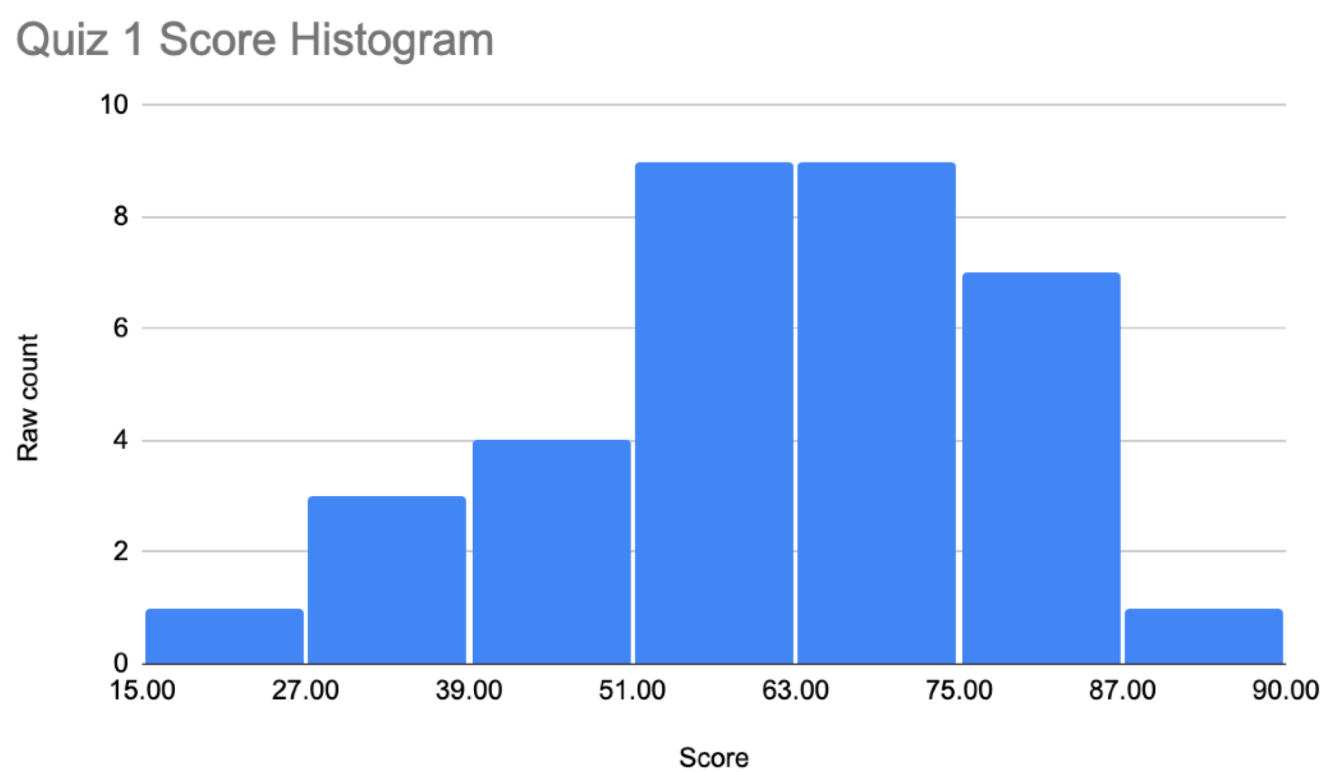
Carnegie Mellon University
Heinz College

Unstructured Data Analysis

Lecture 8: Clustering (cont'd), topic modeling

George Chen

Quiz 1 Results



Mean: 62.1,   std dev: 16.6,   max: 89

This sort of distribution is actually typical for this course!

Letter grades are determined based on a curve

The curve for Section K4 will be different from the A4/B4 sections since Erick is grading everything for K4 (the other sections are graded by other TA's & me)

Extremely rarely do students fail my class (usually this is due to cheating)

Last Time: Automatically Choosing *k*, the Number of Clusters

Simple strategy:
(1) compute a score for each ***k*** you're willing to try
(2) use whichever ***k*** achieves the best score

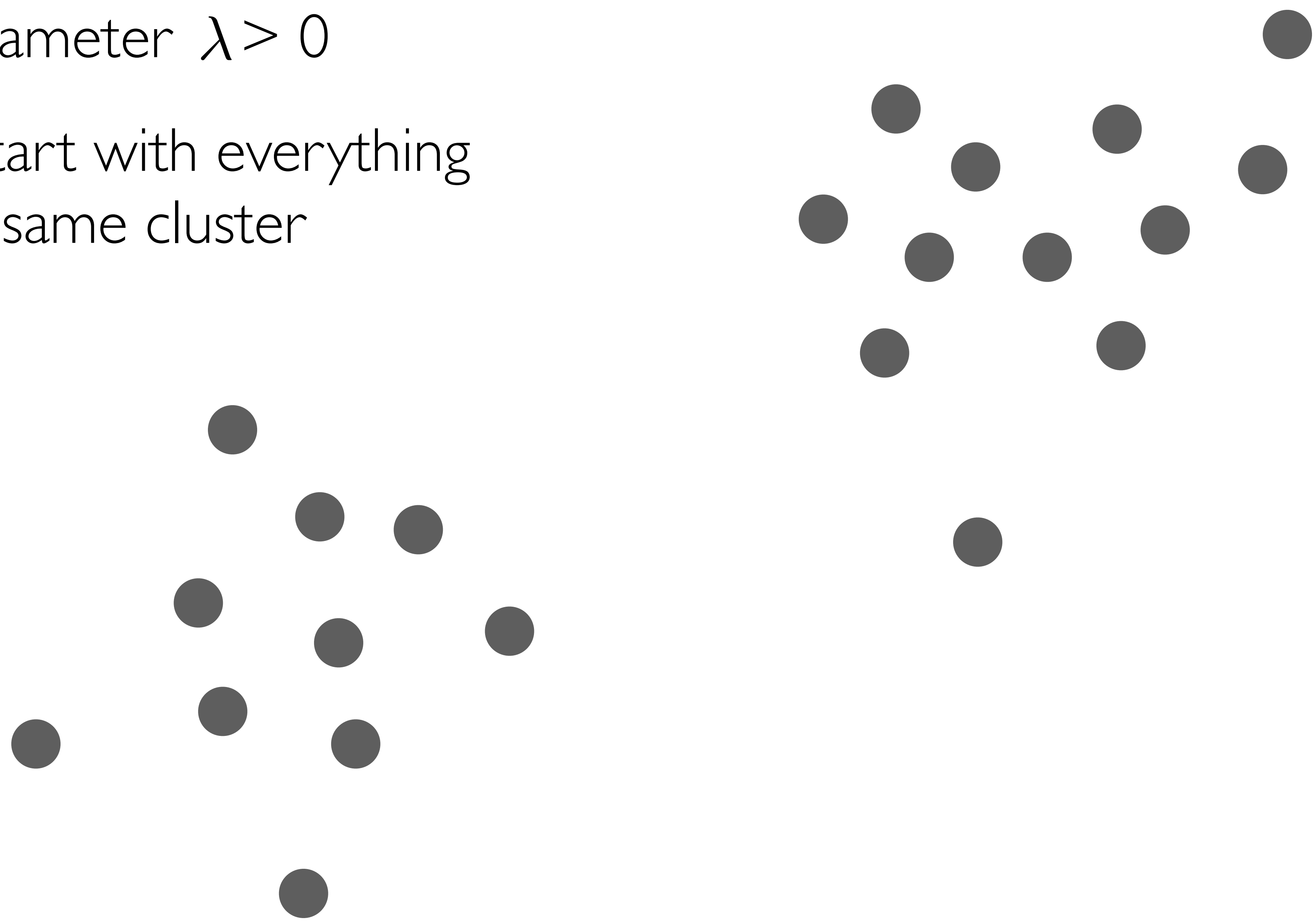
- There is no single best score function
- Fitting k-means/GMMs is in general *random* (for example, in the CH index demo, if we don't set `random_state`, then the CH indices computed will be different every time we run the code, and the best ***k*** could change!)

There are other clustering methods that do not require specifying the number of clusters (e.g., DP-means, DP-GMM, many variants of hierarchical clustering, density-based clustering)

DP-means

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

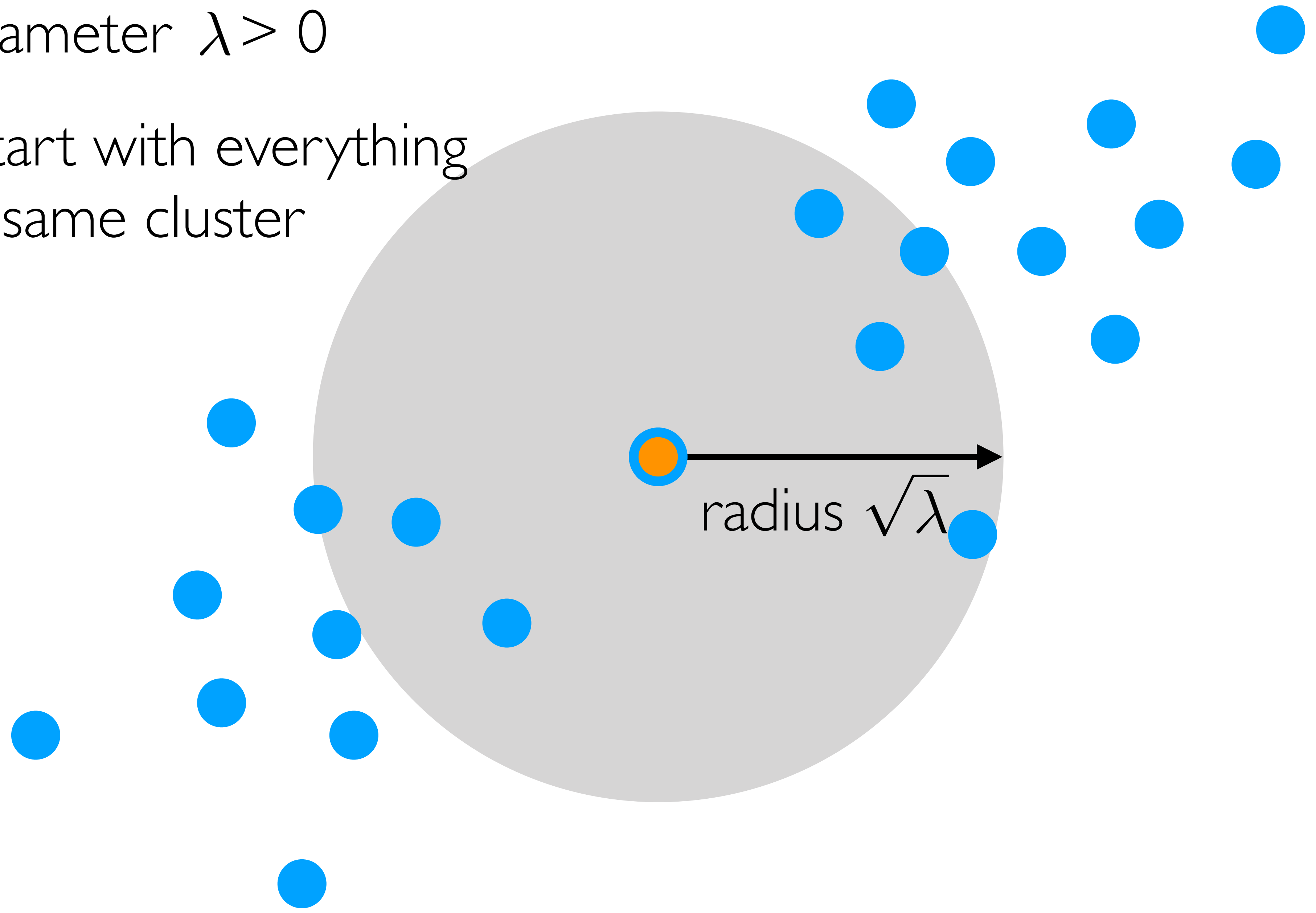
Step 1. Start with everything in same cluster



DP-means

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

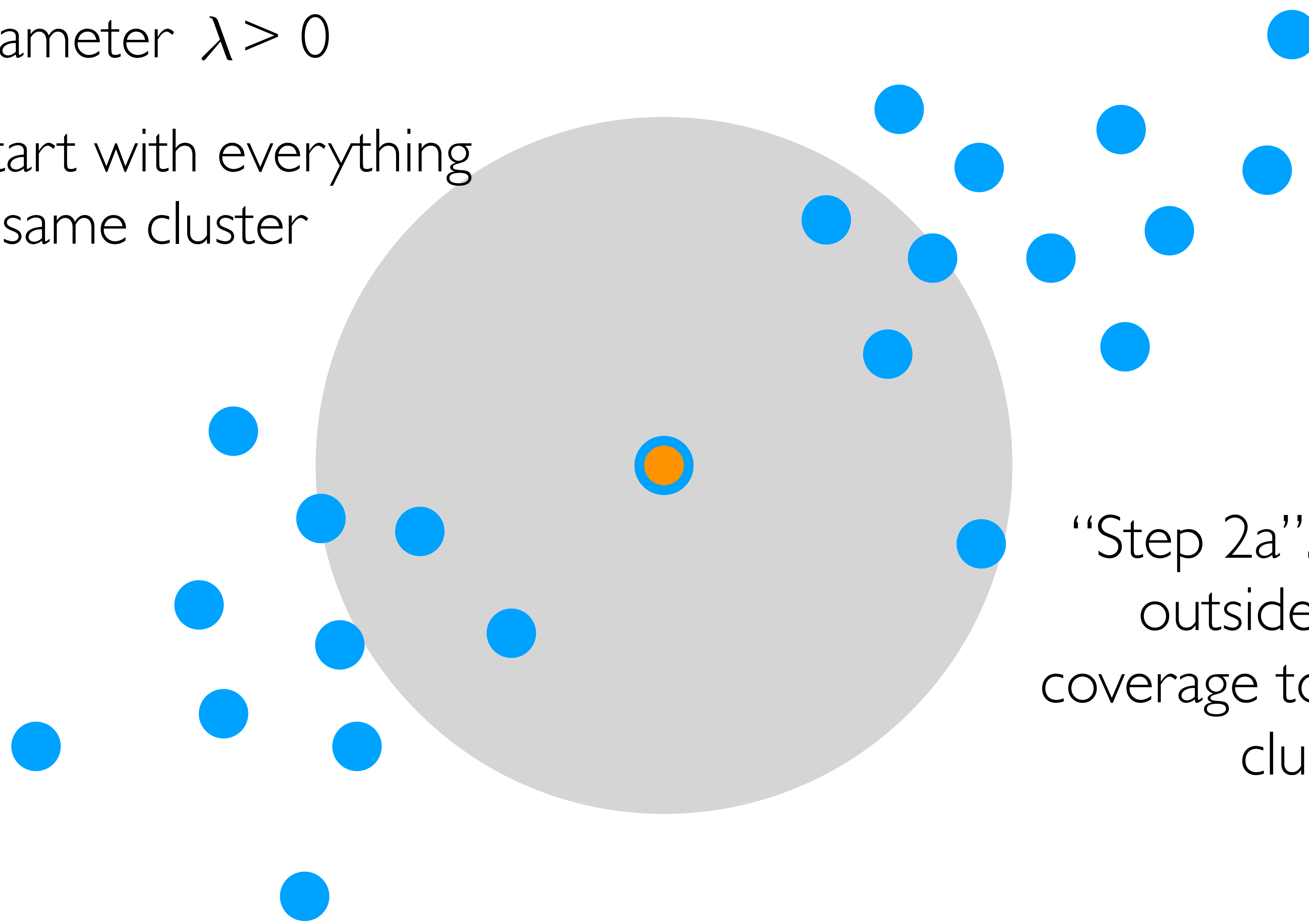
Step 1. Start with everything in same cluster



