E-M method for latent variable models

Define **augmented likelihood**

$$\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R}) := \sum_{i=1}^{n} \sum_{j=1}^{k} R_{ij} \ln \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, y_i = j)}{R_{ij}},$$

with **responsibility matrix** $\boldsymbol{R} \in \mathcal{R}_{n,k} := \{\boldsymbol{R} \in [0,1]^{n \times k} : \boldsymbol{R}\mathbf{1}_k = \mathbf{1}_n\}$.

Alternate two steps:

- E-step: set $(\boldsymbol{R}_t)_{ij} := p_{\boldsymbol{\theta}_{t-1}}(y_i = j | \boldsymbol{x}_i)$.
- M-step: set $\boldsymbol{\theta}_t = \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R}_t)$.

Soon: we'll see this gives nondecreasing likelihood!

E-M for Gaussian mixtures

Initialization: a standard choice is $\pi_j = 1/k$, $\boldsymbol{\Sigma}_j = \boldsymbol{I}$, and $(\mu_j)_{j=1}^k$ given by k-means.

- **E-step:** Set $R_{ij} = p_{\boldsymbol{\theta}}(y_i = j | \boldsymbol{x}_i)$, meaning

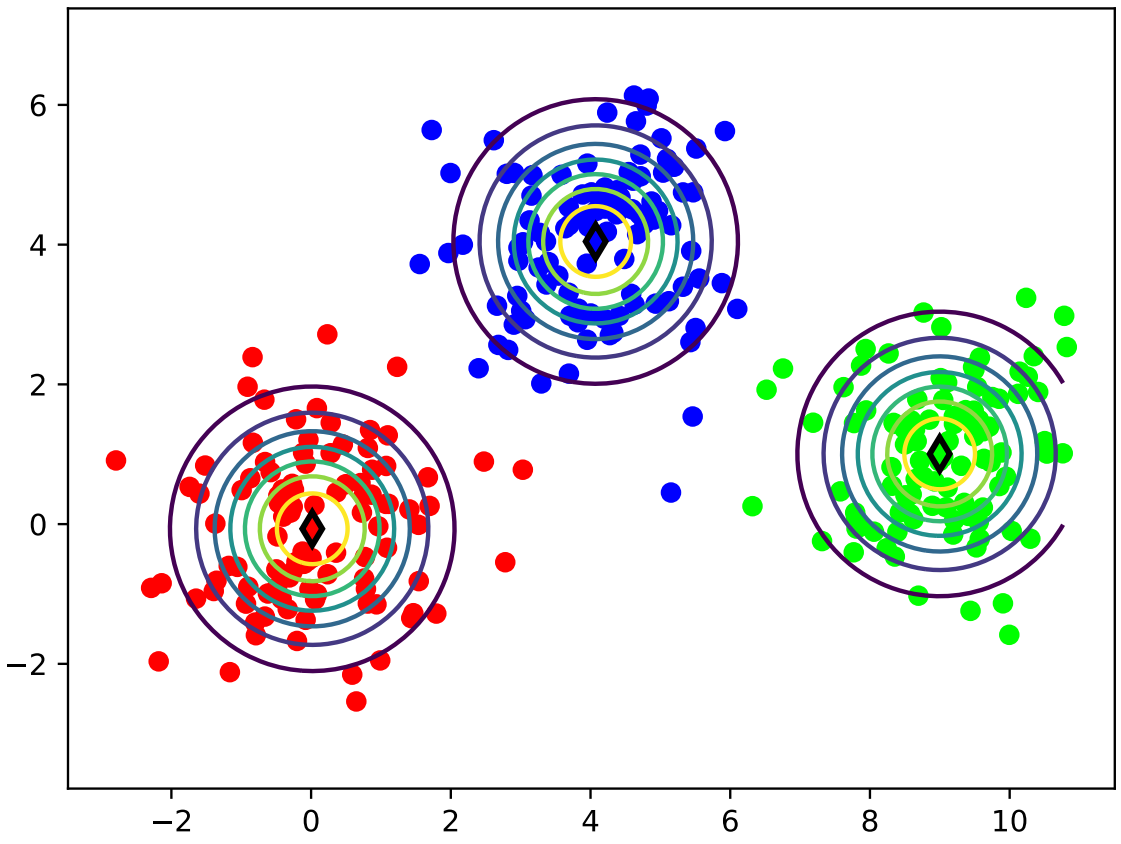
$$R_{ij} = p_{\boldsymbol{\theta}}(y_i = j | \boldsymbol{x}_i) = \frac{p_{\boldsymbol{\theta}}(y_i = j, \boldsymbol{x}_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)} = \frac{\pi_j p_{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j}(\boldsymbol{x}_i)}{\sum_{l=1}^k \pi_l p_{\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l}(\boldsymbol{x}_i)}.$$

- **M-step:** solve $\arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R})$, meaning

$$\begin{aligned}
\pi_j &:= \frac{\sum_{i=1}^n R_{ij}}{\sum_{i=1}^n \sum_{l=1}^k R_{il}} = \frac{\sum_{i=1}^n R_{ij}}{n}, \\
\boldsymbol{\mu}_j &:= \frac{\sum_{i=1}^n R_{ij} \boldsymbol{x}_i}{\sum_{i=1}^n R_{ij}} = \frac{\sum_{i=1}^n R_{ij} \boldsymbol{x}_i}{n \pi_j}, \\
\boldsymbol{\Sigma}_j &:= \frac{\sum_{i=1}^n R_{ij} (\boldsymbol{x}_i - \boldsymbol{\mu}_j)(\boldsymbol{x}_i - \boldsymbol{\mu}_j)^\intercal}{n \pi_j}.
\end{aligned}$$

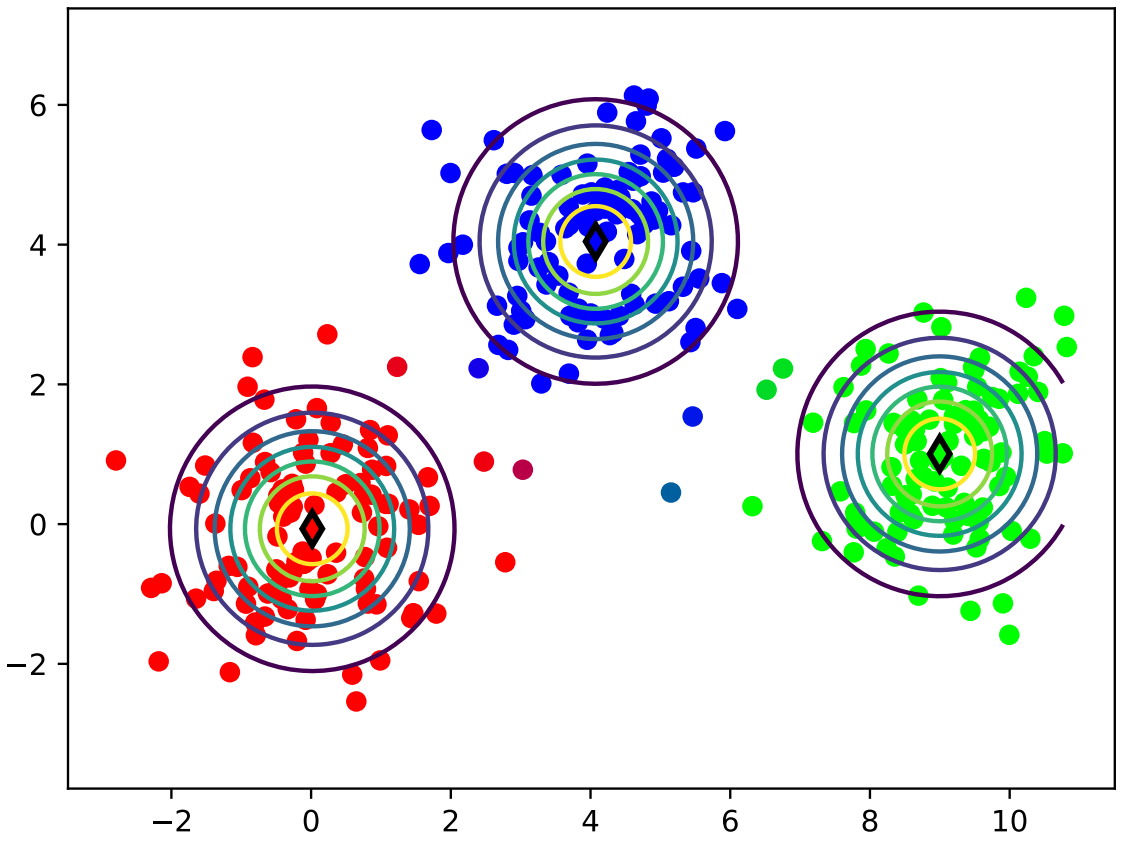
(These are as before.)

