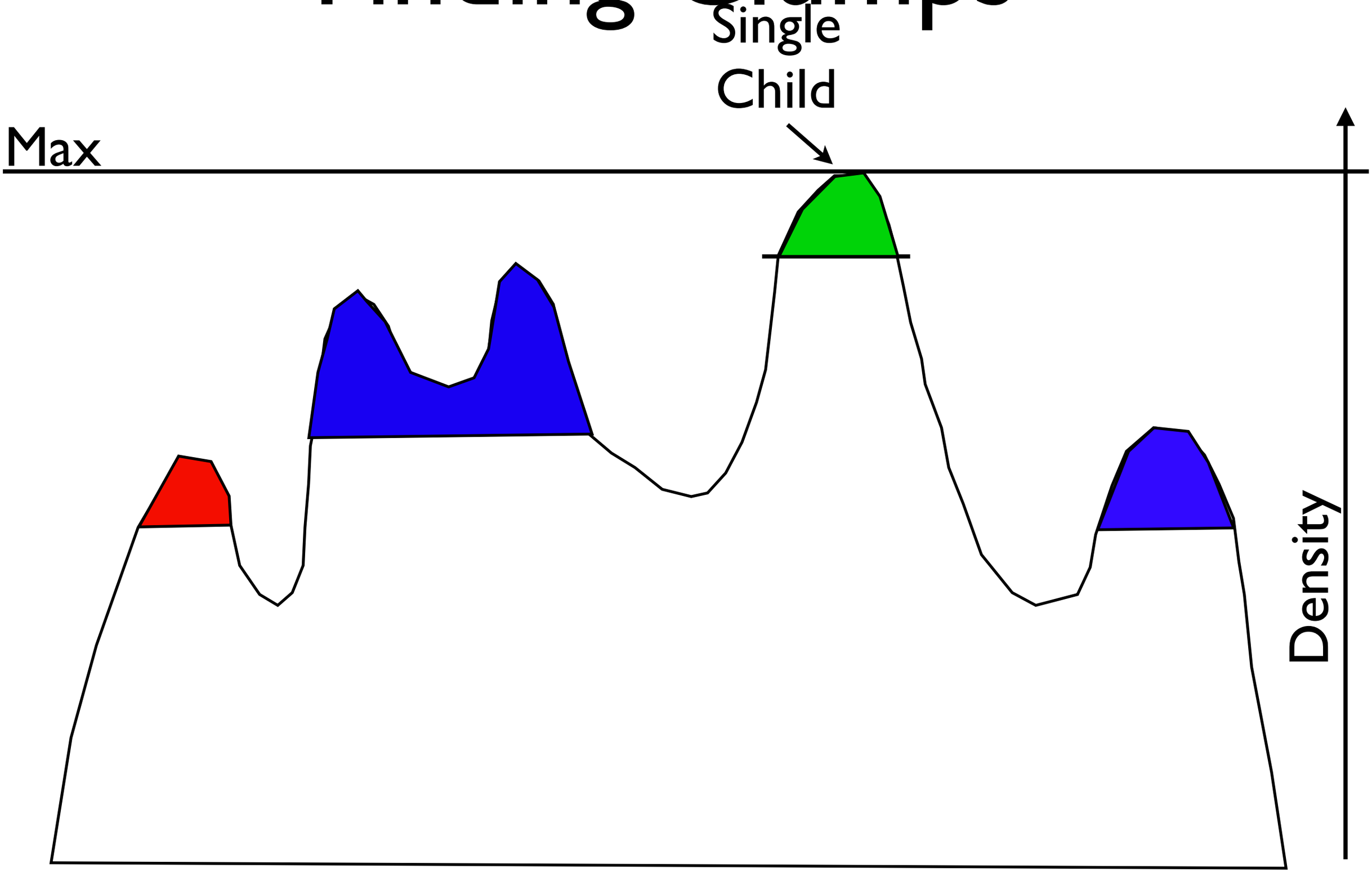
Finding Clumps



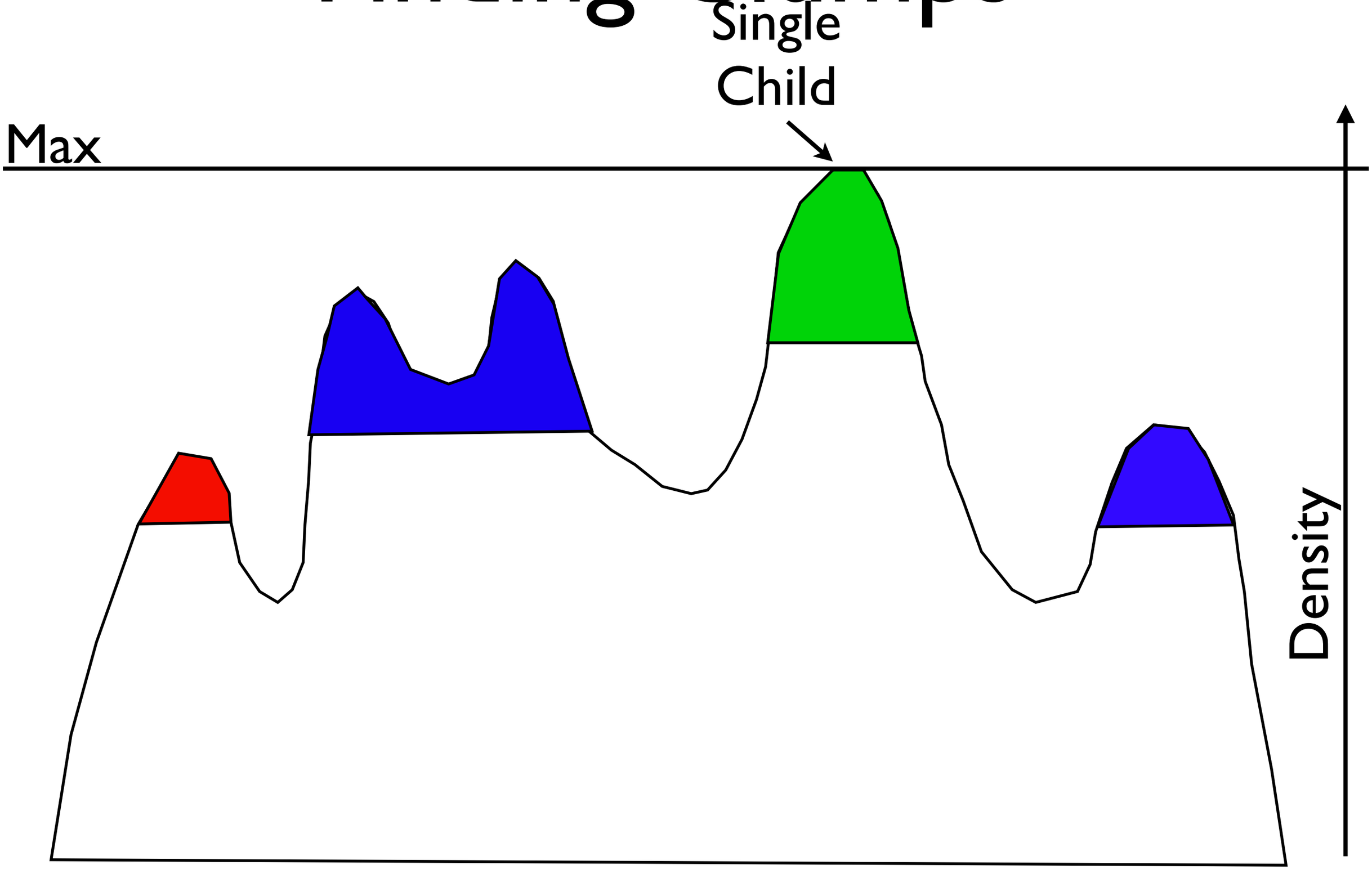
Finding Clumps



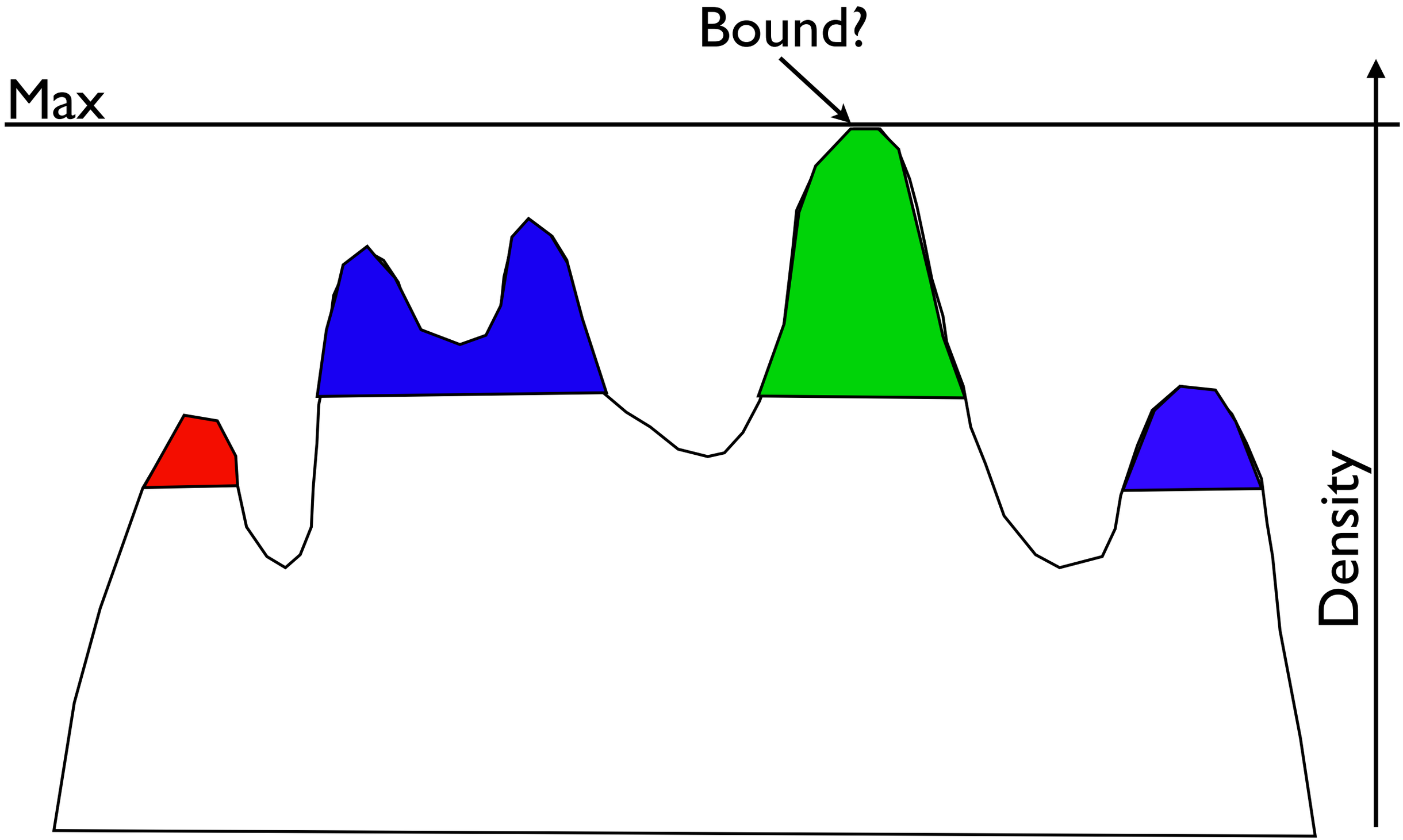