DP-means

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

Step 1. Start with everything in same cluster

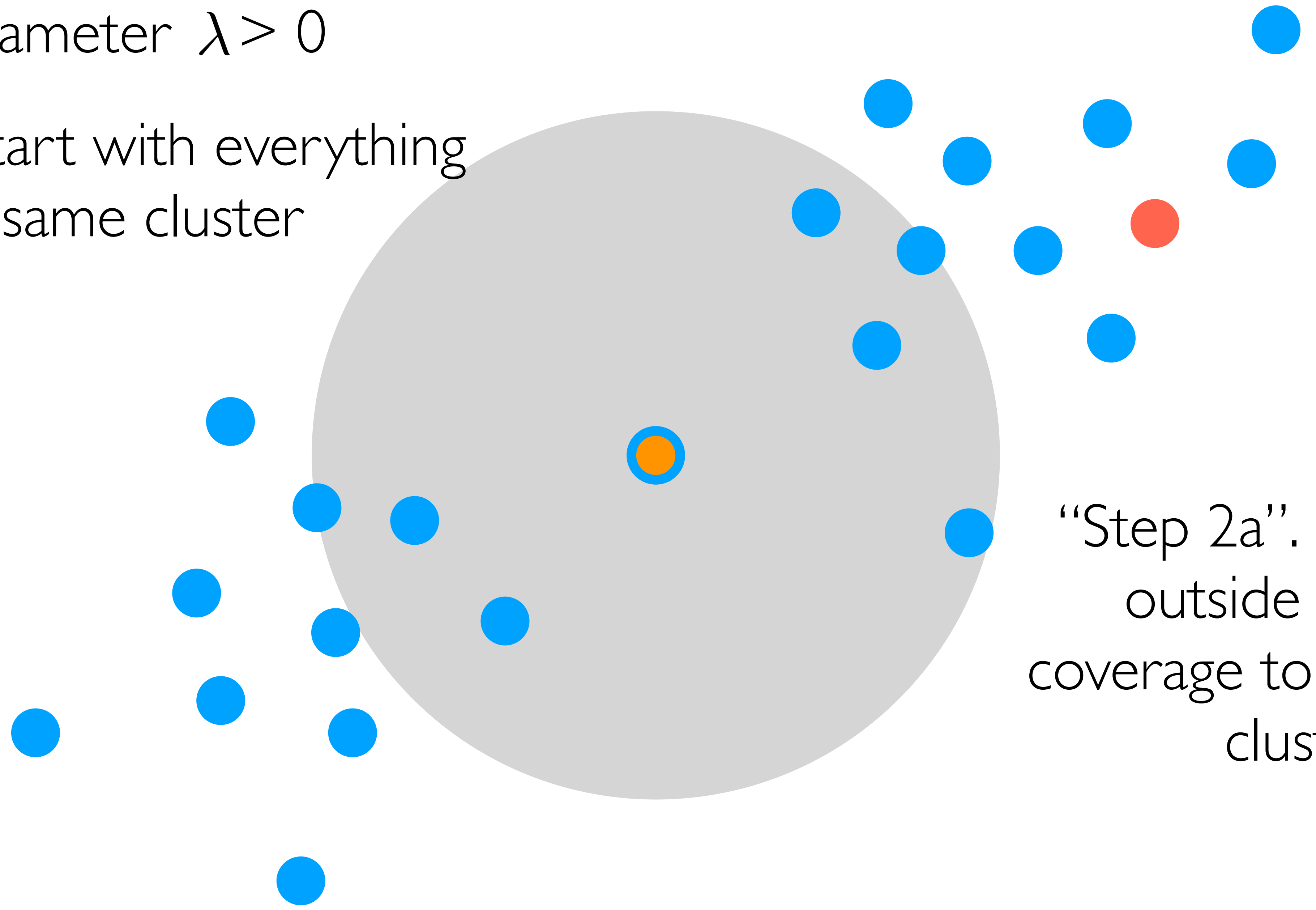


"Step 2a". Pick point outside of gray coverage to make new cluster

DP-means

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

Step 1. Start with everything in same cluster

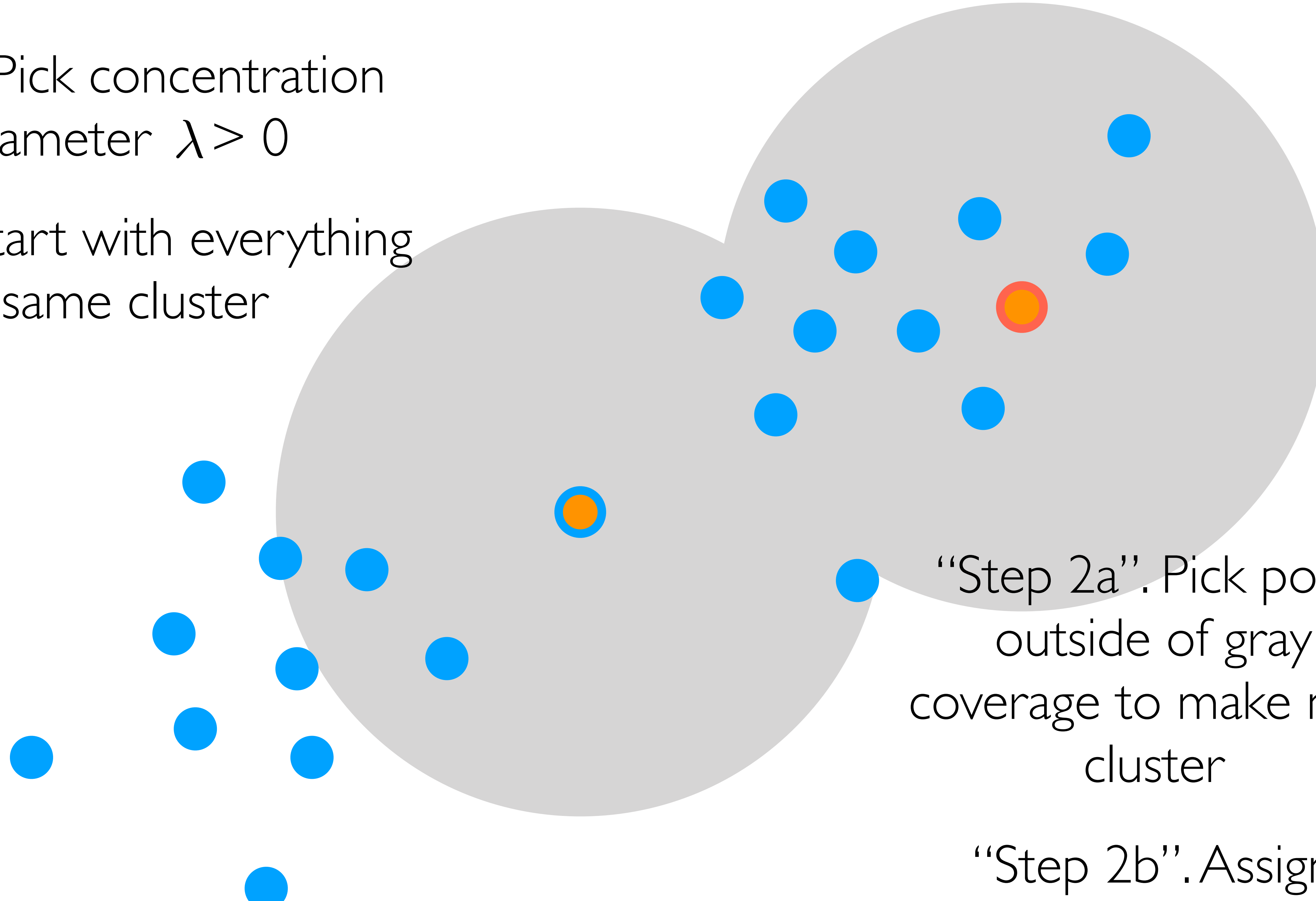


"Step 2a". Pick point outside of gray coverage to make new cluster

DP-means

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

Step 1. Start with everything in same cluster



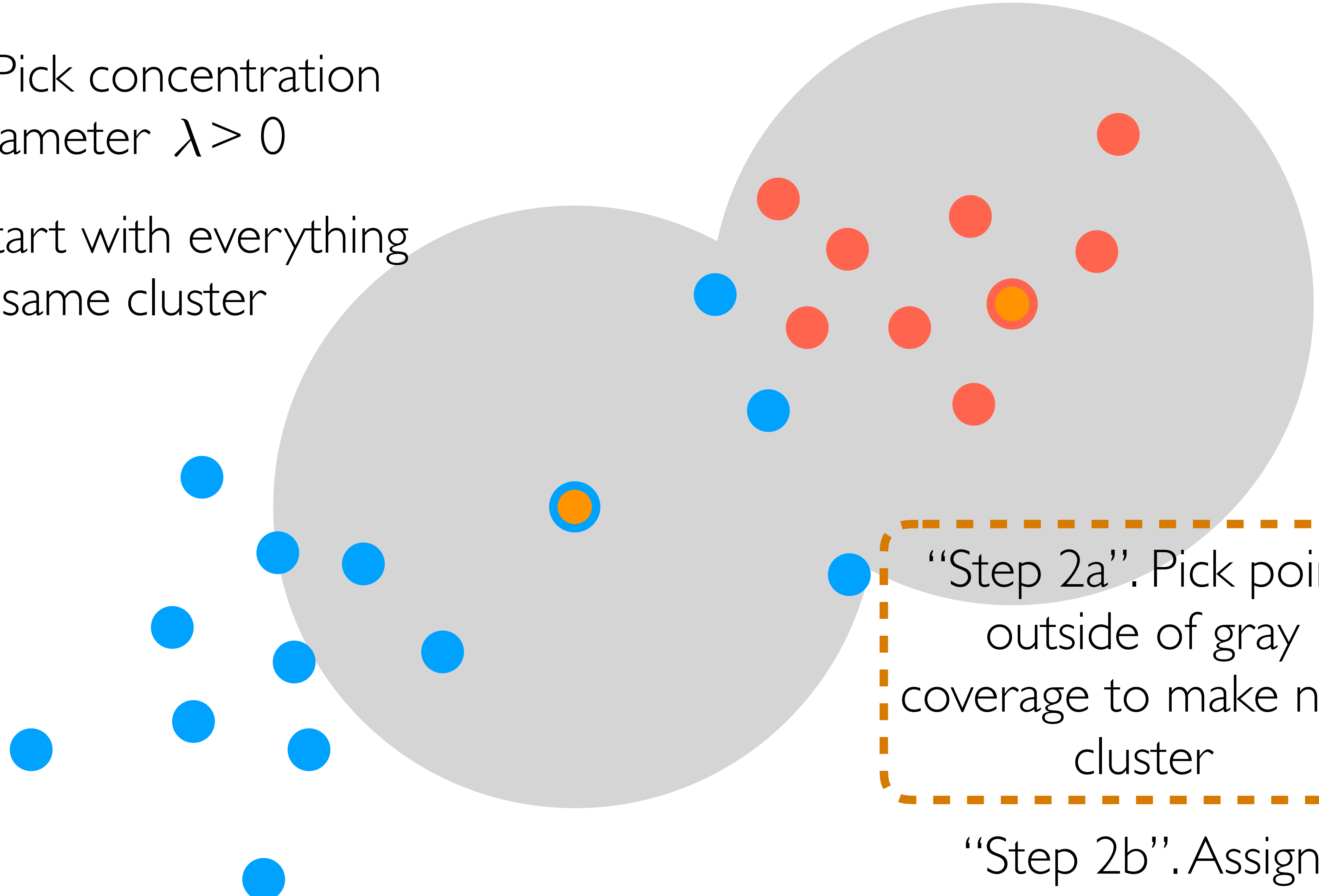
"Step 2a". Pick point outside of gray coverage to make new cluster

"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



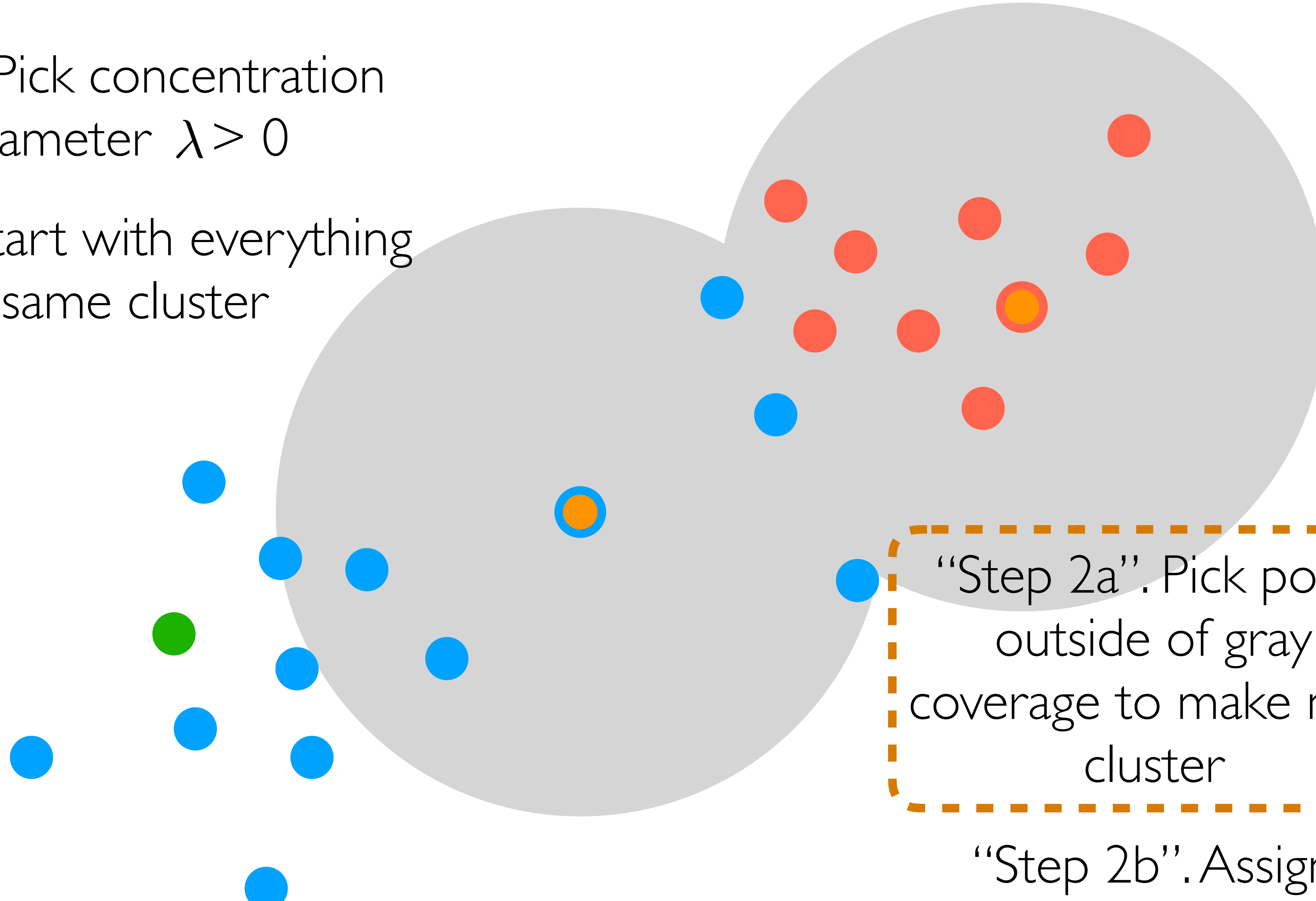
"Step 2a". Pick point outside of gray coverage to make new cluster

"Step 2b". Assign closest points to current clusters

DP-means

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

Step 1. Start with everything in same cluster



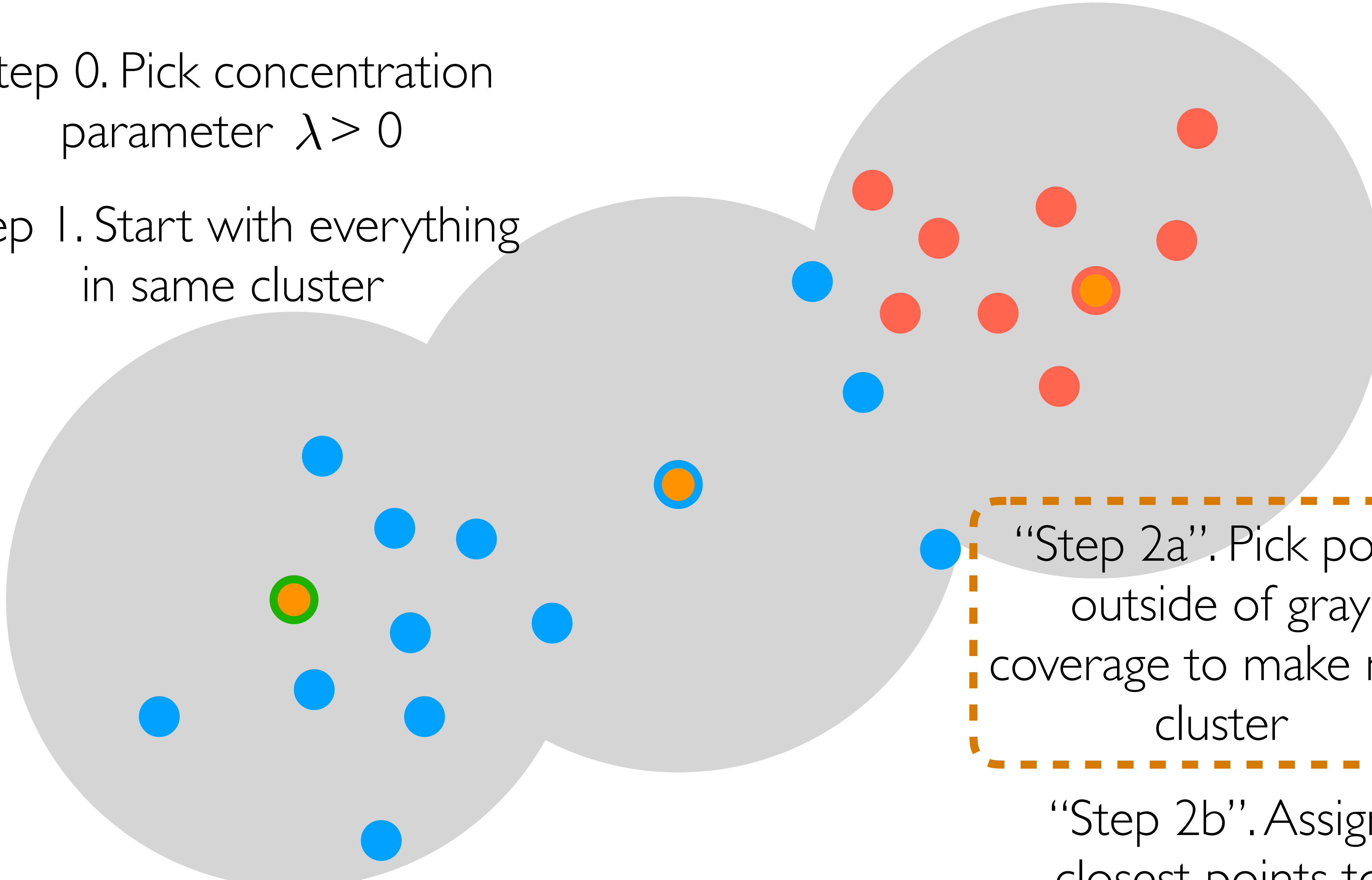
"Step 2a". Pick point outside of gray coverage to make new cluster

"Step 2b". Assign closest points to current clusters

DP-means

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

Step 1. Start with everything in same cluster



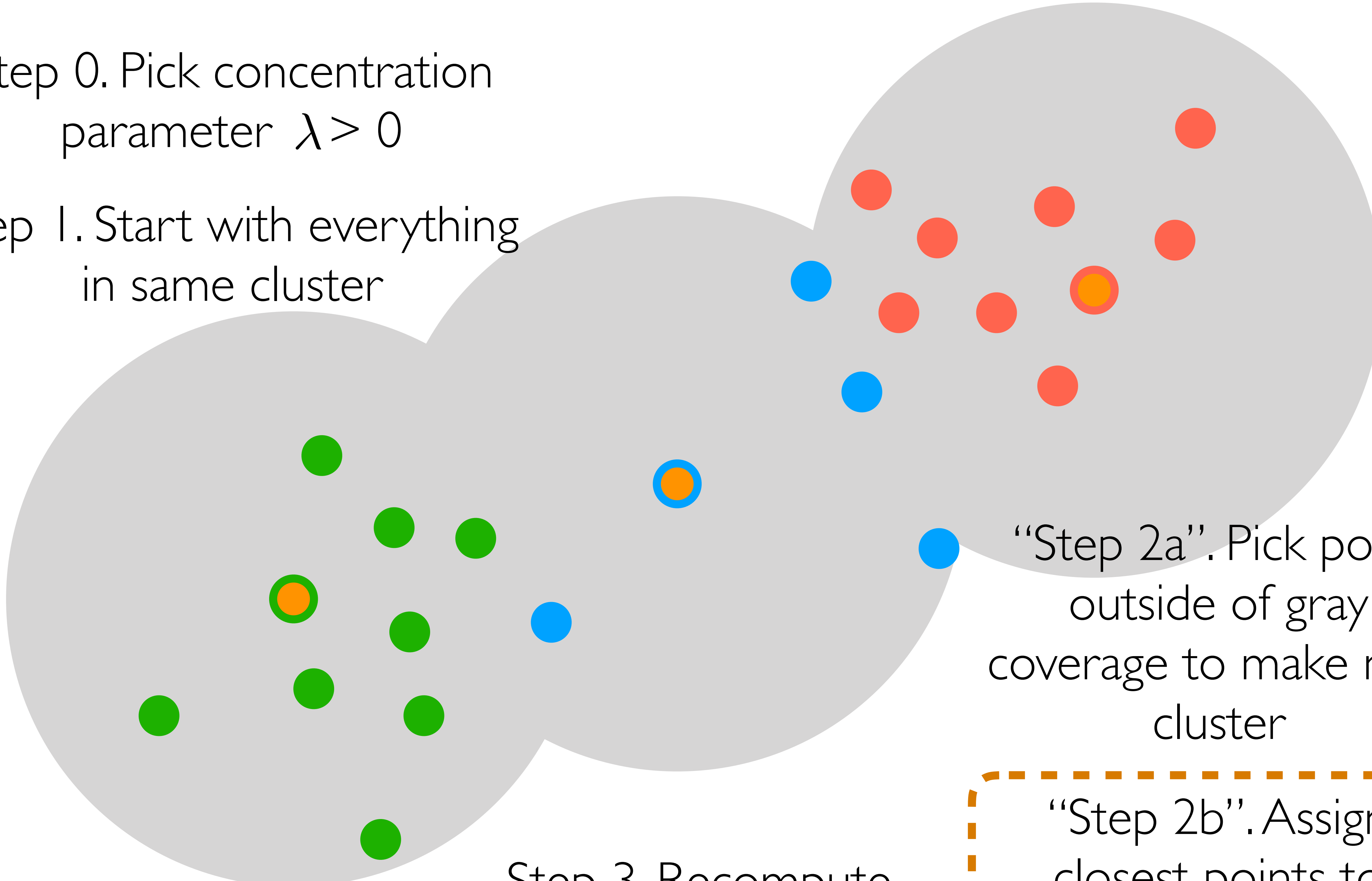
"Step 2a". Pick point outside of gray coverage to make new cluster

"Step 2b". Assign closest points to current clusters

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

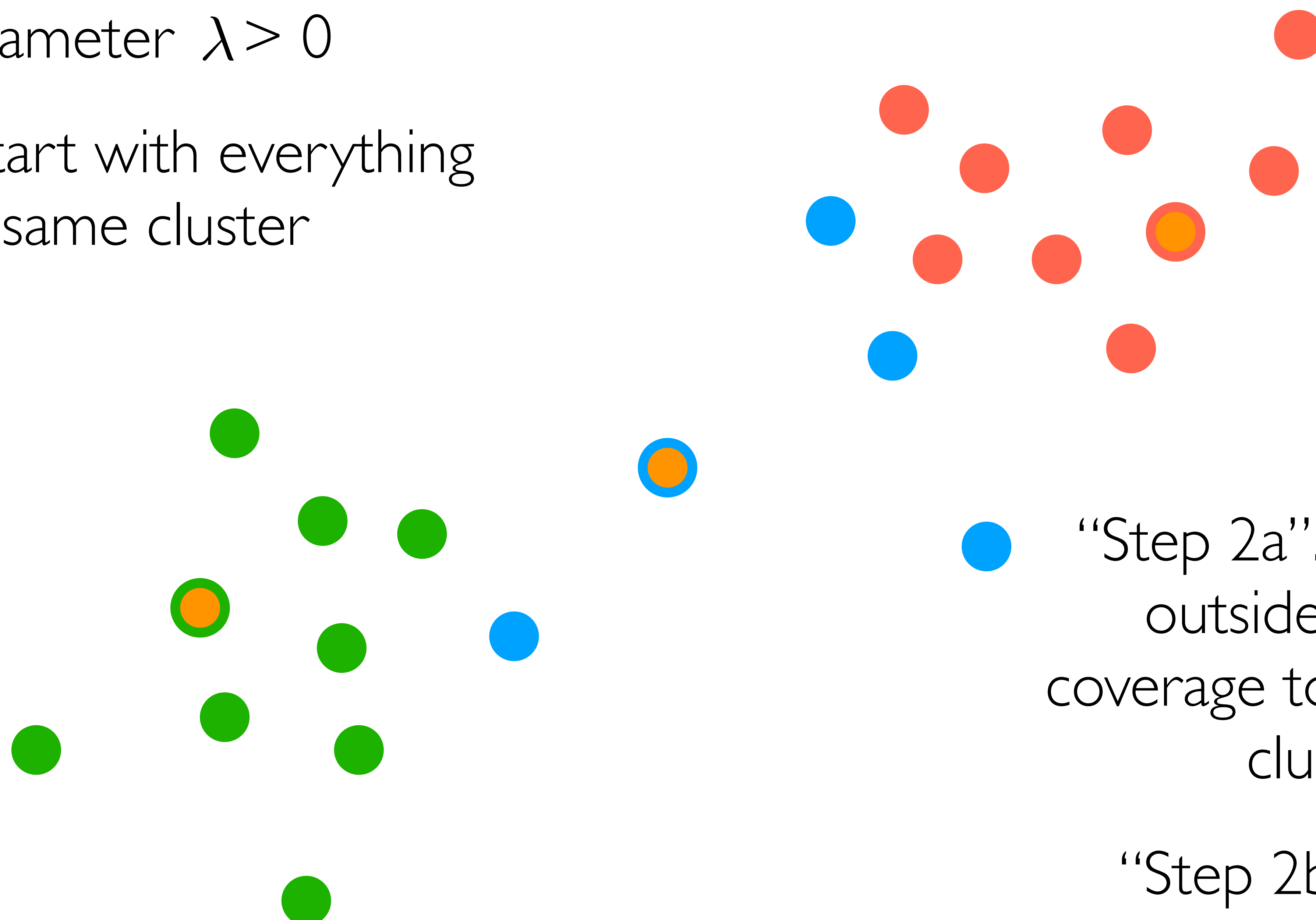
"Step 2b". Assign closest points to current clusters