Demo: spherical clusters



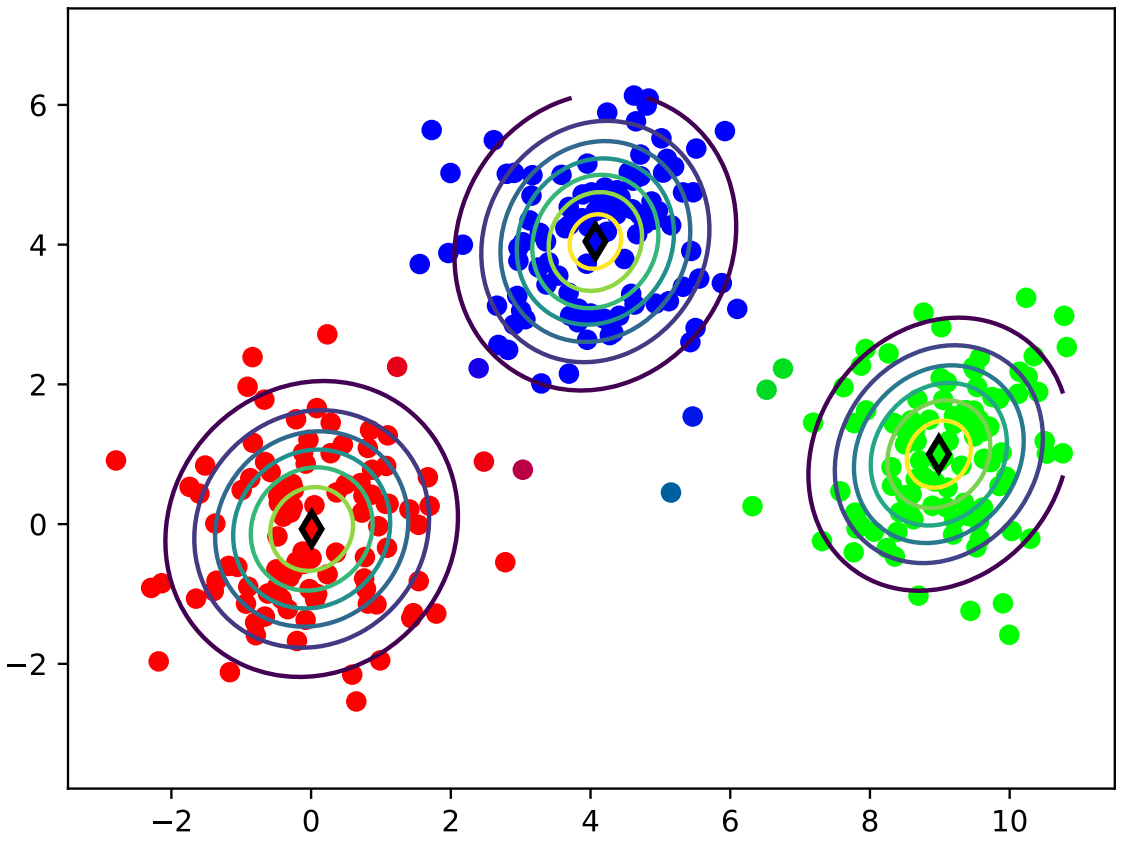
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



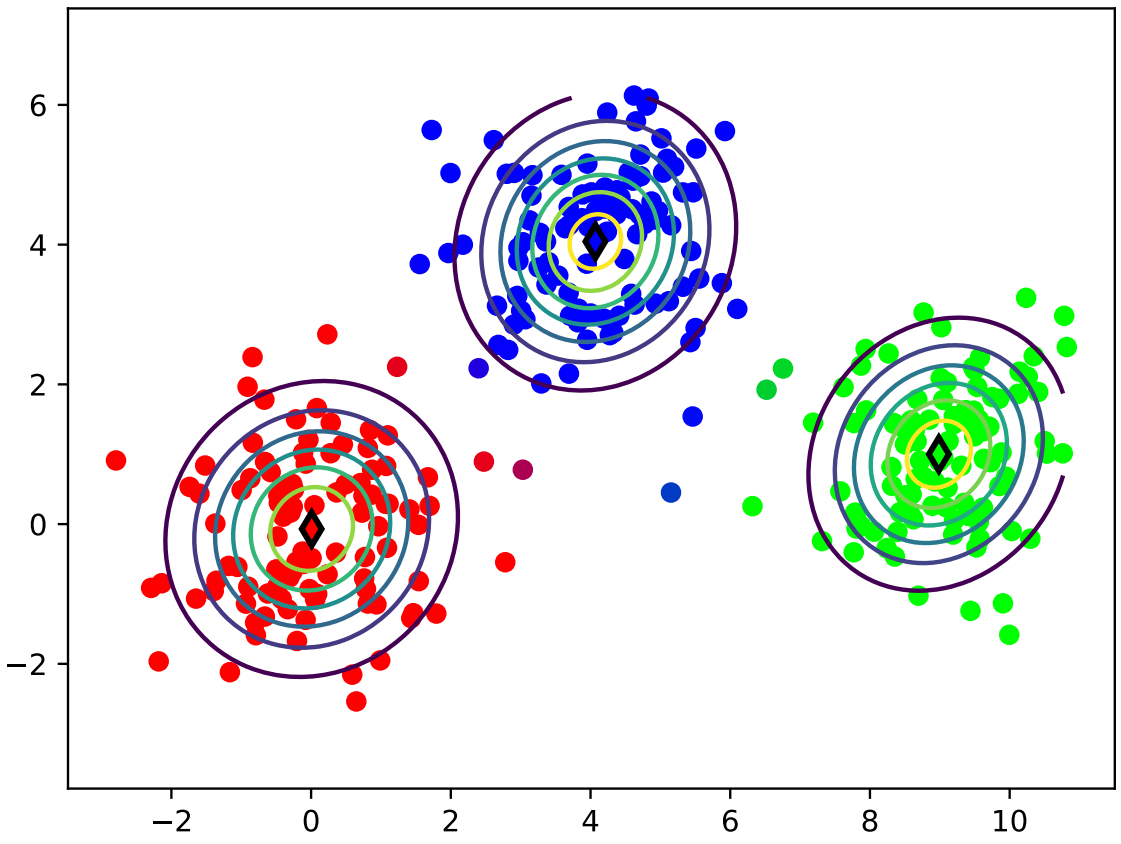
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



