

Unstructured Data Analysis

Lecture 8: More clustering

George Chen

A Sketch of How to Interpret Clusters

Demo

Automatically Choosing k

For $k = 2, 3, \dots$ up to some user-specified max value:

Fit model using k

Compute a score for the model

But what score function should we use?

Use whichever k has the best score

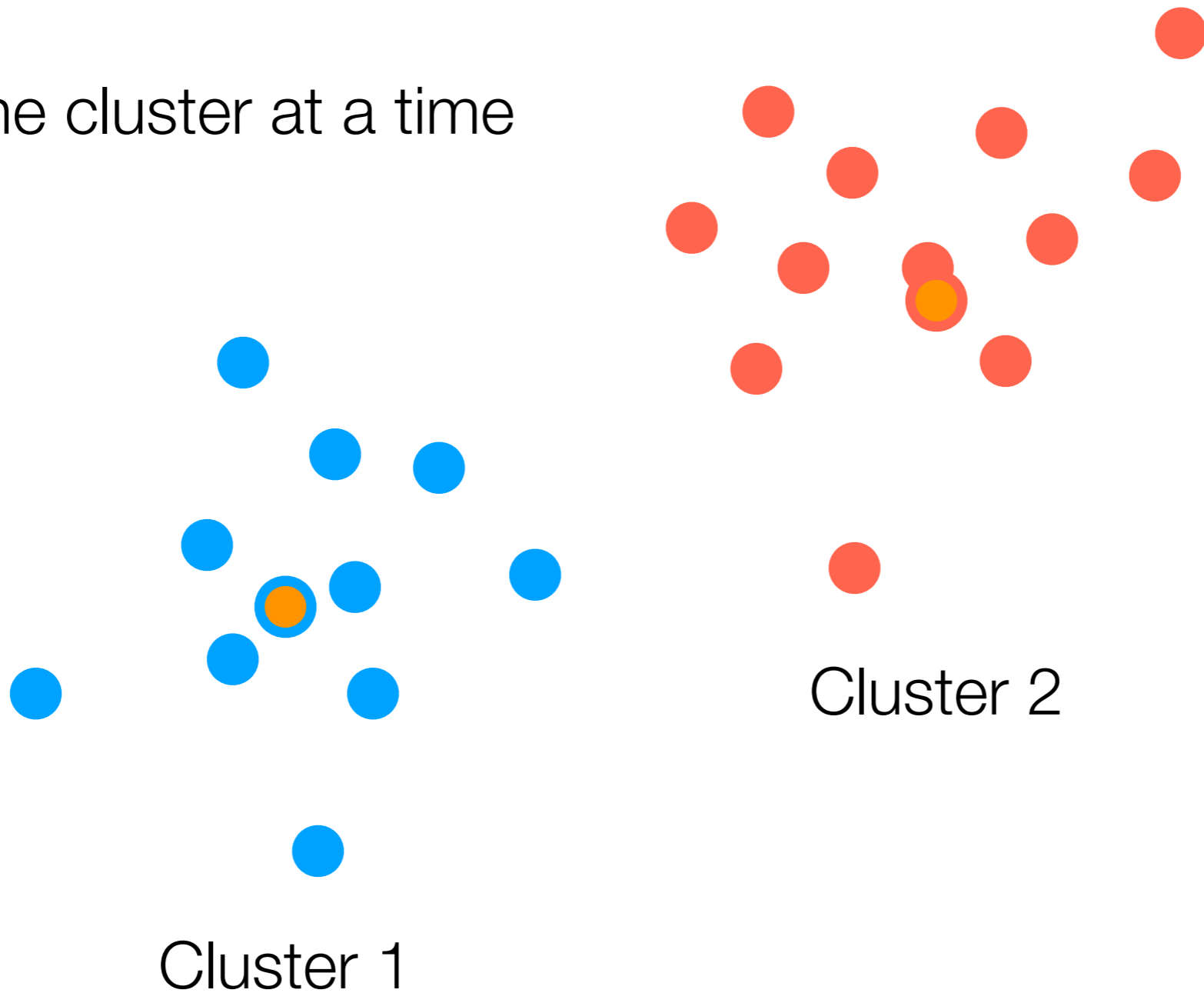
No single way of choosing k is the “best” way

**Here's an example of a score
function you don't want to use**

But hey it's worth a shot

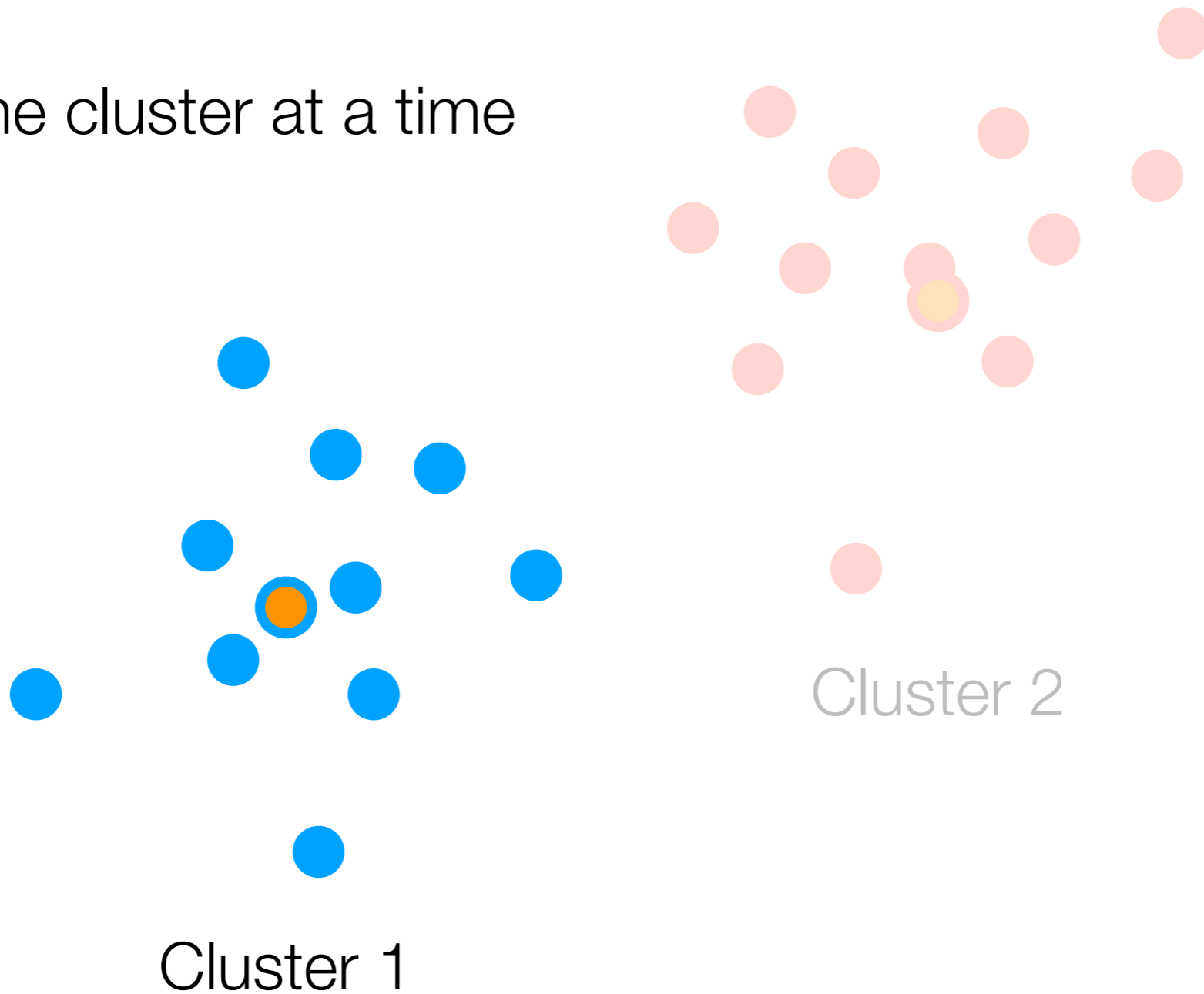
Residual Sum of Squares

Look at one cluster at a time



Residual Sum of Squares

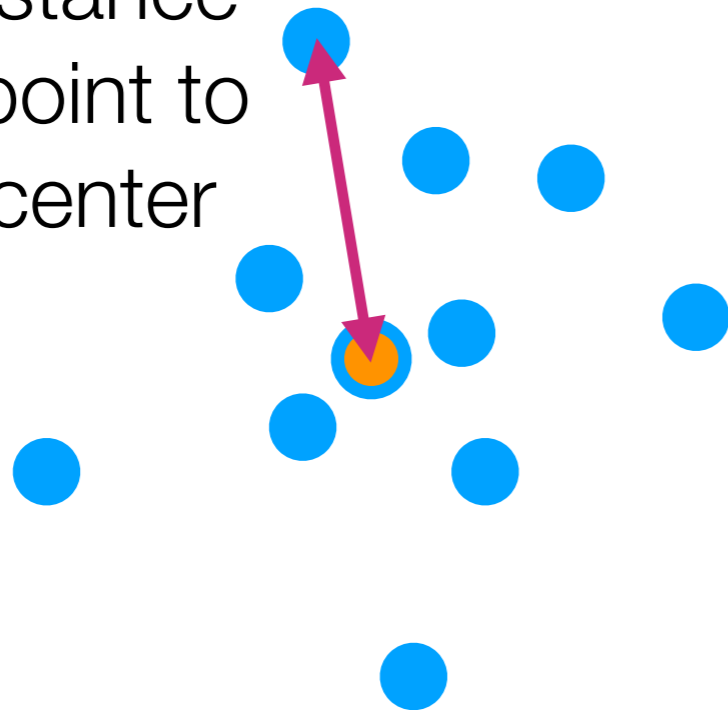
Look at one cluster at a time



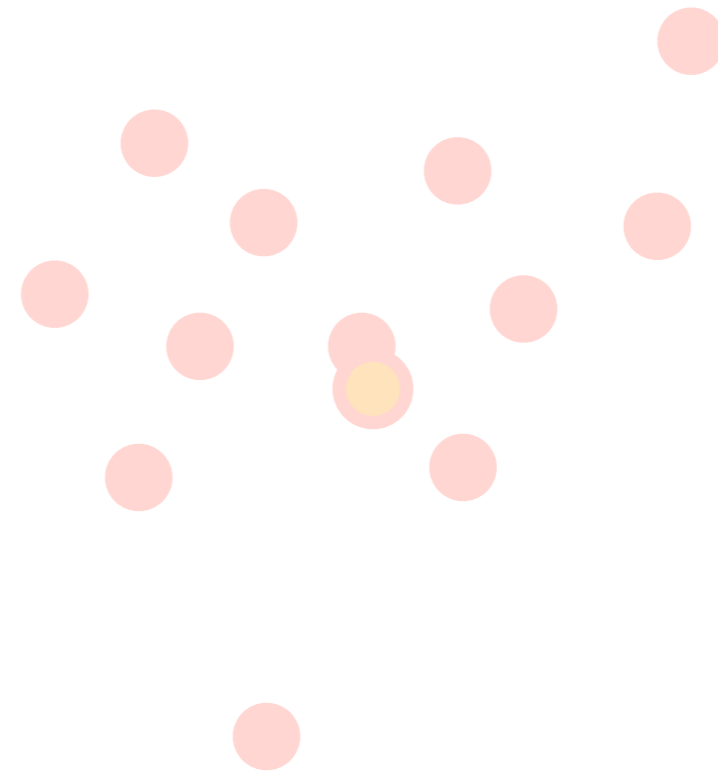
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

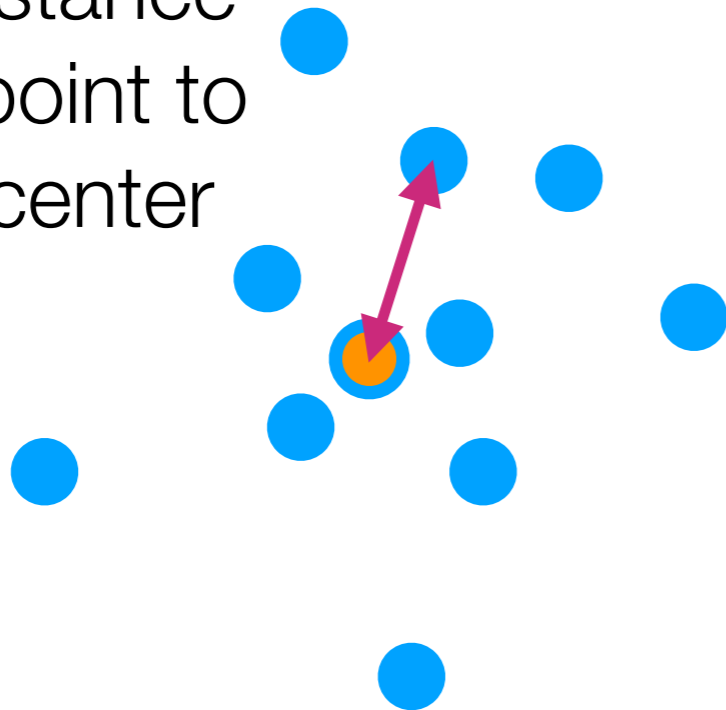


Cluster 2

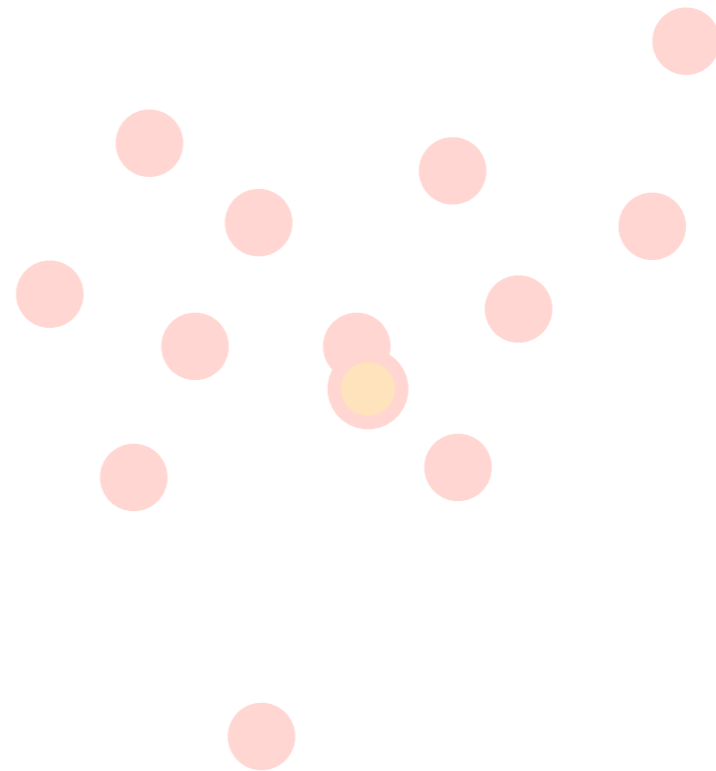
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

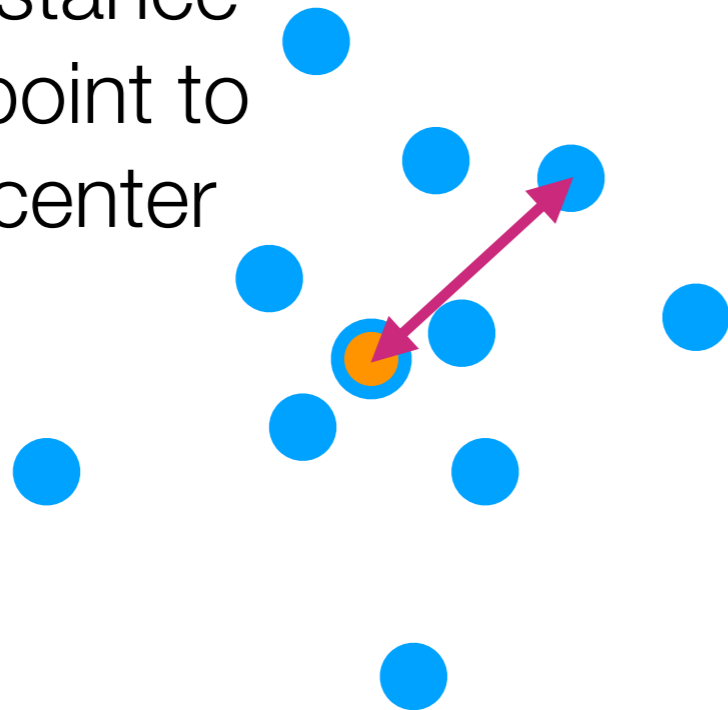


Cluster 2

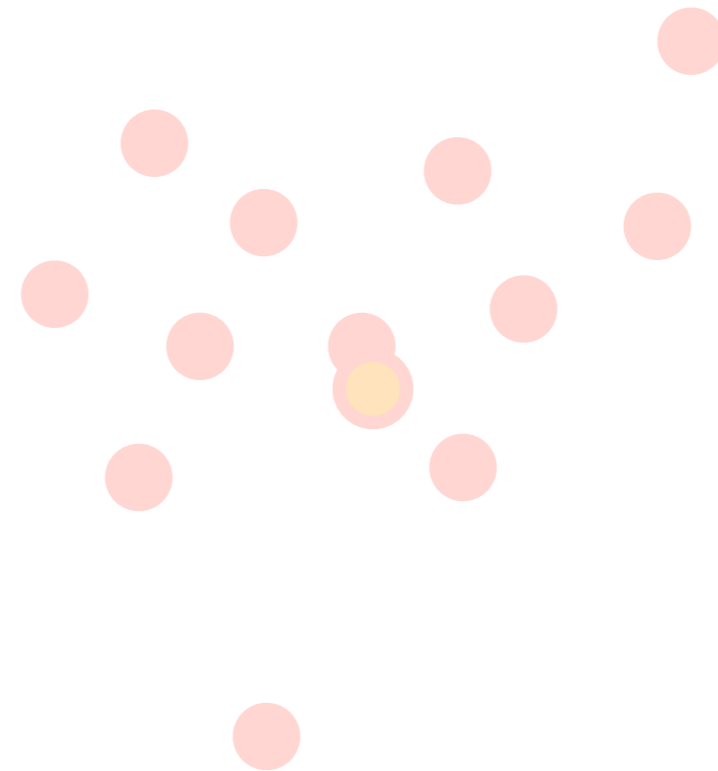
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

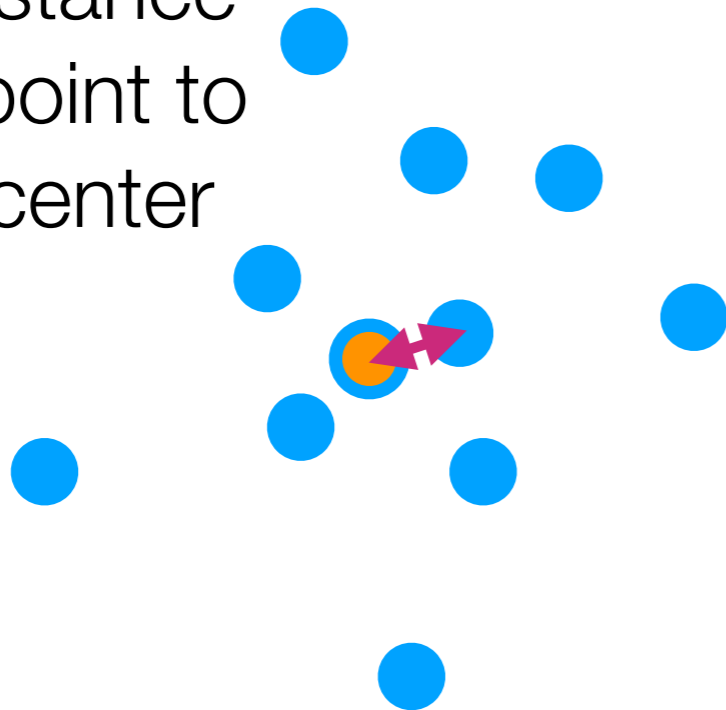


Cluster 2

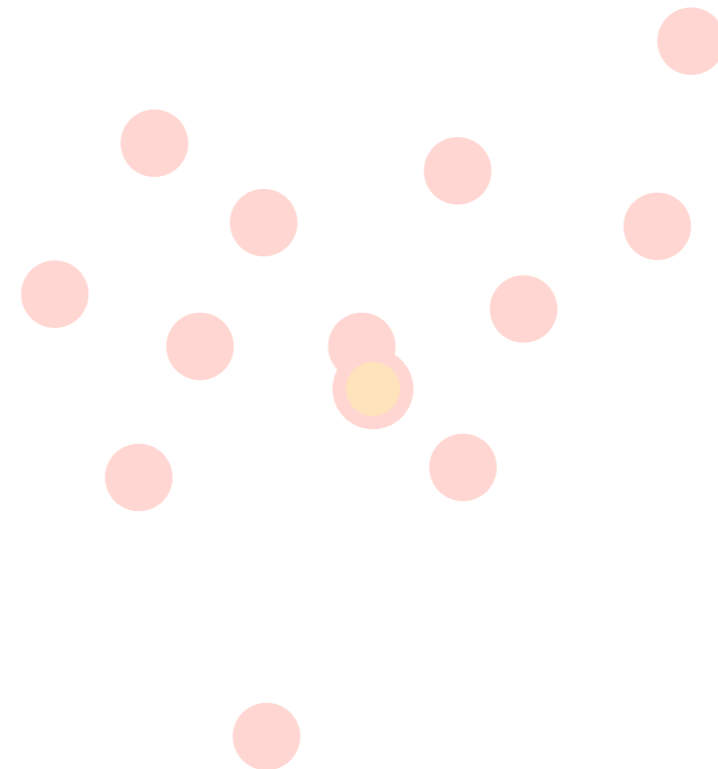
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

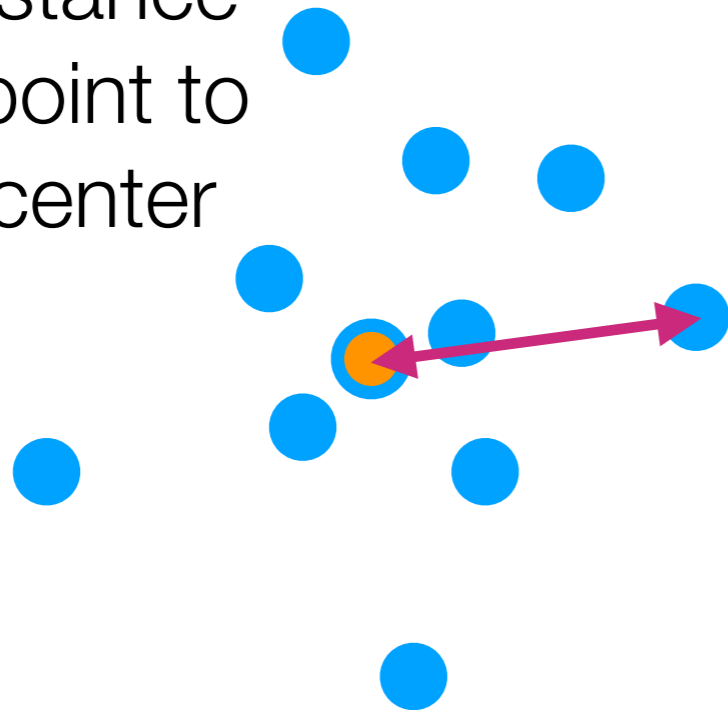


Cluster 2

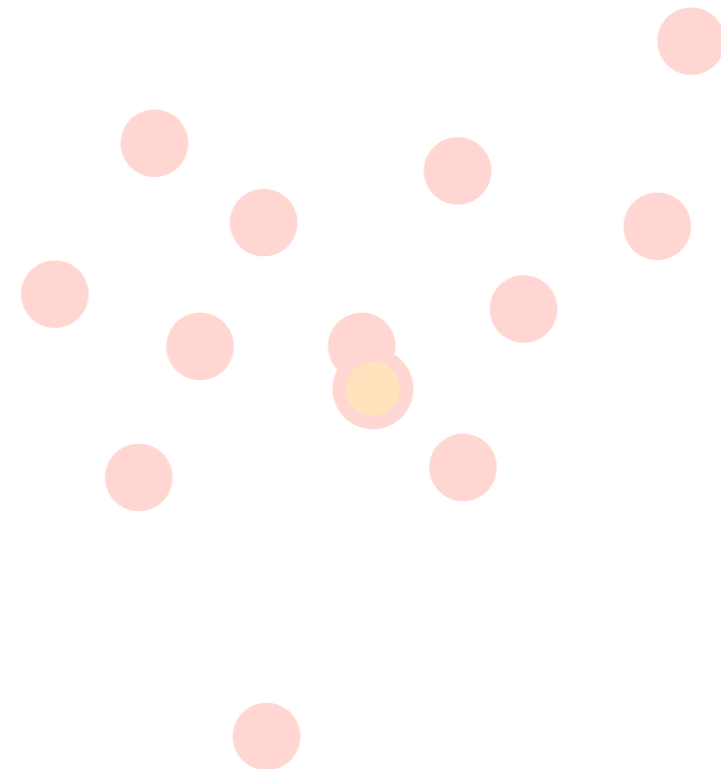
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

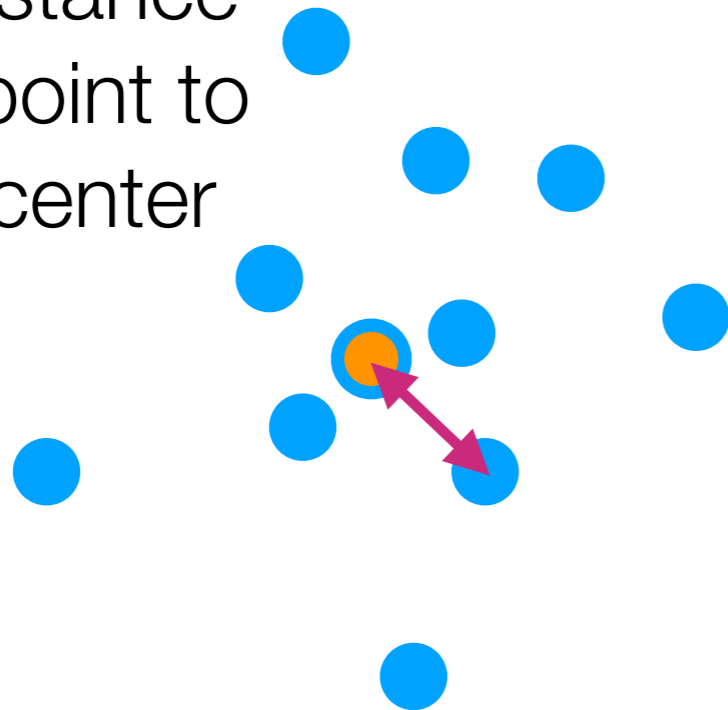


Cluster 2

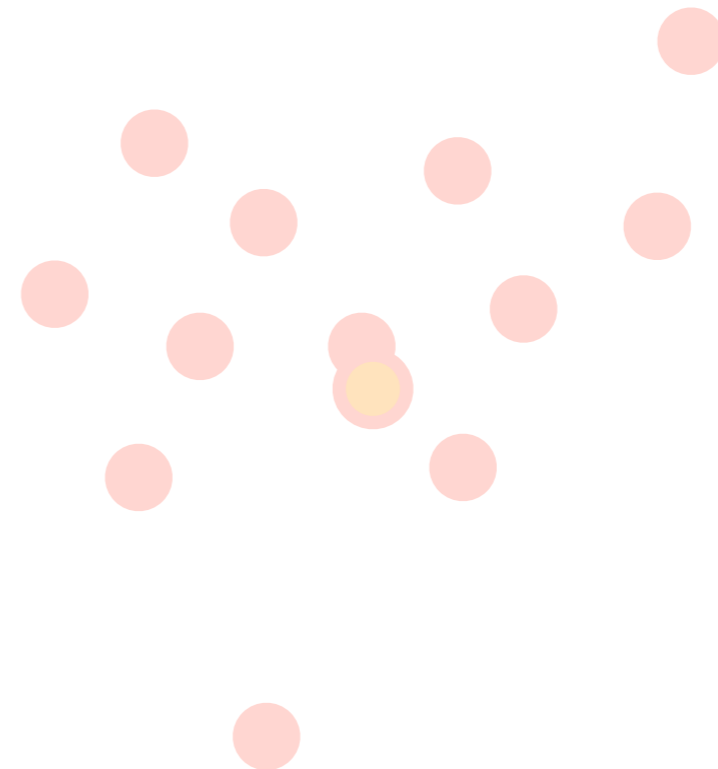
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