Step 3. Recompute cluster centers

DP-means

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

Step 1. Start with everything in same cluster



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

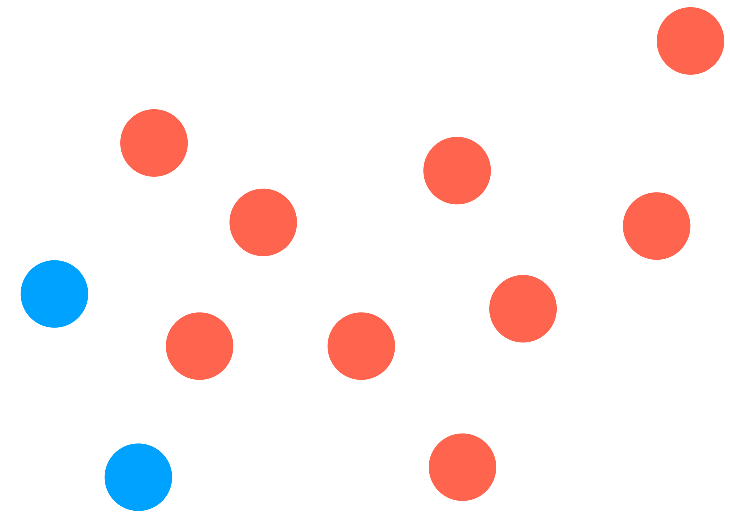
"Step 2b". Assign closest points to current clusters

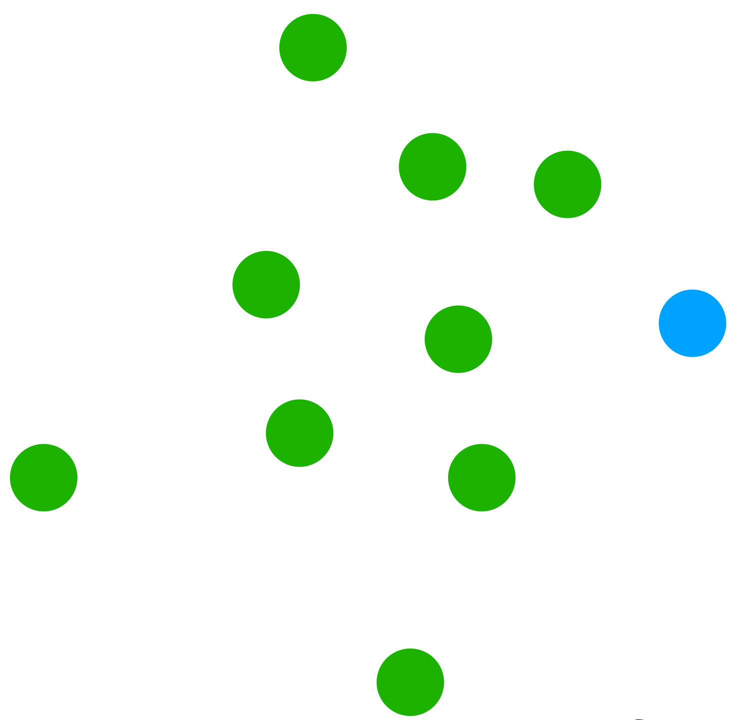
Step 3. Recompute cluster centers

DP-means

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

Step 1. Start with everything in same cluster





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

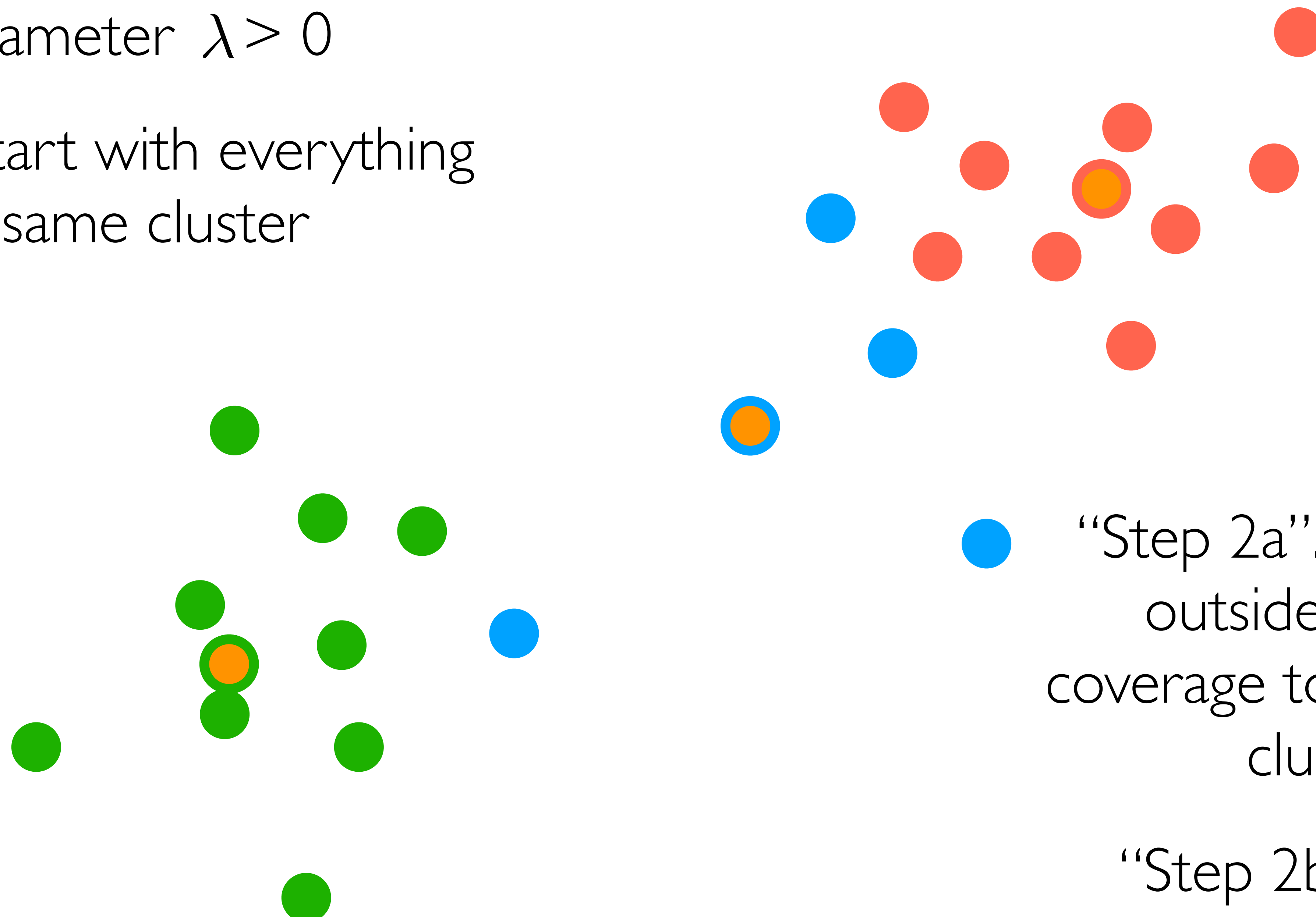
"Step 2b". Assign closest points to current clusters

Step 3. Recompute cluster centers

DP-means

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

Step 1. Start with everything in same cluster



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

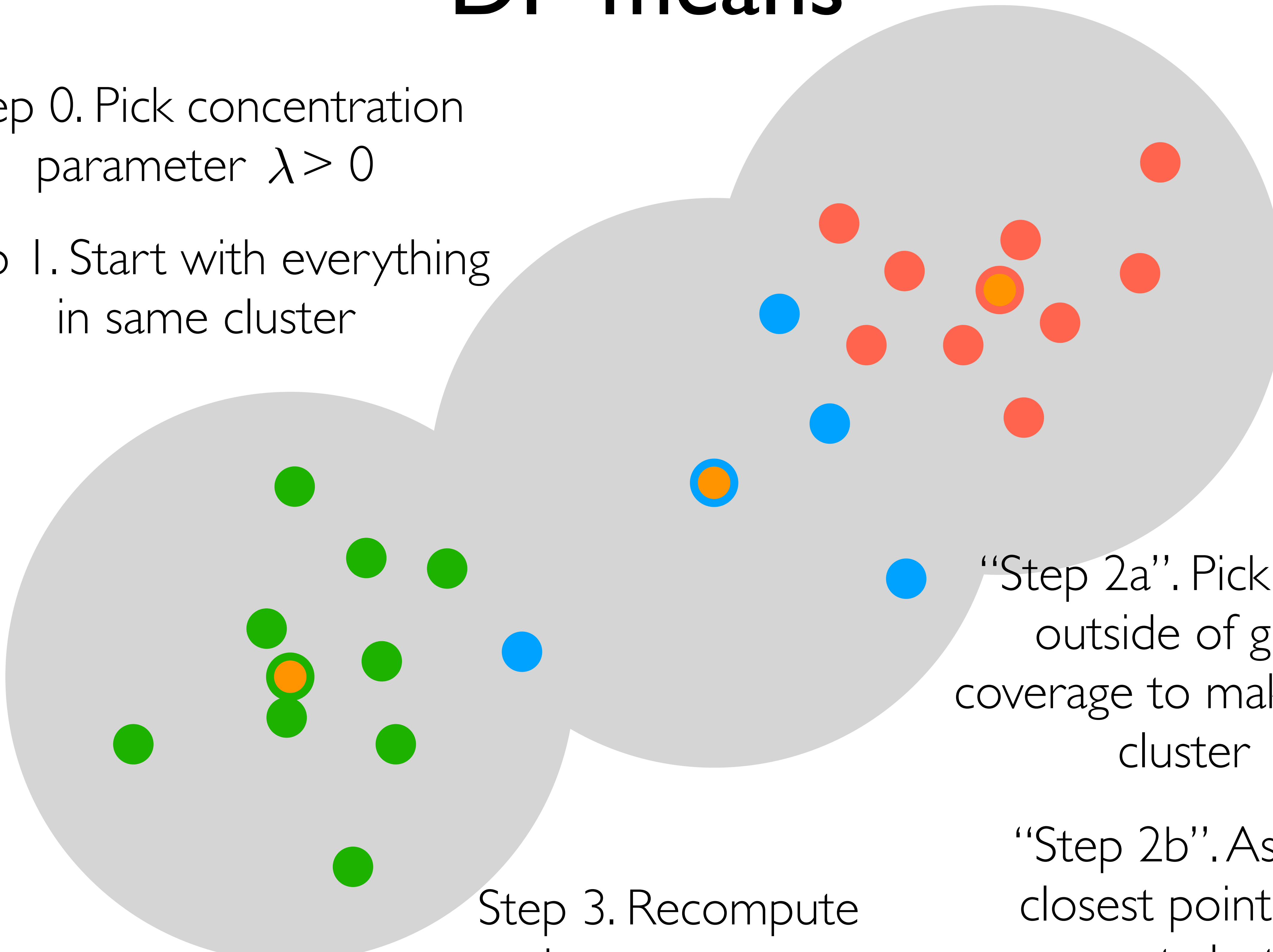
"Step 2b". Assign closest points to current clusters

Step 3. Recompute cluster centers

DP-means

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

Step 1. Start with everything in same cluster



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

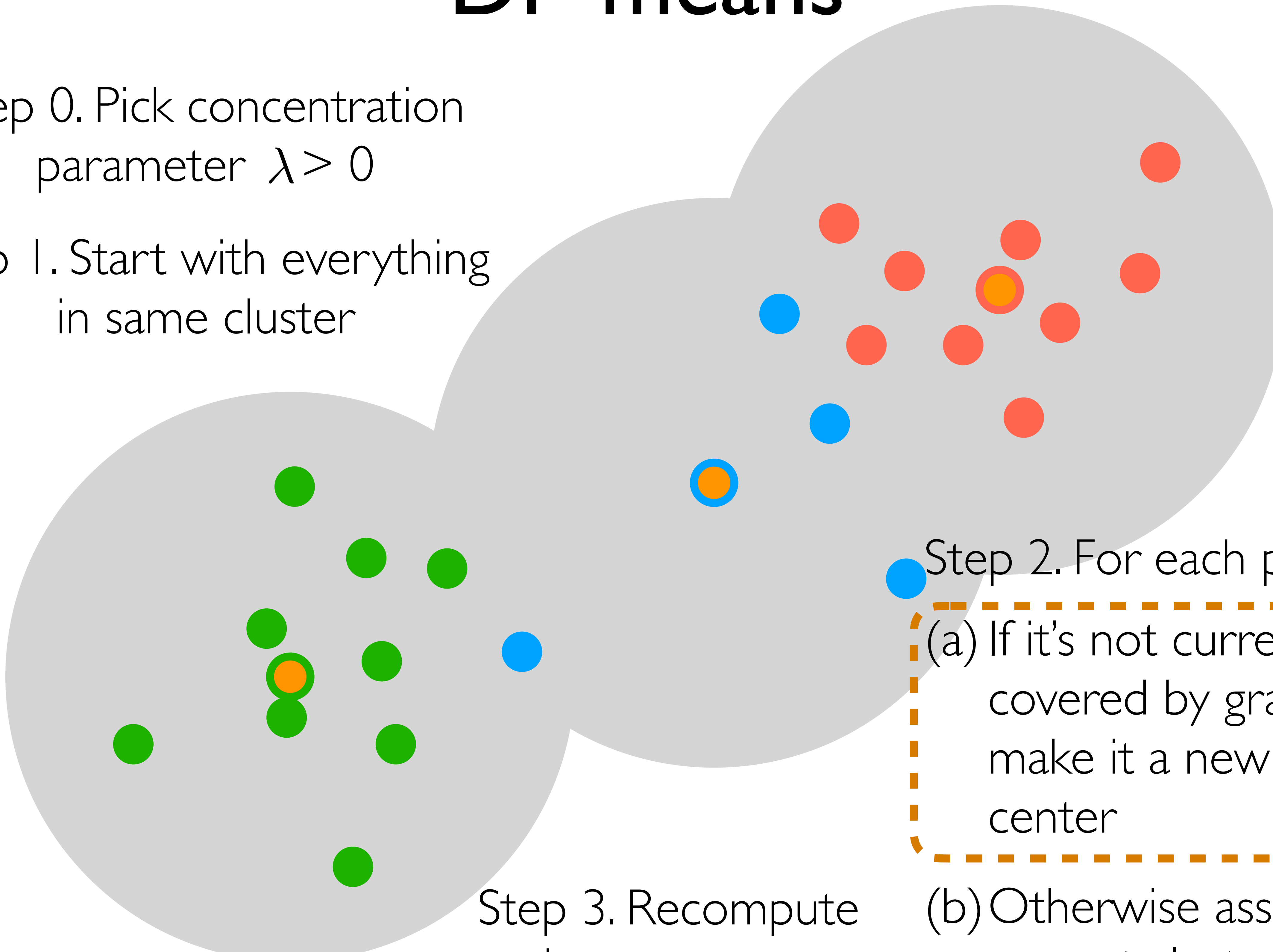
"Step 2b". Assign closest points to current clusters

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

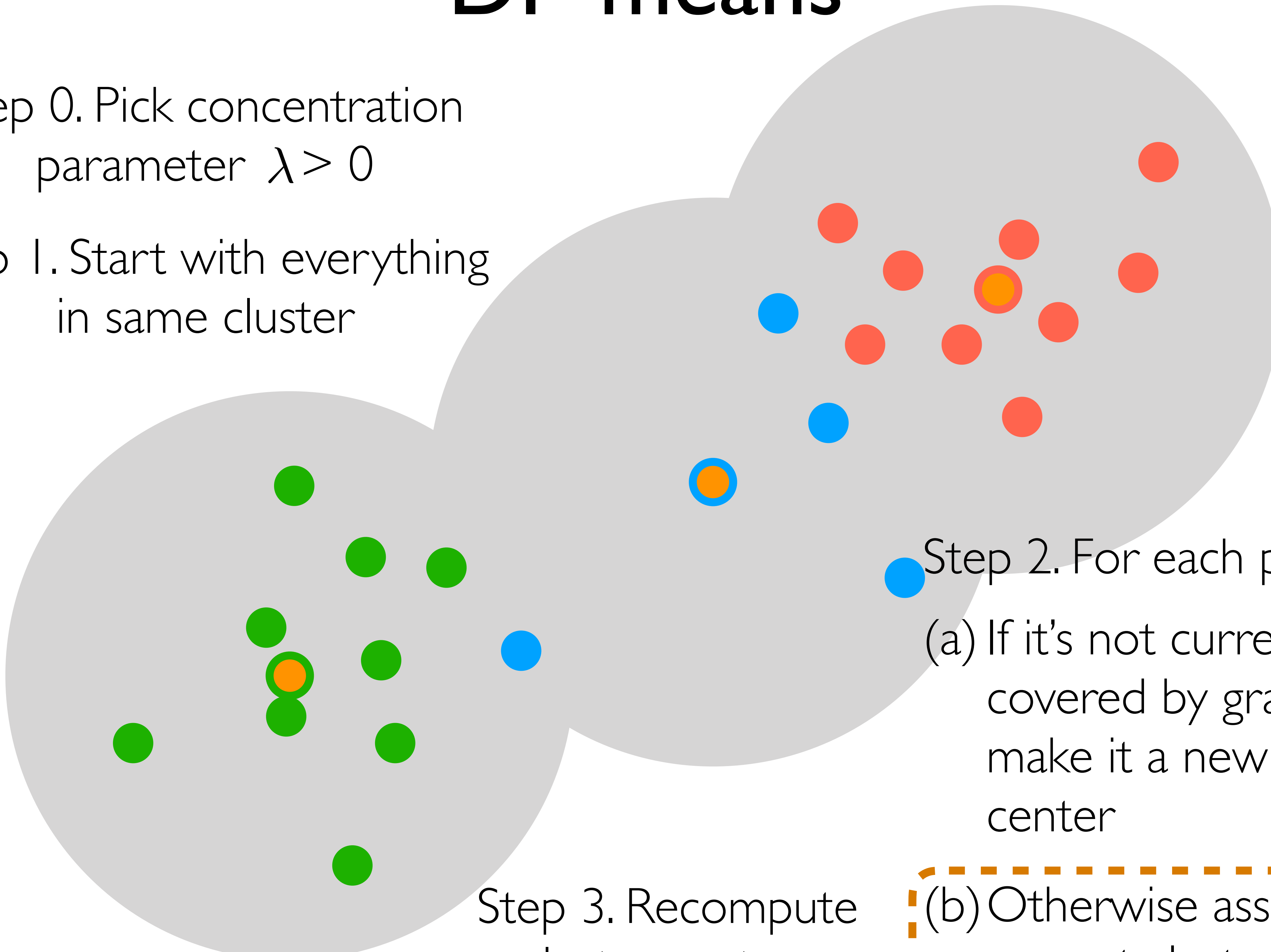
(b) Otherwise assign it to nearest cluster