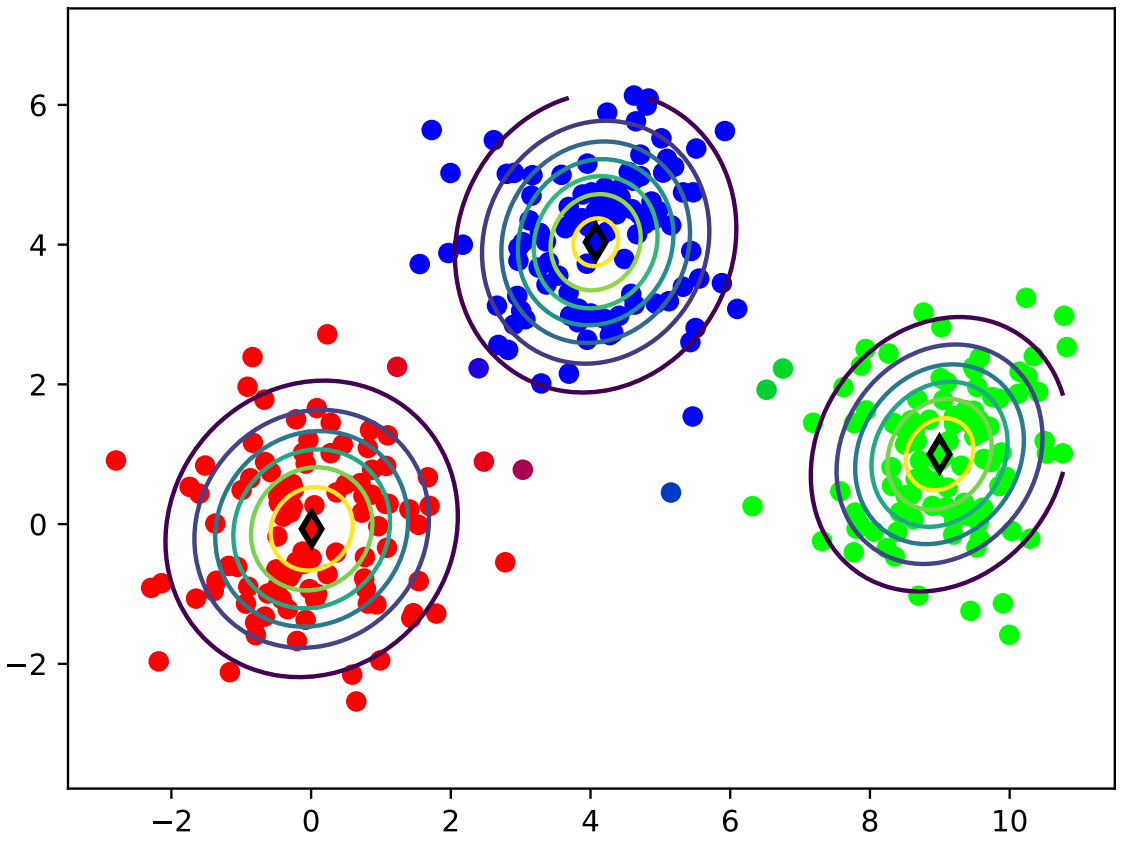
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



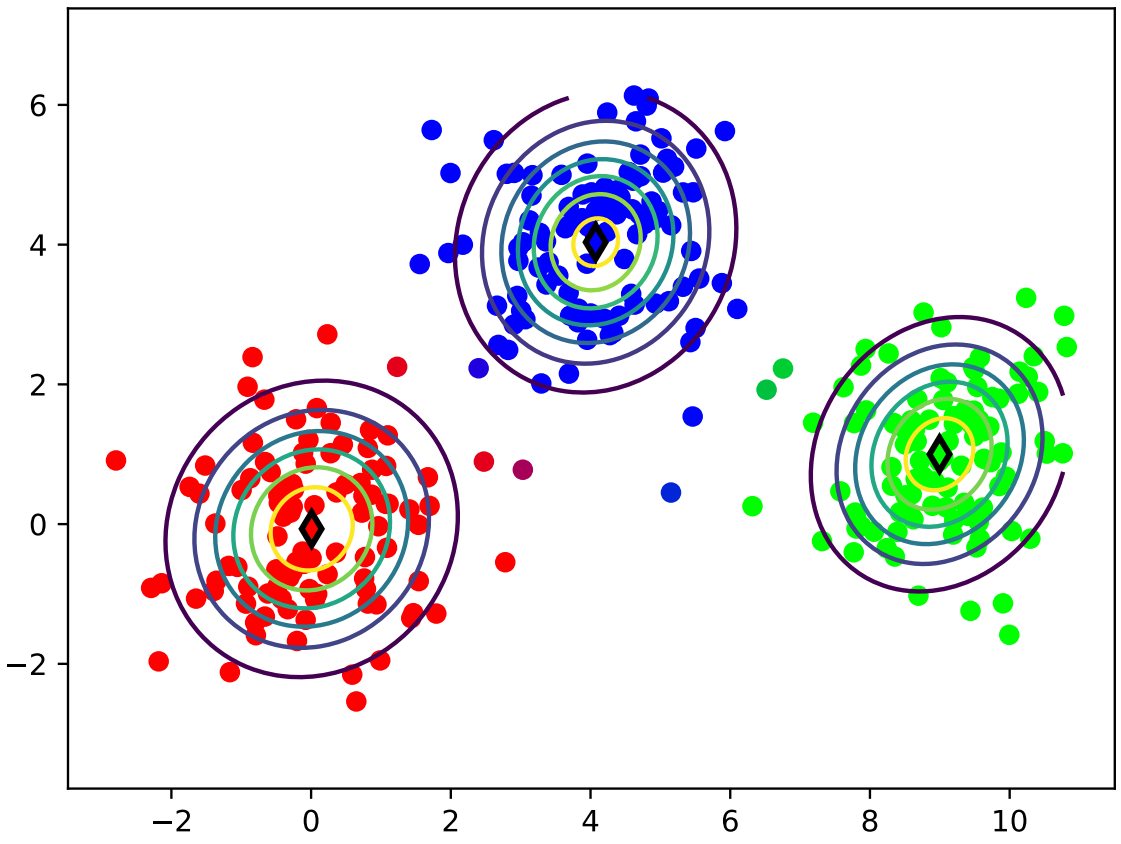
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