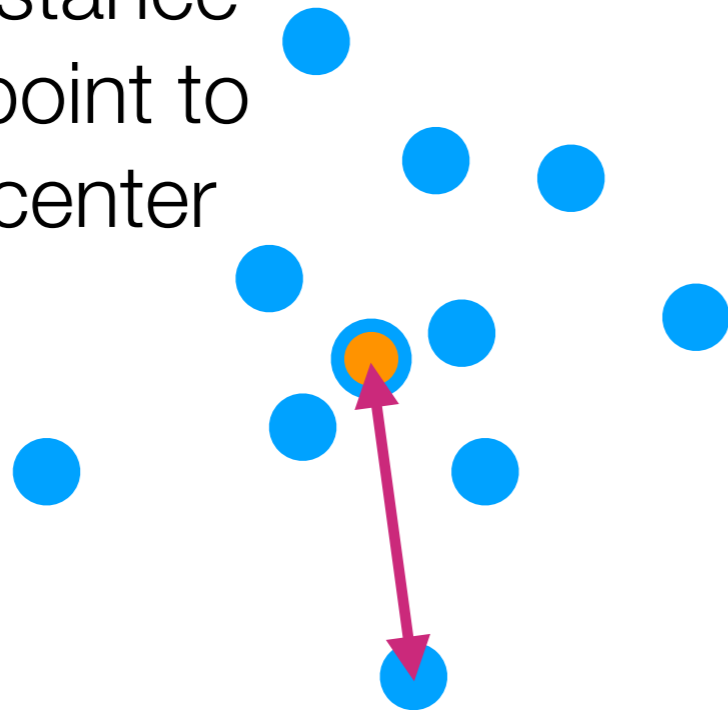


Cluster 2

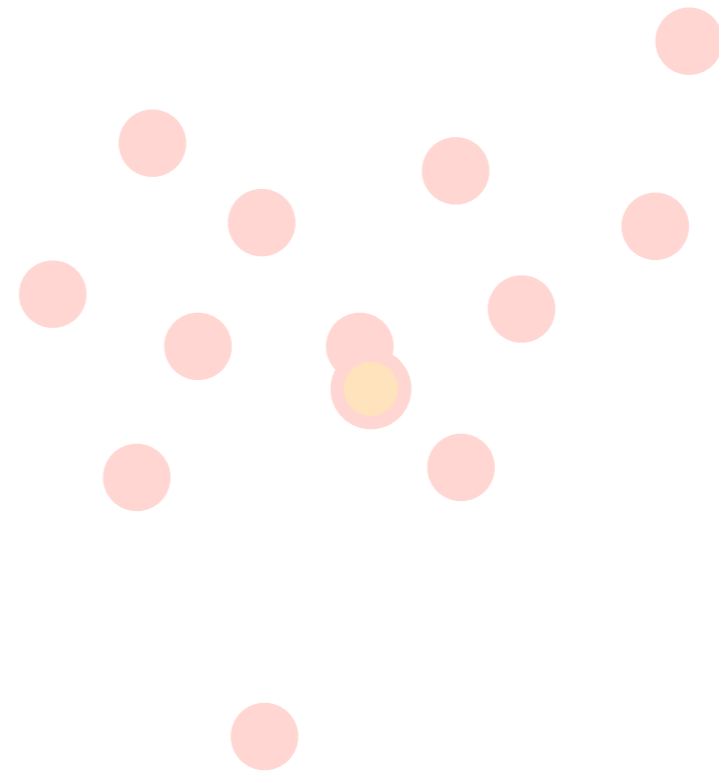
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

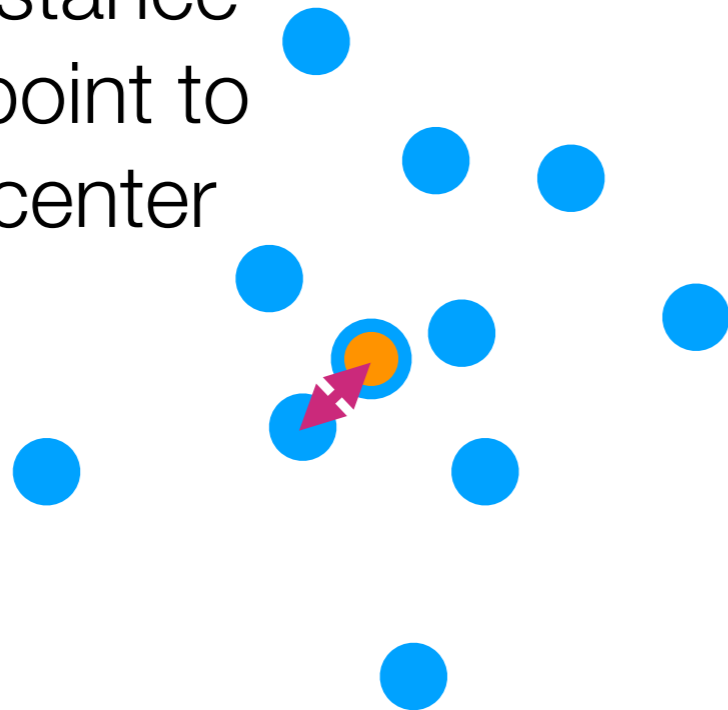


Cluster 2

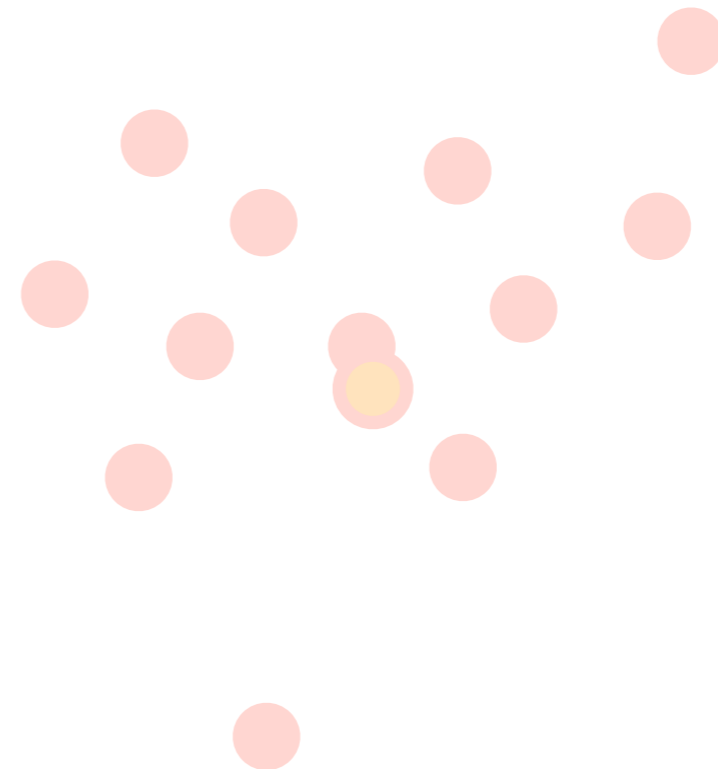
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

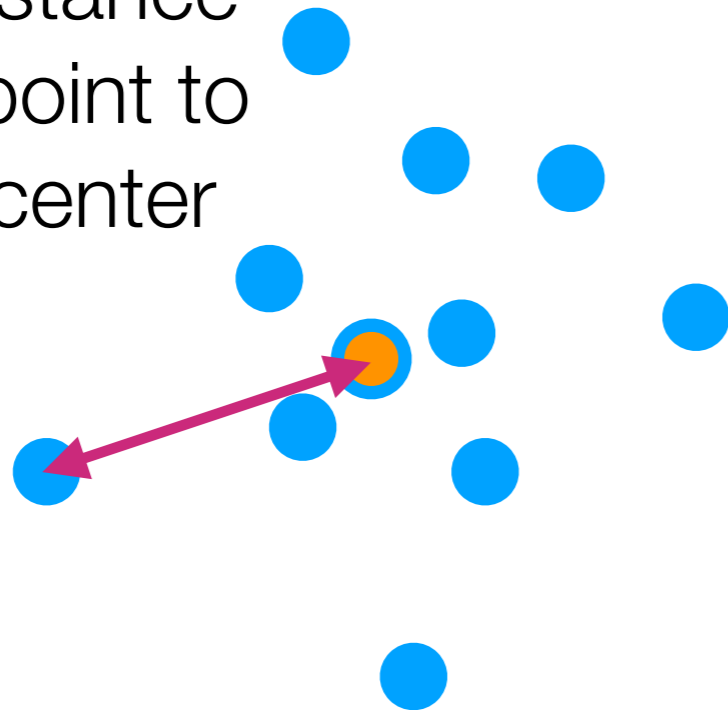


Cluster 2

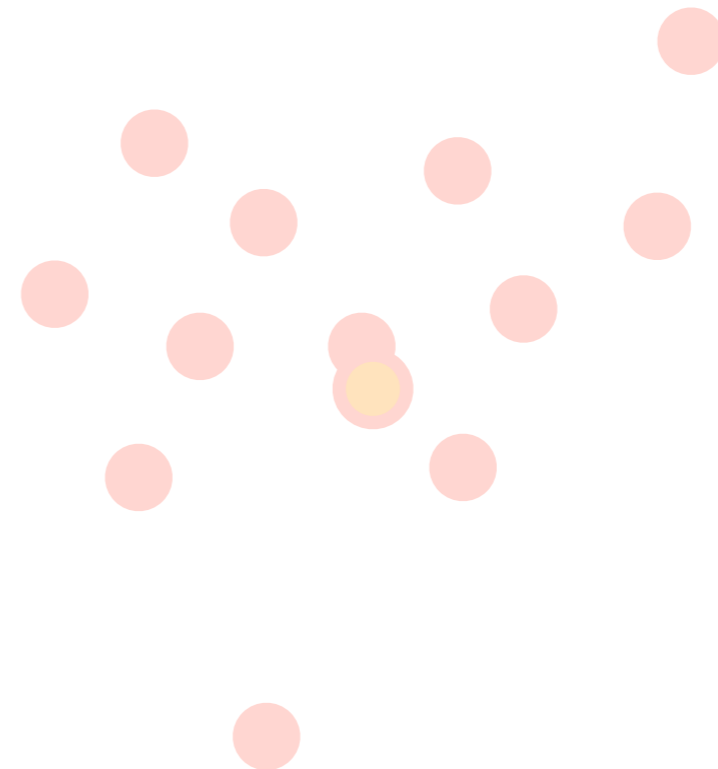
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

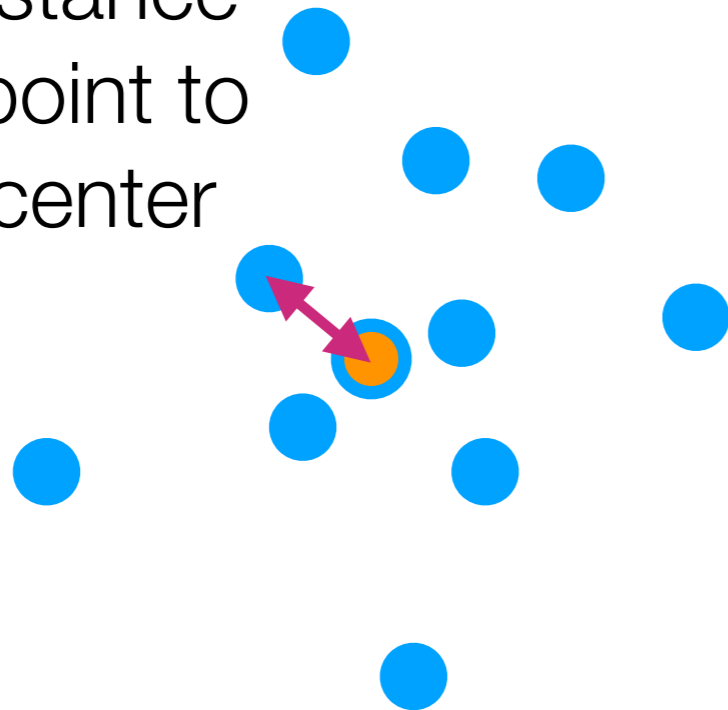


Cluster 2

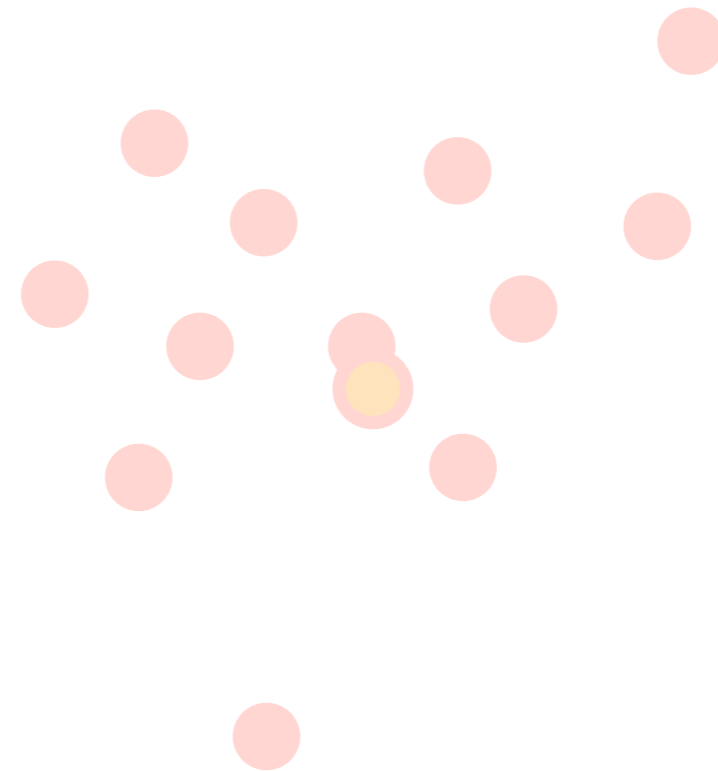
Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1

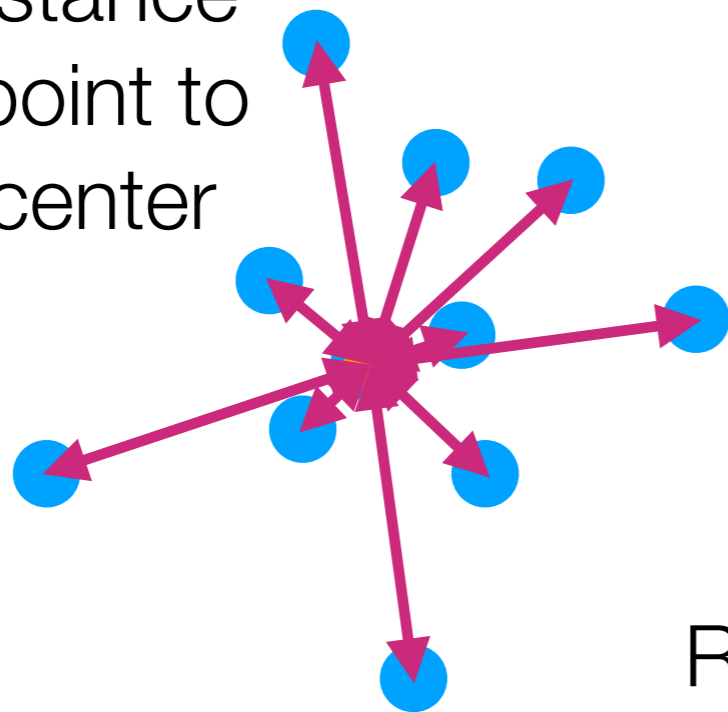


Cluster 2

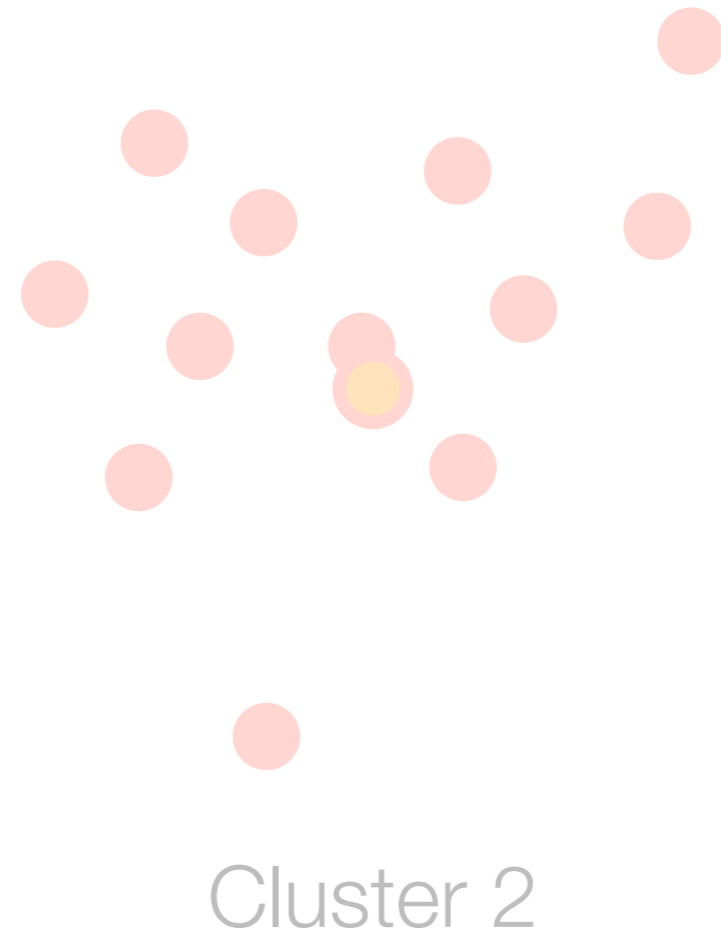
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center



Cluster 1

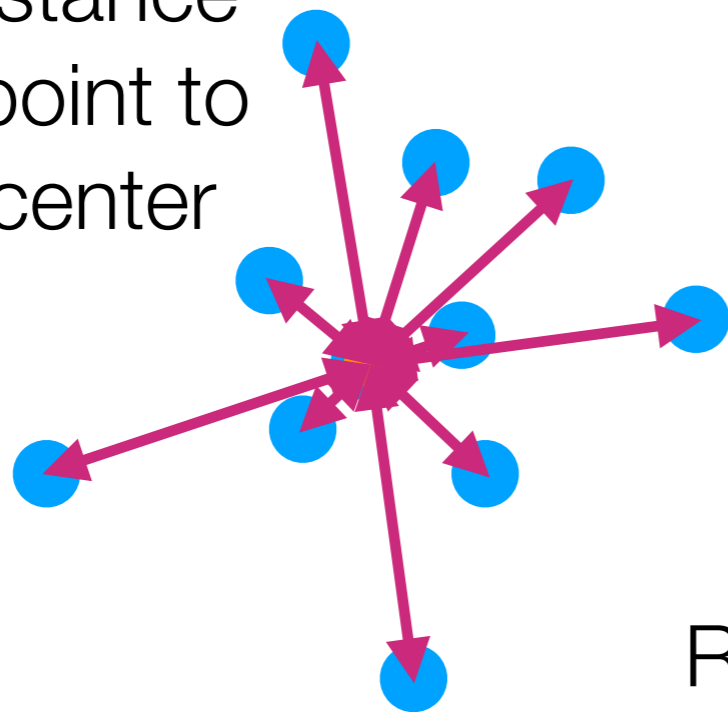


Residual sum of squares for cluster 1:
sum of *squared* purple lengths

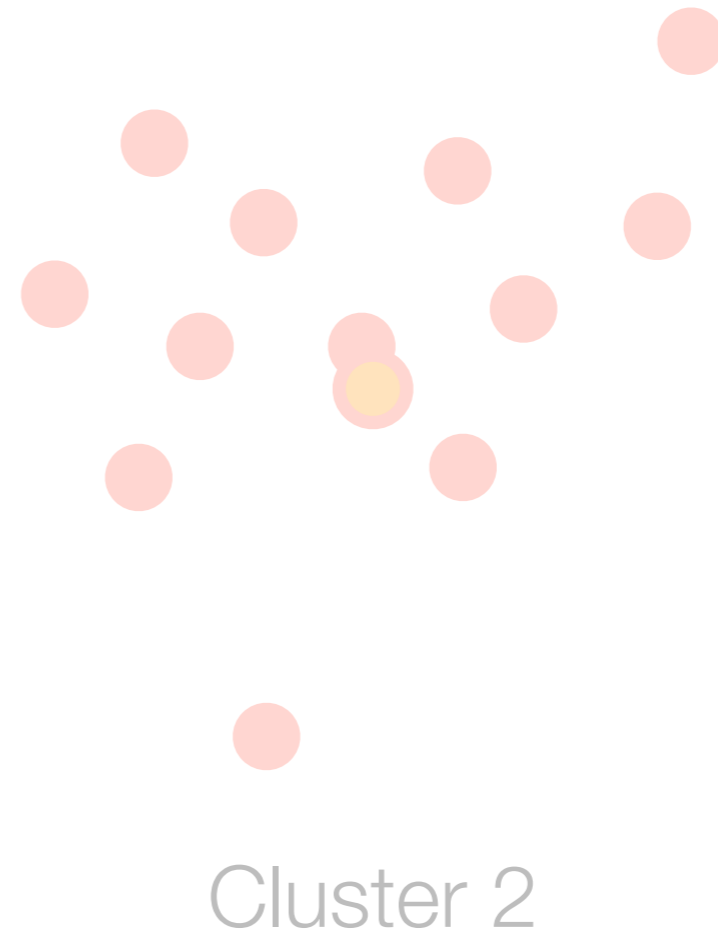
Residual Sum of Squares

Look at one cluster at a time

Measure distance from each point to its cluster center



Cluster 1



Residual sum of squares for cluster 1:

$$RSS_1 = \sum_{x \in \text{cluster 1}} \|x - \mu_1\|^2$$

Residual Sum of Squares

Look at one cluster at a time

Measure distance
from each point to
its cluster center



Cluster 1



Repeat similar calculation
for other cluster

Cluster 2

Residual sum of squares for cluster 2:

$$RSS_2 = \sum_{x \in \text{cluster 2}} \|x - \mu_2\|^2$$

Residual Sum of Squares

$$\text{RSS} = \text{RSS}_1 + \text{RSS}_2 = \sum_{x \in \text{cluster 1}} \|x - \mu_1\|^2 + \sum_{x \in \text{cluster 2}} \|x - \mu_2\|^2$$

Measure distance
from each point to
its cluster center

In general if there are k clusters:

$$\text{RSS} = \sum_{g=1}^k \text{RSS}_g = \sum_{g=1}^k \sum_{x \in \text{cluster } g} \|x - \mu_g\|^2$$

repeat similar calculation
for other cluster

Remark: k -means *tries* to minimize RSS

(it does so *approximately*, with no guarantee of optimality)

Cluster 1

RSS only really makes sense for clusters that look like circles

Why is minimizing RSS a bad way to choose k ?

What happens when k is equal to the number of data points?

A Good Way to Choose k

RSS measures *within-cluster variation*

$$W = \text{RSS} = \sum_{g=1}^k \text{RSS}_g = \sum_{g=1}^k \sum_{x \in \text{cluster } g} \|x - \mu_g\|^2$$

Want to also measure *between-cluster variation*

$$B = \sum_{g=1}^k (\# \text{ points in cluster } g) \|\mu_g - \mu\|^2$$

Called the **CH index**

mean of *all* points

[Calinski and Harabasz 1974]

A good score function to use for choosing k :

$$\text{CH}(k) = \frac{B \cdot (n - k)}{W \cdot (k - 1)}$$

n = total # points

Pick k with highest $\text{CH}(k)$

(Choose k among 2, 3, ... up to pre-specified max)

Automatically Choosing k

Demo

More on Automatic Selection of k

Dirichlet Process Gaussian Mixture Model (DP-GMM):

- Number of clusters is effectively random, and *can grow with the amount of data you have!*
- While you don't have to choose k , you have to choose a different parameter which says how large clusters are

DP-GMM High-Level Idea

Cluster 1

Probability of generating a point from cluster 1 = π_1

Gaussian mean = μ_1

Gaussian covariance = Σ_1

Cluster 2

π_2

μ_2

Σ_2

Cluster 3

π_3

μ_3

Σ_3

...

It goes on forever!

There is a parameter that controls how these π values roughly decay

There are an infinite number of parameters

(Rough idea) How to generate points from this DP-GMM:

1. Flip biased ∞ -sided coin (the sides have probabilities $\pi_1, \pi_2, \pi_3, \dots$)
2. Let Z be the side that we got (it is a positive integer)
3. Sample 1 point from Gaussian mean μ_Z , covariance Σ_Z

Remark: For any given dataset, when learning the DP-GMM, there aren't going to be an infinite number of clusters found

More on Automatic Selection of k

Dirichlet Process Gaussian Mixture Model (DP-GMM):

- Number of clusters is effectively random, and *can grow with the amount of data you have!*
- While you don't have to choose k , you have to choose a different parameter which says how large clusters are
- An example of a *Bayesian nonparametric model* (roughly: a generative model with an *infinite number of parameters*, where the *parameters are random*)

Learning a DP-GMM

Common approach: finite approximation where you specify some maximum number of clusters (the algorithm will find up to that many clusters)

- Algorithm is somewhat similar to k -means/EM for GMMs
- Algorithm output: very similar to regular GMM fitting

Learning a DP-GMM

Demo

***k*-means approximates
(a special case of) learning GMM's.**

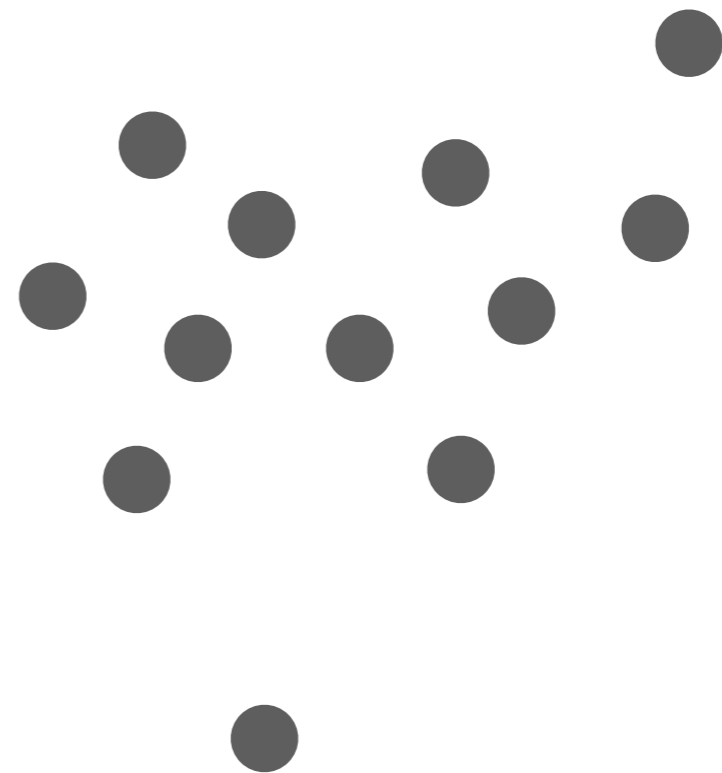
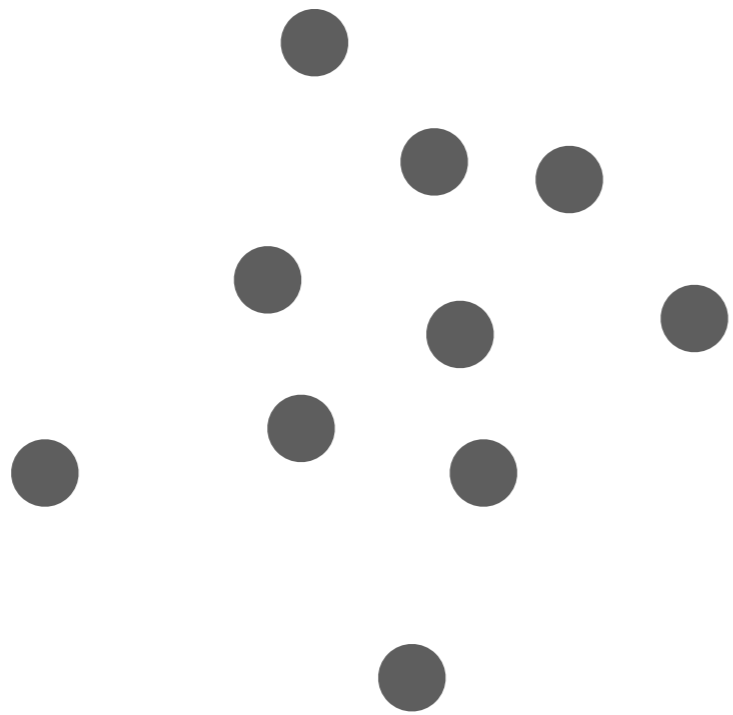
What approximates learning DP-GMMs?