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

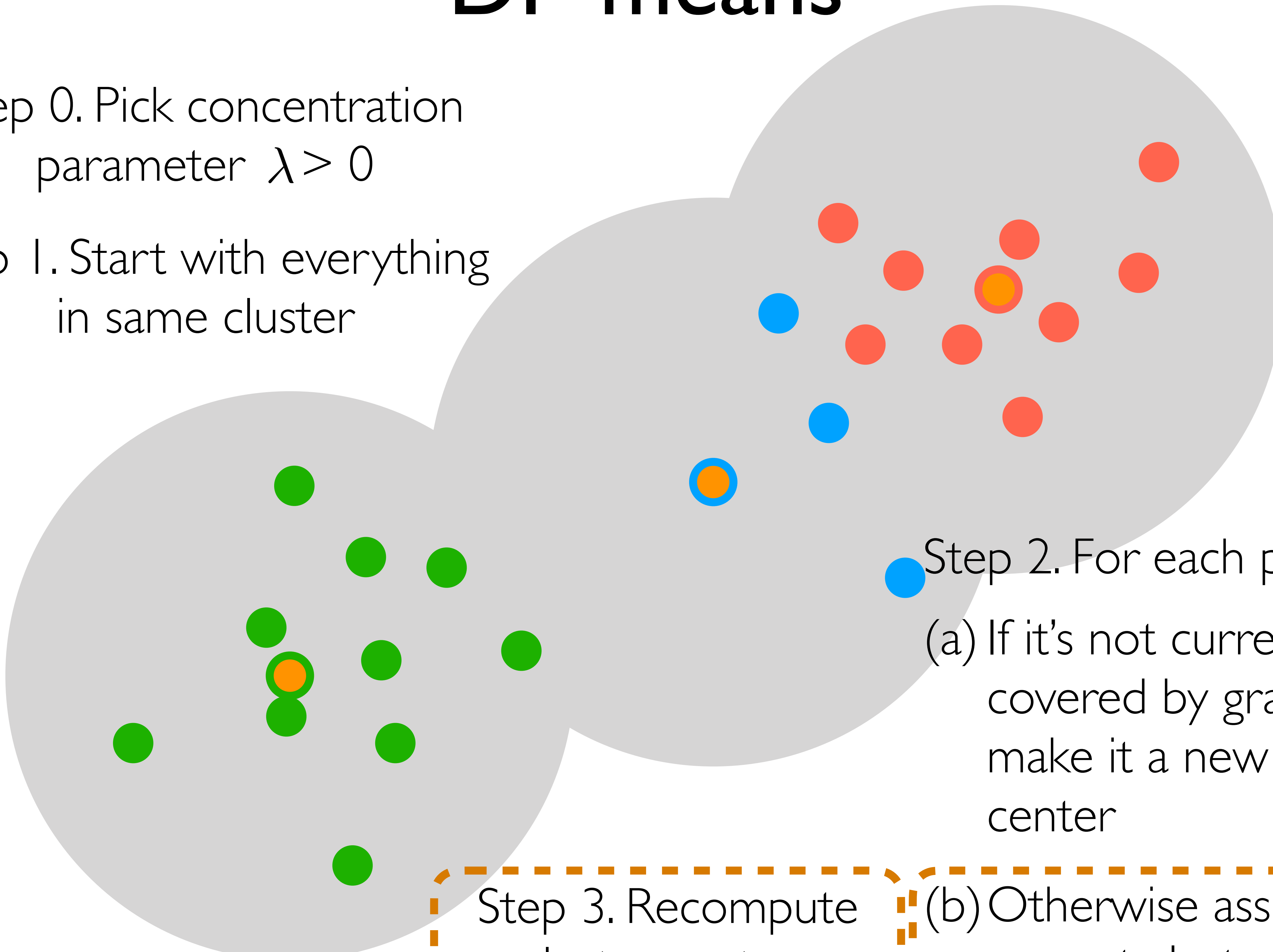
(b) Otherwise assign it to nearest cluster

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

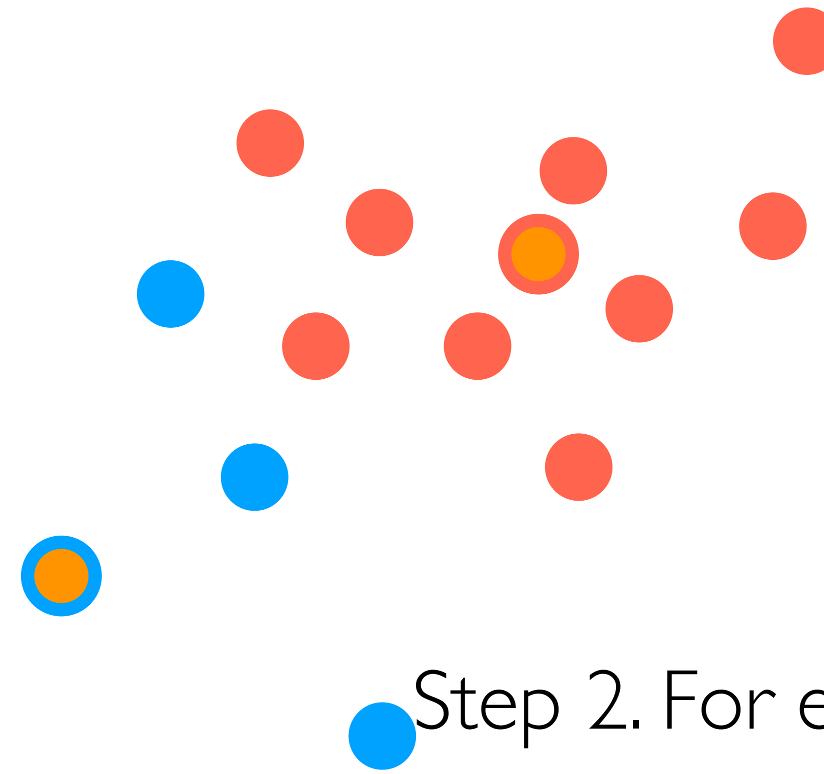
(b) Otherwise assign it to nearest cluster

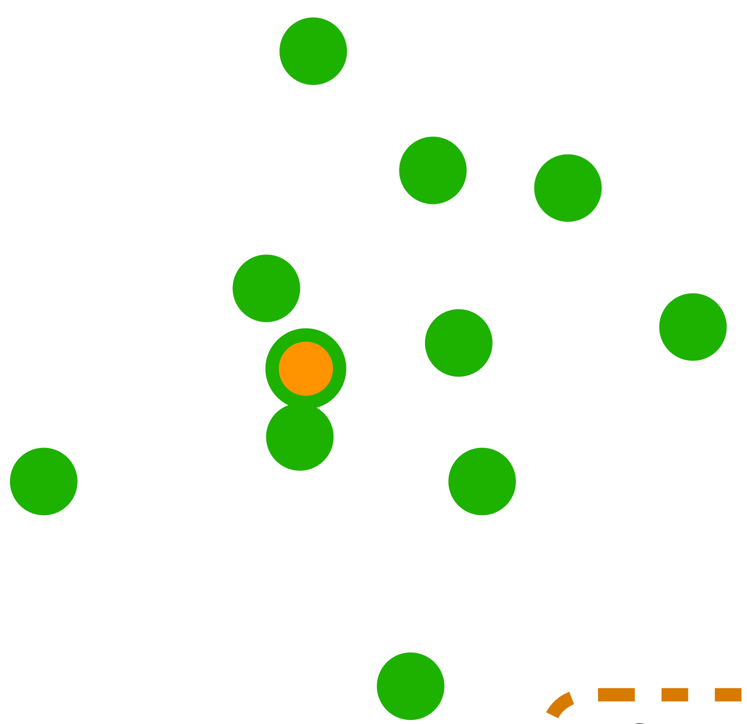
Step 3. Recompute cluster centers

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

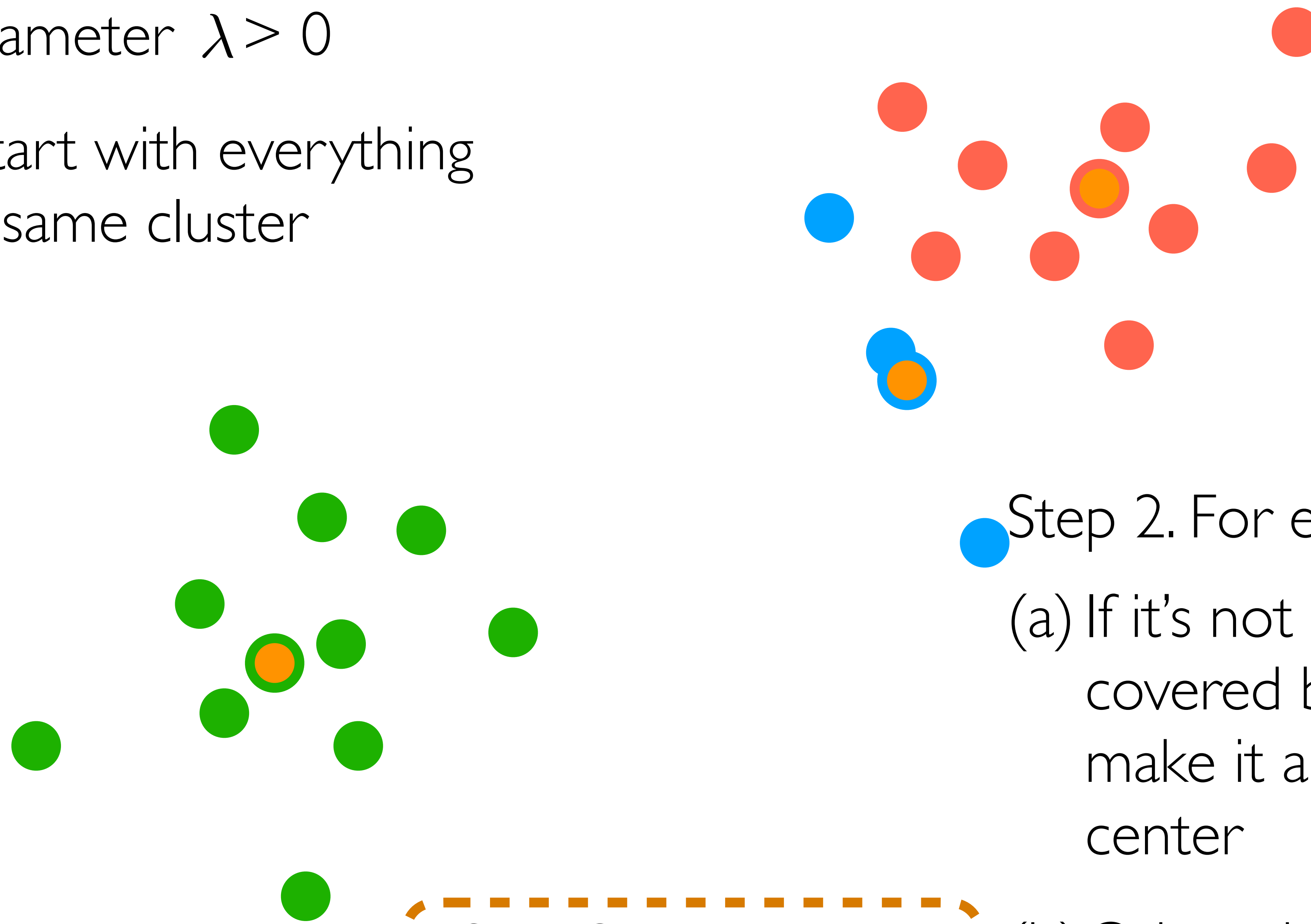
(b) Otherwise assign it to nearest cluster

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

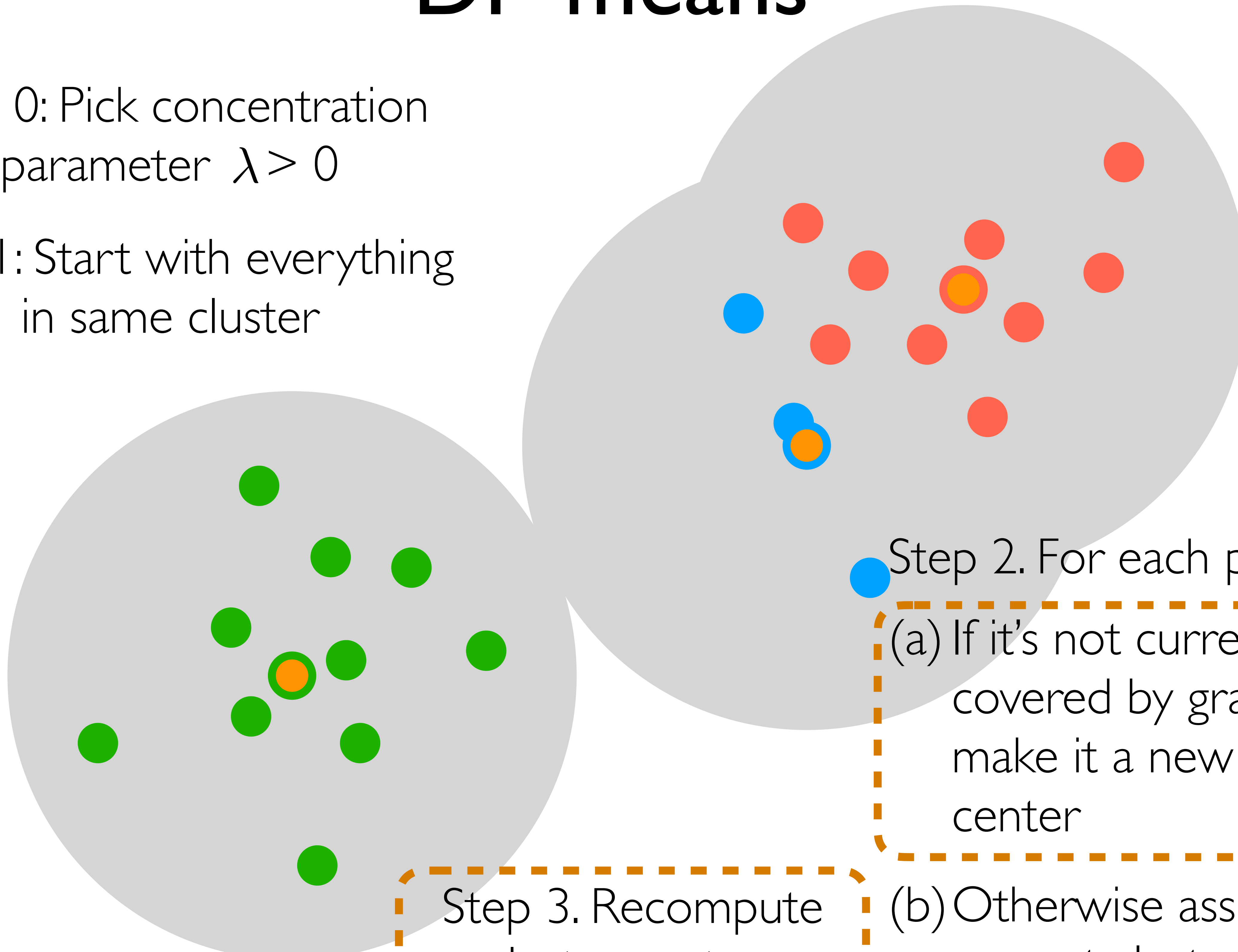
(b) Otherwise assign it to nearest cluster

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

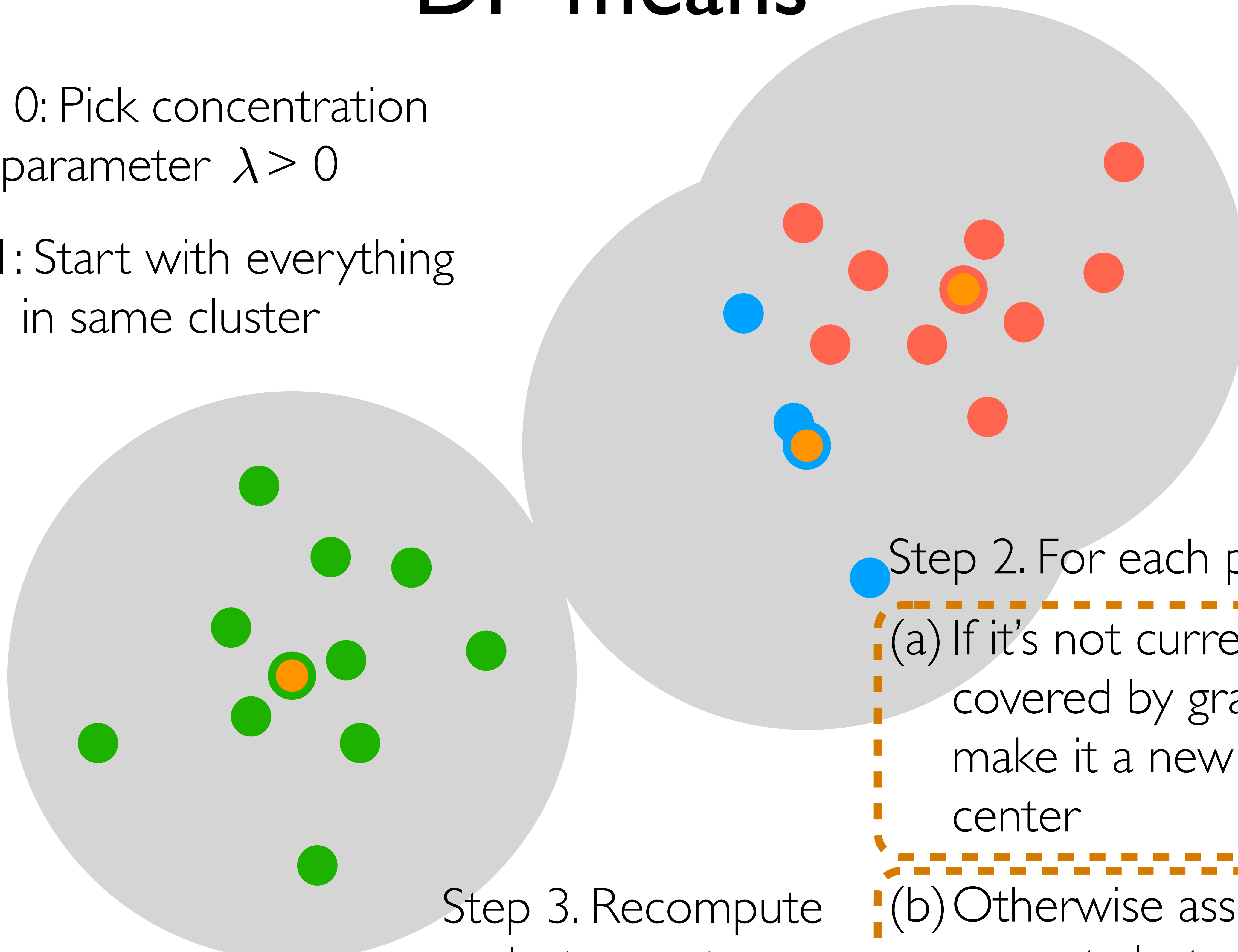
(b) Otherwise assign it to nearest cluster