Finding Clumps



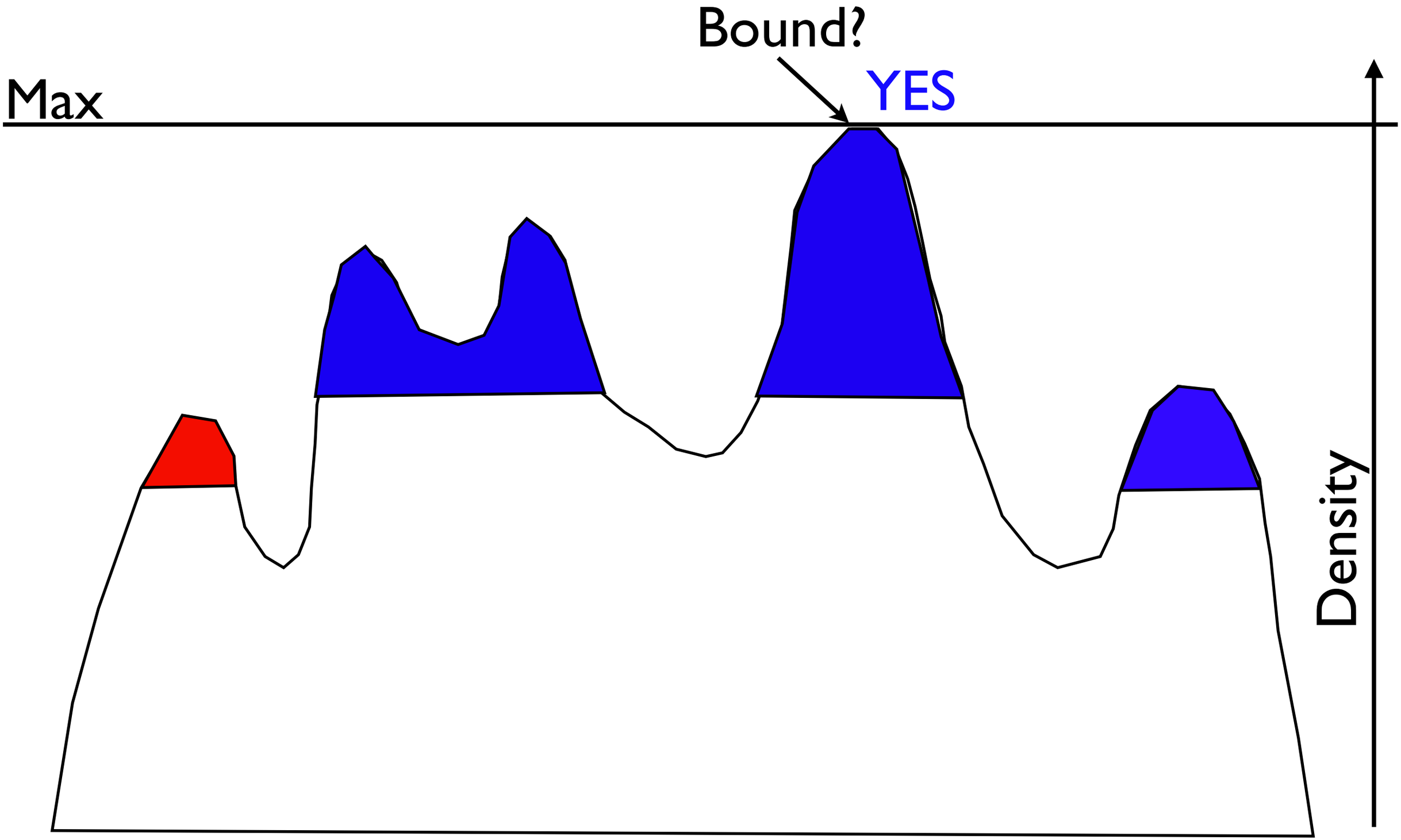
Finding Clumps



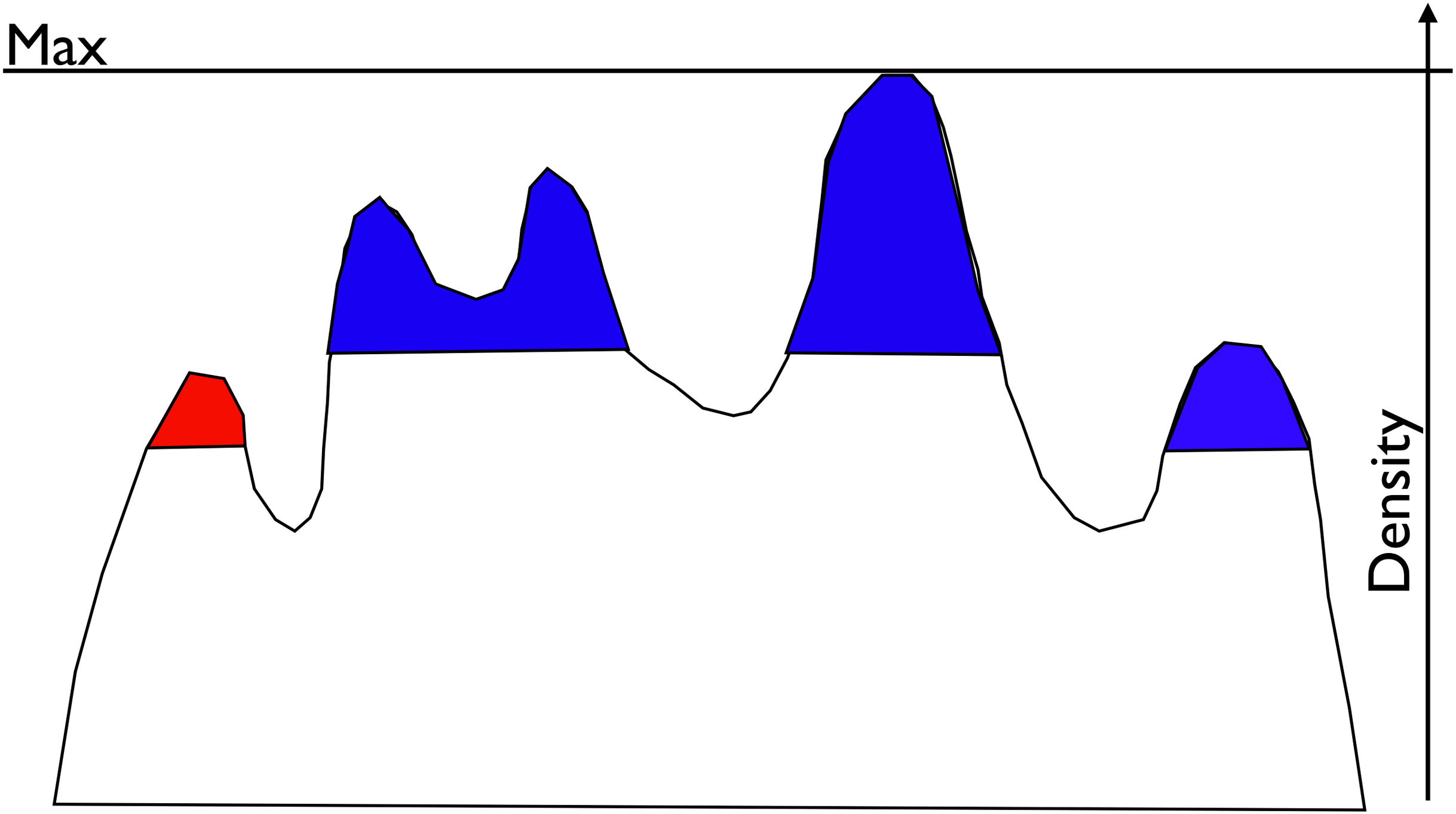
Finding Clumps



Finding Clumps



Finding Clumps



Clump Finding Recipe

cp /mnt/iscsi5/enzo_workshop/yt-x86_64/src/yt-hg/doc/source/cookbook/find_clumps.py .

The Cookbook

ls /mnt/iscsi5/enzo_workshop/yt-x86_64/src/yt-hg/doc/source/cookbook/

or

yt-project.org/doc/cookbook/

Supported Codes

Simulation Codes/Formats

ART
Athena
Boxlib
Chombo
Enzo
Flash
Eagle
OWLS
Gadget
GDF
Piernik
RAMSES
Tipsy

Halo Finders

FoF
HOP
Rockstar

Other Formats

array data
fits