This next algorithm will give you a sense of how we get around
specifying the number of clusters directly

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

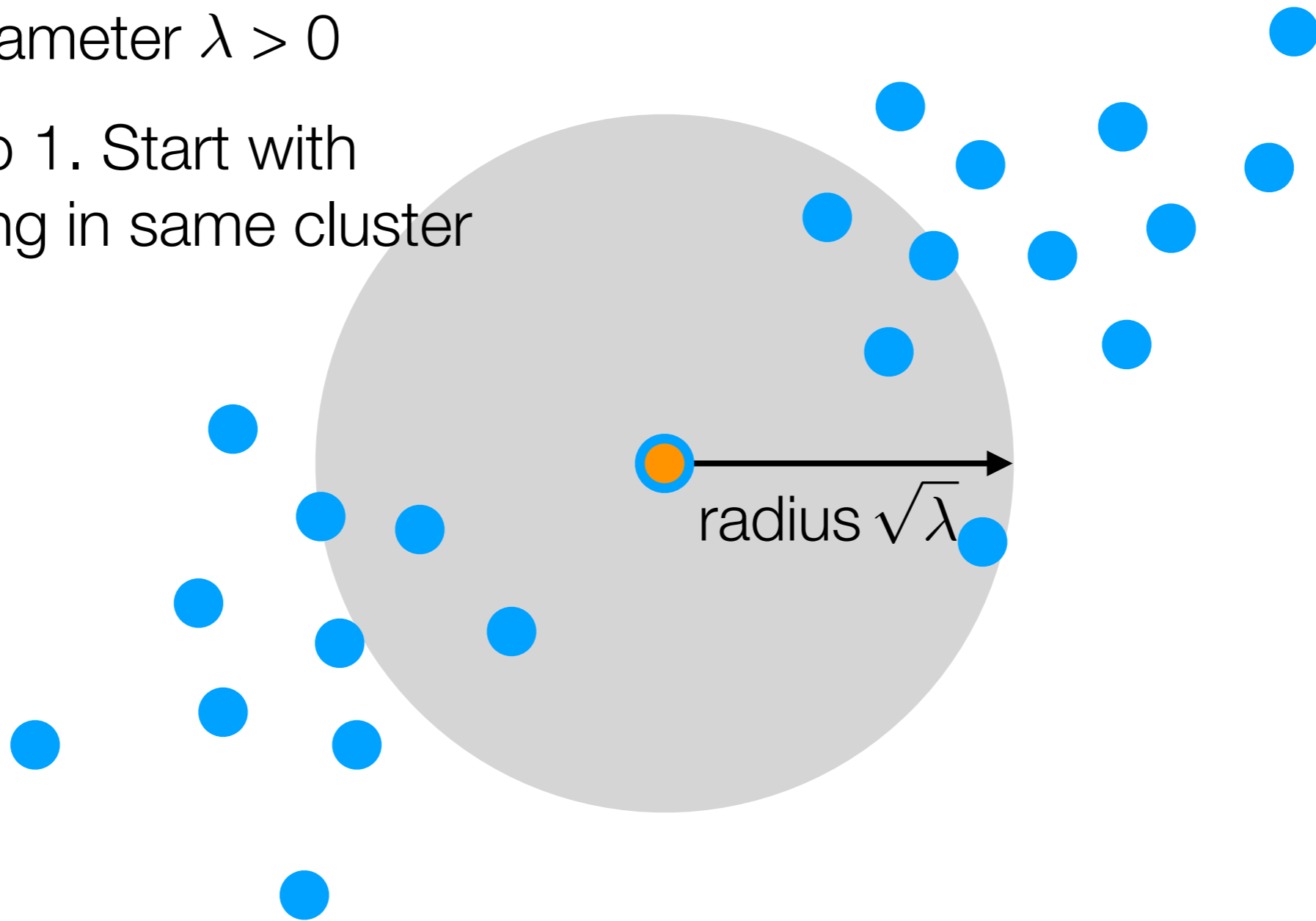
Step 1. Start with everything in same cluster



DP-means

Step 0. Pick concentration parameter $\lambda > 0$

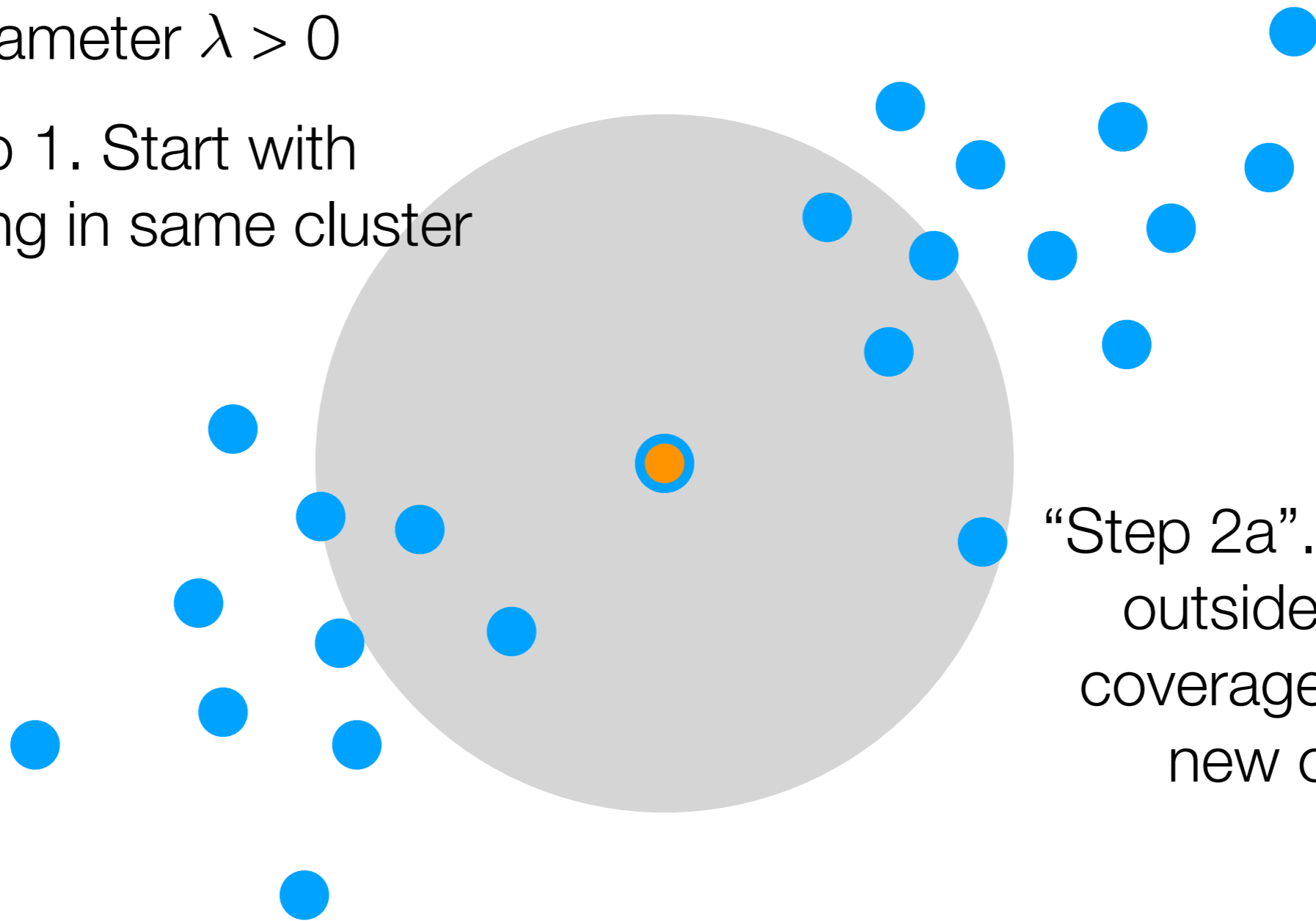
Step 1. Start with everything in same cluster



DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster

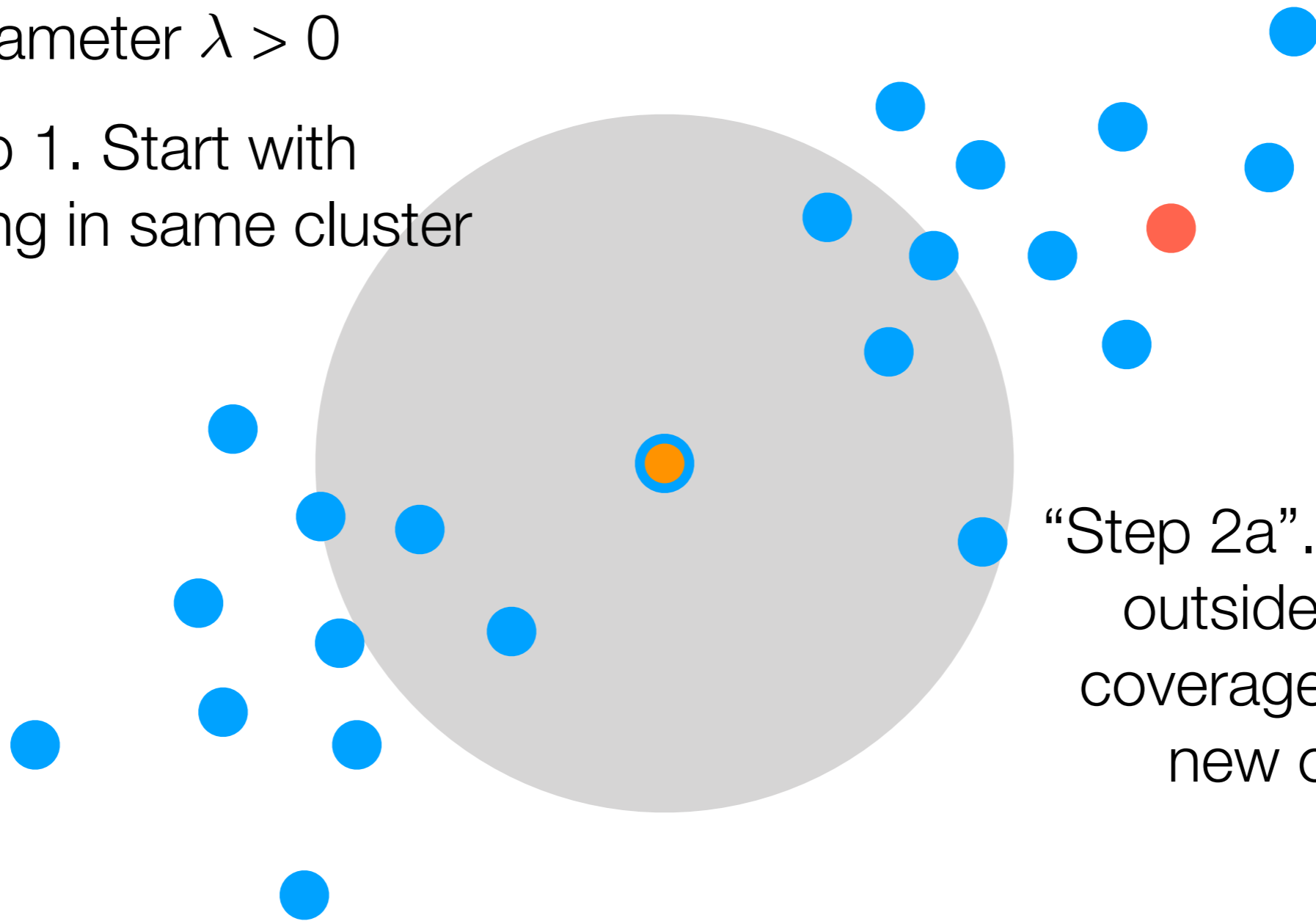


“Step 2a”. Pick point outside of gray coverage to make new cluster

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster

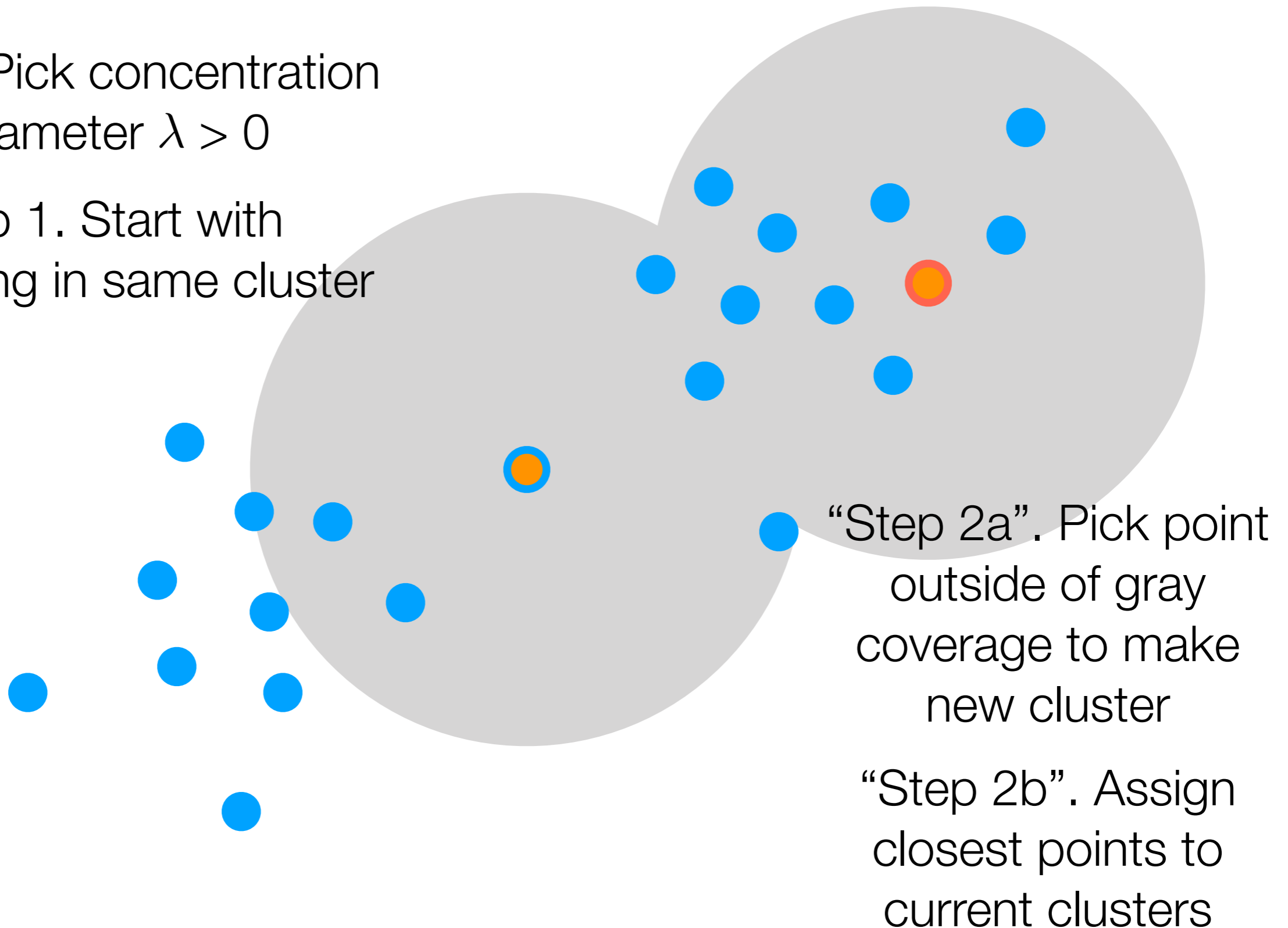


“Step 2a”. Pick point outside of gray coverage to make new cluster

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

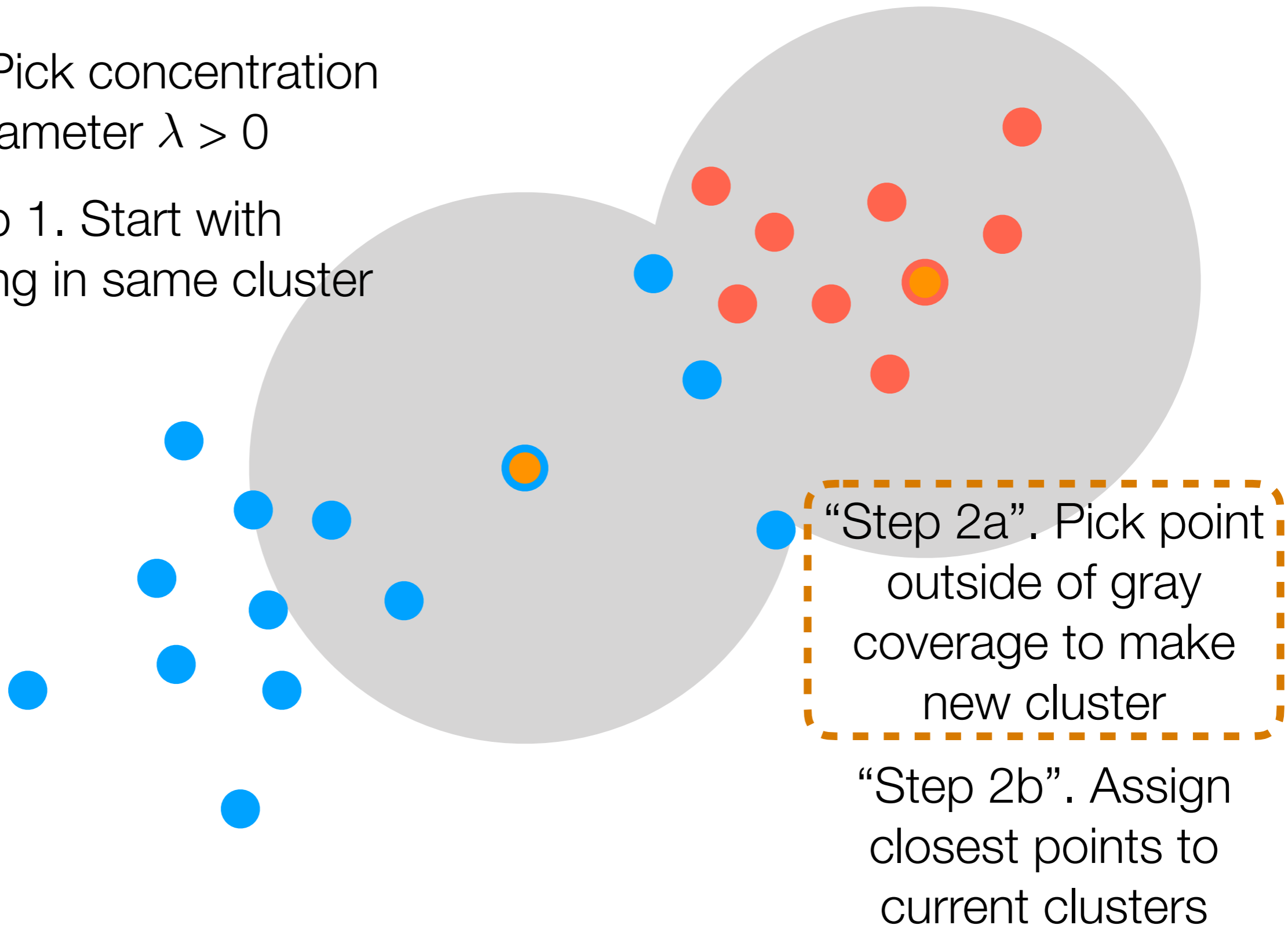
Step 1. Start with everything in same cluster



DP-means

Step 0. Pick concentration parameter $\lambda > 0$

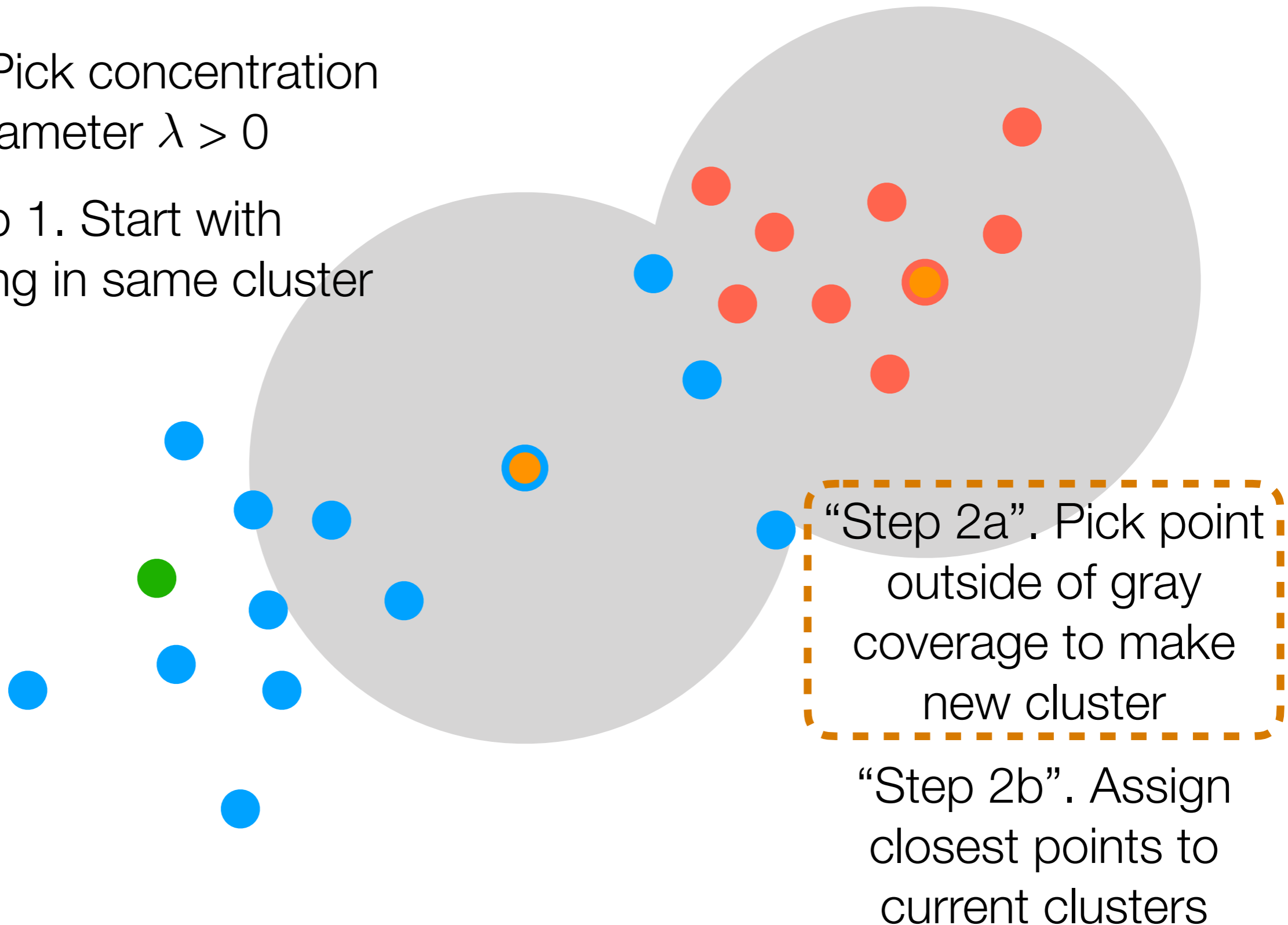
Step 1. Start with everything in same cluster



DP-means

Step 0. Pick concentration parameter $\lambda > 0$

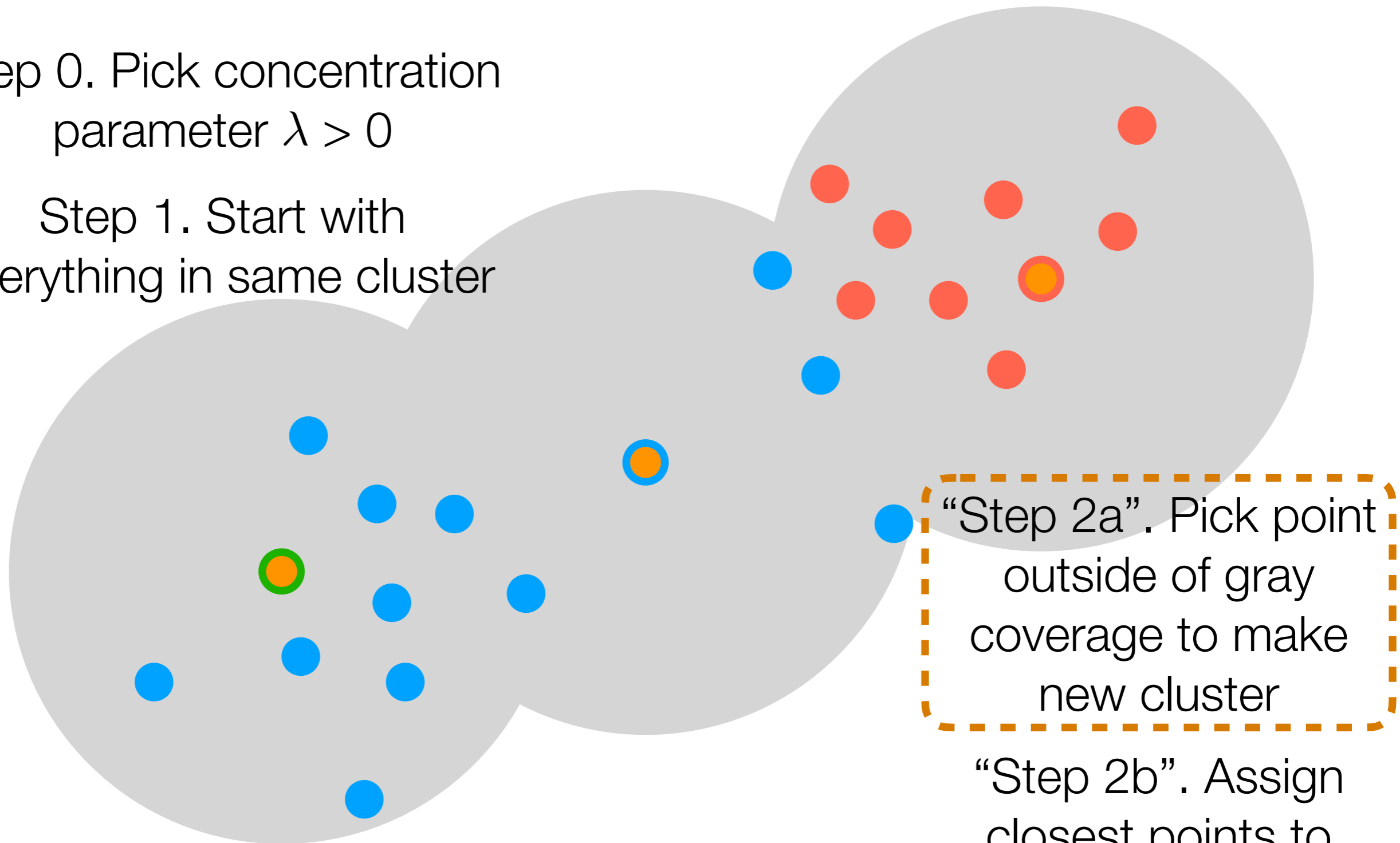
Step 1. Start with everything in same cluster