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

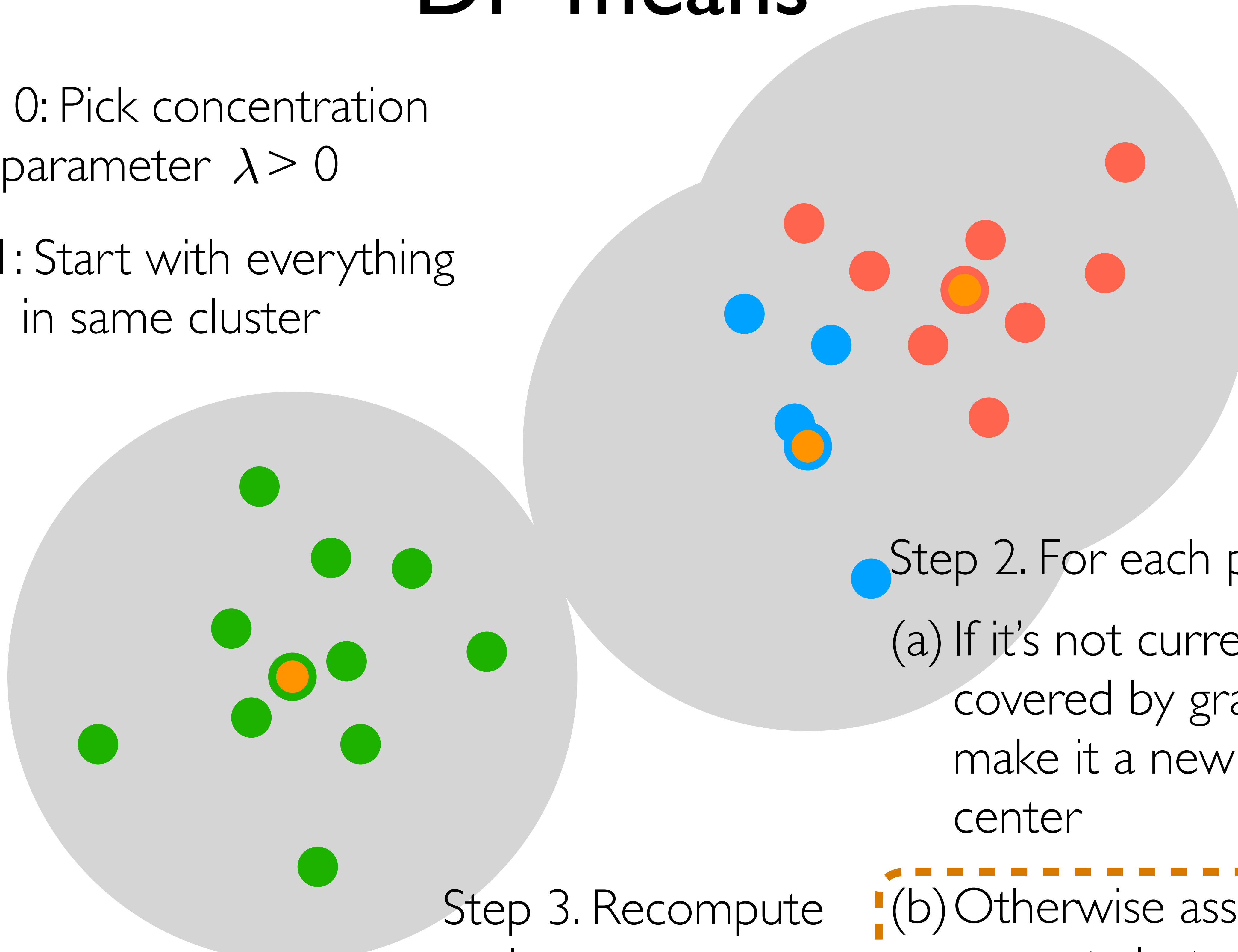
(b) Otherwise assign it to nearest cluster

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

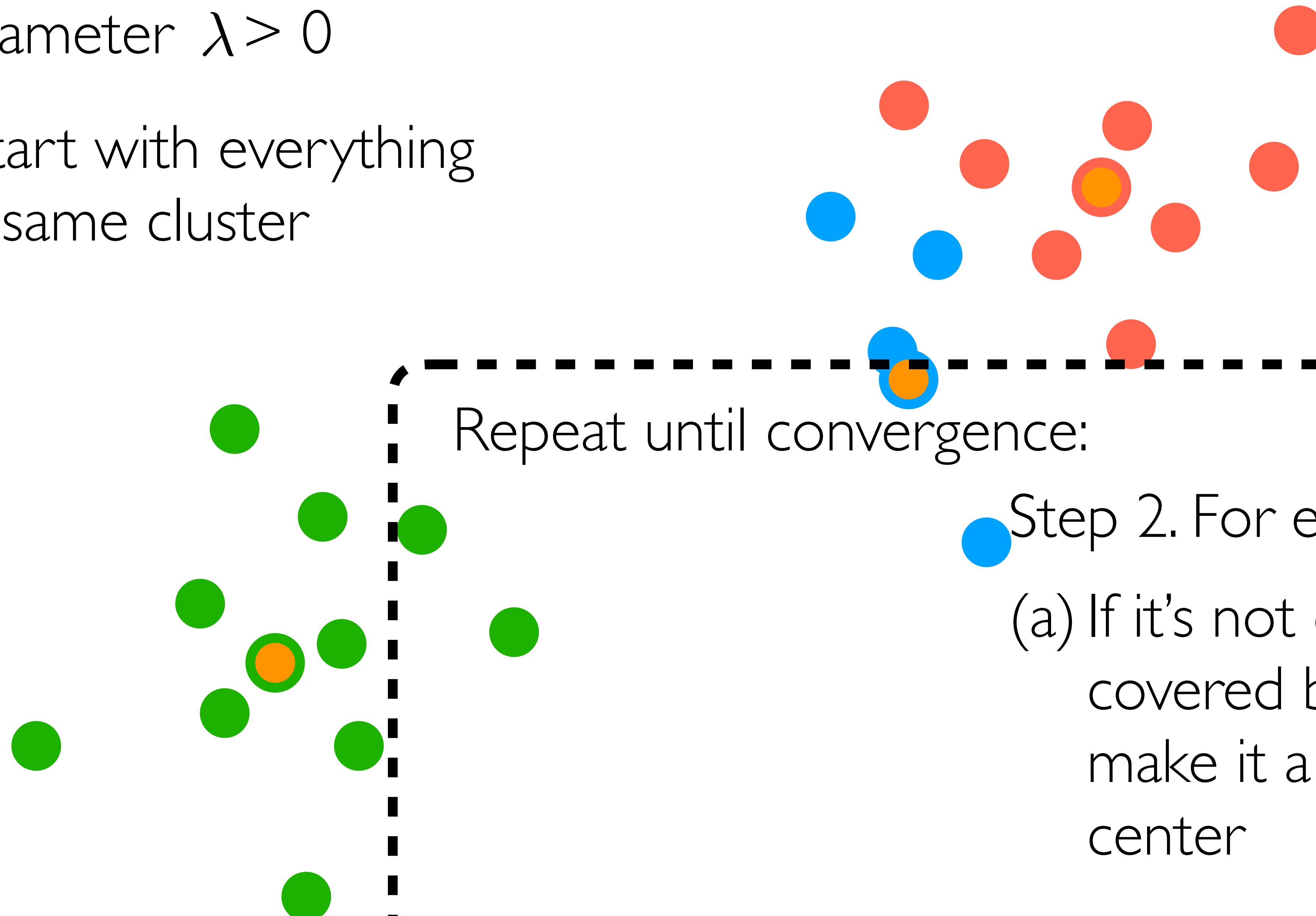
Repeat until convergence:

Step 2. For each point:

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

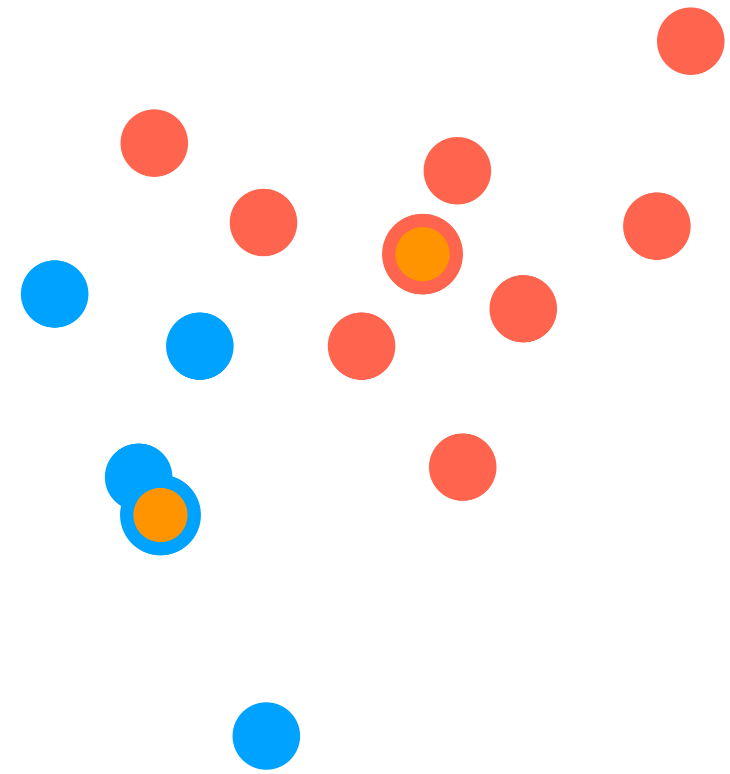
(b) Otherwise assign it to nearest cluster

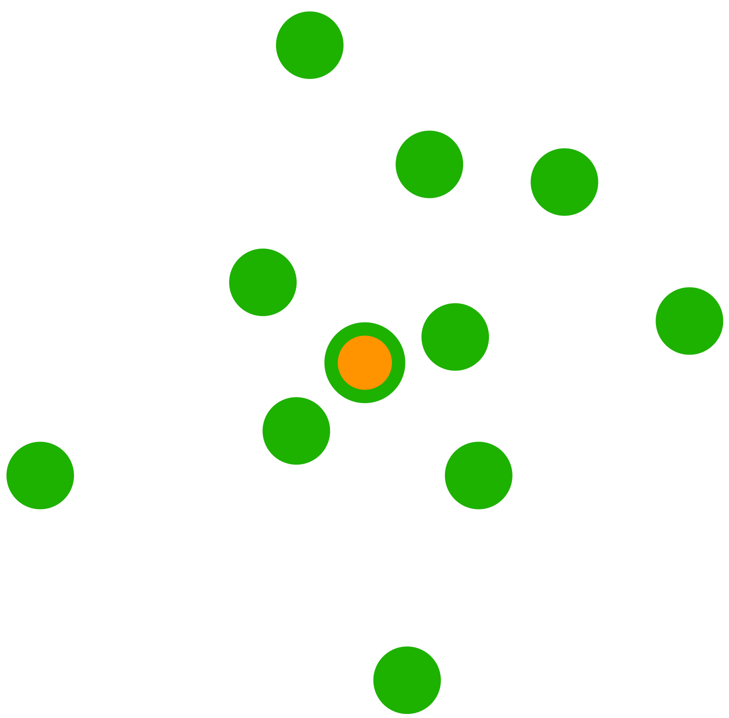
Step 3. Recompute cluster centers



DP-means

DP-means can produce a few extra small clusters

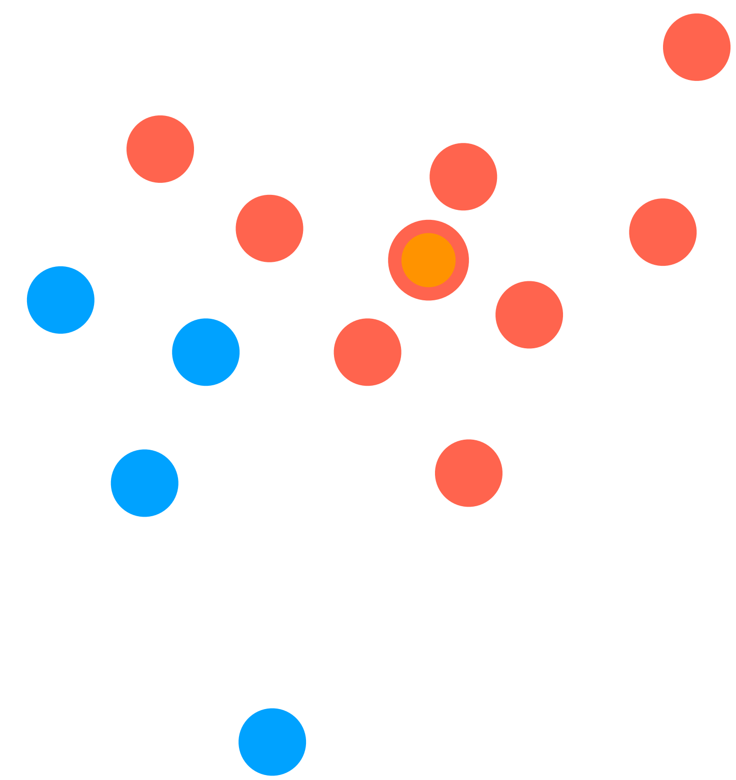


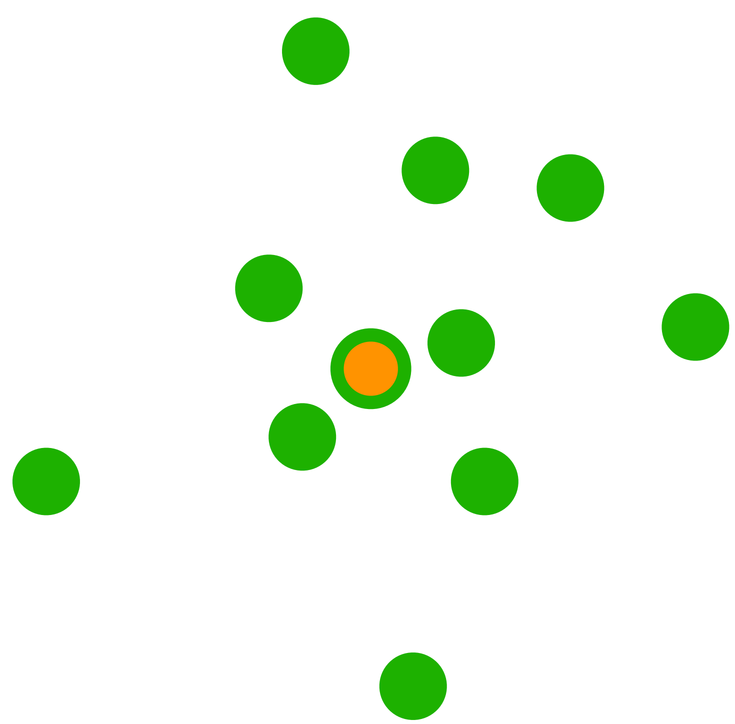


In practice: can reassign points in small clusters to bigger clusters

DP-means

DP-means can produce a few extra small clusters

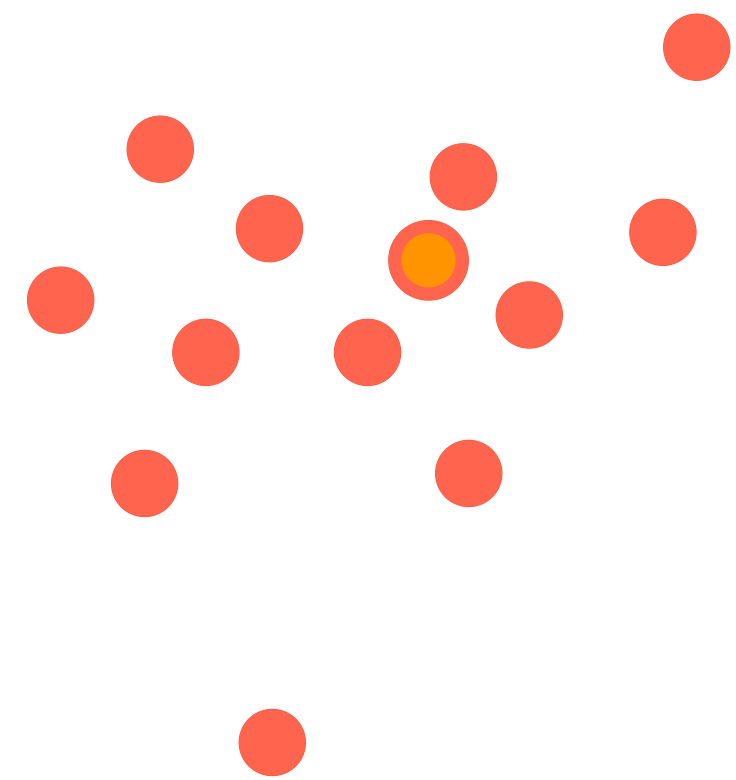


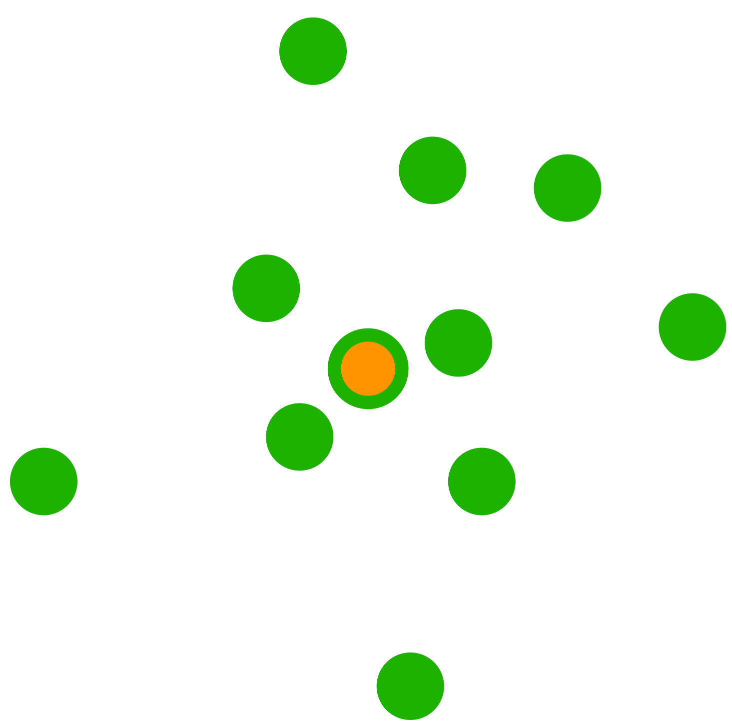


In practice: can reassign points in small clusters to bigger clusters

DP-means

DP-means can produce a few extra
small clusters



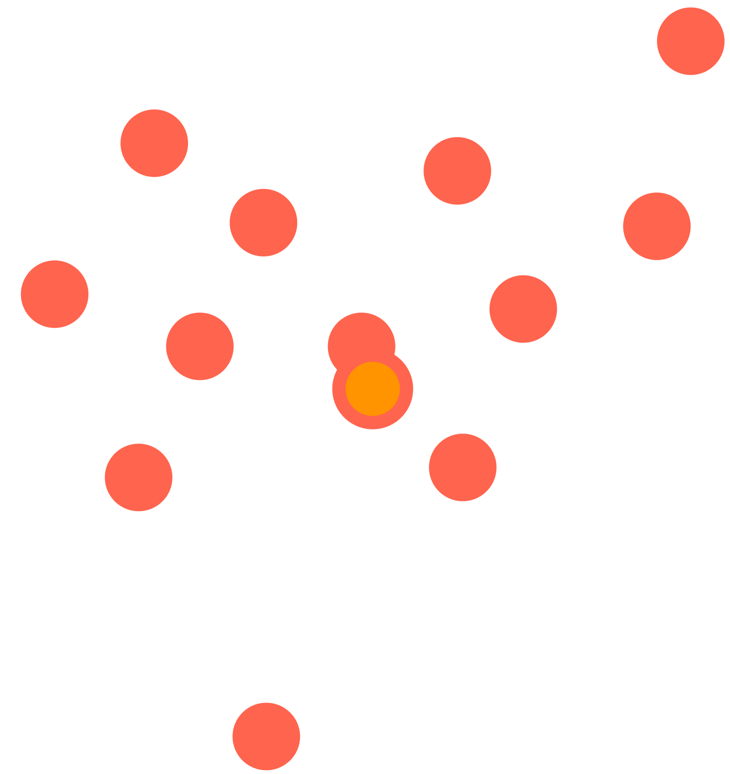


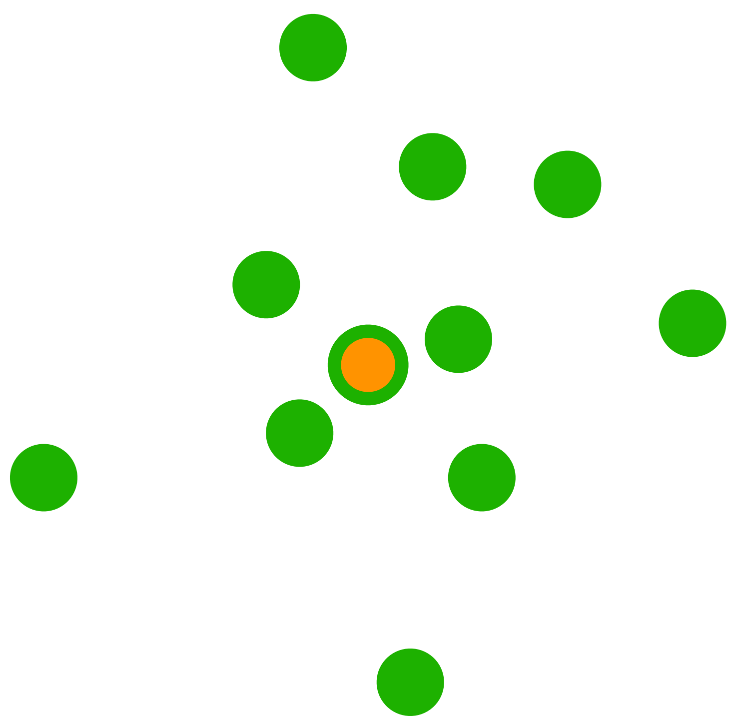
In practice: can reassign points in small
clusters to bigger clusters

Can recompute cluster centers

DP-means

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

- Big picture: DP-means has a parameter controlling the size (radius) of clusters rather than number of clusters
- If your problem can more naturally be phrased as having cluster sizes that should not be too large, can use DP-means instead of k-means

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!