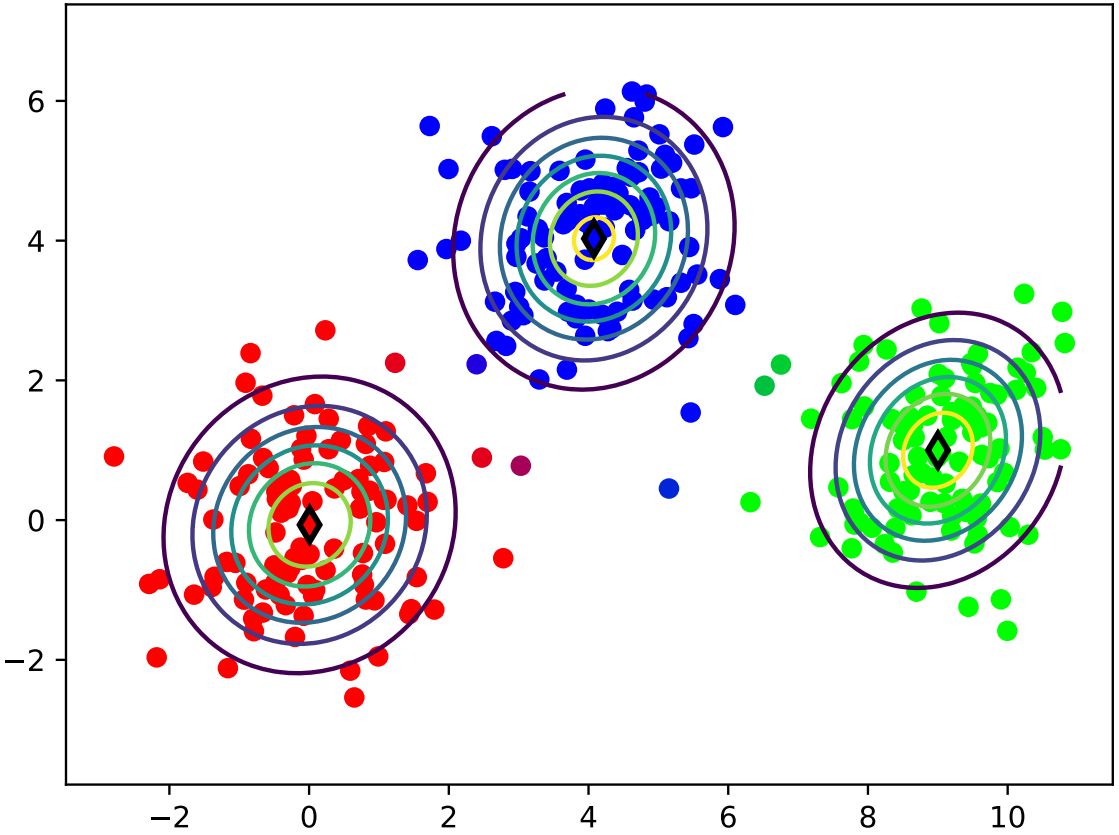
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



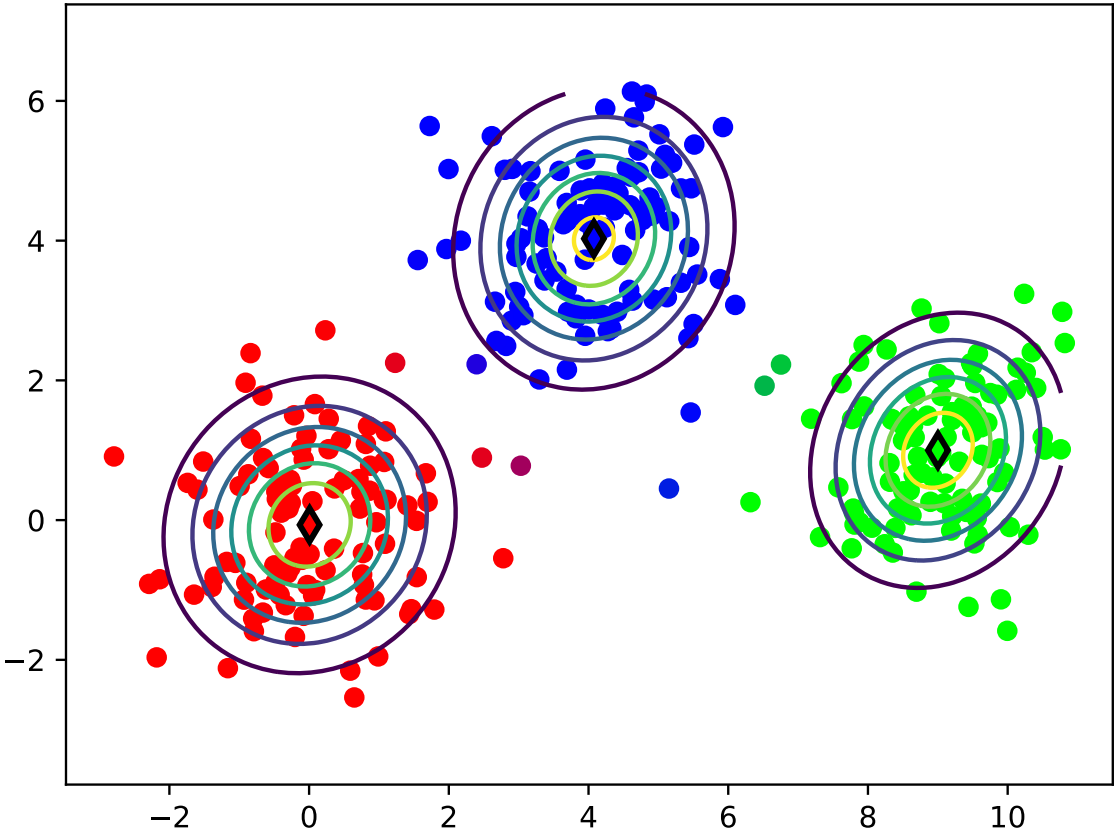
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



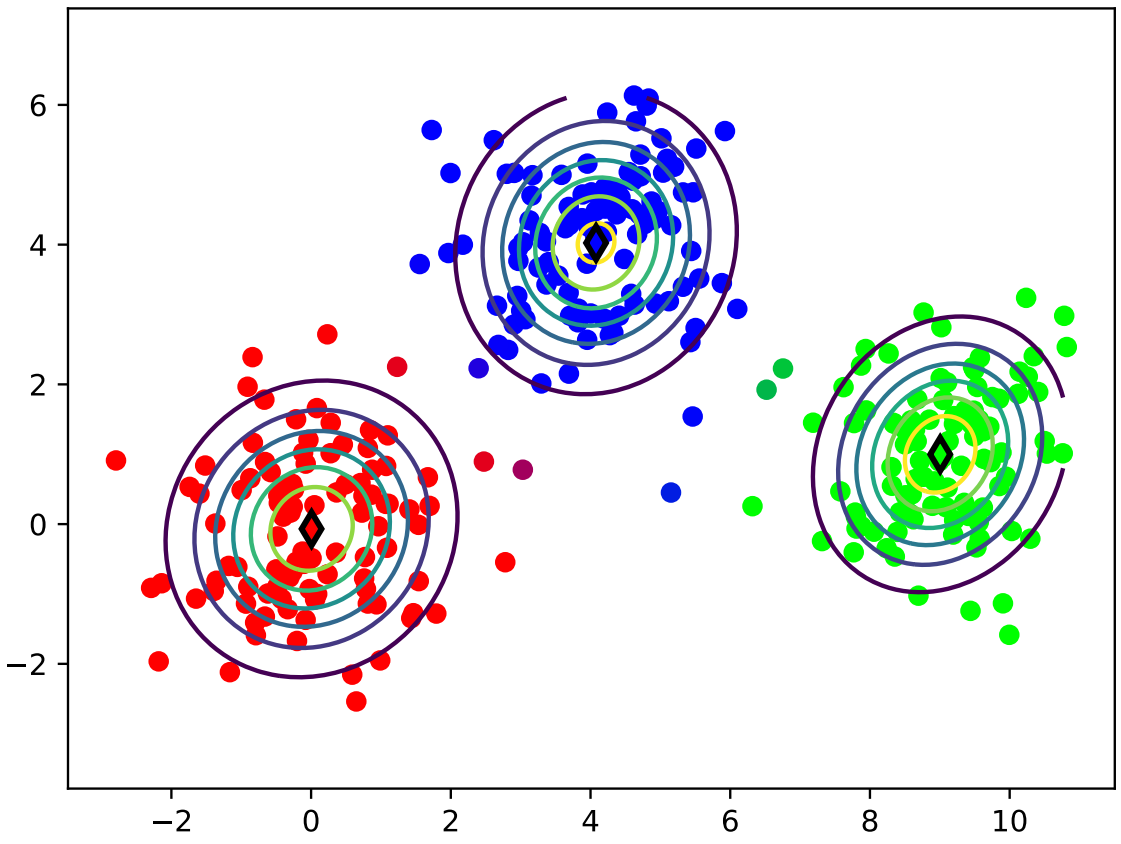
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