DP-means

Step 0. Pick concentration parameter $\lambda > 0$

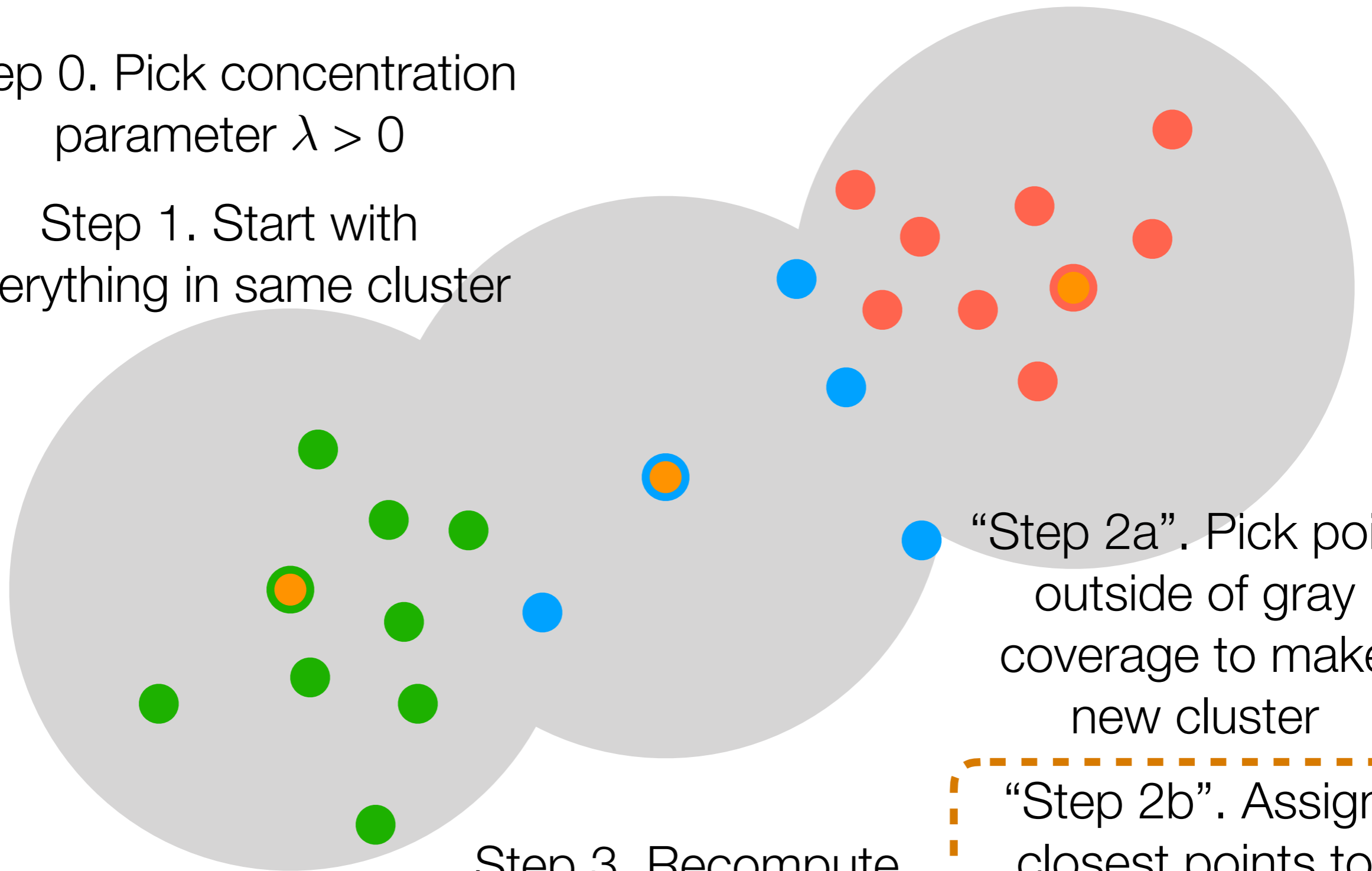
Step 1. Start with everything in same cluster



DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



“Step 2a”. Pick point outside of gray coverage to make new cluster

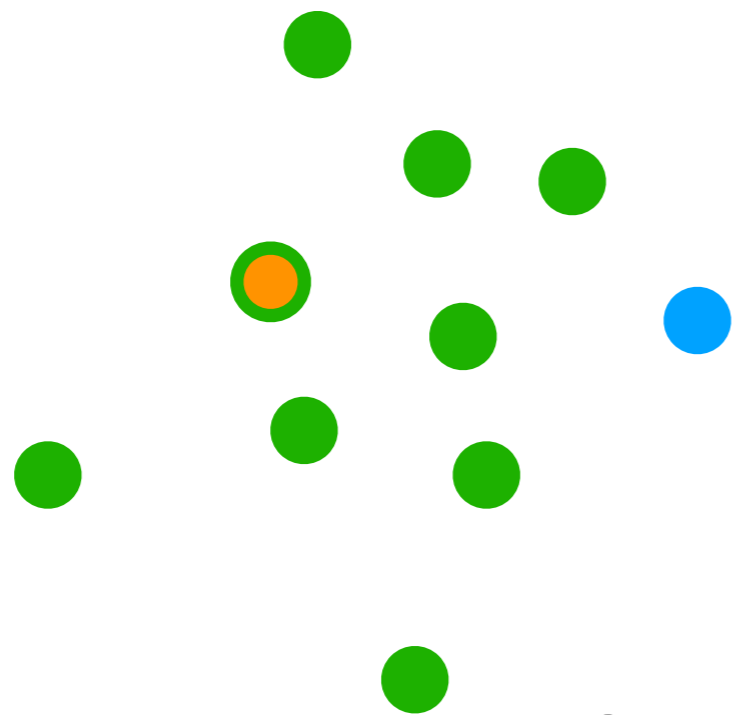
“Step 2b”. Assign closest points to current clusters

Step 3. Recompute cluster centers

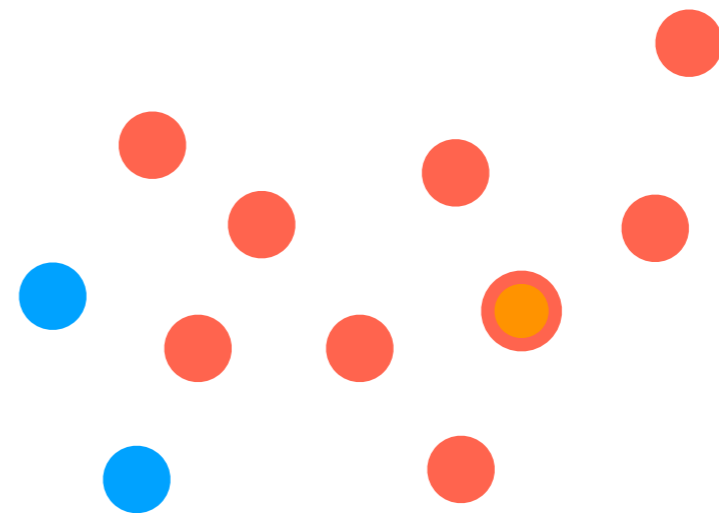
DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 3. Recompute cluster centers



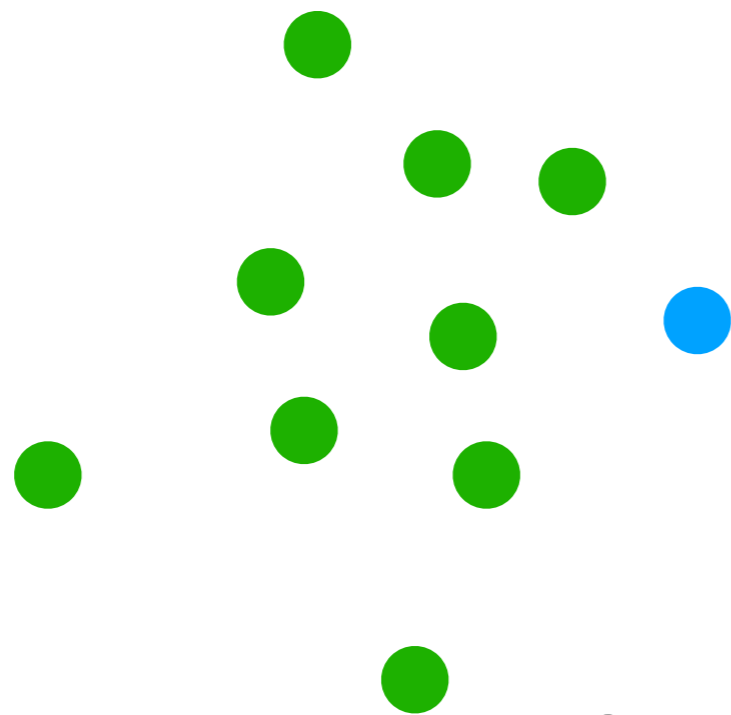
“Step 2a”. Pick point outside of gray coverage to make new cluster

“Step 2b”. Assign closest points to current clusters

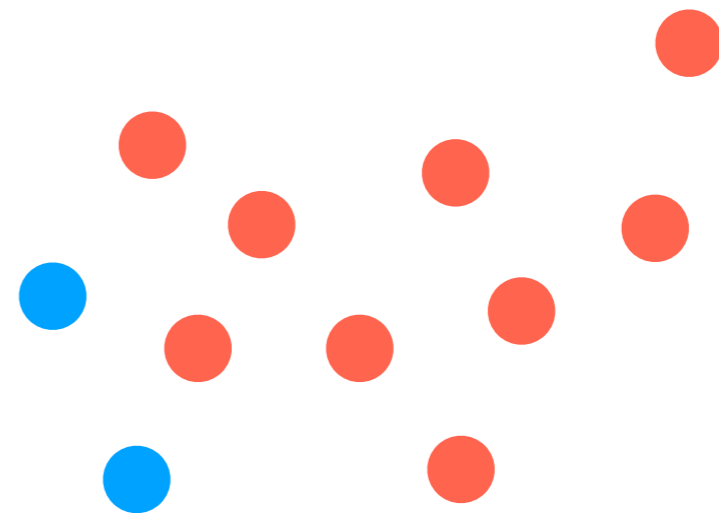
DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 3. Recompute cluster centers



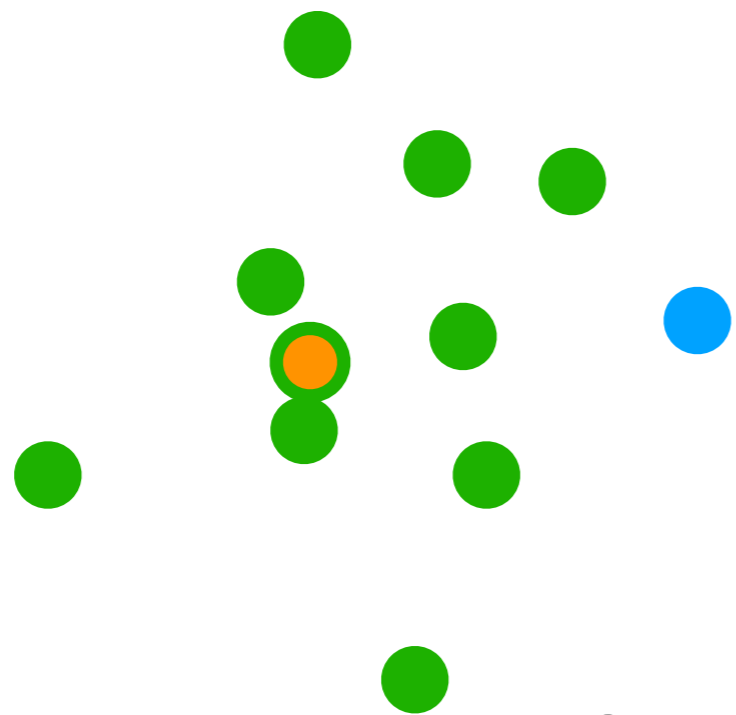
“Step 2a”. Pick point outside of gray coverage to make new cluster

“Step 2b”. Assign closest points to current clusters

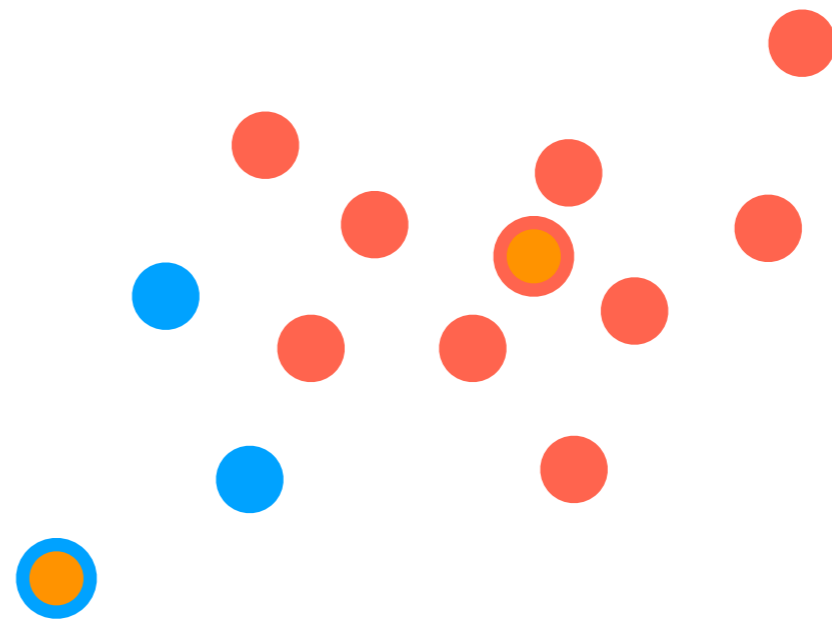
DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 3. Recompute cluster centers



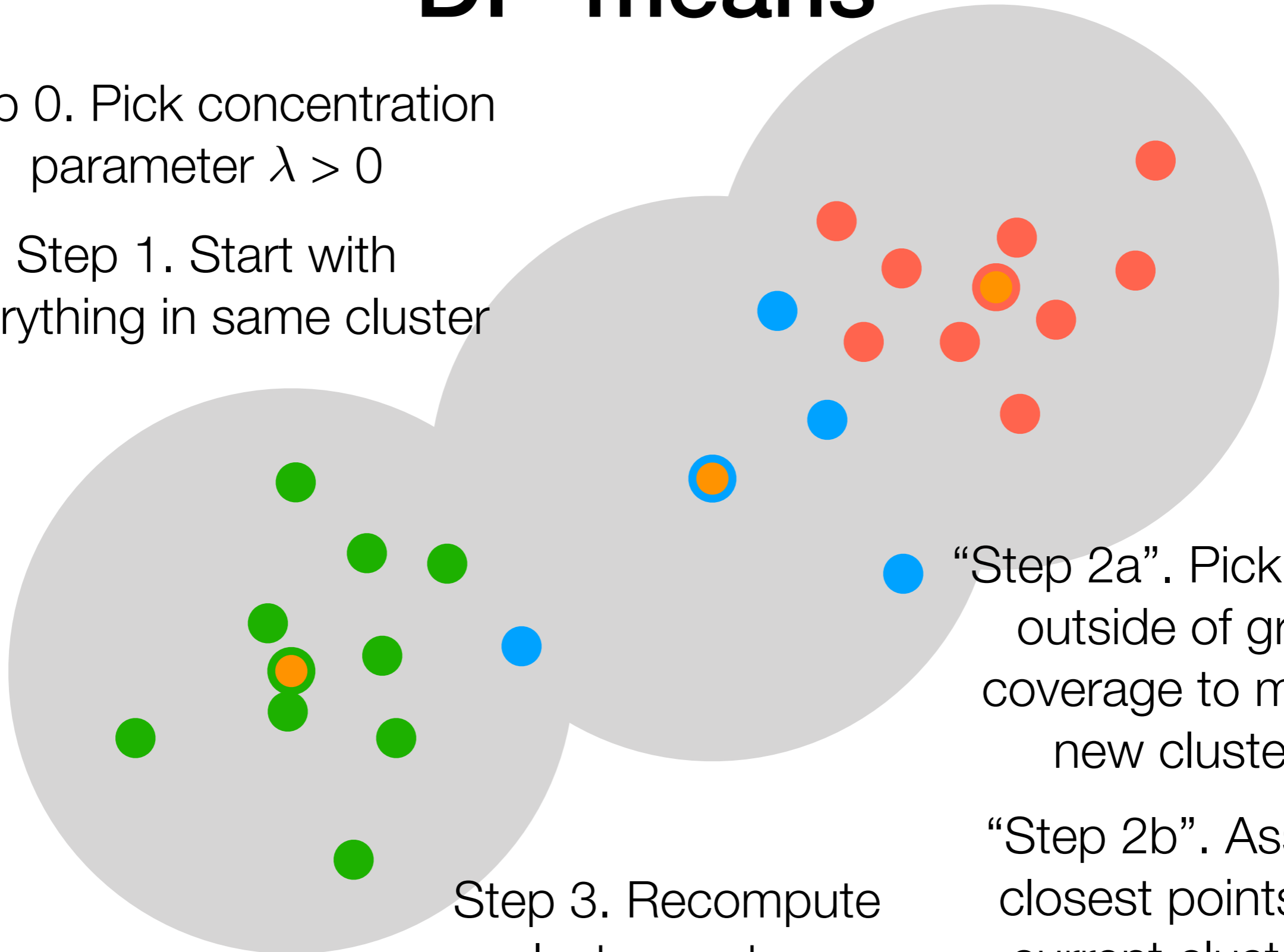
“Step 2a”. Pick point outside of gray coverage to make new cluster

“Step 2b”. Assign closest points to current clusters

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



“Step 2a”. Pick point outside of gray coverage to make new cluster

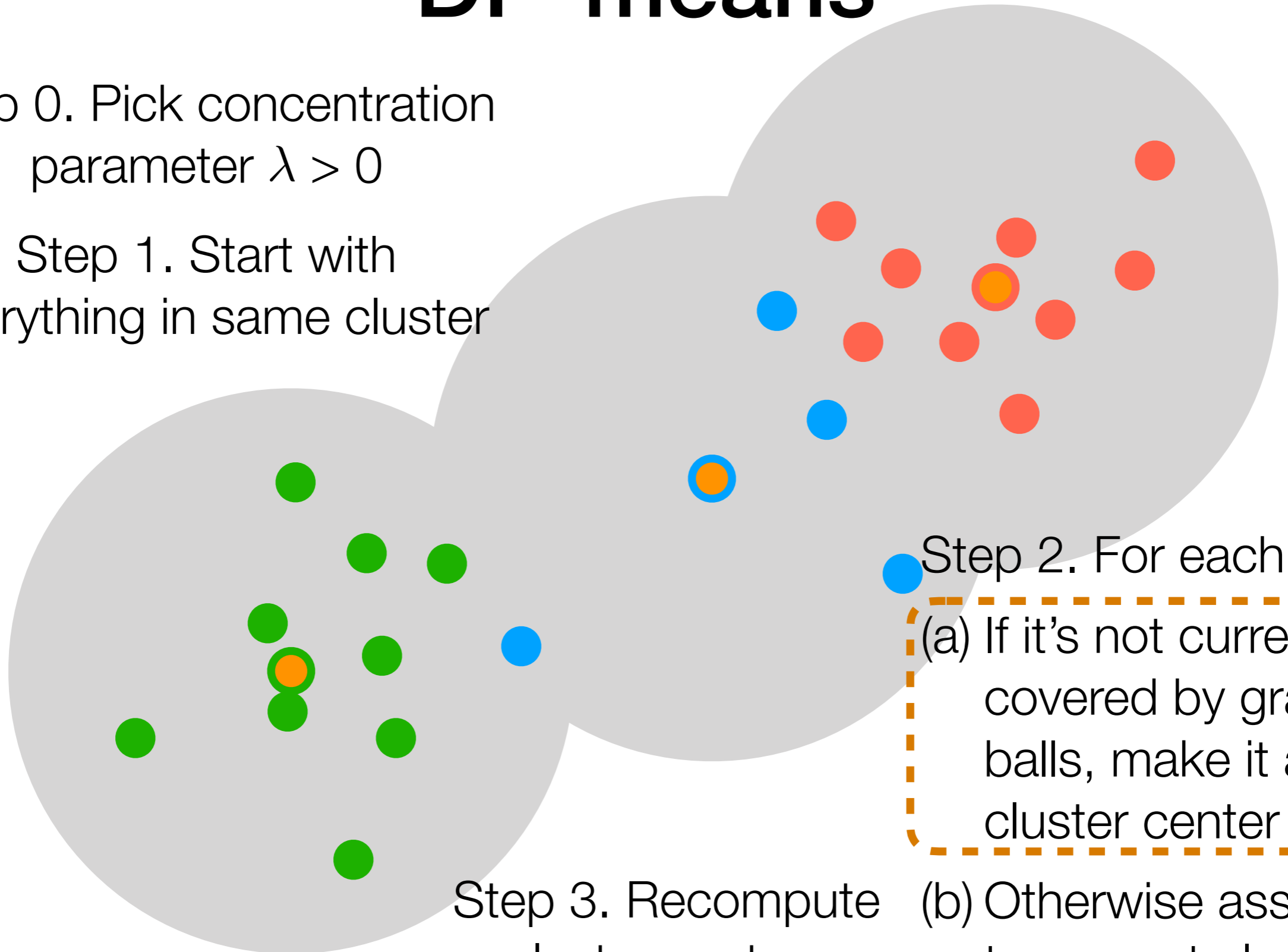
“Step 2b”. Assign closest points to current clusters

Step 3. Recompute cluster centers

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 2. For each point:

(a) If it's not currently covered by gray balls, make it a new cluster center

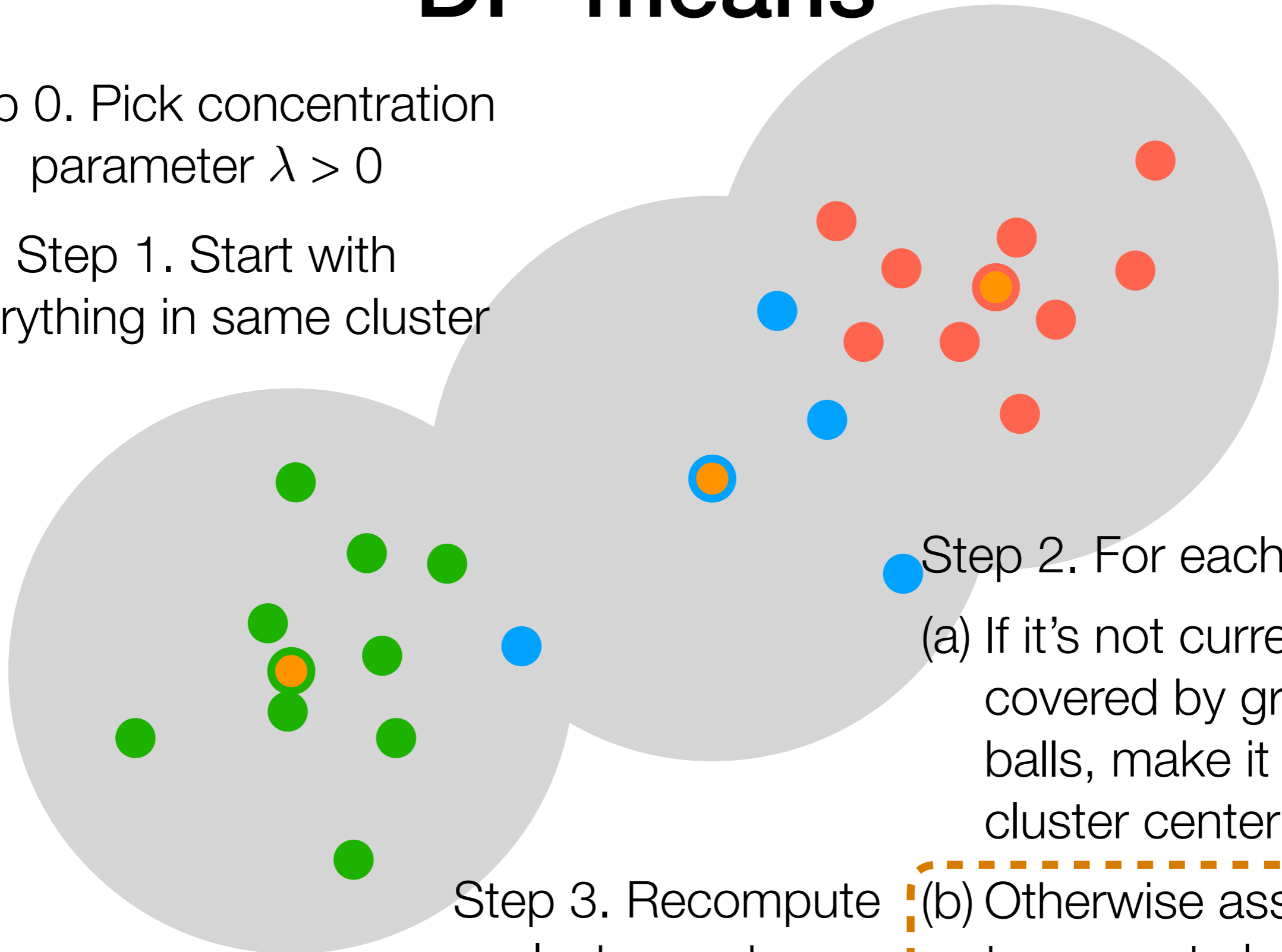
(b) Otherwise assign it to nearest cluster

Step 3. Recompute cluster centers

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 2. For each point:
(a) If it's not currently covered by gray balls, make it a new cluster center