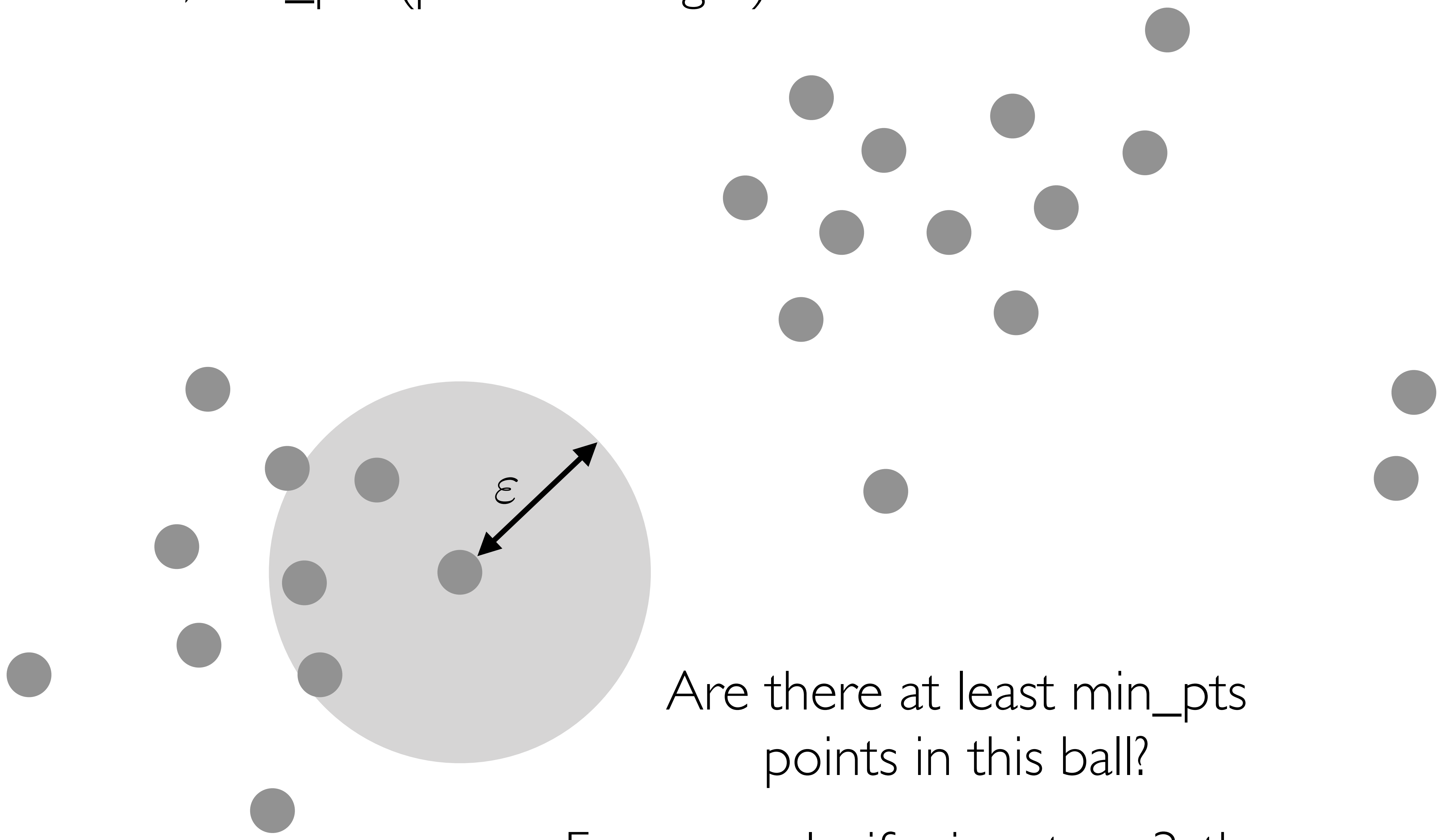
Technical remark: k-means approximates learning a GMM,
DP-means approximates learning what's called a *Dirichlet-Process GMM*

Density-based Clustering

DBSCAN

Let's choose min_pts = 3

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



Are there at least min_pts points in this ball?

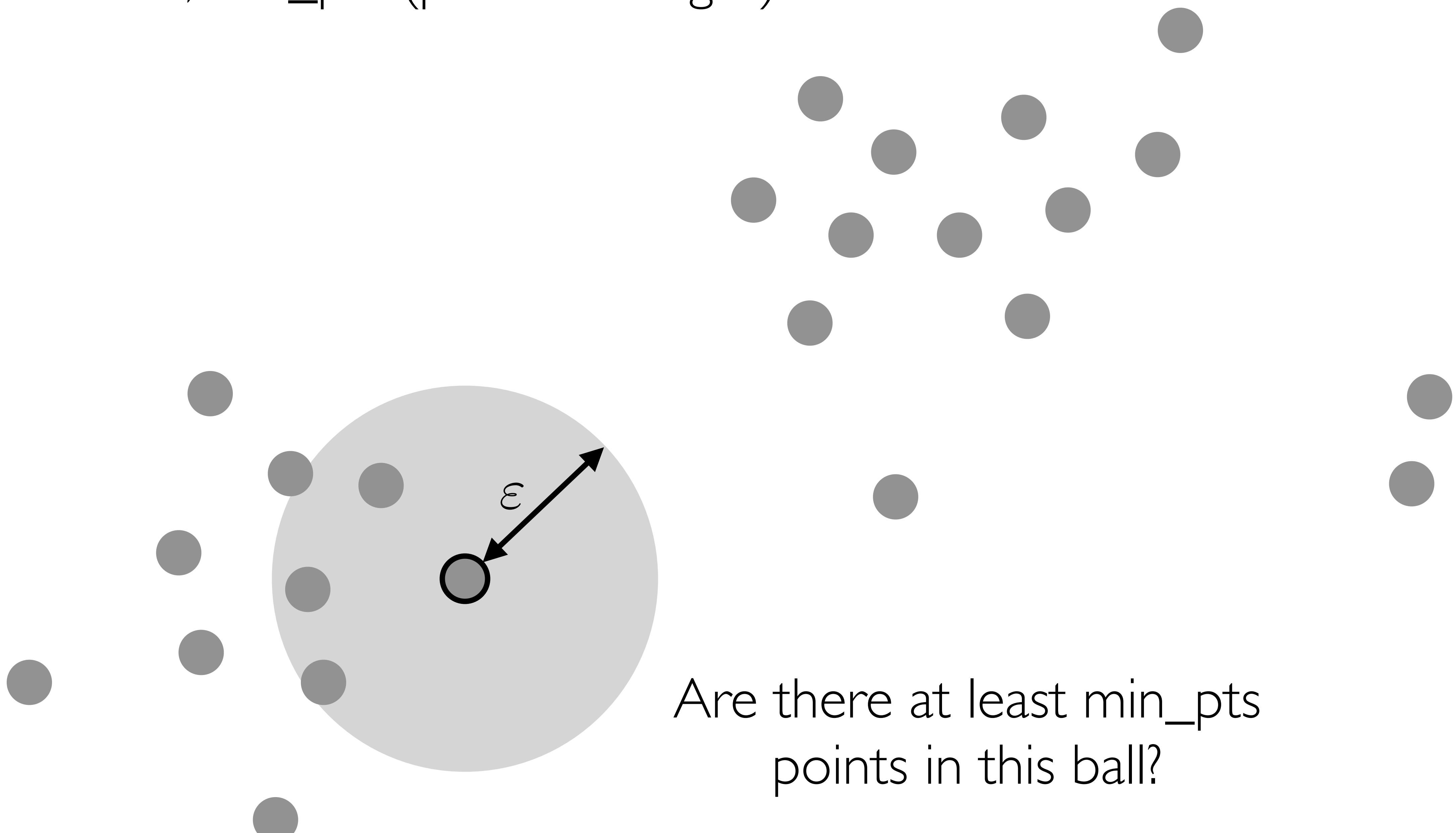
For example, if min_pts = 3, then yes

For example, if min_pts = 10, then no

DBSCAN

Let's choose min_pts = 3

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



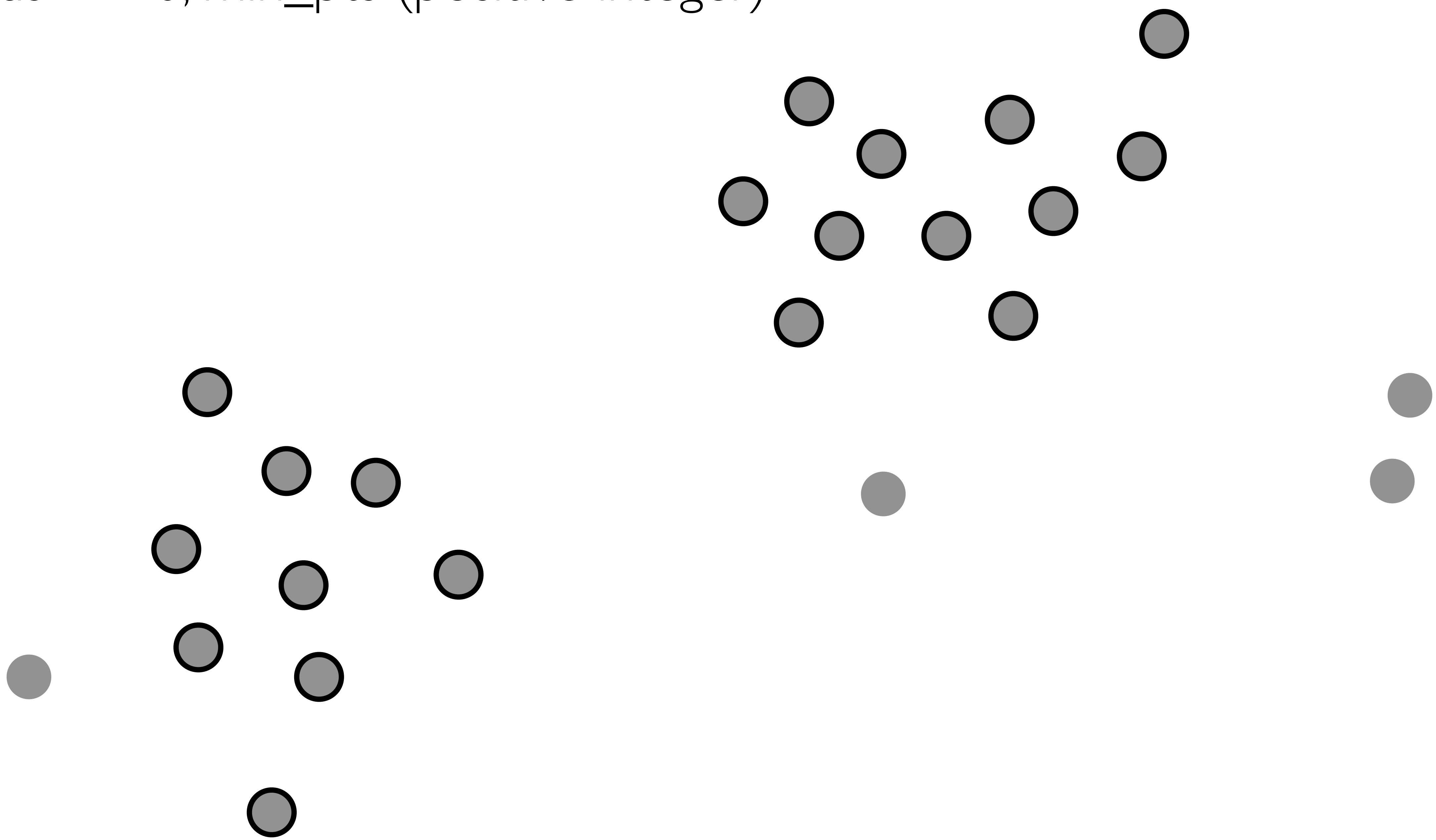
Are there at least min_pts points in this ball?

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

DBSCAN

Let's choose min_pts = 3

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

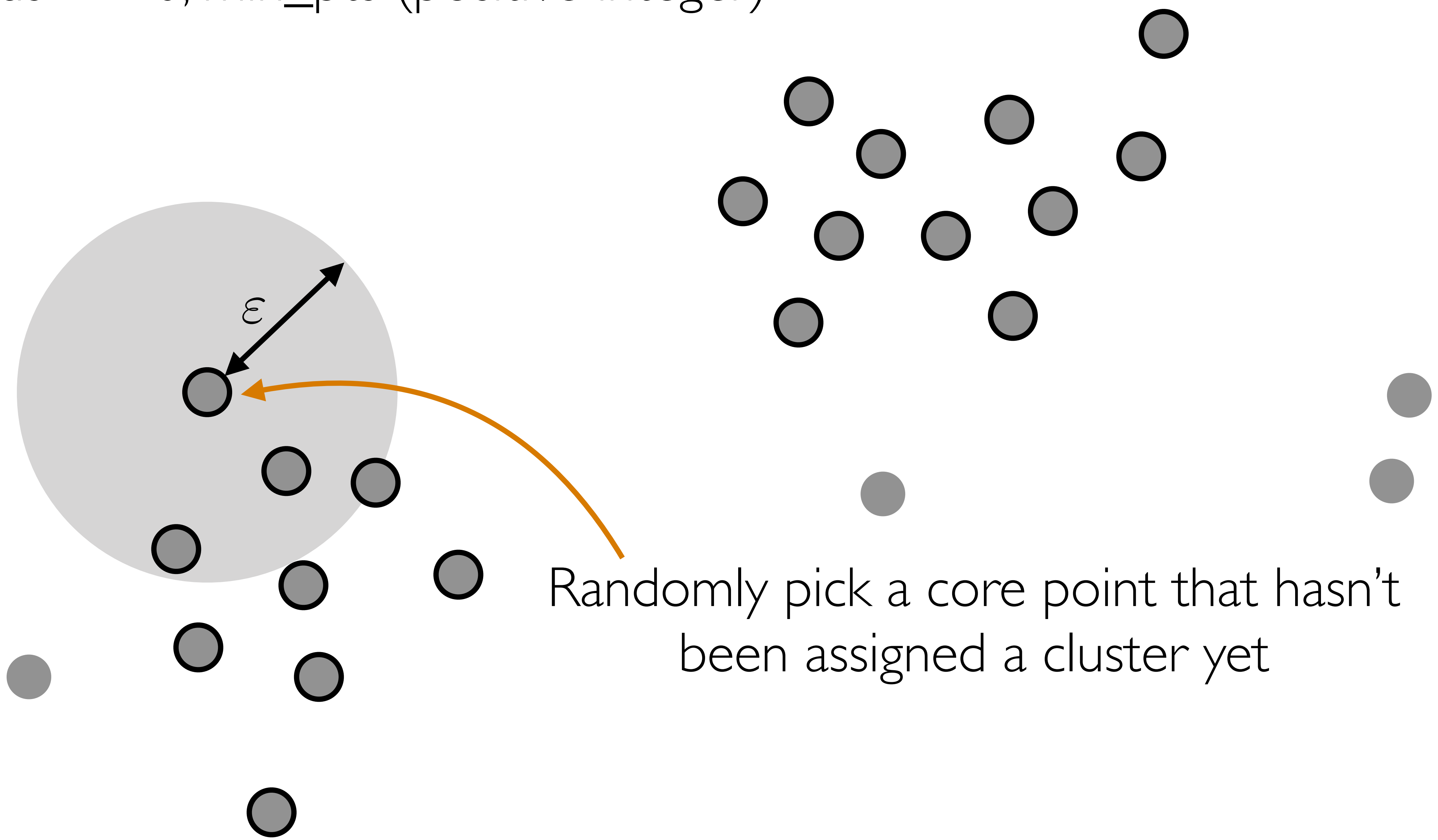


Core points (**outlined**)

DBSCAN

Let's choose min_pts = 3

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



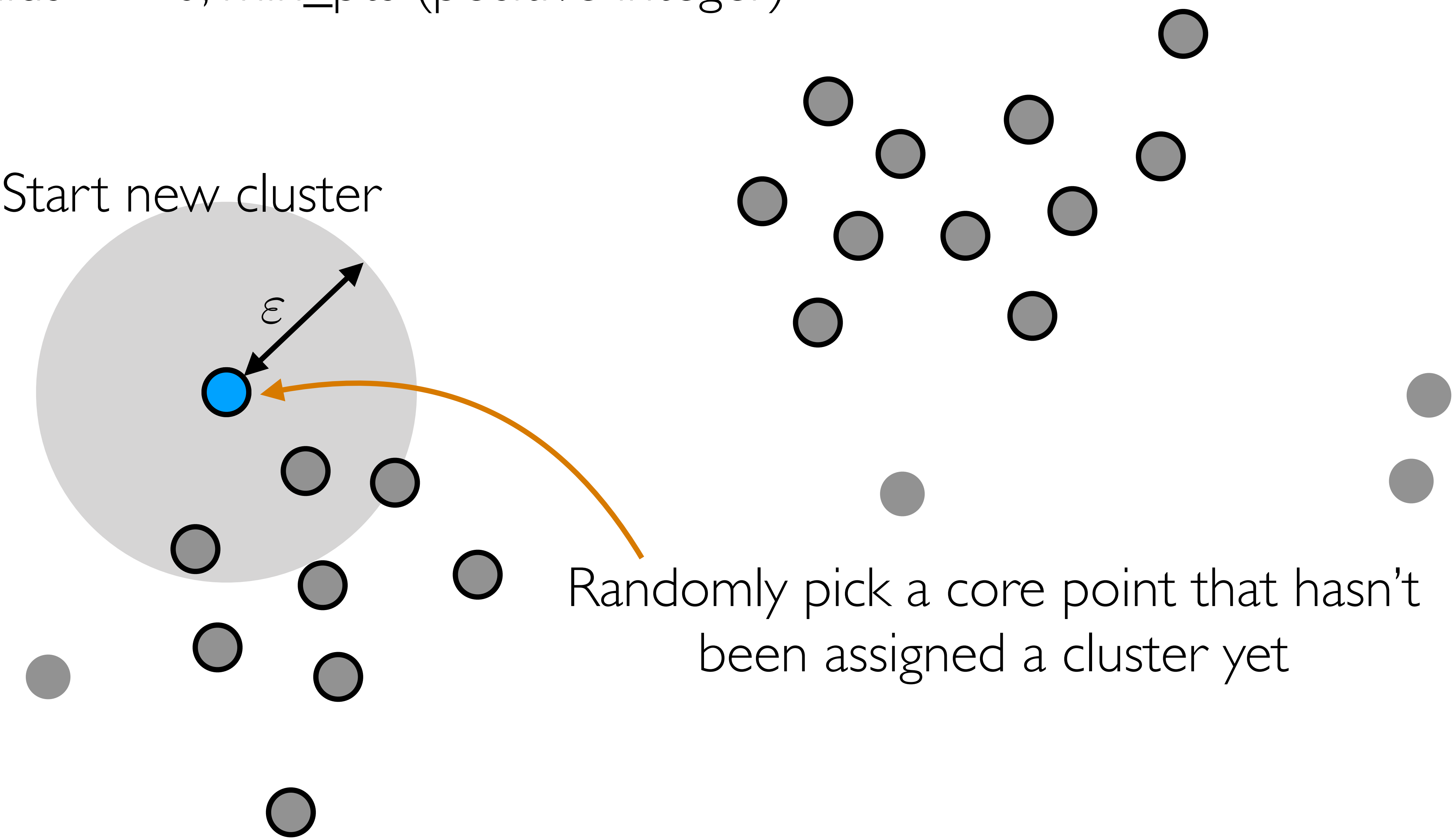
Randomly pick a core point that hasn't been assigned a cluster yet

Core points (**outlined**)