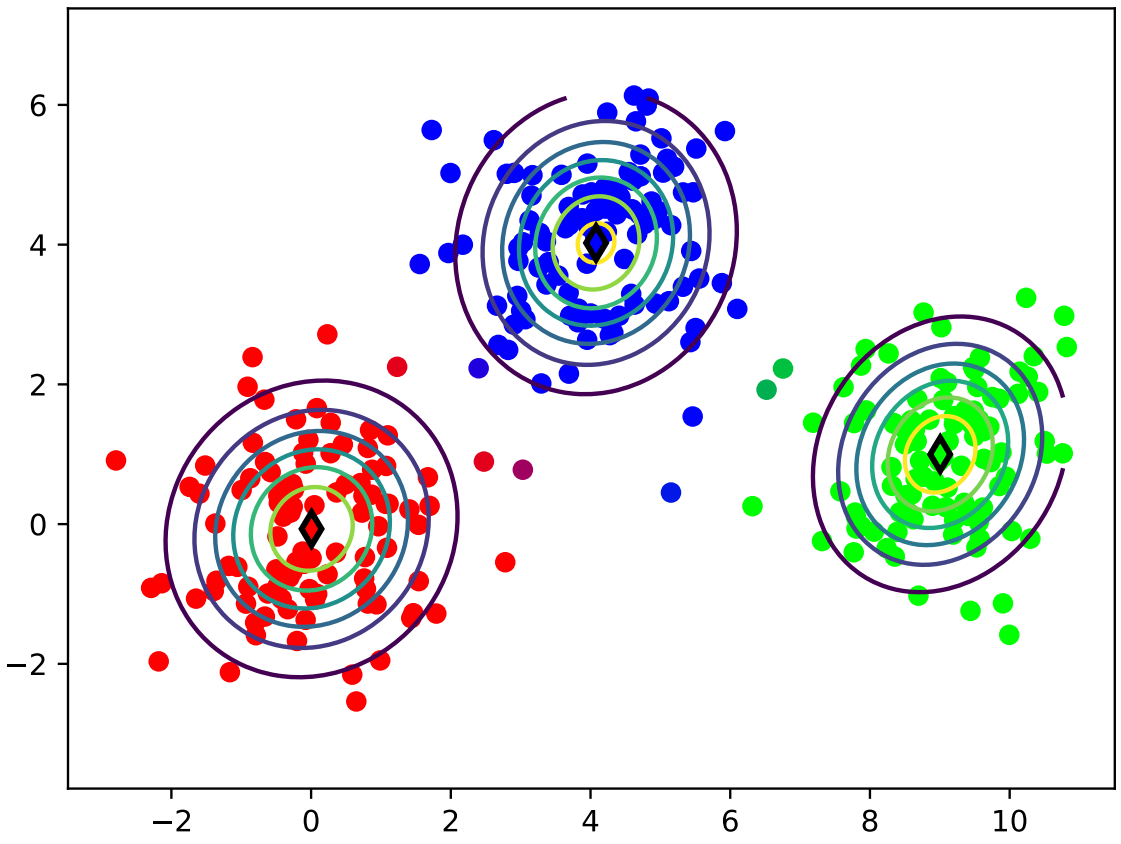
(Initialized with k-means, thus not so dramatic.)

Demo: spherical clusters



(Initialized with k-means, thus not so dramatic.)

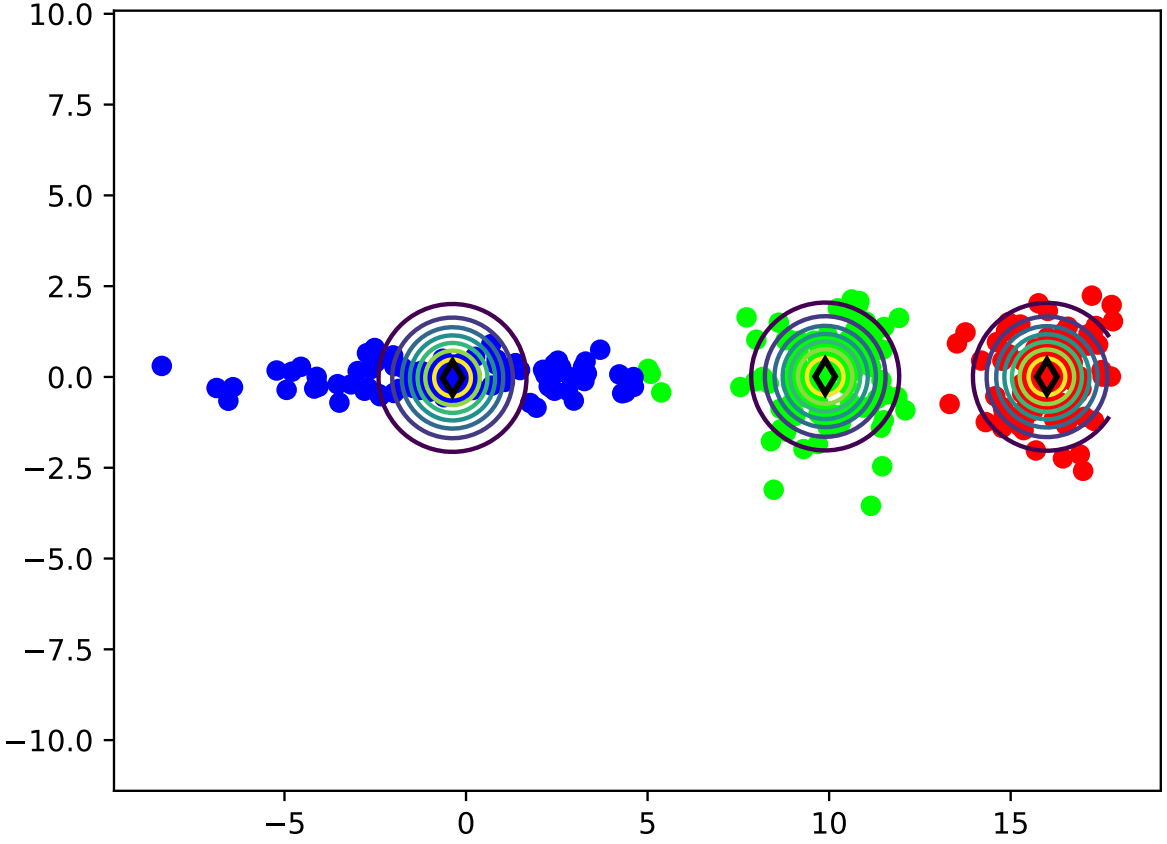
Demo: spherical clusters



(Initialized with k-means, thus not so dramatic.)