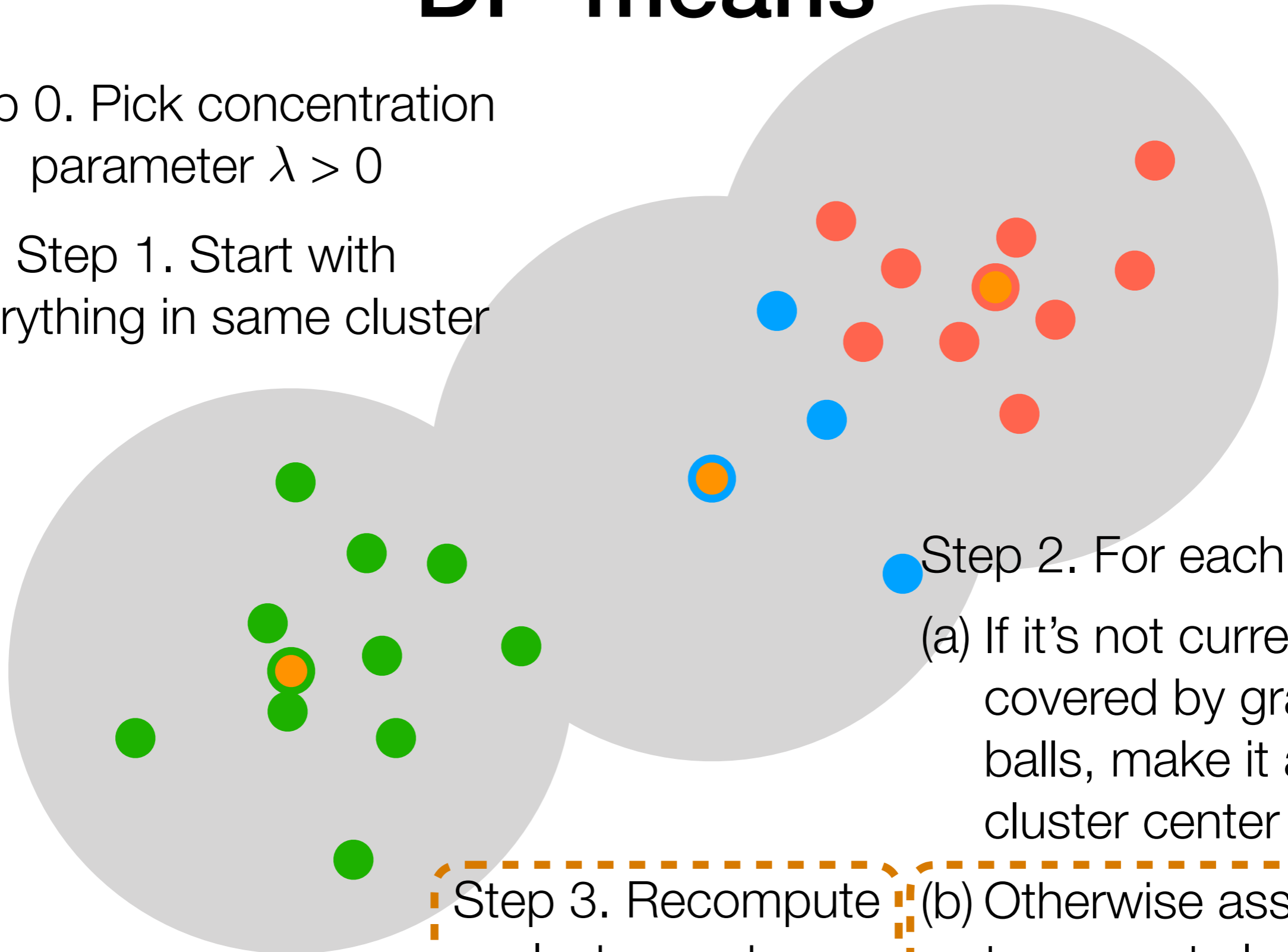
(b) Otherwise assign it to nearest cluster

Step 3. Recompute cluster centers

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 2. For each point:
(a) If it's not currently covered by gray balls, make it a new cluster center

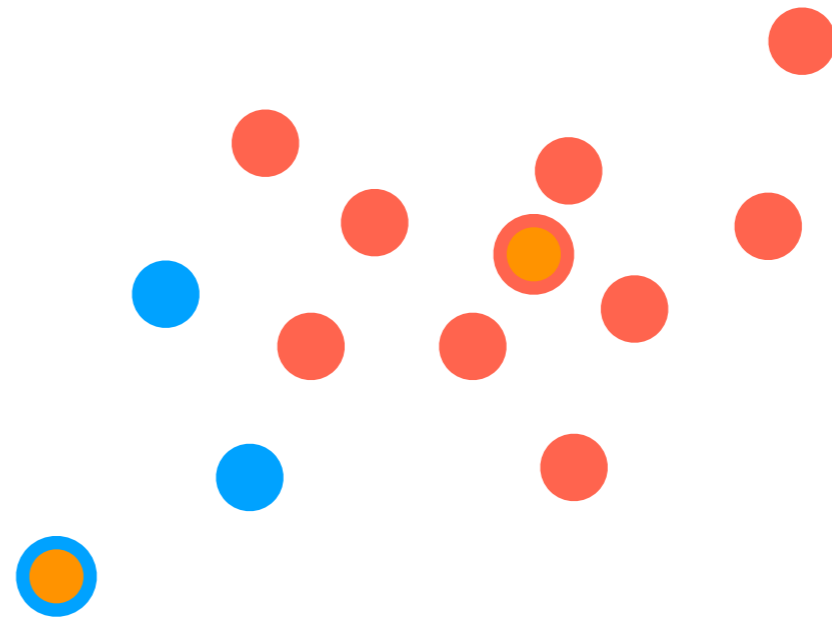
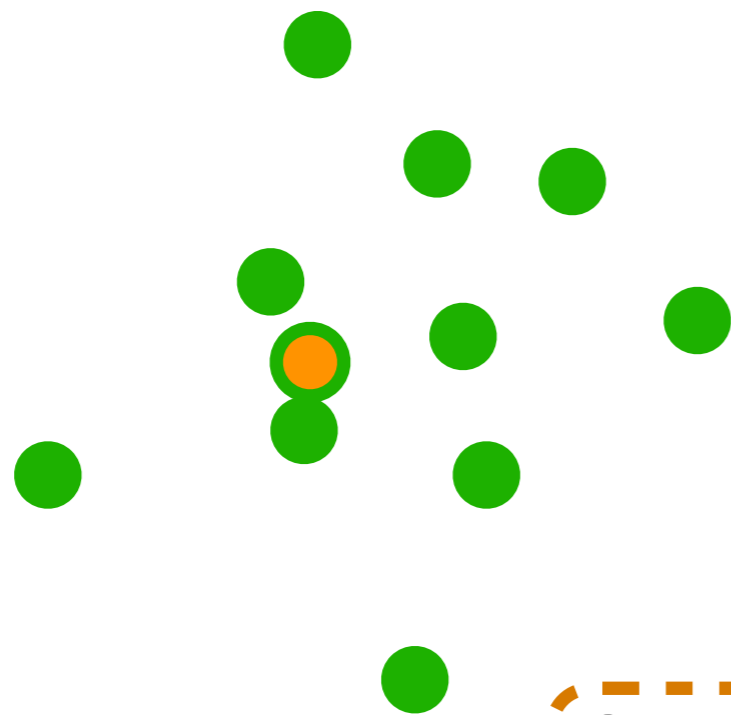
Step 3. Recompute cluster centers

(b) Otherwise assign it to nearest cluster

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 2. For each point:
(a) If it's not currently covered by gray balls, make it a new cluster center

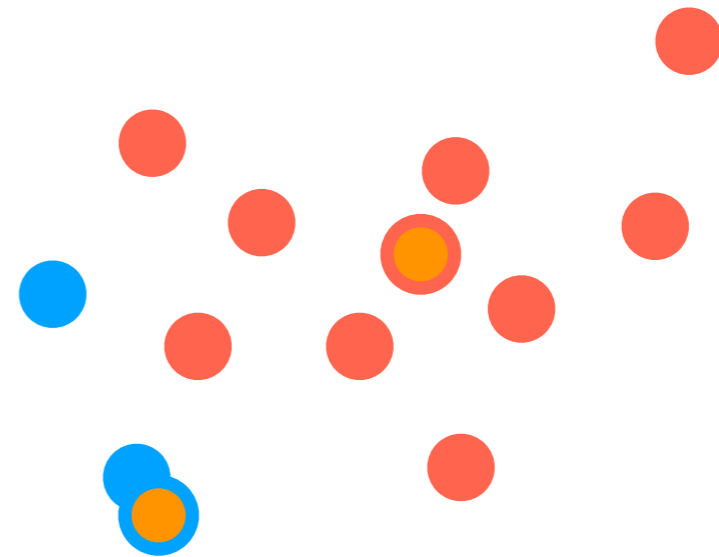
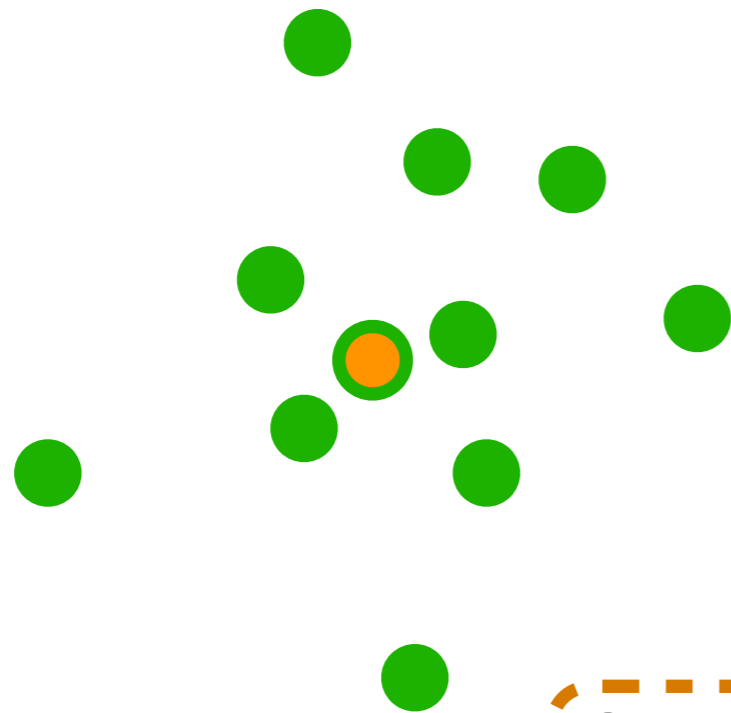
(b) Otherwise assign it to nearest cluster

Step 3. Recompute cluster centers

DP-means

Step 0. Pick concentration parameter $\lambda > 0$

Step 1. Start with everything in same cluster



Step 2. For each point:
(a) If it's not currently covered by gray balls, make it a new cluster center

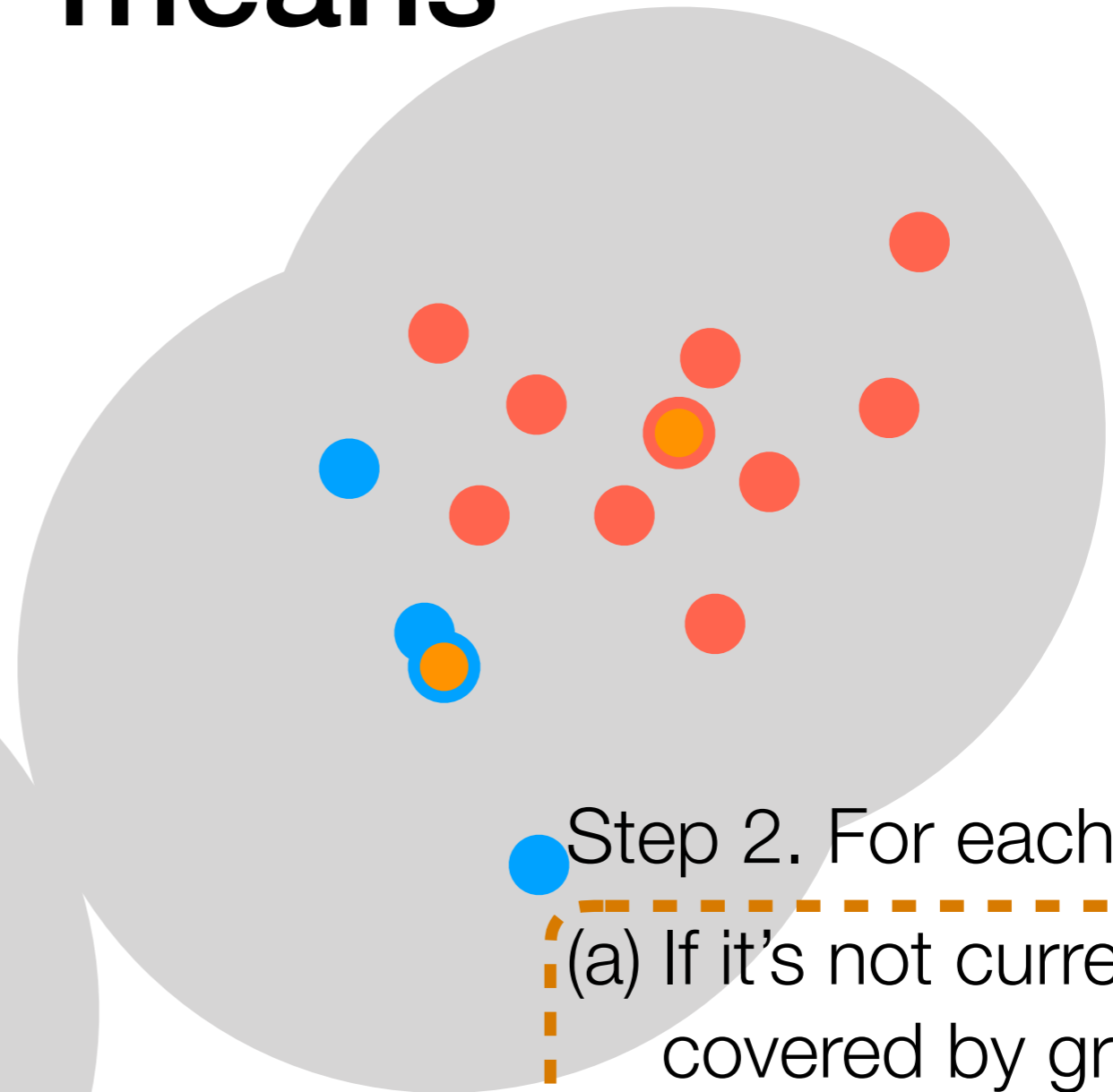
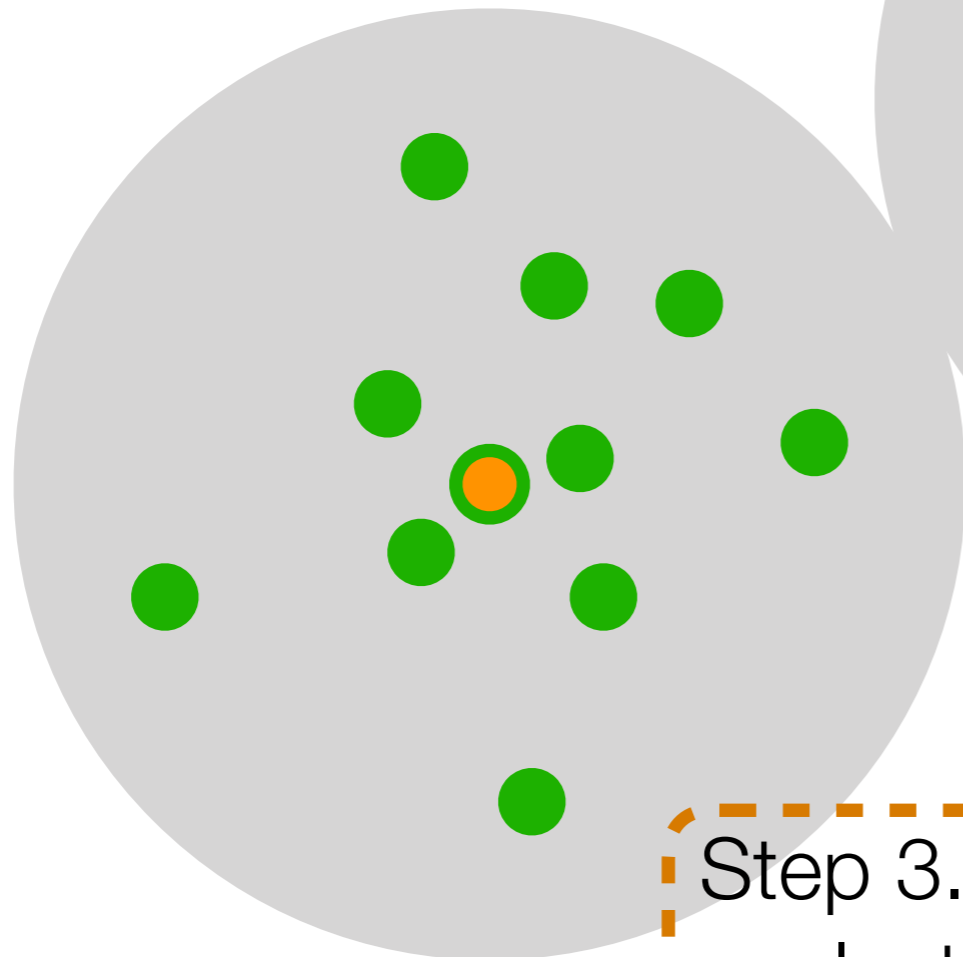
Step 3. Recompute cluster centers

(b) Otherwise assign it to nearest cluster

DP-means

Step 0: Pick concentration parameter $\lambda > 0$

Step 1: Start with everything in same cluster



Step 2. For each point:

(a) If it's not currently covered by gray balls, make it a new cluster center

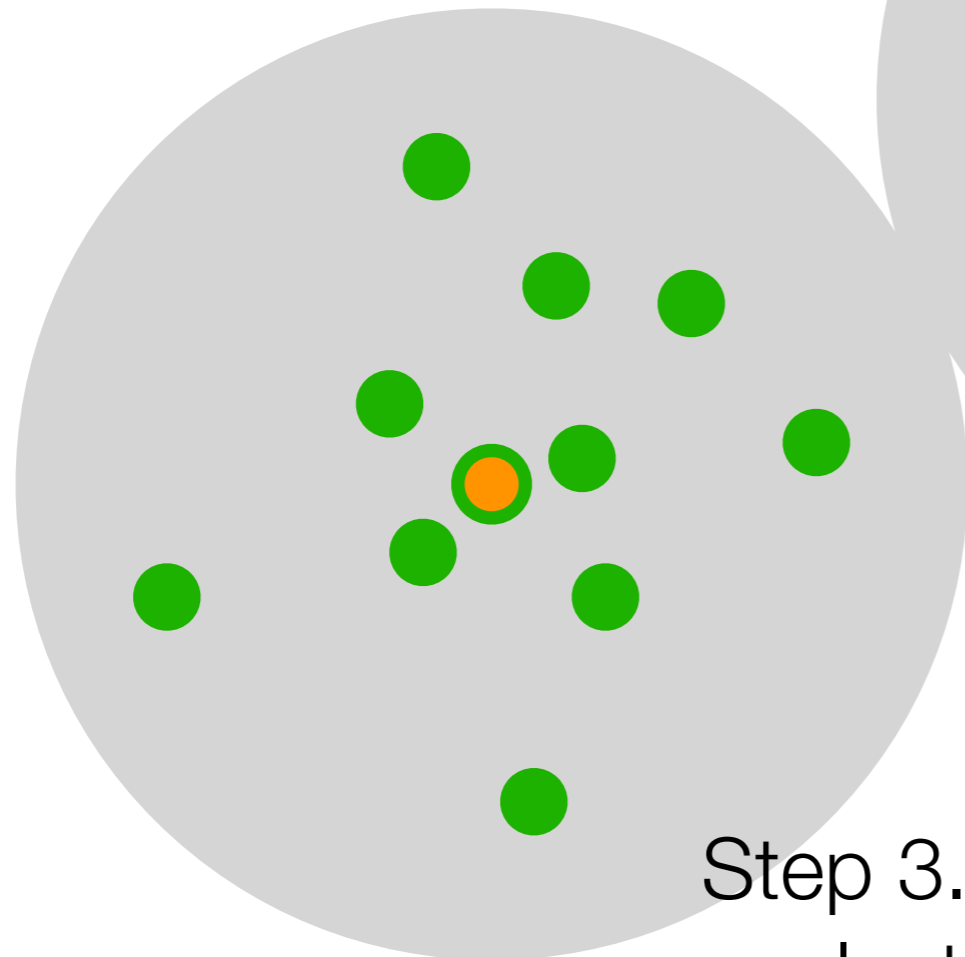
(b) Otherwise assign it to nearest cluster

Step 3. Recompute cluster centers

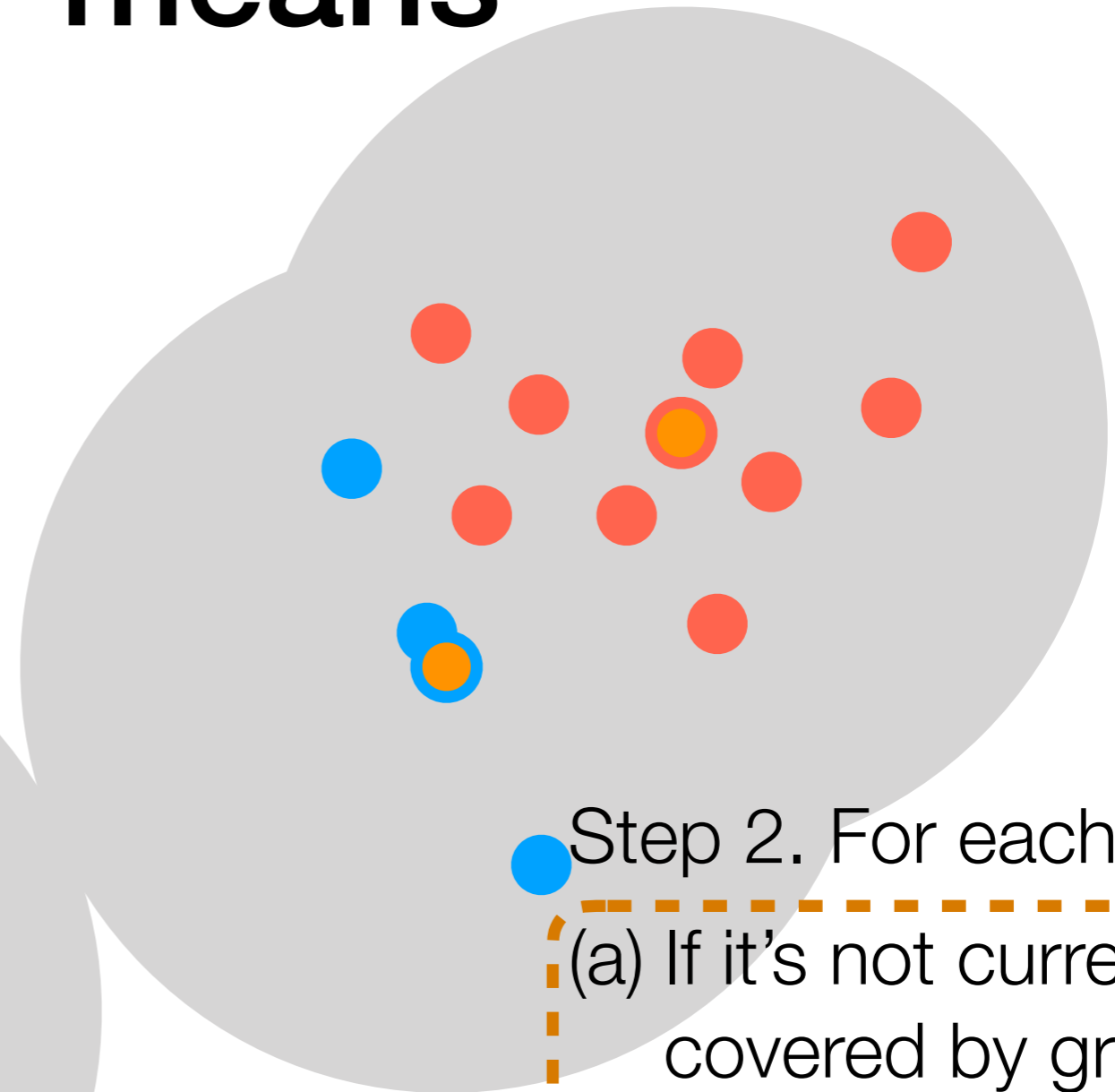
DP-means

Step 0: Pick concentration parameter $\lambda > 0$

Step 1: Start with everything in same cluster



Step 3. Recompute cluster centers



Step 2. For each point:

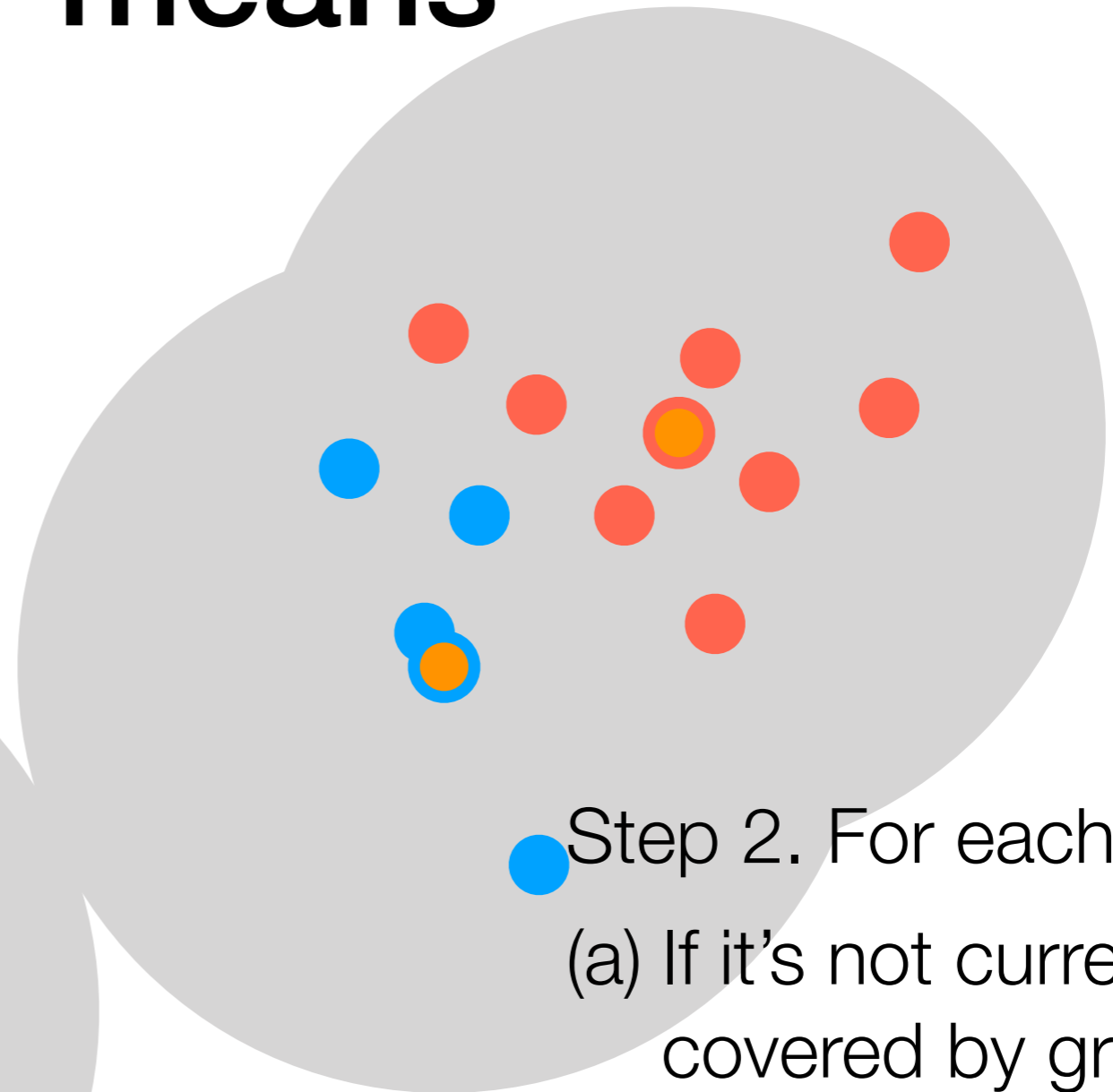
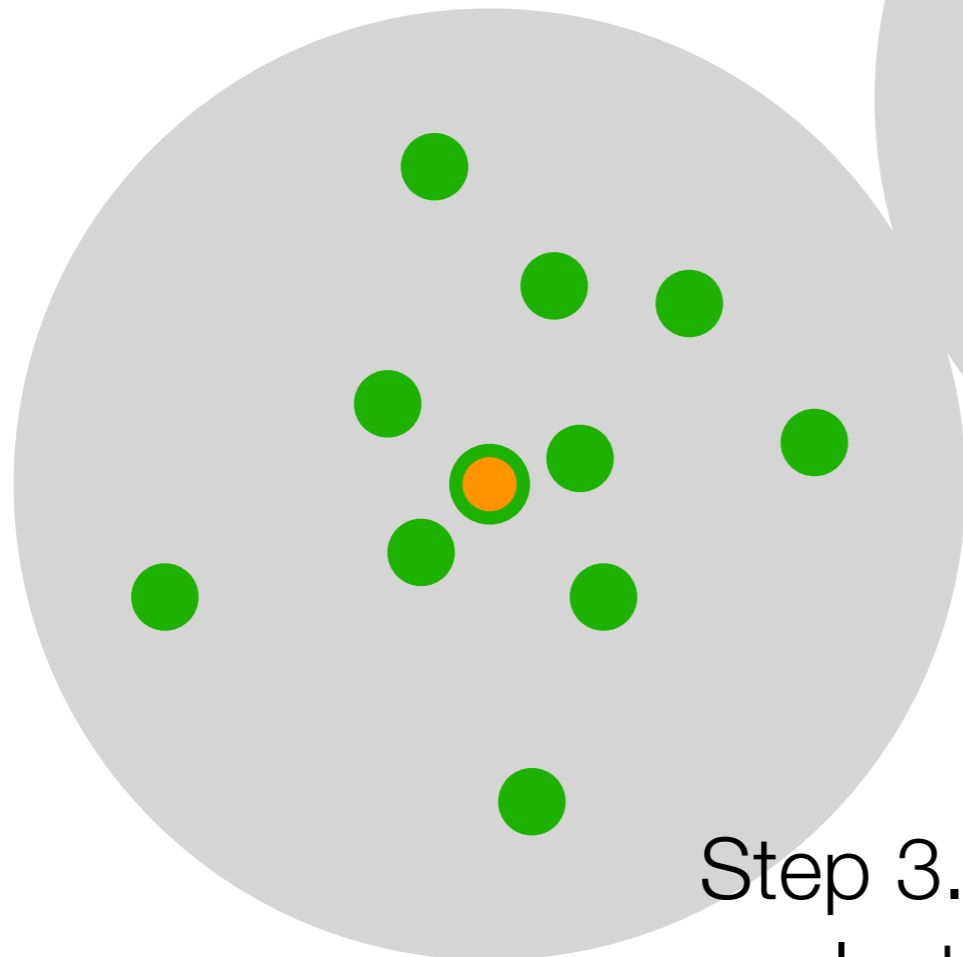
(a) If it's not currently covered by gray balls, make it a new cluster center

(b) Otherwise assign it to nearest cluster

DP-means

Step 0: Pick concentration parameter $\lambda > 0$

Step 1: Start with everything in same cluster



Step 2. For each point:
(a) If it's not currently covered by gray balls, make it a new cluster center

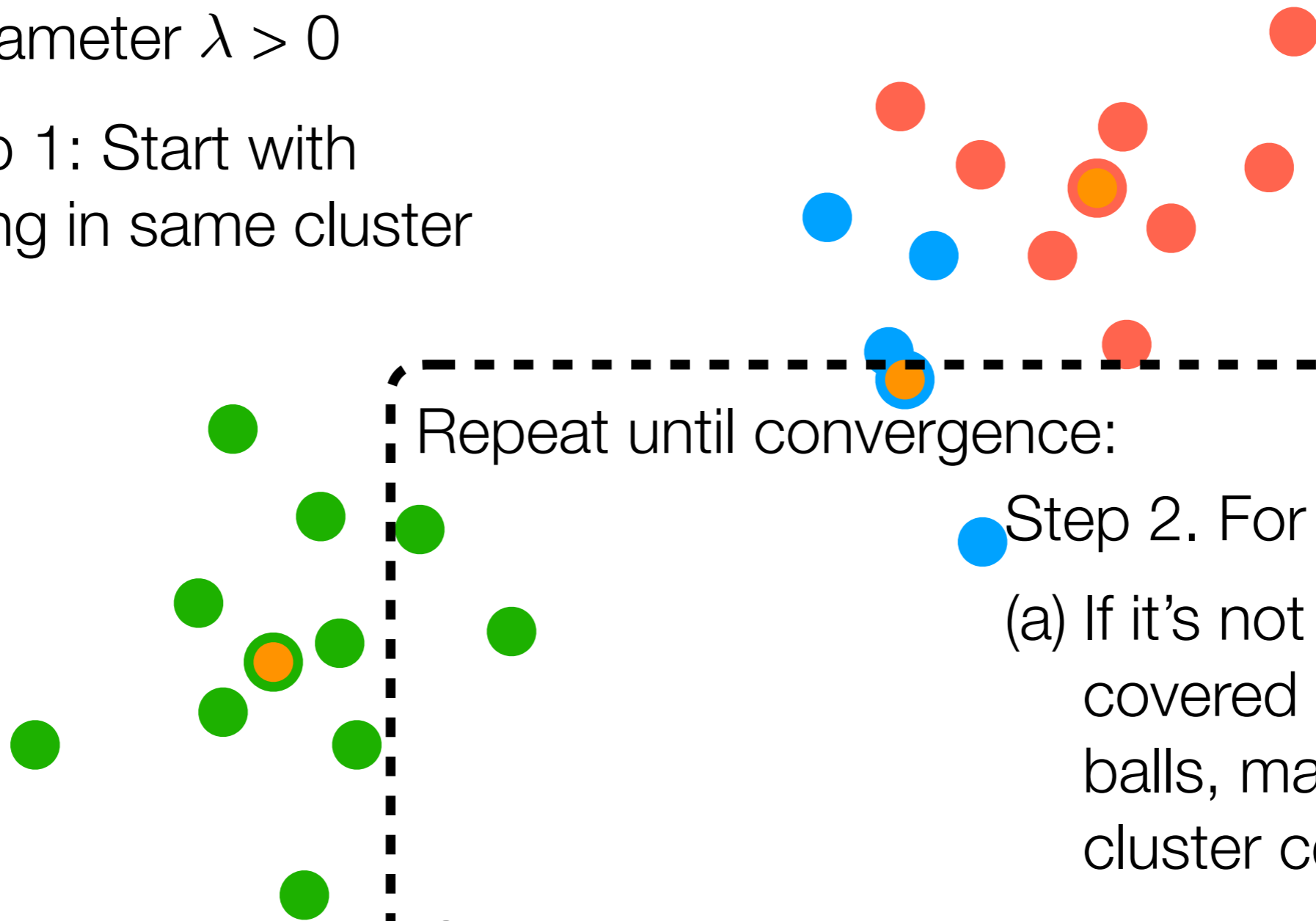
Step 3. Recompute cluster centers

(b) Otherwise assign it to nearest cluster

DP-means

Step 0: Pick concentration parameter $\lambda > 0$

Step 1: Start with everything in same cluster



Repeat until convergence:

Step 2. For each point:

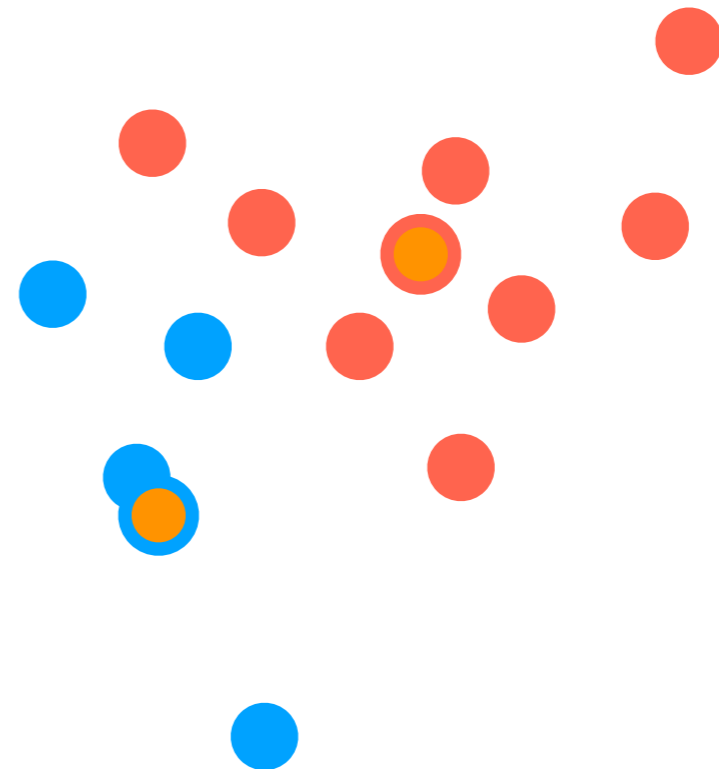
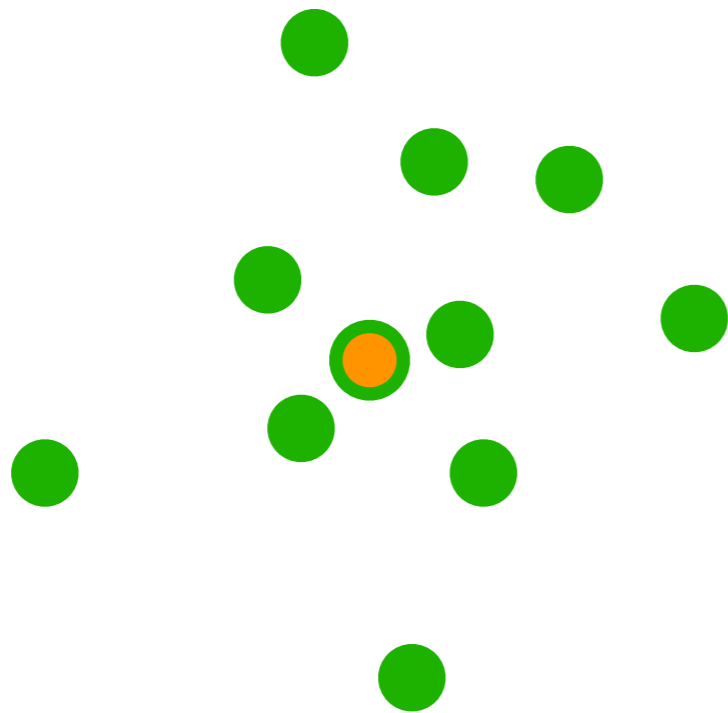
(a) If it's not currently covered by gray balls, make it a new cluster center

(b) Otherwise assign it to nearest cluster

Step 3. Recompute cluster centers

DP-means

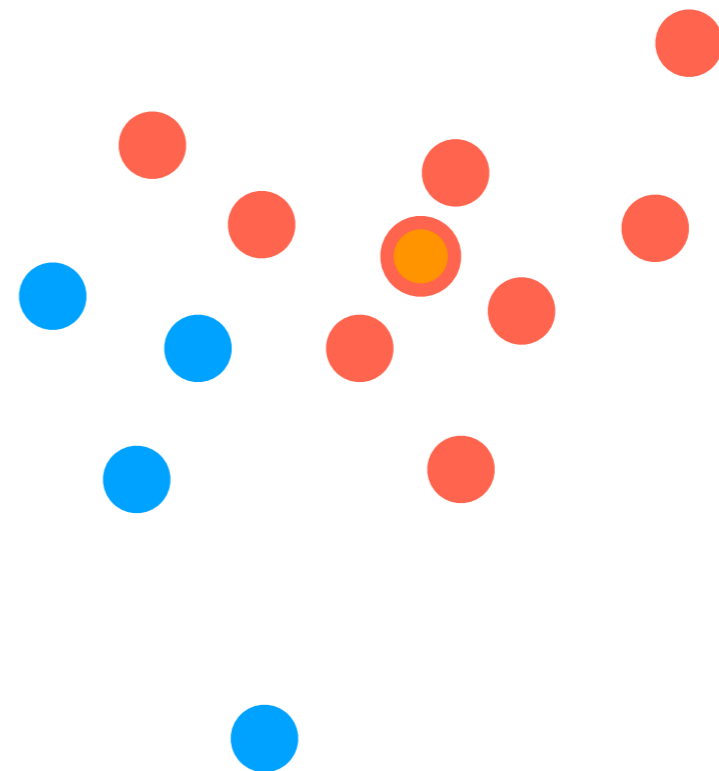
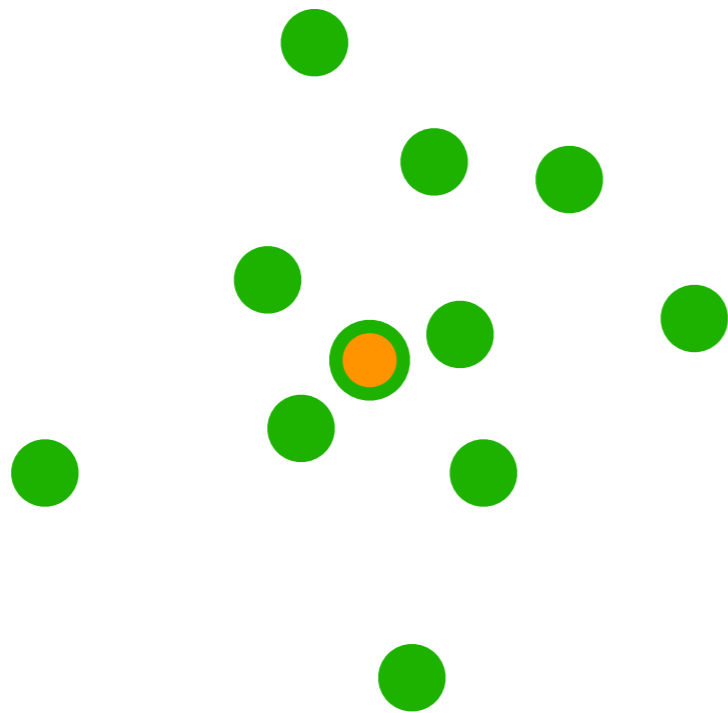
As you saw in the DP-GMM demo
(and is similar with DP-means),
DP-means can produce a few
extra small clusters



In practice: can reassign points in
small clusters to bigger clusters

DP-means

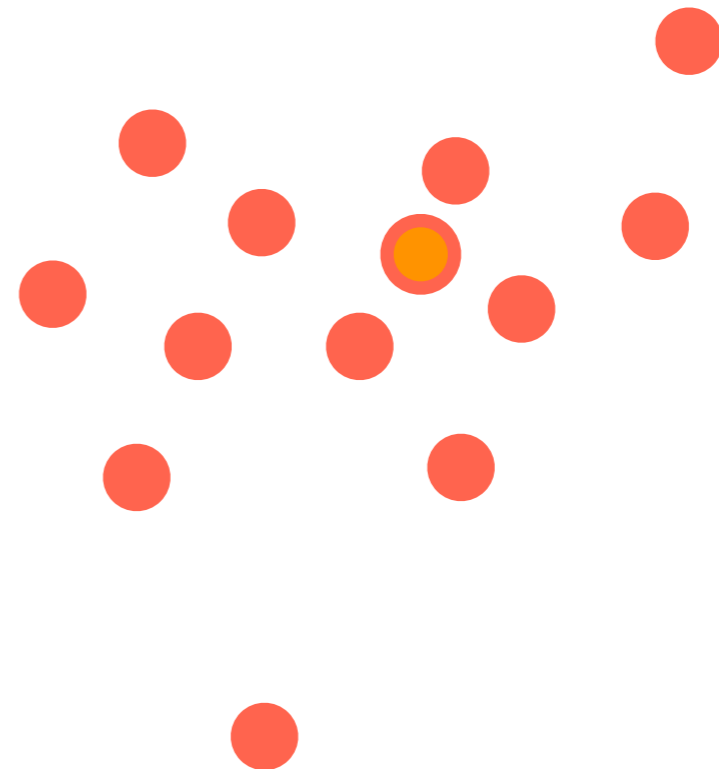
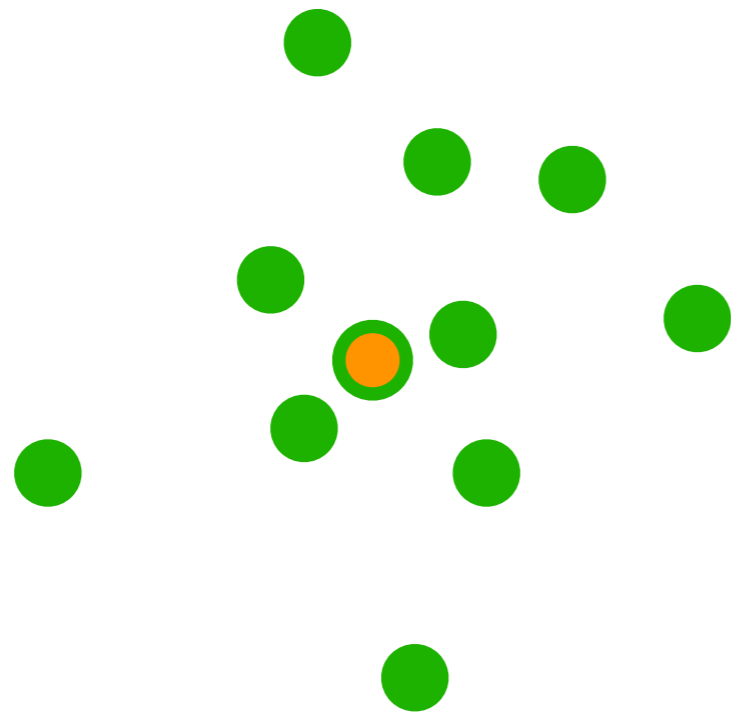
As you saw in the DP-GMM demo
(and is similar with DP-means),
DP-means can produce a few
extra small clusters



In practice: can reassign points in
small clusters to bigger clusters

DP-means

As you saw in the DP-GMM demo
(and is similar with DP-means),
DP-means can produce a few
extra small clusters

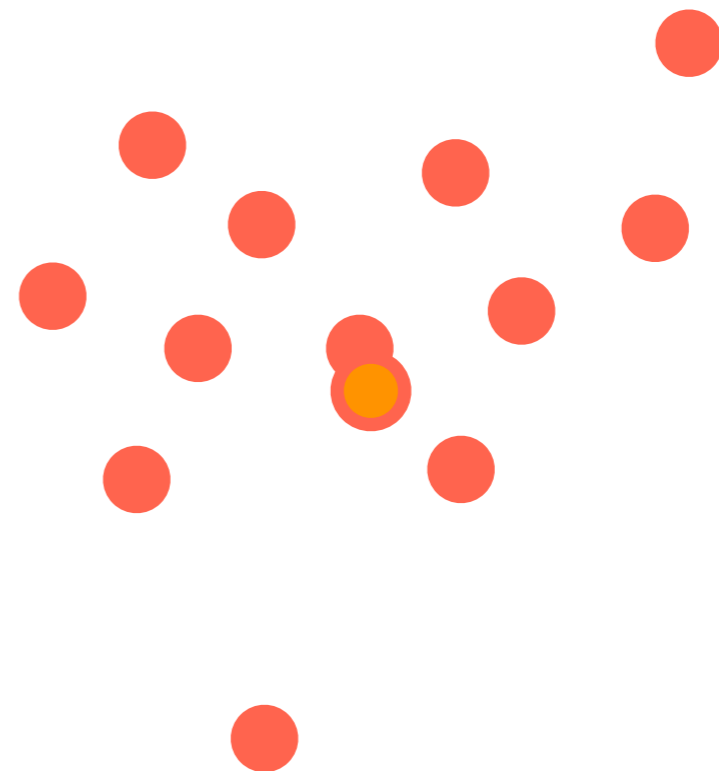
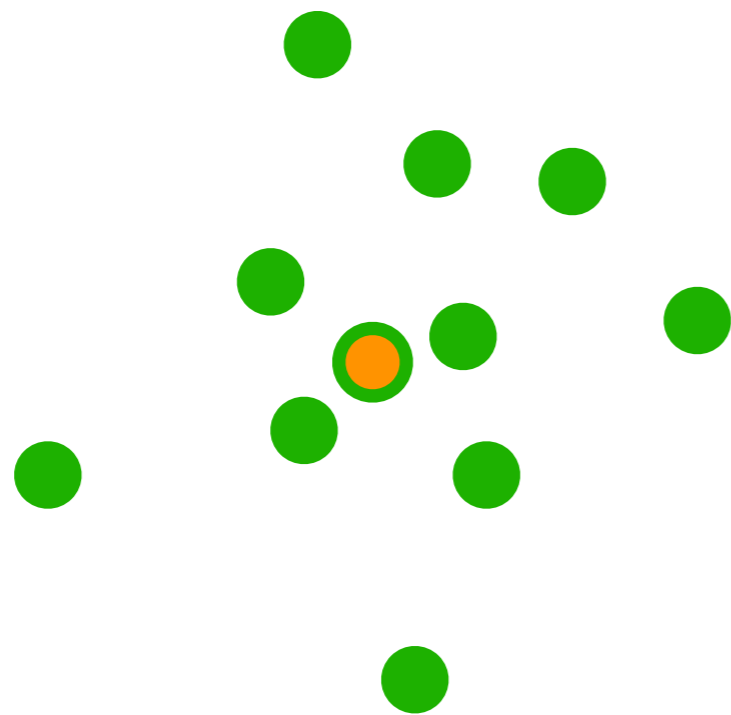


In practice: can reassign points in
small clusters to bigger clusters

Can recompute cluster centers

DP-means

As you saw in the DP-GMM demo
(and is similar with DP-means),
DP-means can produce a few
extra small clusters



In practice: can reassign points in
small clusters to bigger clusters

Can recompute cluster centers

Big picture: DP-means & DP-GMM have a parameter roughly controlling *size* of clusters rather than *number* of clusters

If your problem can more naturally be thought of as having cluster sizes that should not be too large, can use DP-means/DP-GMM instead of k-means/GMM

Real example. *Satellite image analysis of rural India to find villages*

Each cluster is a village: don't know how many villages there are total but rough upper bound on radius of village can be specified

→ DP-means can provide a decent solution!

Going from Similarities to Clusters

There's a whole zoo of clustering methods

Several main categories (although there are other categories!):

Generative models

1. Pretend data generated by specific model with parameters
2. Learn the parameters ("fit model to data")
3. Use fitted model to determine cluster assignments

Hierarchical clustering

Top-down: Start with everything in 1 cluster and decide on how to recursively split

Bottom-up: Start with everything in its own cluster and decide on how to iteratively merge clusters

Density-based clustering

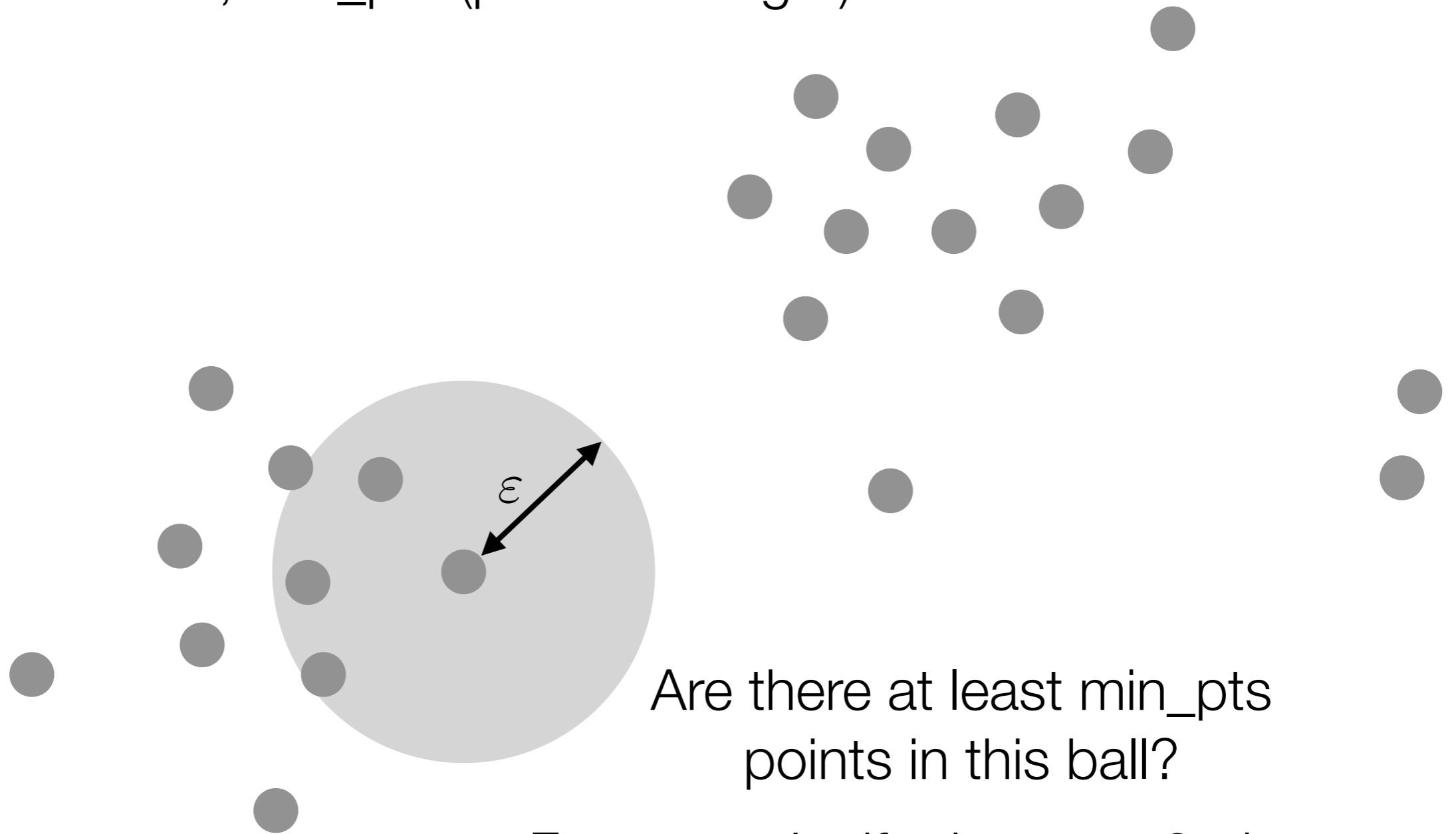
Based on finding parts of the data with higher density

Density-based Clustering

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)