DBSCAN

Let's choose min_pts = 3

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

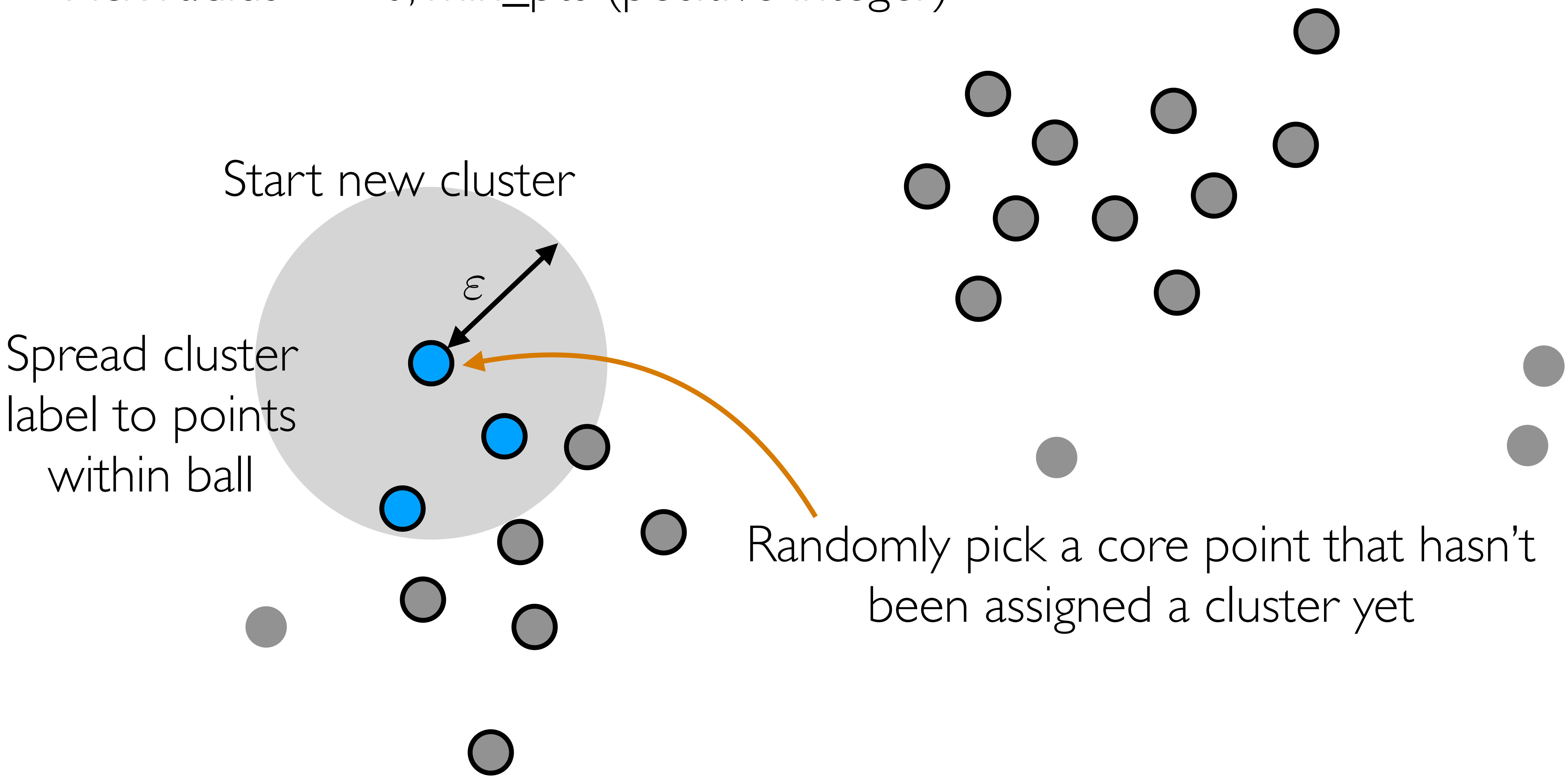


Core points (**outlined**)

DBSCAN

Let's choose min_pts = 3

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

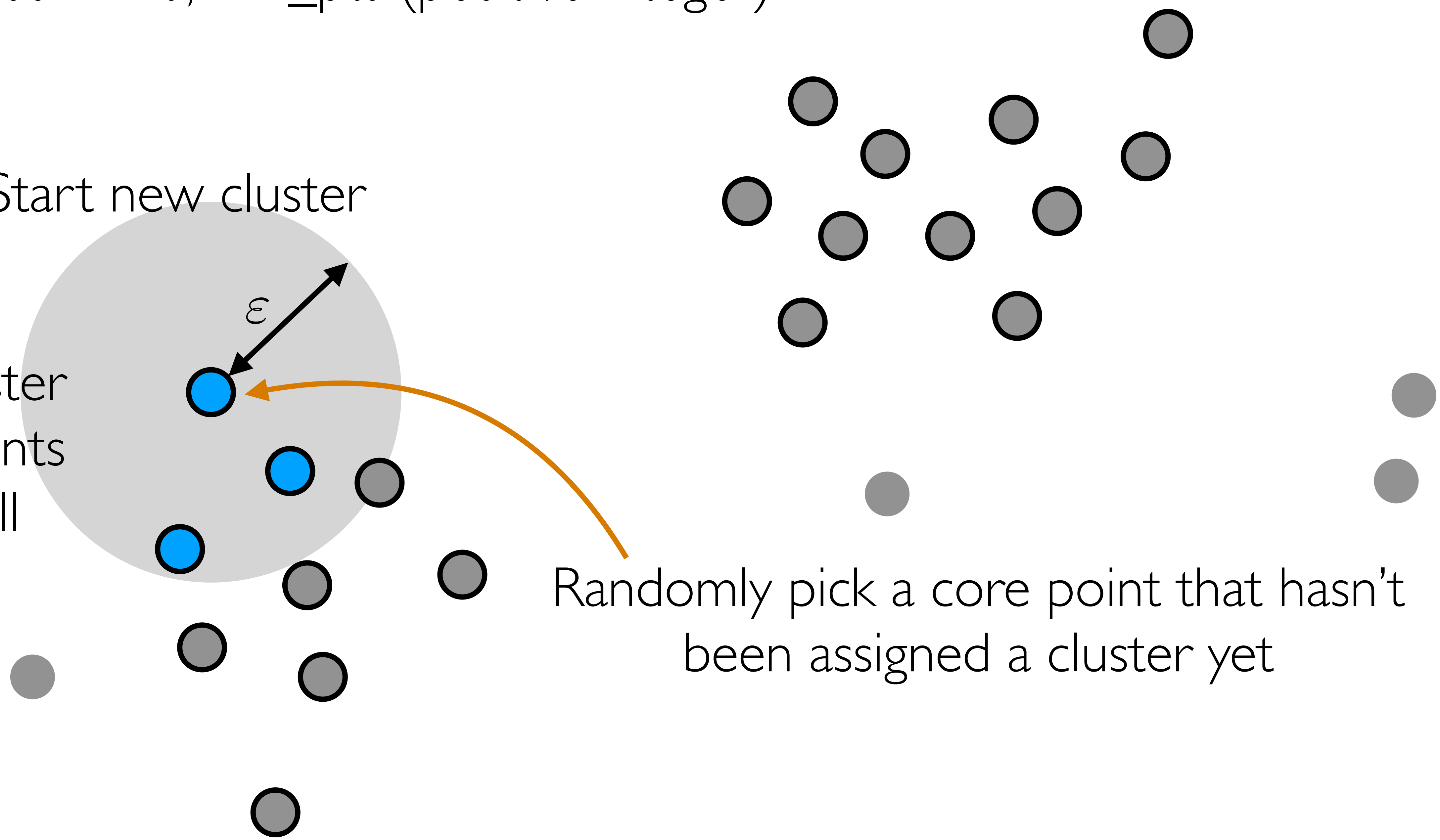


Core points (**outlined**)

DBSCAN

Let's choose min_pts = 3

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



Spread cluster label to points within ball

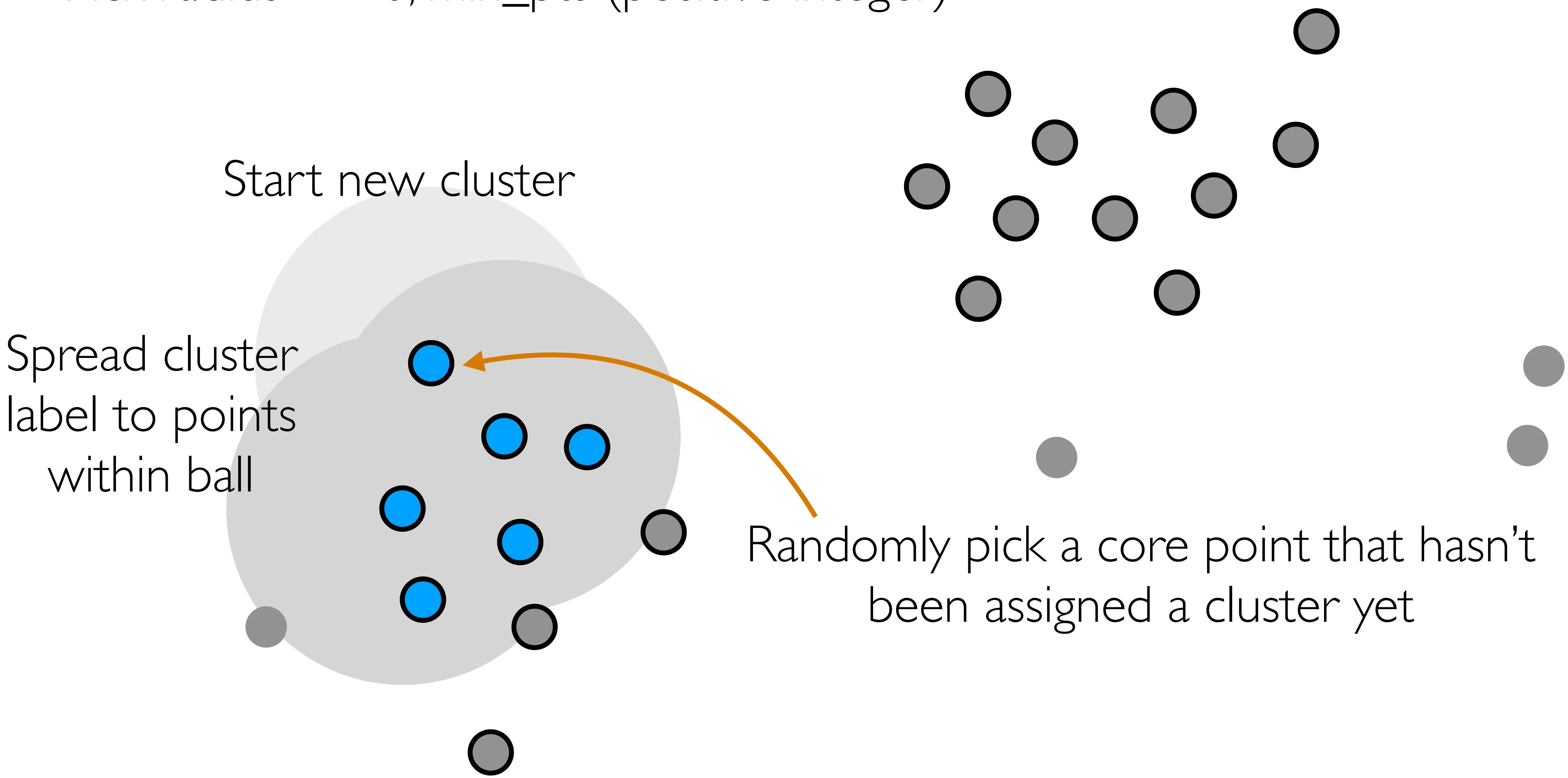
For core points within the ball: spread the cluster label

Core points (**outlined**)

DBSCAN

Let's choose min_pts = 3

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



For core points within the ball: spread the cluster label

Core points (**outlined**)

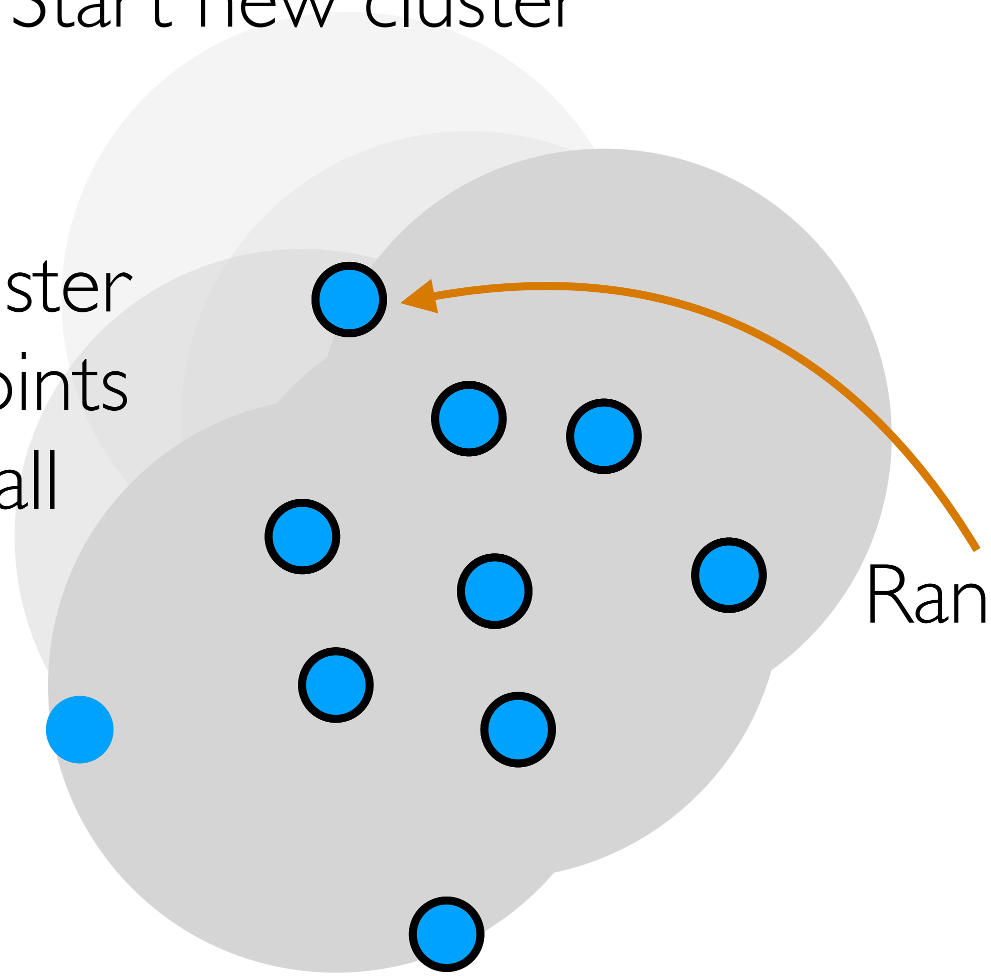
DBSCAN

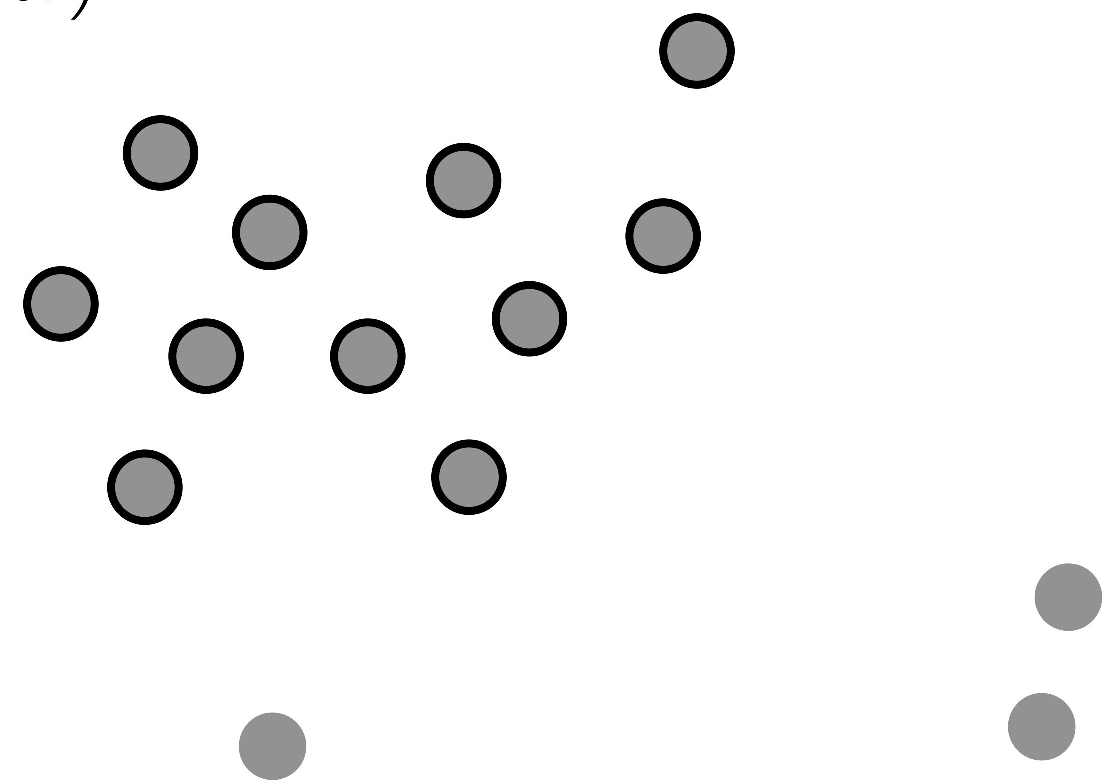
Let’s choose min_pts = 3

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

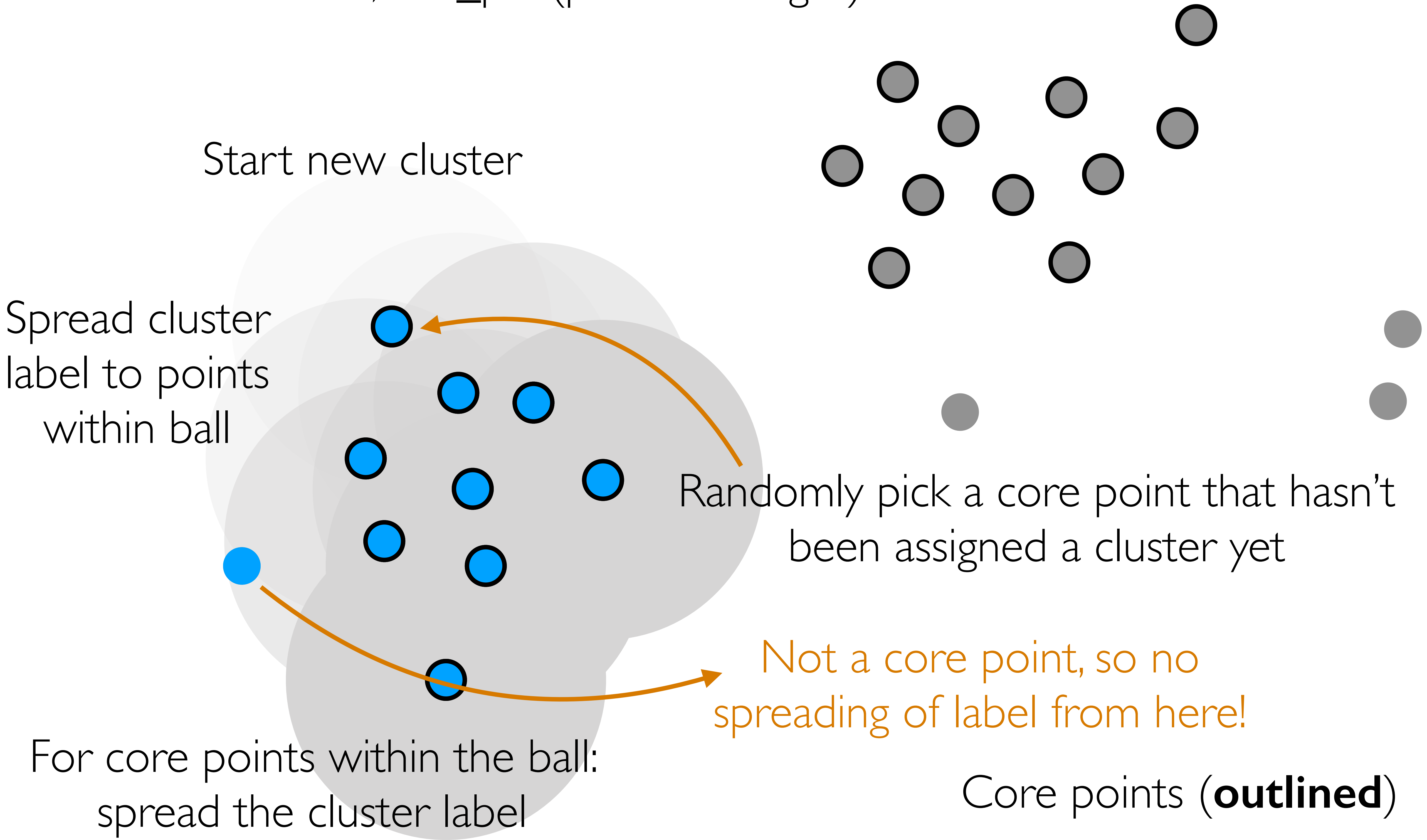
For core points within the ball: spread the cluster label

Core points (**outlined**)