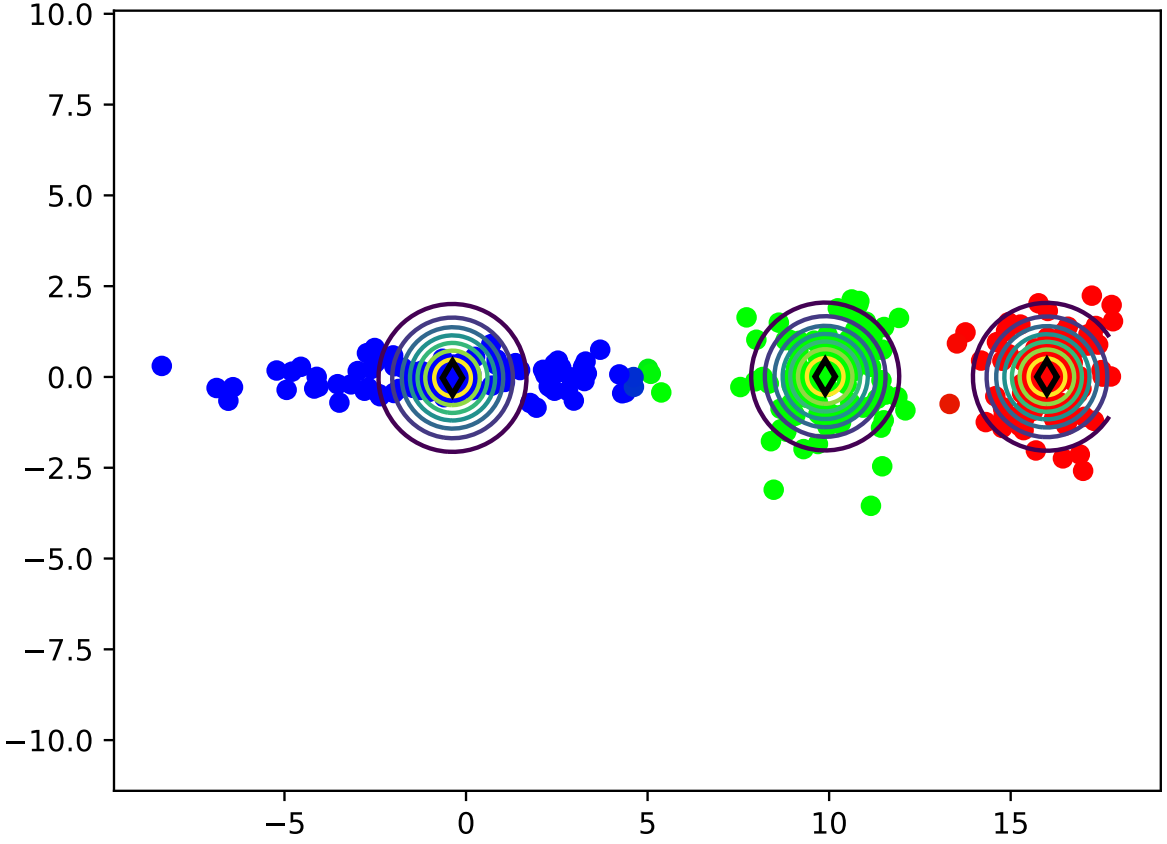
Demo: elliptical clusters

E. . .



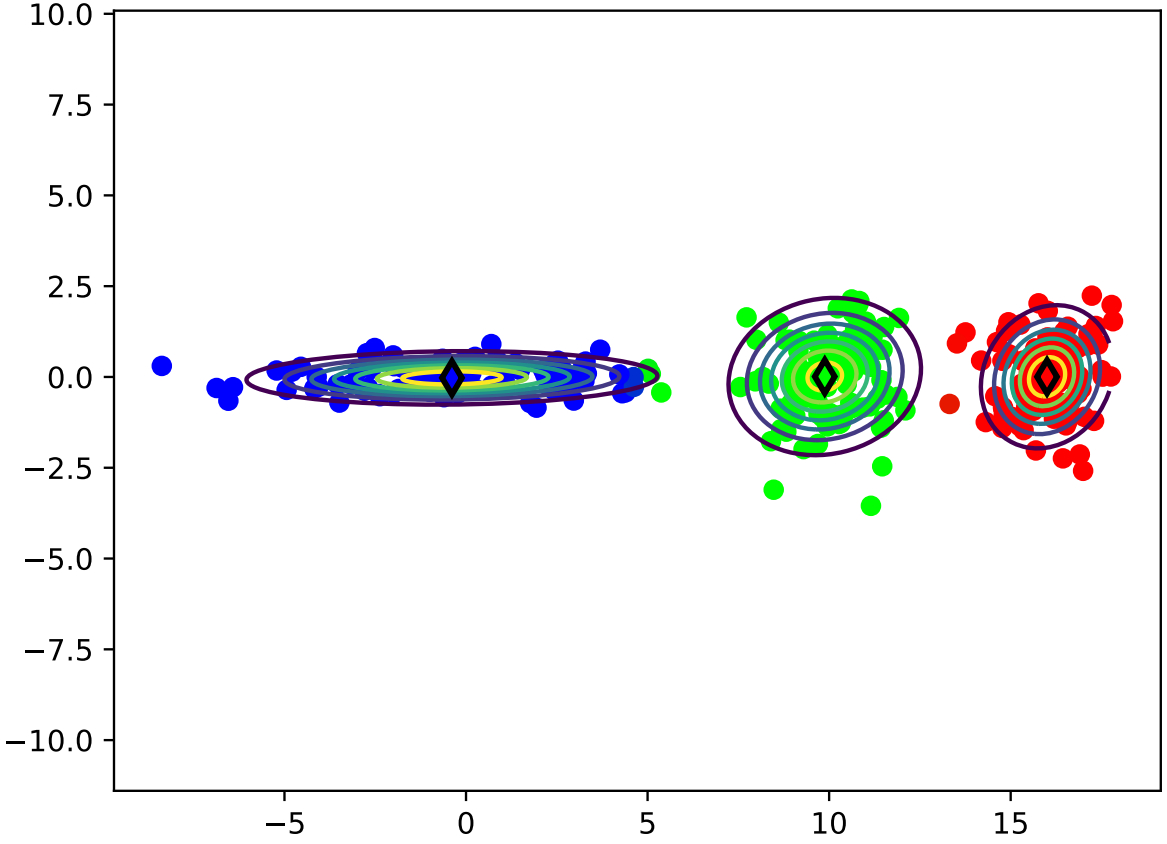
Demo: elliptical clusters

E... M...