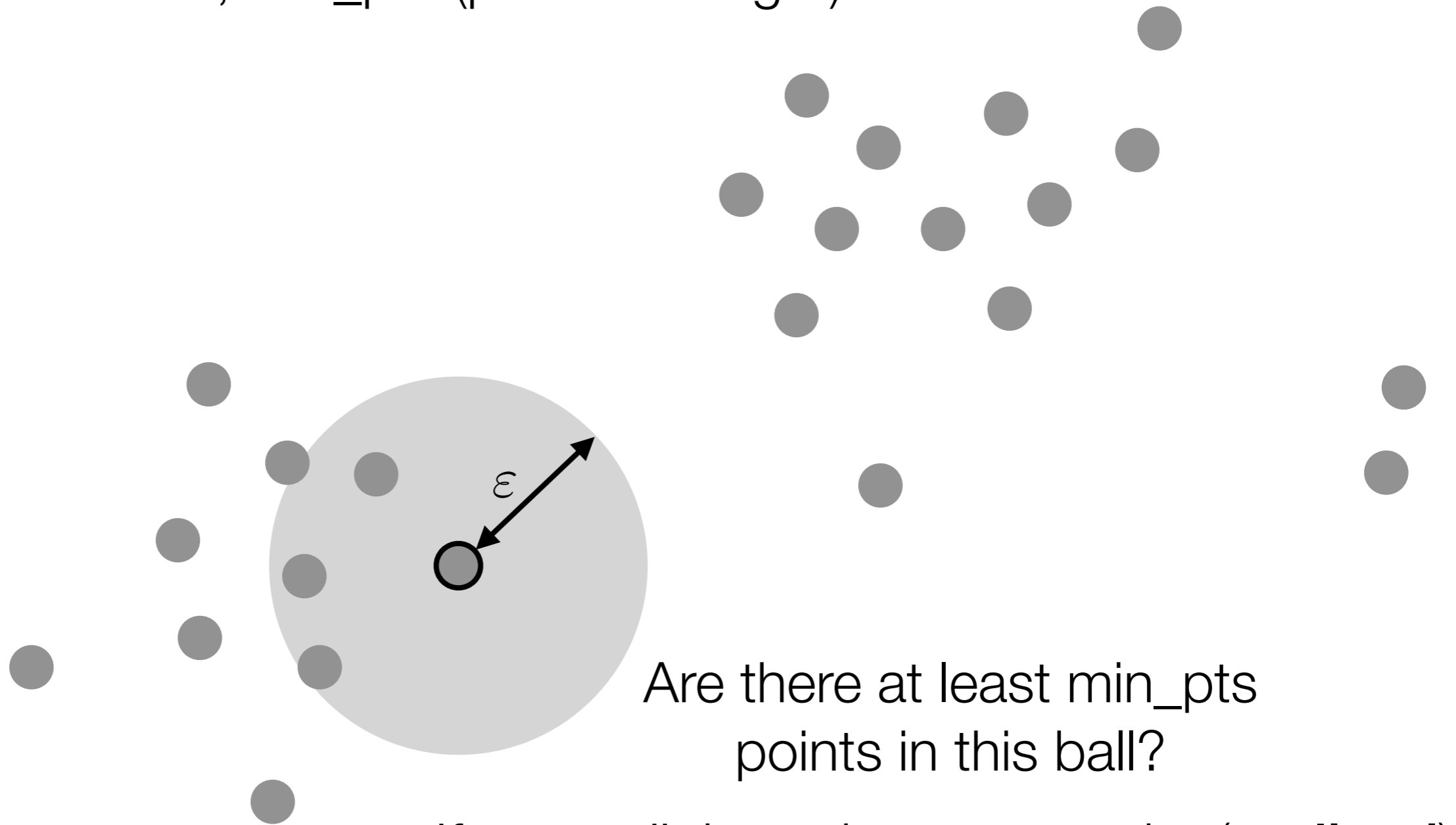
For example, if $\text{min_pts} = 3$, then yes

For example, if $\text{min_pts} = 10$, then no

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

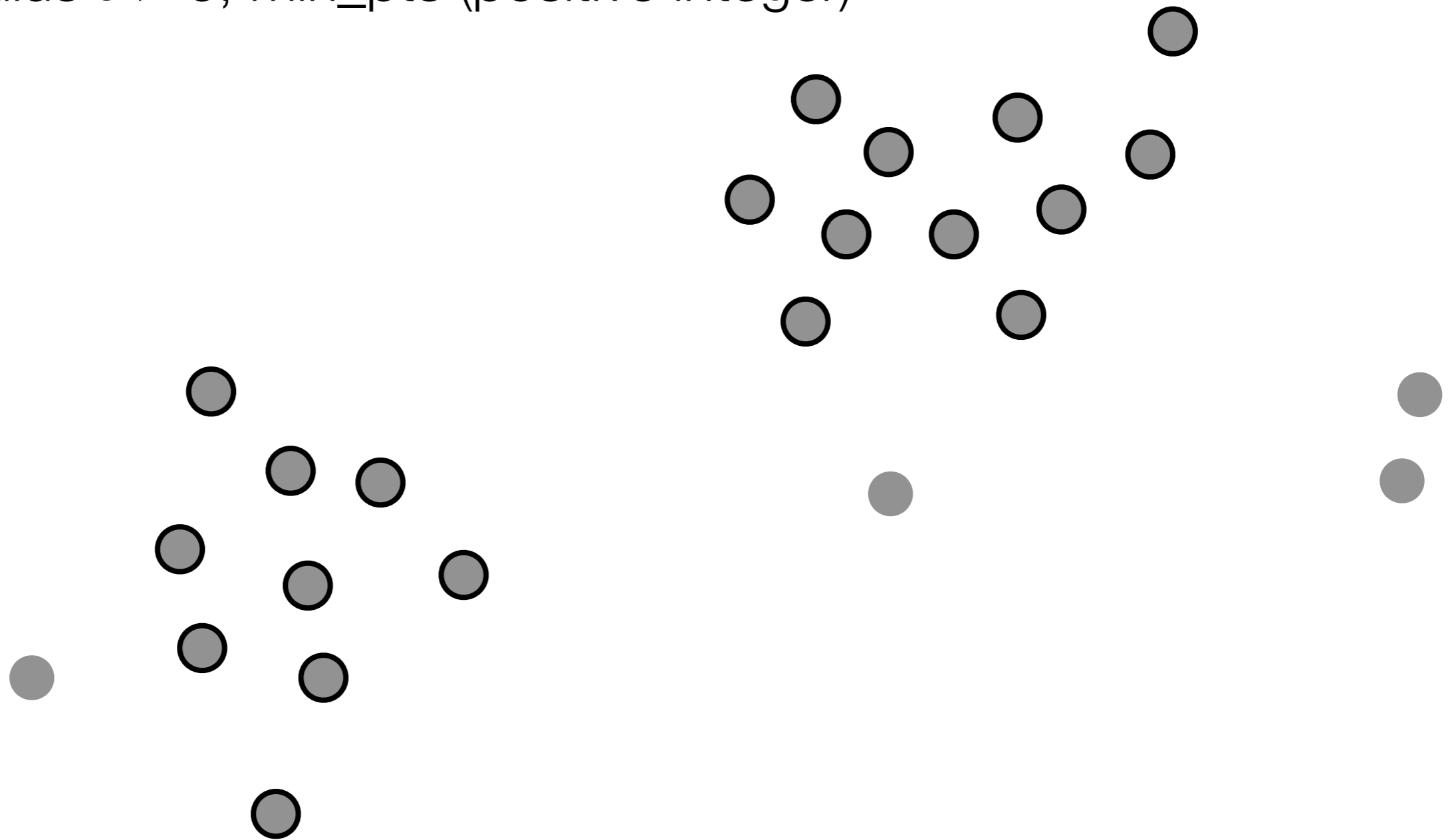


If yes: call the point a core point (**outlined**)

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

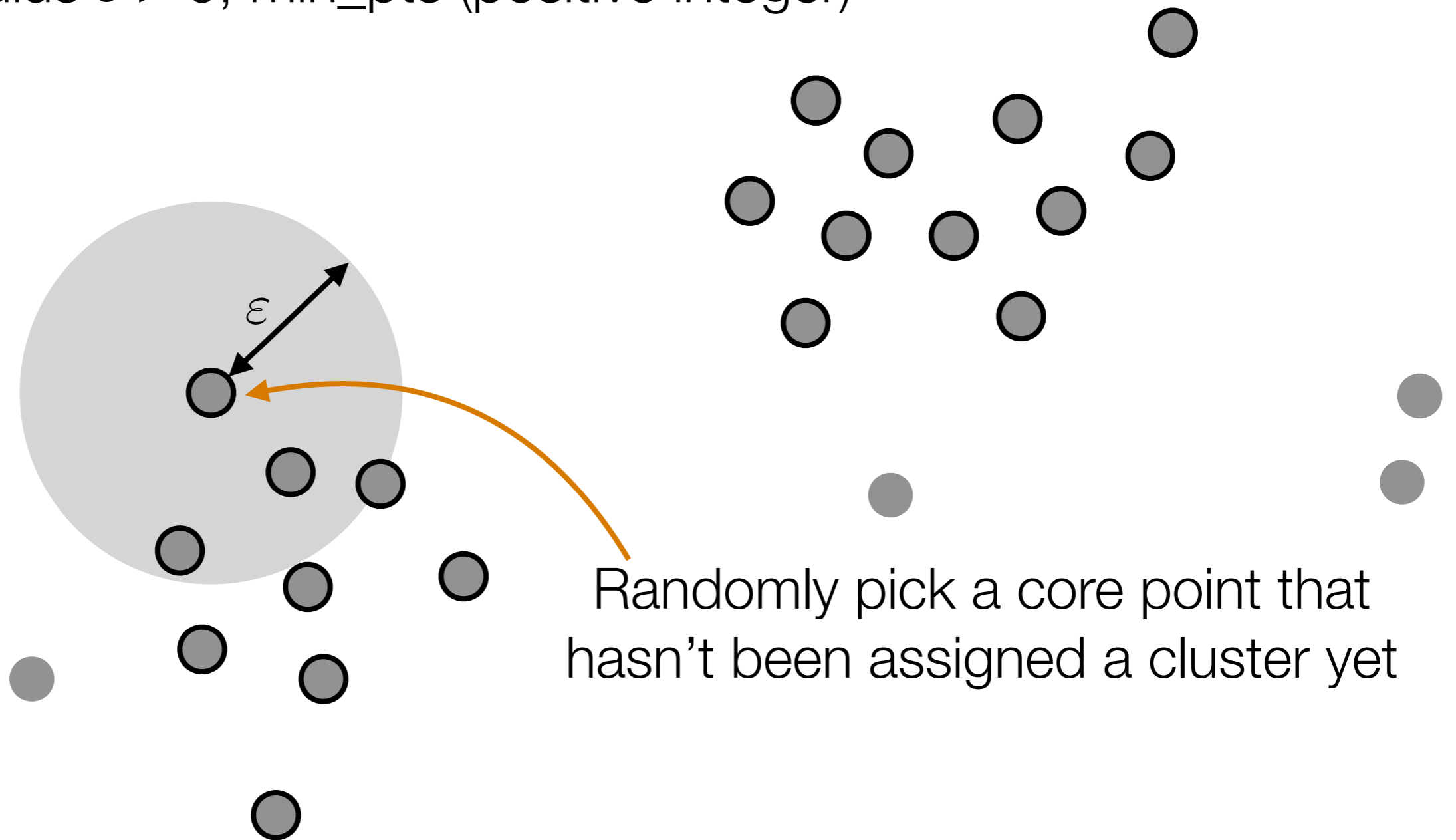


Core points (**outlined**)

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)



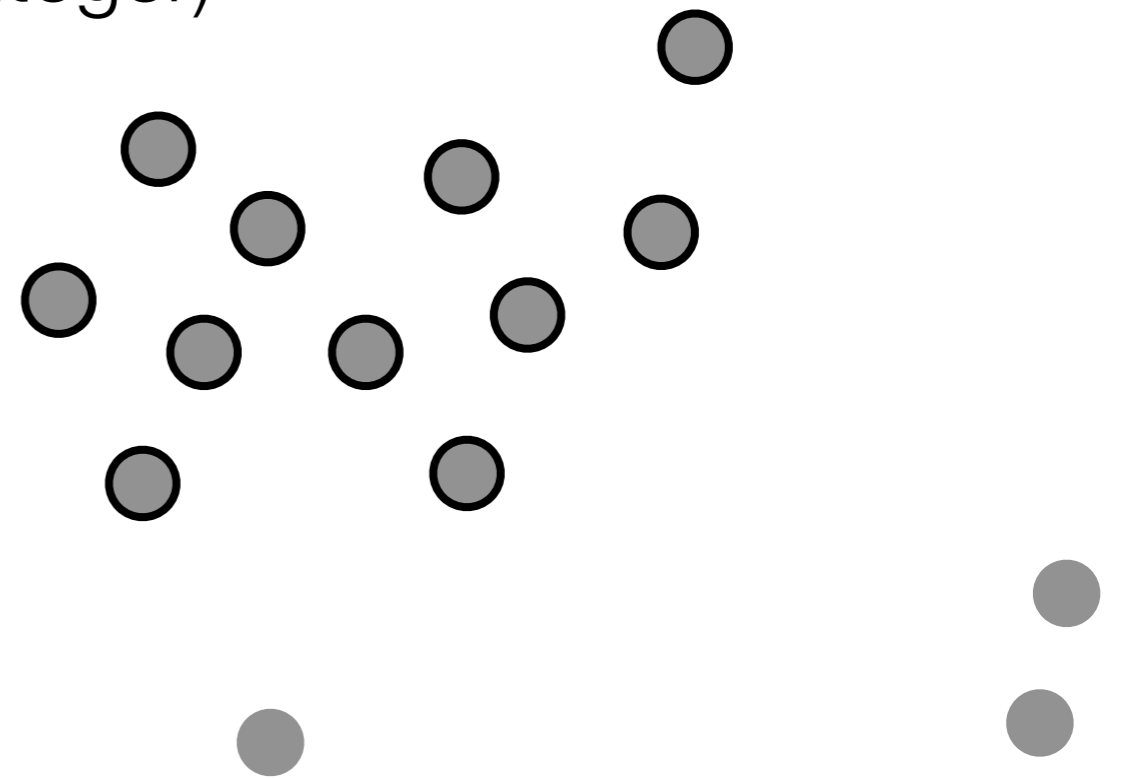
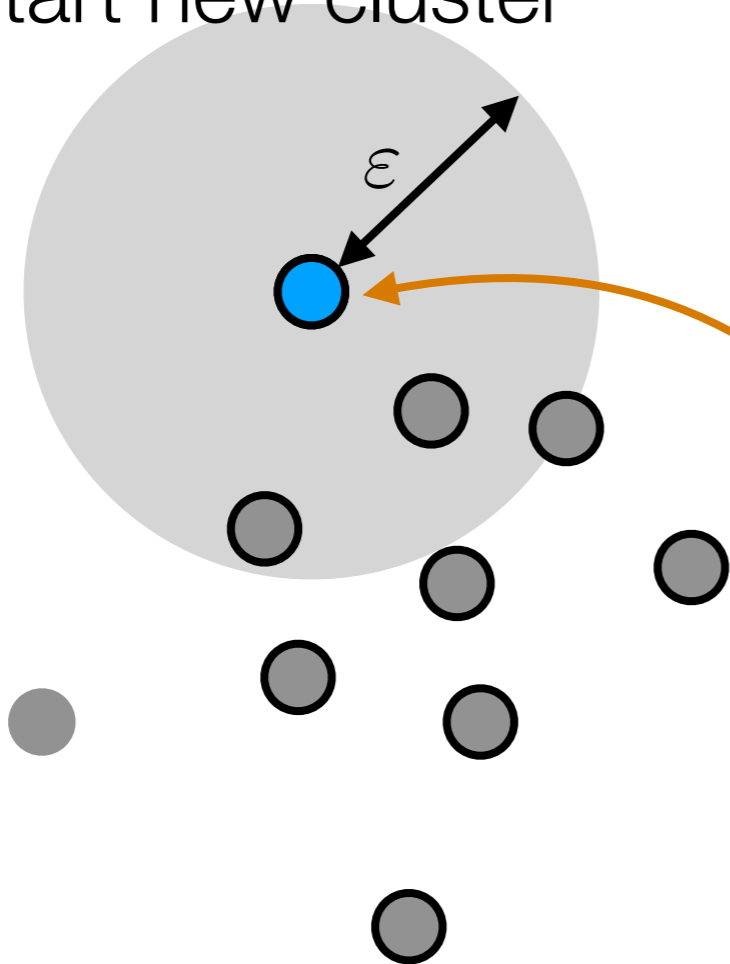
Core points (**outlined**)

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

Start new cluster



Randomly pick a core point that hasn't been assigned a cluster yet

Core points (**outlined**)

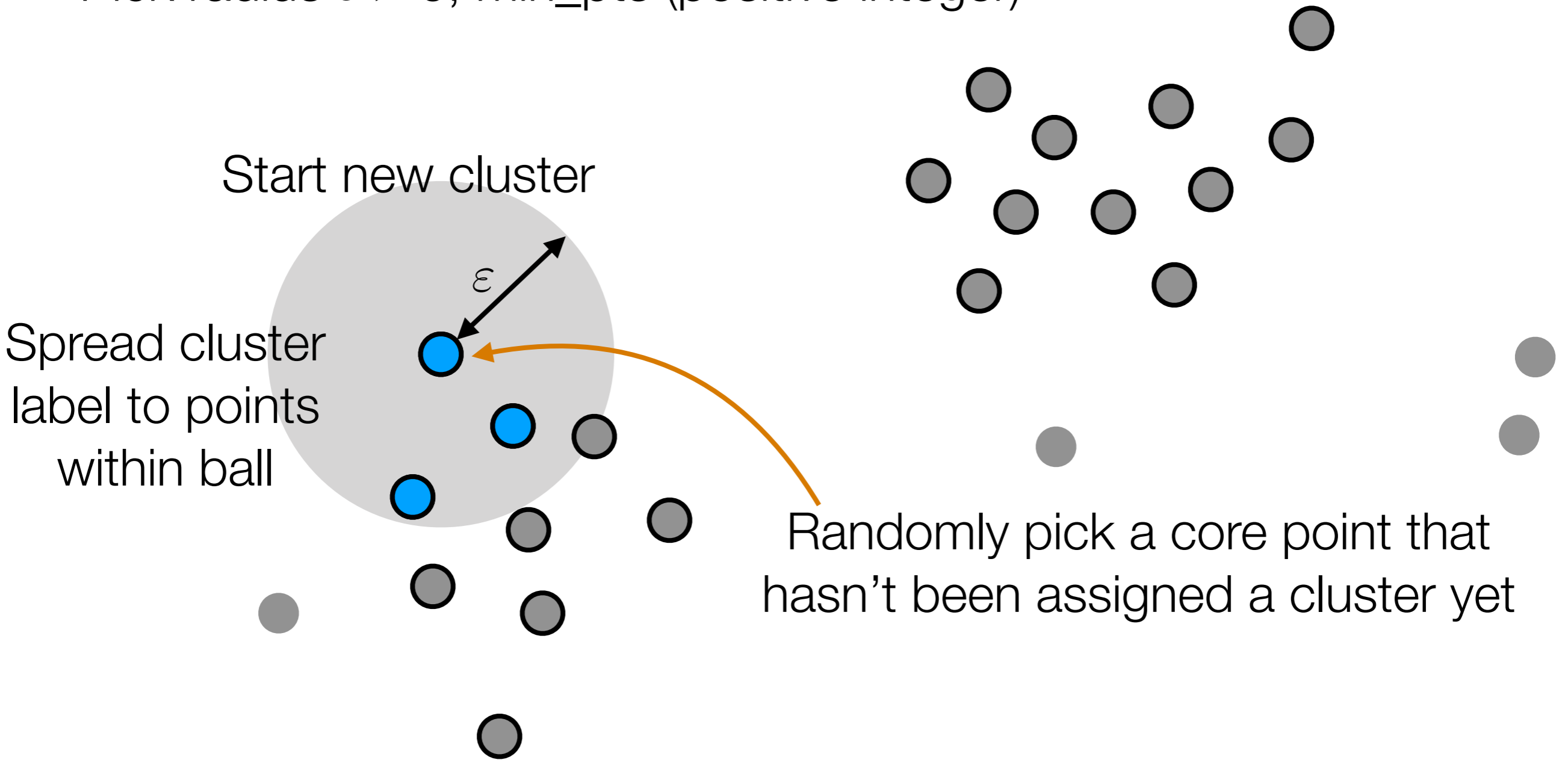
DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

Start new cluster

Spread cluster label to points within ball



Core points (**outlined**)

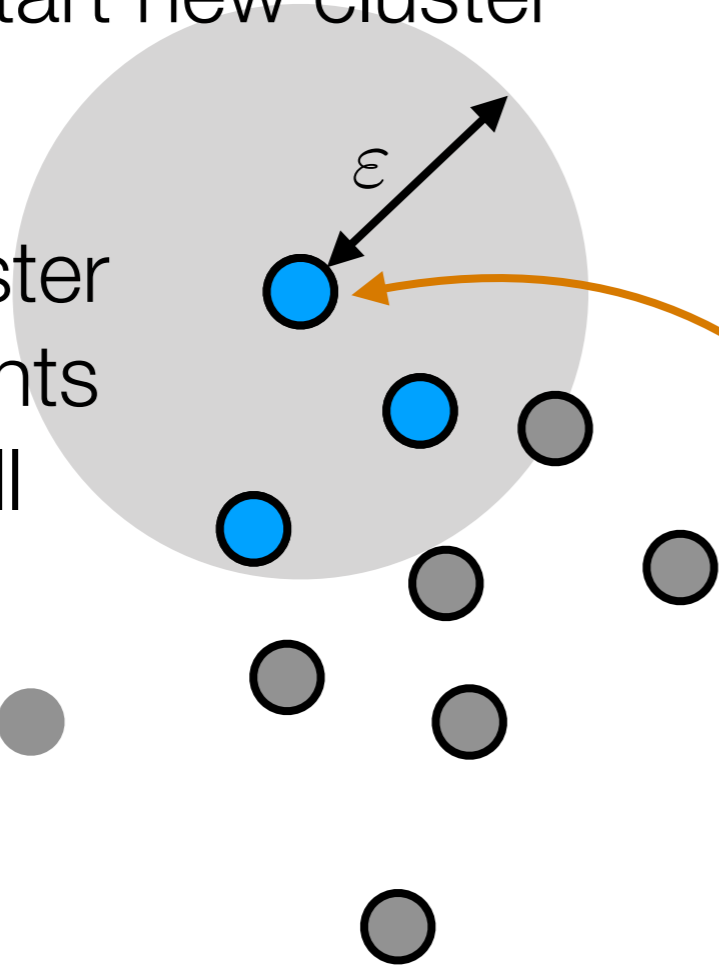
DBSCAN

Let's choose $\text{min_pts} = 3$

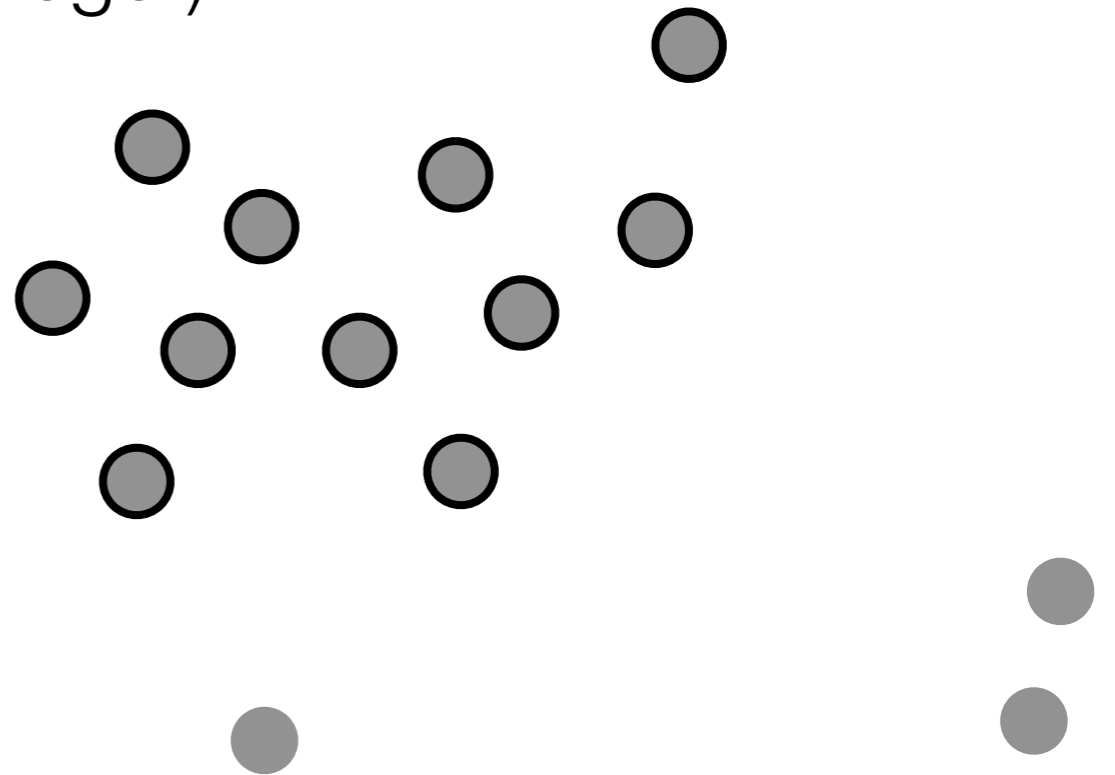
Pick radius $\varepsilon > 0$, min_pts (positive integer)

Start new cluster

Spread cluster label to points within ball



Randomly pick a core point that hasn't been assigned a cluster yet



For core points within the ball:
spread the cluster label

Core points (**outlined**)