DBSCAN

Let's choose min_pts = 3

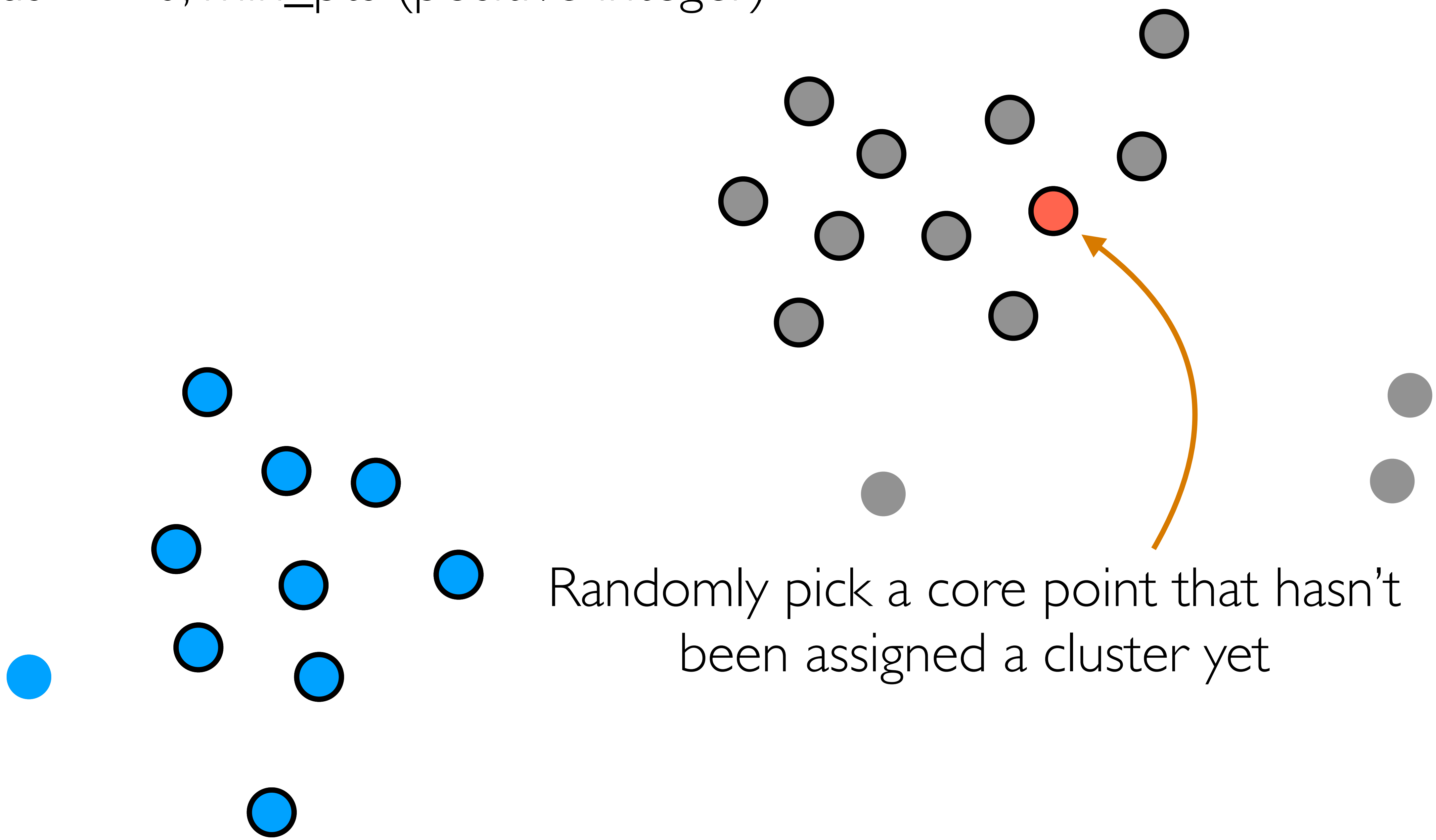
Pick radius $\varepsilon > 0$, min_pts (positive integer)



DBSCAN

Let's choose min_pts = 3

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



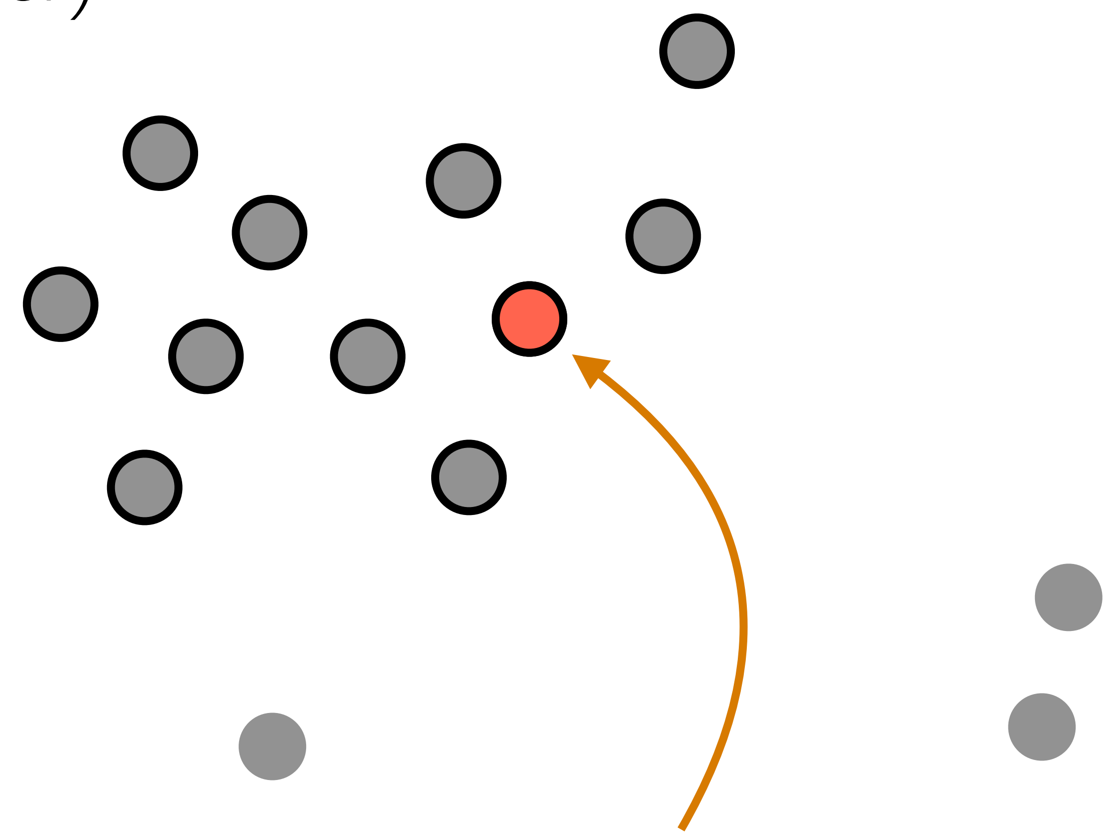
Core points (**outlined**)

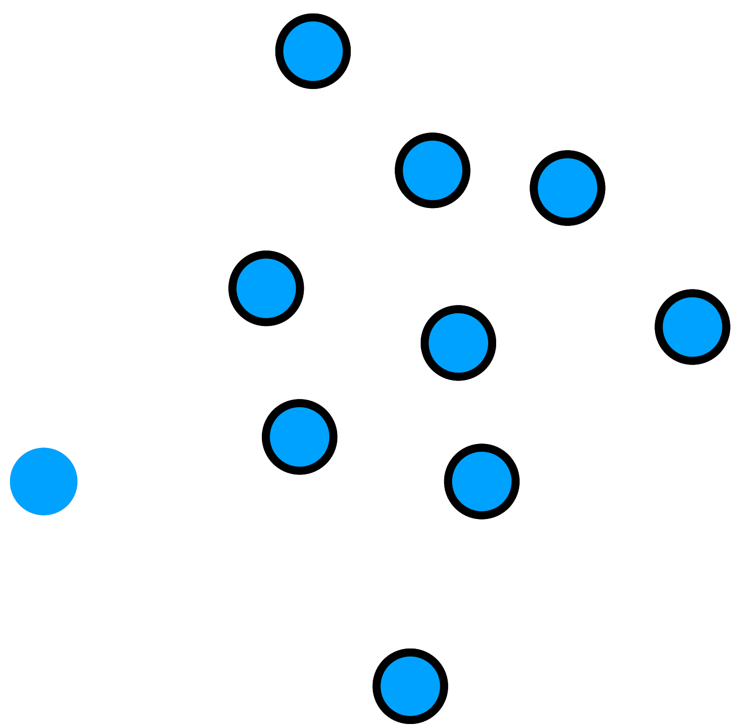
DBSCAN

Let's choose 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

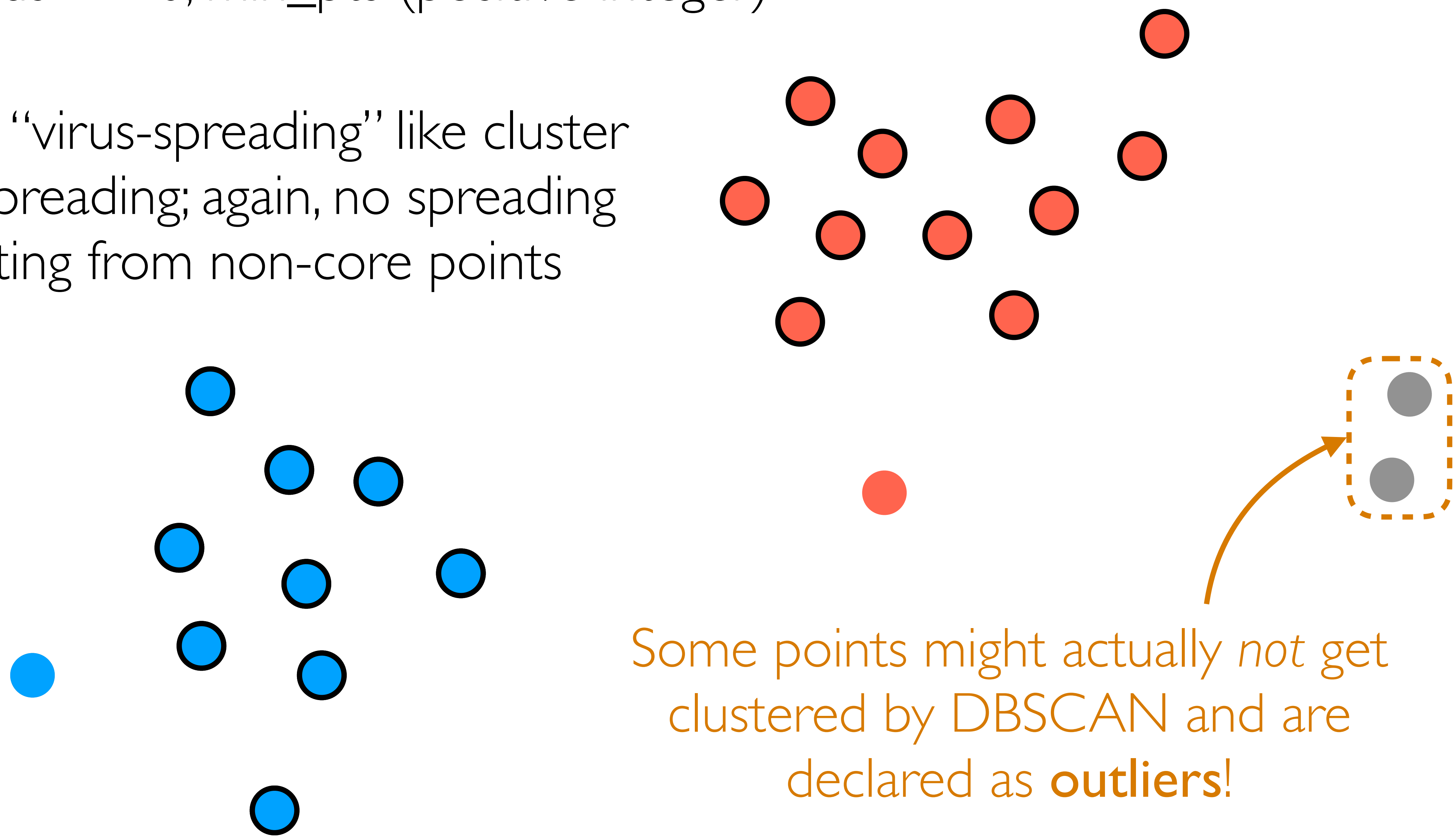
Core points (**outlined**)

DBSCAN

Let's choose 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 are declared as **outliers**!

Core points (**outlined**)

Some Last Remarks on Clustering

Demo for DP-GMM & DBSCAN are at the end of prev. lecture's demo

What about clustering unstructured data?

- Covered this Friday April 8 in Adelaide's recitation
- CMU Pittsburgh students: watch the video recording of the Adelaide recitation next week (since CMU Pittsburgh has Spring Carnival)

Important takeaway: ultimately, *you* have to decide on which clustering method and number of clusters make sense for your data

- After you run a clustering algorithm, make visualizations to interpret the clusters *in the context of your application*!
- Do not just blindly rely on numerical metrics (e.g., CH index)

If you can set up a prediction task, then you can use the prediction task to guide the clustering

Is clustering structure enough?

(Flashback) GMM with *k* Clusters

Cluster 1

Probability of generating a point from cluster 1 $= \pi_1$

Gaussian mean $= \mu_1$

Gaussian covariance $= \Sigma_1$

$\dots$

Cluster ***k***

Probability of generating a point from cluster ***k*** $= \pi_k$

Gaussian mean $= \mu_k$

Gaussian covariance $= \Sigma_k$

How to generate points from this GMM:

1. Flip biased ***k***-sided coin (the sides have probabilities $\pi_1, \dots, \pi_k$)
2. Let ***Z*** be the side that we got (it is some value 1, …, ***k***)
3. Sample 1 point from the Gaussian for cluster ***Z***

Each data point has a single true cluster assignment ***Z***
& is generated from the Gaussian for cluster ***Z***

In reality, a data point could have “mixed” membership and belong to multiple clusters

How do we model this?

Topic Modeling

Text

Each document is part of multiple topics

Each topic consists of a bunch of regularly co-occurring words (example topics: "sports", "medicine", "movies", "finance")

Movie recommendation

Each user is part of multiple "clusters"/topics

Each cluster/topic consists of a bunch of movies (example clusters: "sci-fi epics", "cheesy rom-coms")

Health care

Each patient's health records explained by multiple "topics"

Each topic consists of co-occurring "events" (example topics: "heart condition", "severe pancreatitis")

Topic Modeling

Text

Each document is part of multiple topics

Each topic consists of a bunch of regularly co-occurring words
(example topics: "sports", "medicine", "movies", "finance")

In all of these examples:

- Each data point (a feature vector) is part of multiple topics
- Each topic corresponds to specific feature values in the feature vector likely appearing

Health care

Each patient's health records explained by multiple "topics"

Each topic consists of co-occurring "events"
(example topics: "heart condition", "severe pancreatitis")

Latent Dirichlet Allocation (LDA)

- For text
- A generative model
- Input: "document-word" matrix, and pre-specified # topics ***k***

Word

1 2 . . . ***d***

Document 1 2 ⋮ ***n***

Each row is a feature vector representing a raw counts histogram!

i-th row, ***j***-th column: # times word ***j*** appears in doc ***i***

- Output: what the ***k*** topics are (details on this shortly)

LDA Generative Model Example

Topic

Document		weather	food
	Alice's text	0.1	0.9
	Bob's text	0.5	0.5

Word

Topic		cold	hot	apple	pie
	weather	0.3	0.7	0.0	0.0
	food	0.1	0.3	0.5	0.1

Each word in Alice's text is generated by:

1. Flip 2-sided coin for Alice
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

LDA Generative Model Example

Document	**Topic:** weather	food
Alice's text	0.1	0.9
Bob's text	0.5	0.5

Topic	**Word:** cold	hot	apple	pie
weather	0.3	0.7	0.0	0.0
food	0.1	0.3	0.5	0.1

Each word in Bob's text is generated by:

1. Flip 2-sided coin for Bob
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

LDA Generative Model Example

		Topic	
		weather	food
Document	Alice's text	0.1	0.9
	Bob's text	0.5	0.5

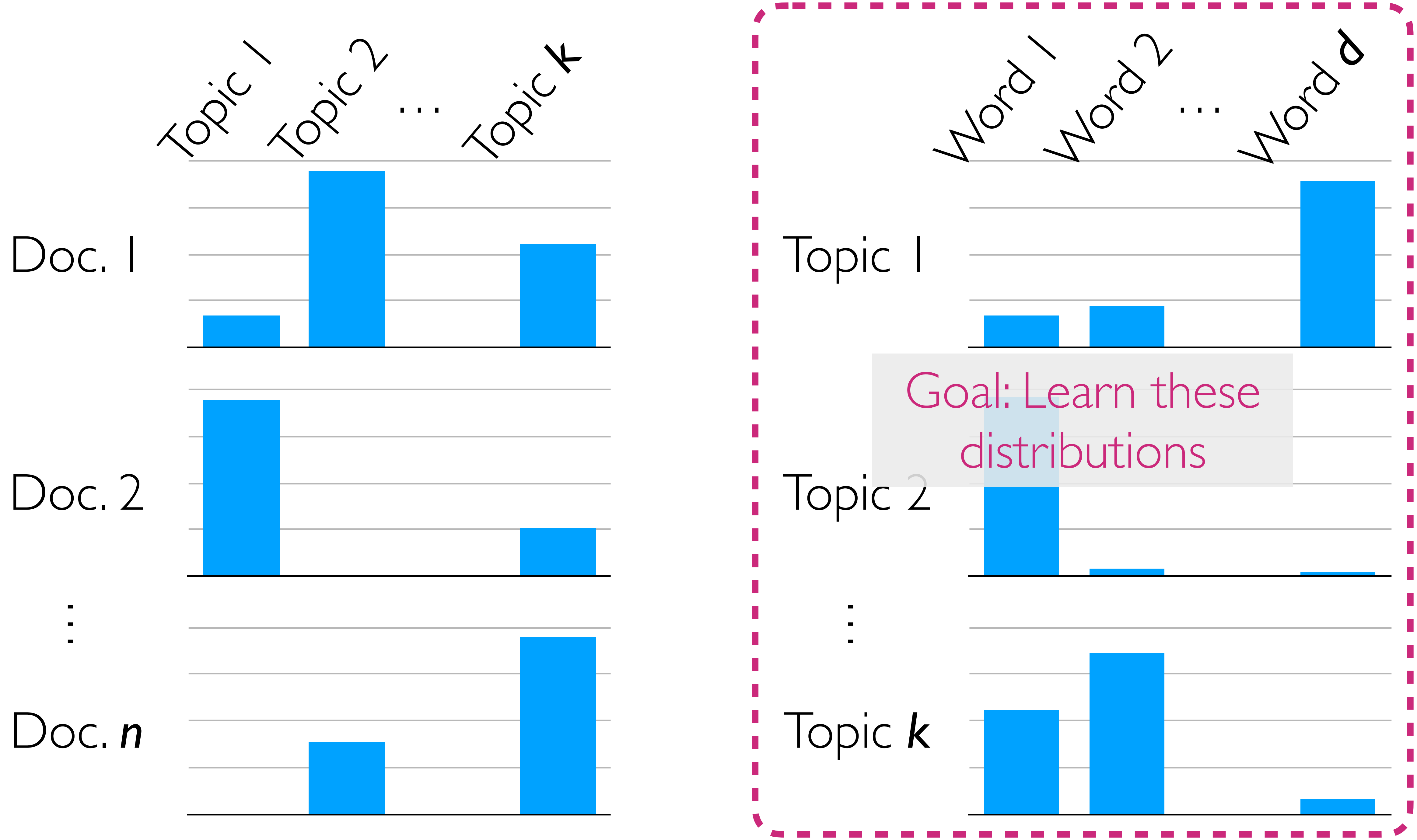
		Word			
		cold	hot	apple	pie
Topic	weather	0.3	0.7	0.0	0.0
	food	0.1	0.3	0.5	0.1

Each word in doc ***i*** is generated by:

1. Flip 2-sided coin for doc ***i***
2. If weather: flip 4-sided coin for weather
   If food: flip 4-sided coin for food

"Learning the topics" means figuring out these 4-sided coin probabilities

LDA Generative Model



LDA models each word in document ***i*** to be generated as:

- Randomly choose a topic ***Z*** (use topic distribution for doc ***i***)
- Randomly choose a word (use word distribution for topic ***Z***)

LDA

- For text
- A generative model
- Input: "document-word" matrix, and pre-specified # topics ***k***

Word

1 2 … ***d***

Document 1, 2, ⋮, ***n***

Each row is a feature vector representing a raw counts histogram!

i-th row, ***j***-th column: # times word ***j*** appears in doc ***i***

- Output: the ***k*** topics' distribution of words

LDA

Demo