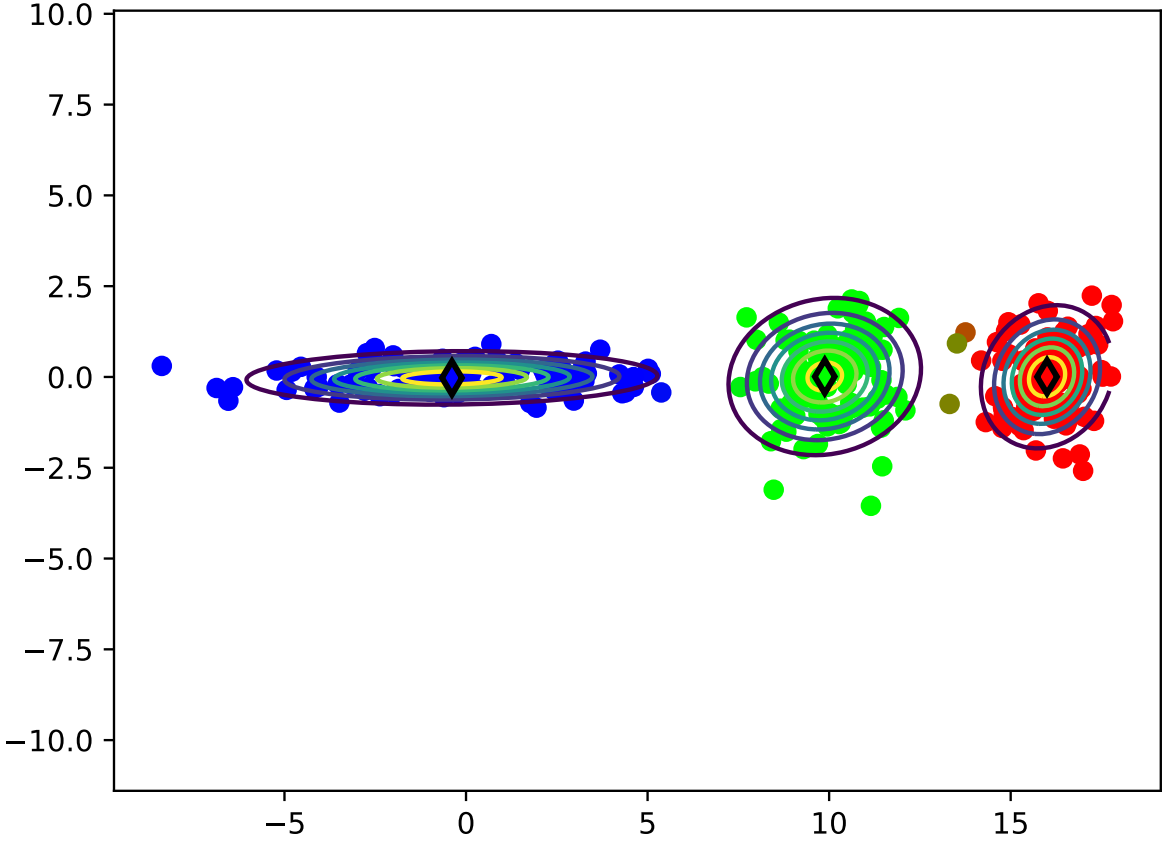
Demo: elliptical clusters

E... M... E...



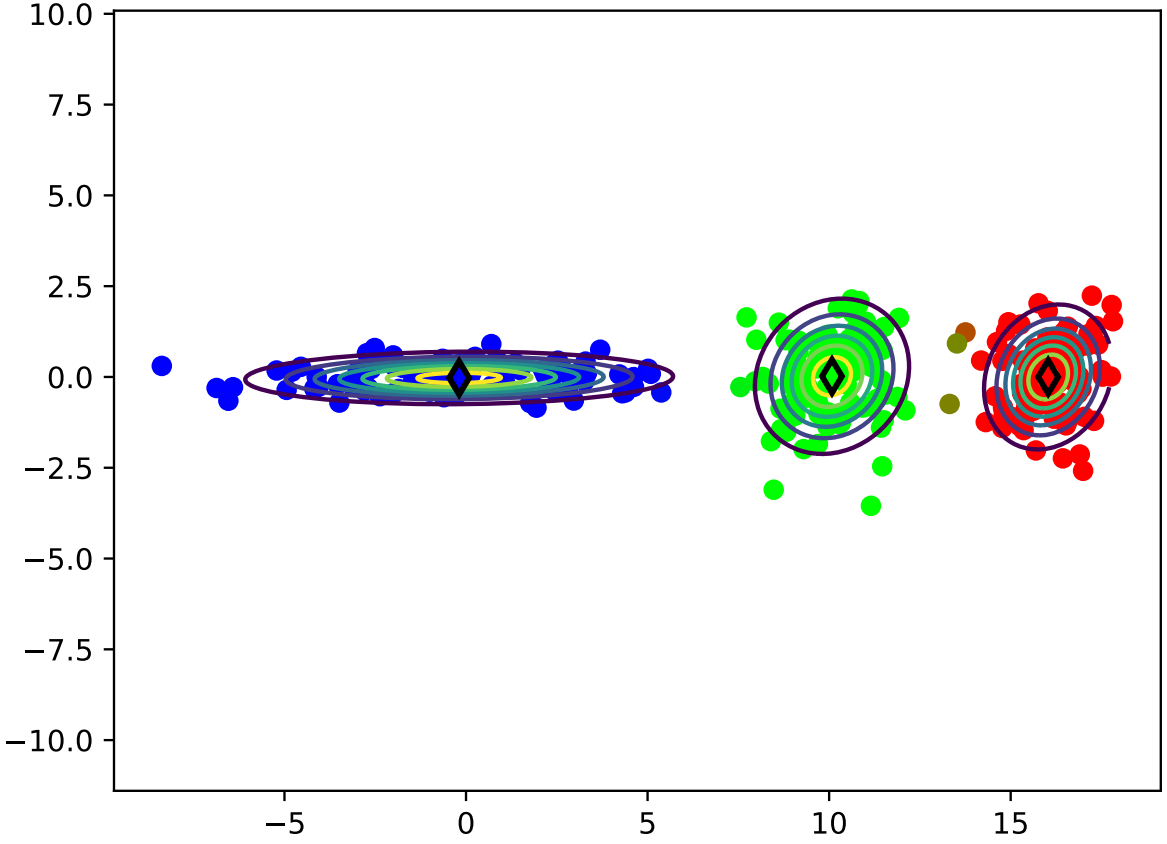
Demo: elliptical clusters

E... M... E... M...



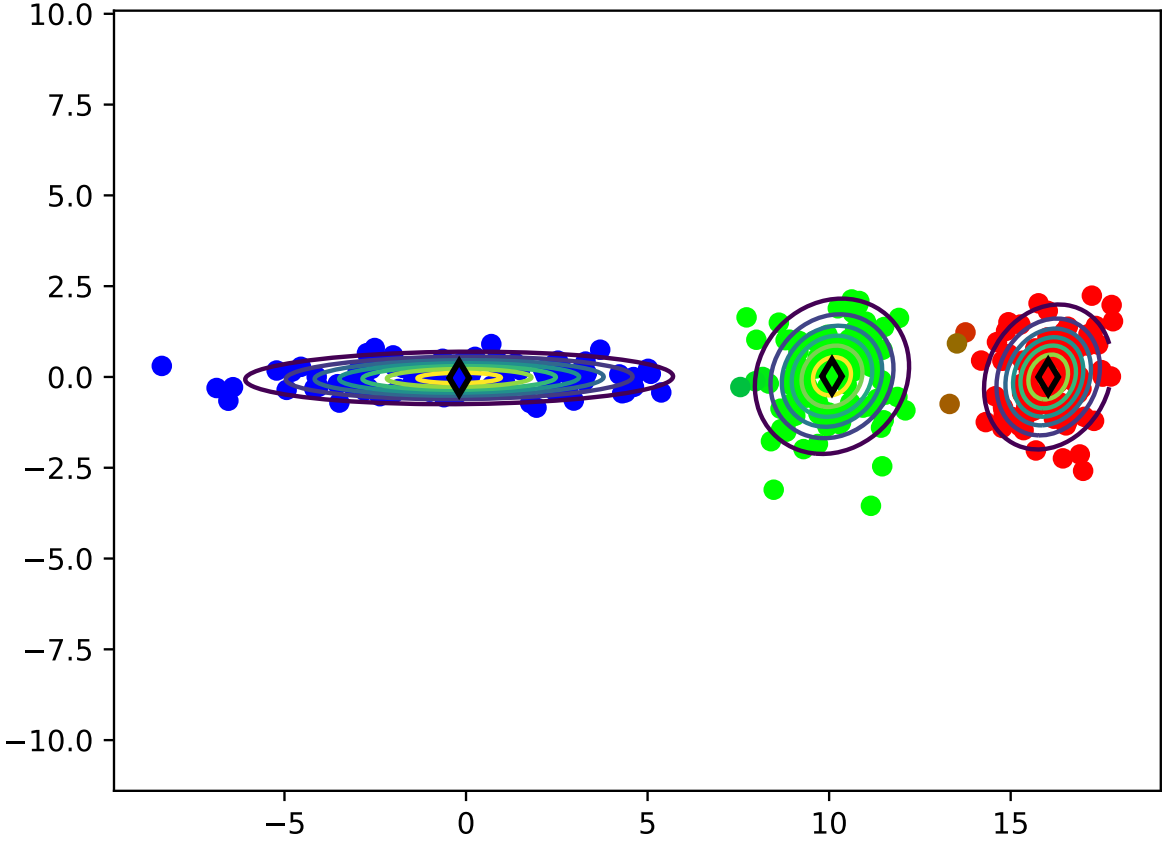
Demo: elliptical clusters

E... M... E... M... E...