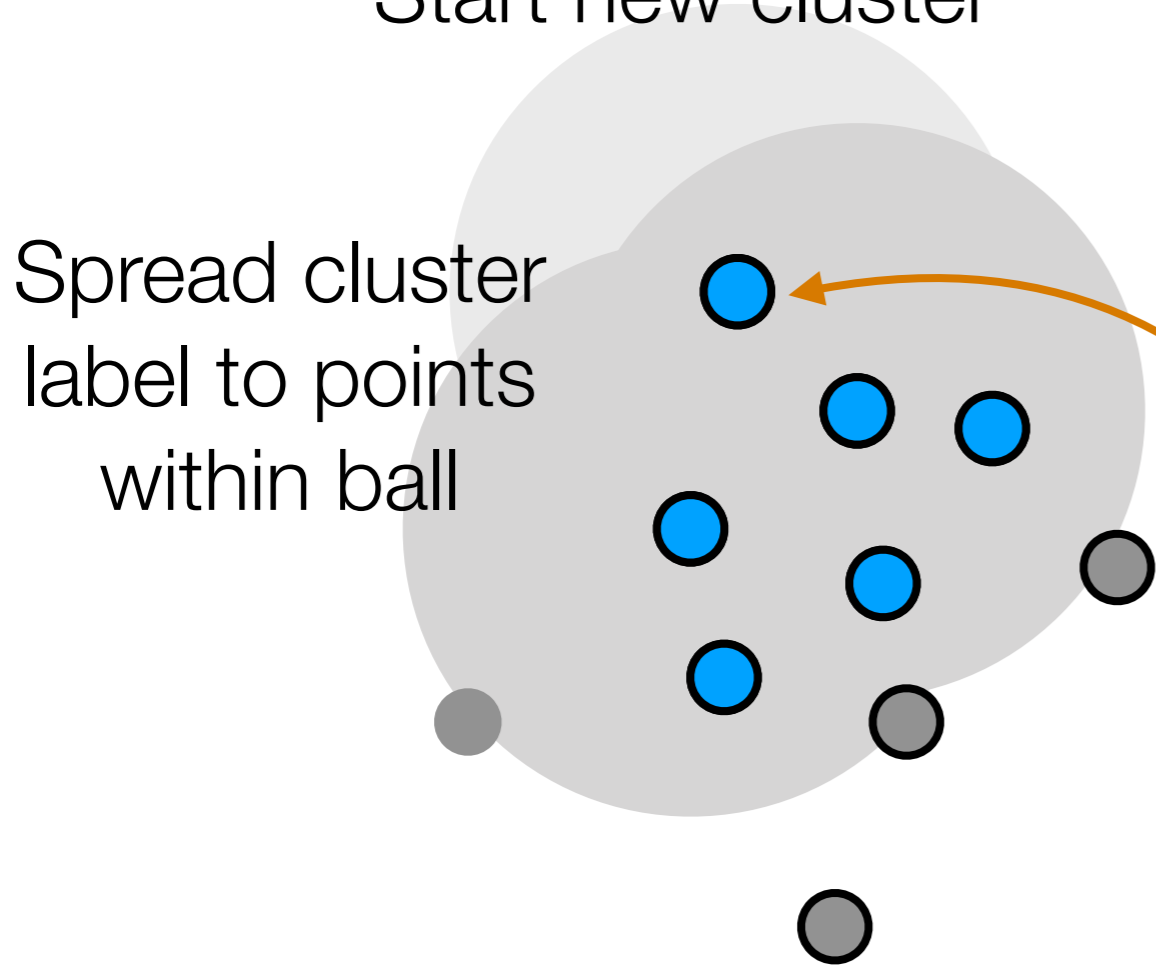
DBSCAN

Let's choose $\text{min_pts} = 3$

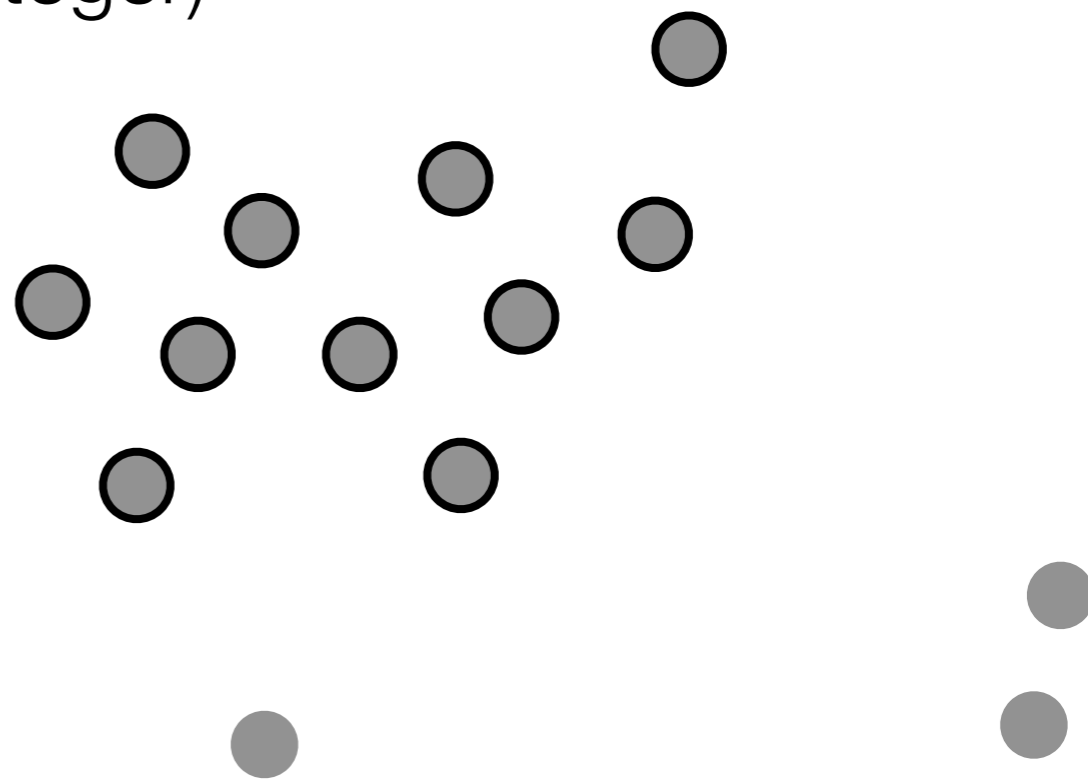
Pick radius $\varepsilon > 0$, min_pts (positive integer)

Start new cluster

Spread cluster label to points within ball



Randomly pick a core point that hasn't been assigned a cluster yet



For core points within the ball:
spread the cluster label

Core points (**outlined**)

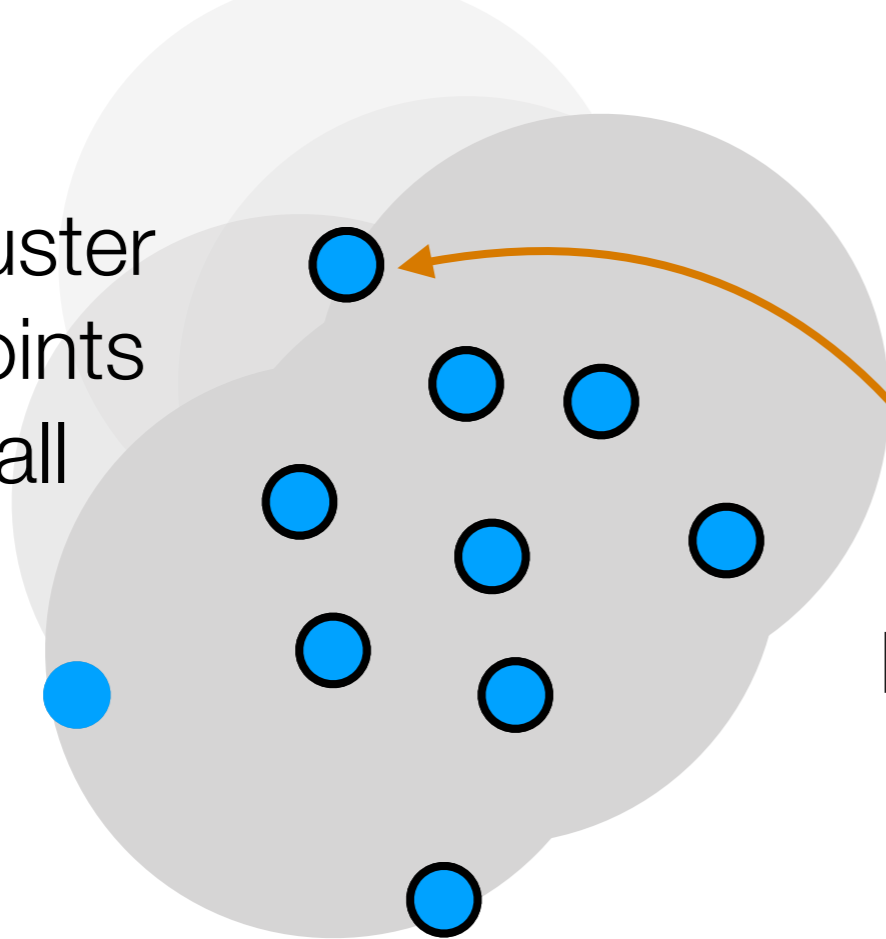
DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

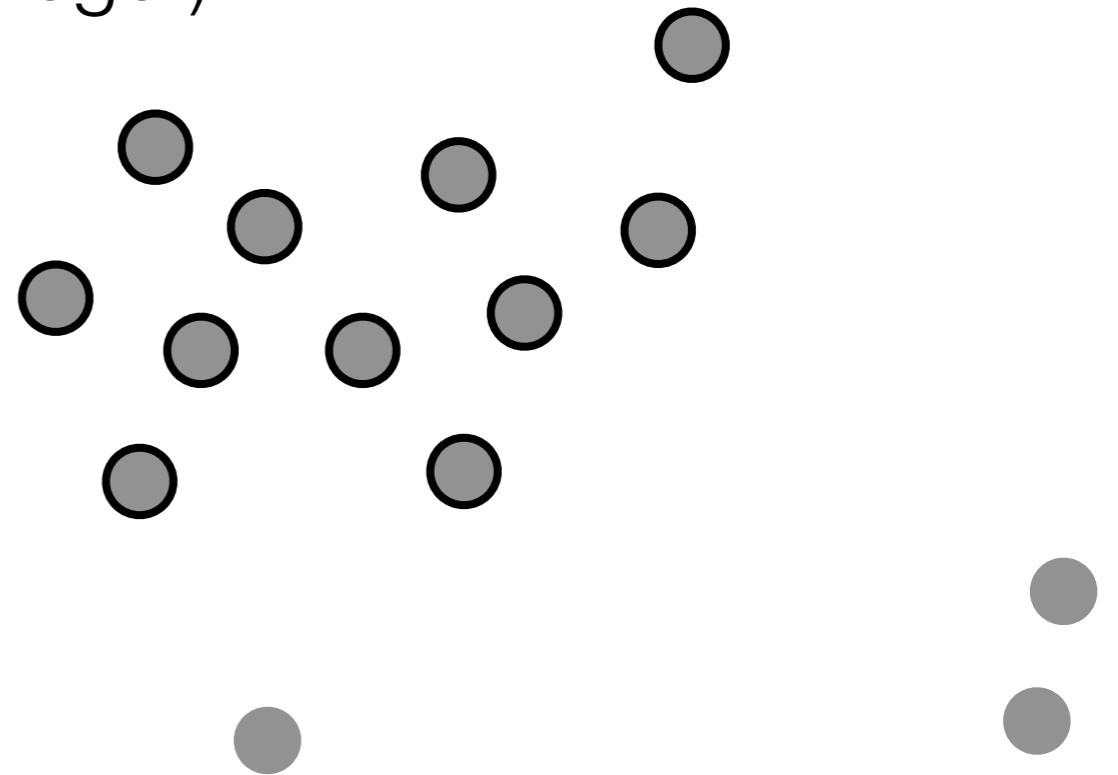
Start new cluster

Spread cluster label to points within ball



Randomly pick a core point that hasn't been assigned a cluster yet

For core points within the ball:
spread the cluster label



Core points (**outlined**)

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

Start new cluster

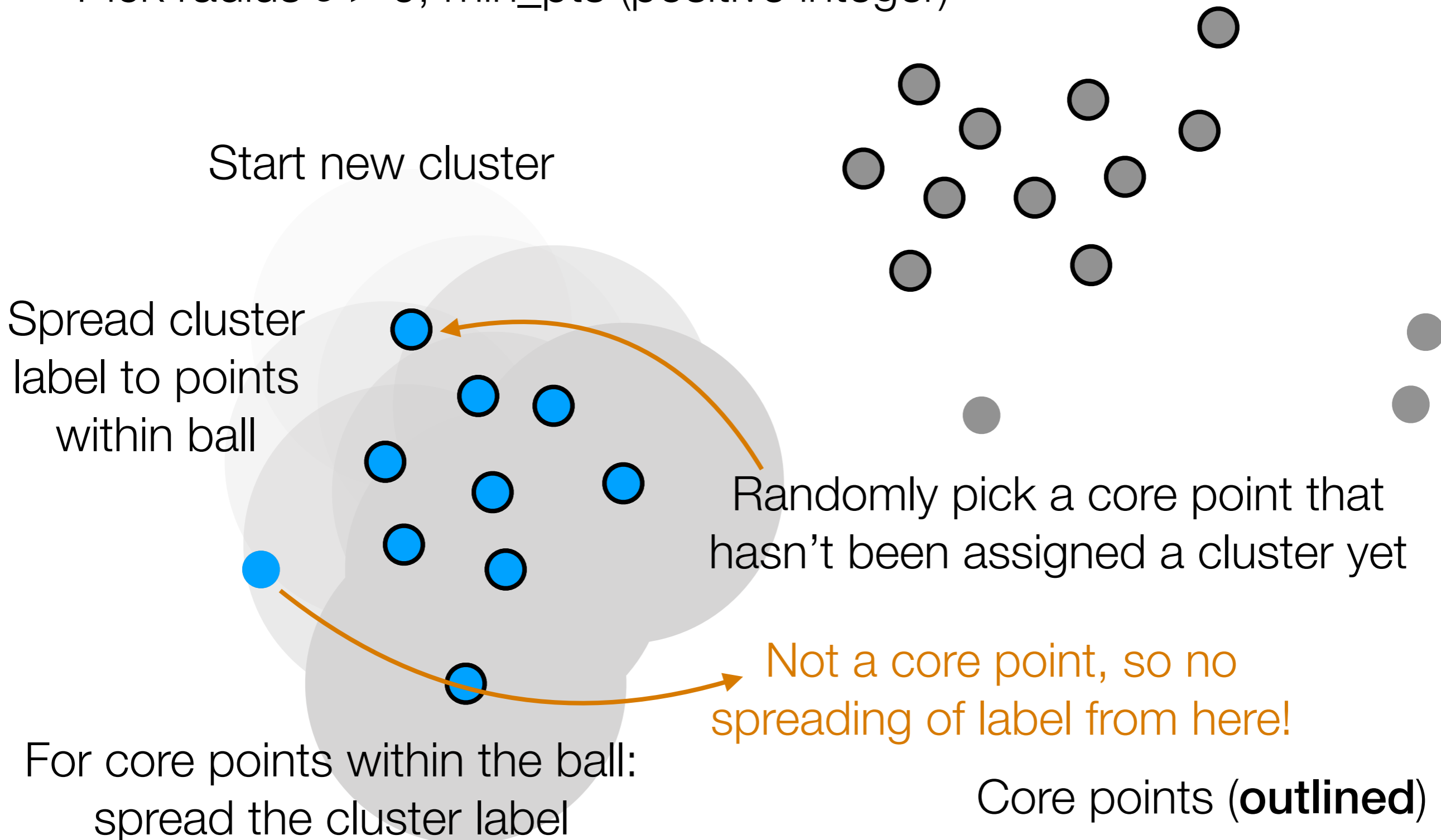
Spread cluster label to points within ball

Randomly pick a core point that hasn't been assigned a cluster yet

Not a core point, so no spreading of label from here!

For core points within the ball: spread the cluster label

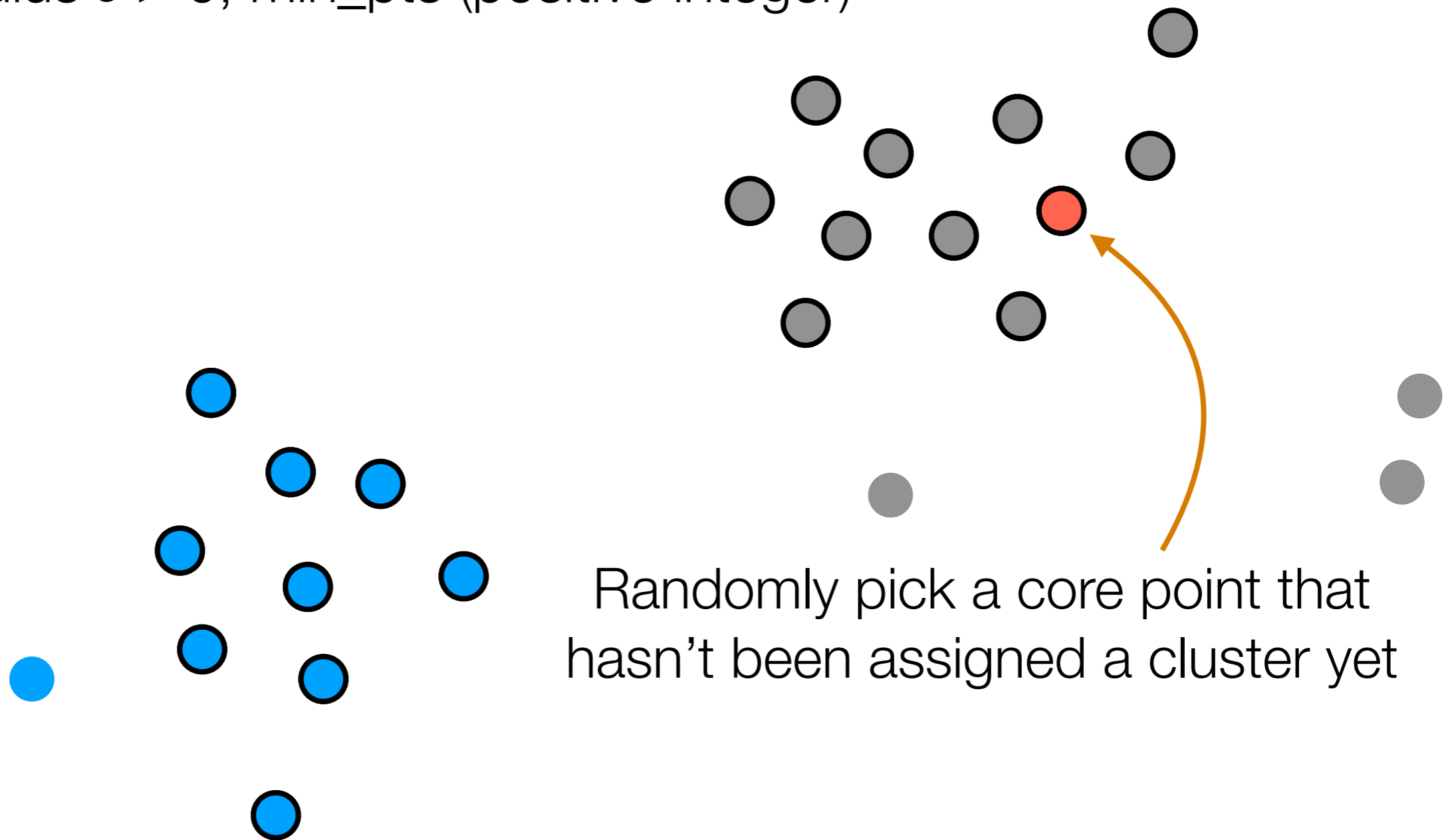
Core points (**outlined**)



DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)



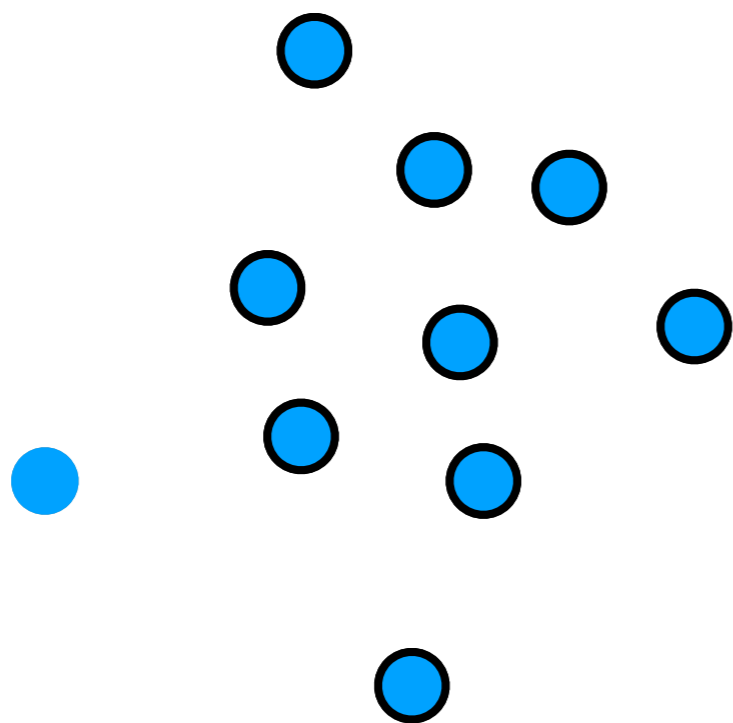
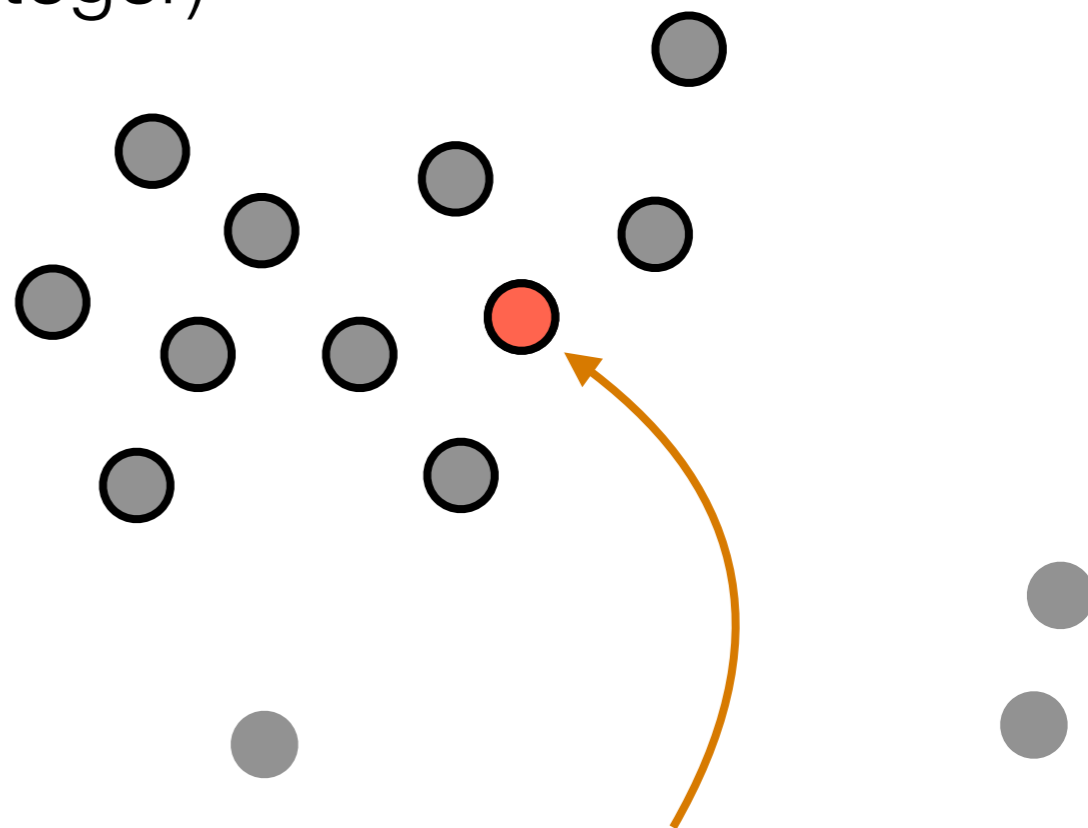
Core points (**outlined**)

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

Repeat “virus-spreading” like cluster label spreading; again, no spreading starting from non-core points



Randomly pick a core point that hasn't been assigned a cluster yet

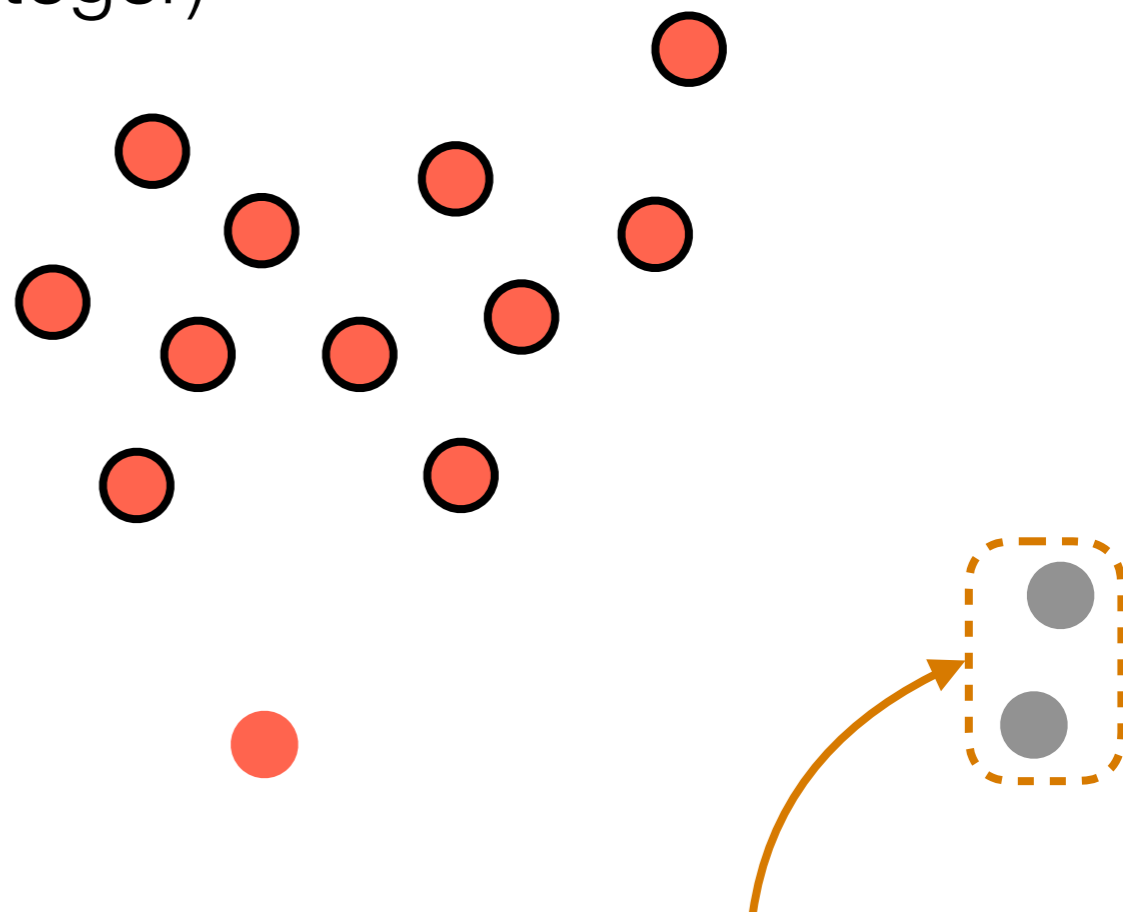
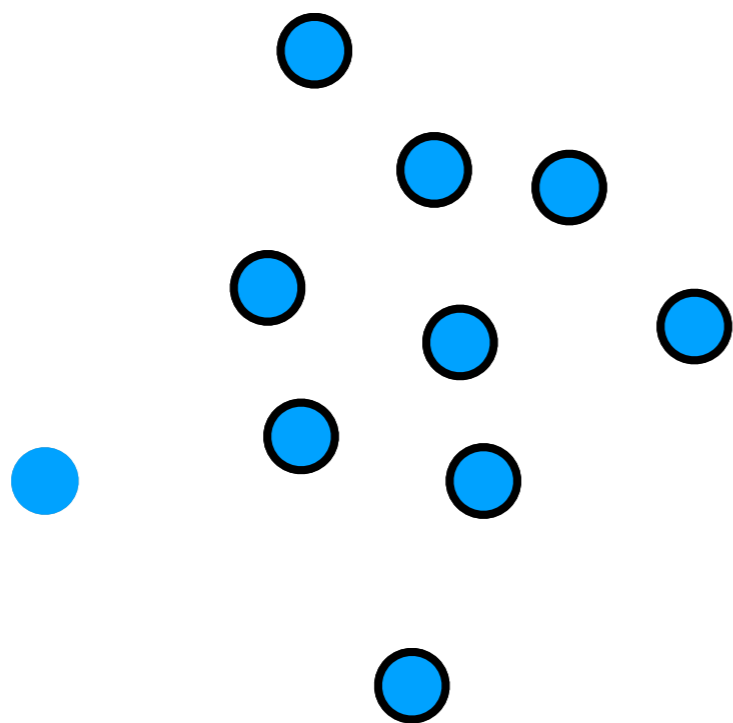
Core points (**outlined**)

DBSCAN

Let's choose $\text{min_pts} = 3$

Pick radius $\varepsilon > 0$, min_pts (positive integer)

Repeat “virus-spreading” like cluster label spreading; again, no spreading starting from non-core points



Some points might actually *not* get clustered by DBSCAN and be declared as outliers!

Core points (**outlined**)

DBSCAN

Demo