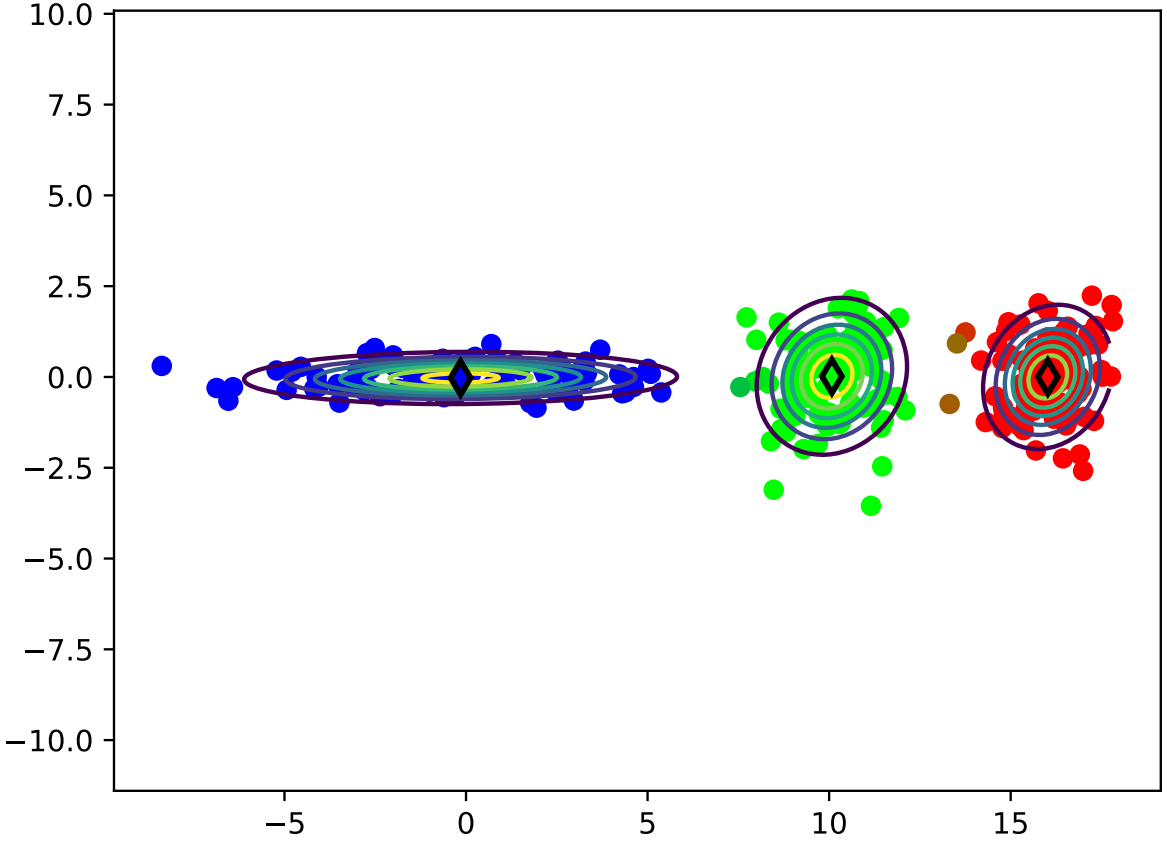
Demo: elliptical clusters

E... M... E... M... E... M...



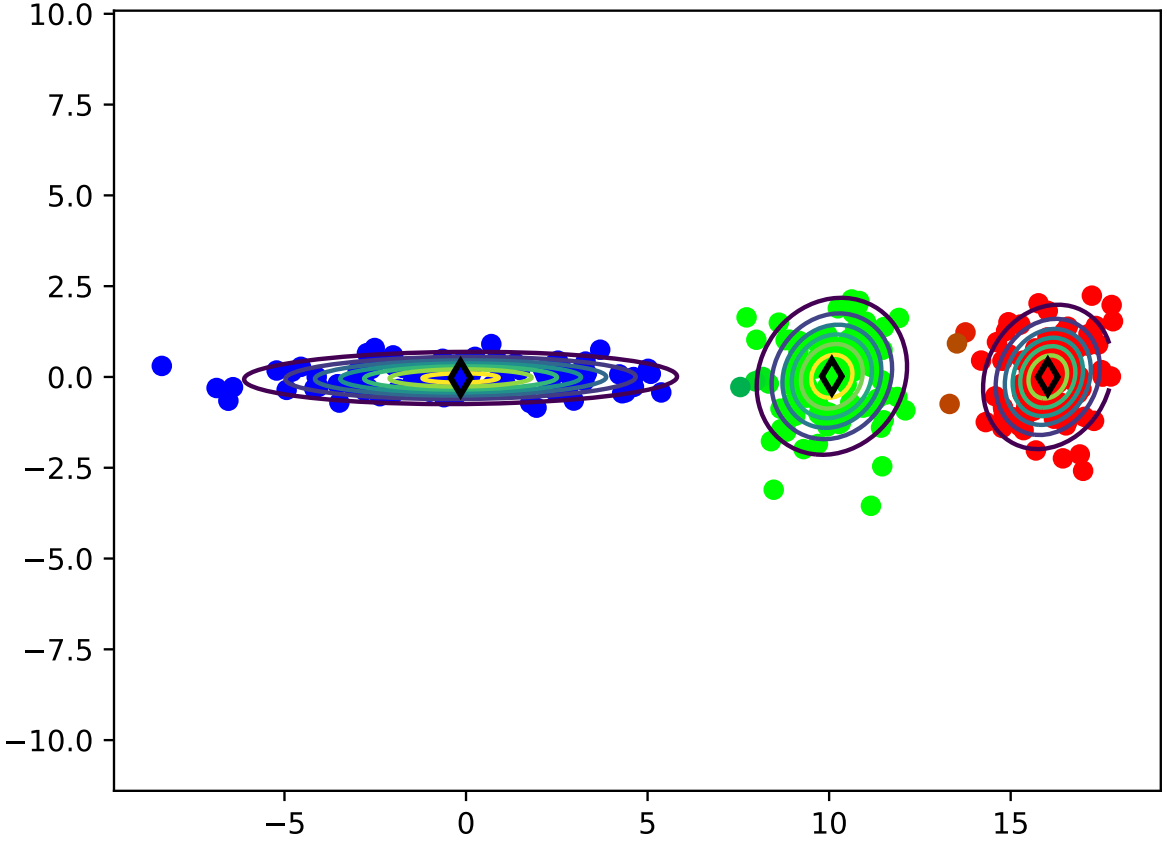
Demo: elliptical clusters

E... M... E... M... E... M... E...



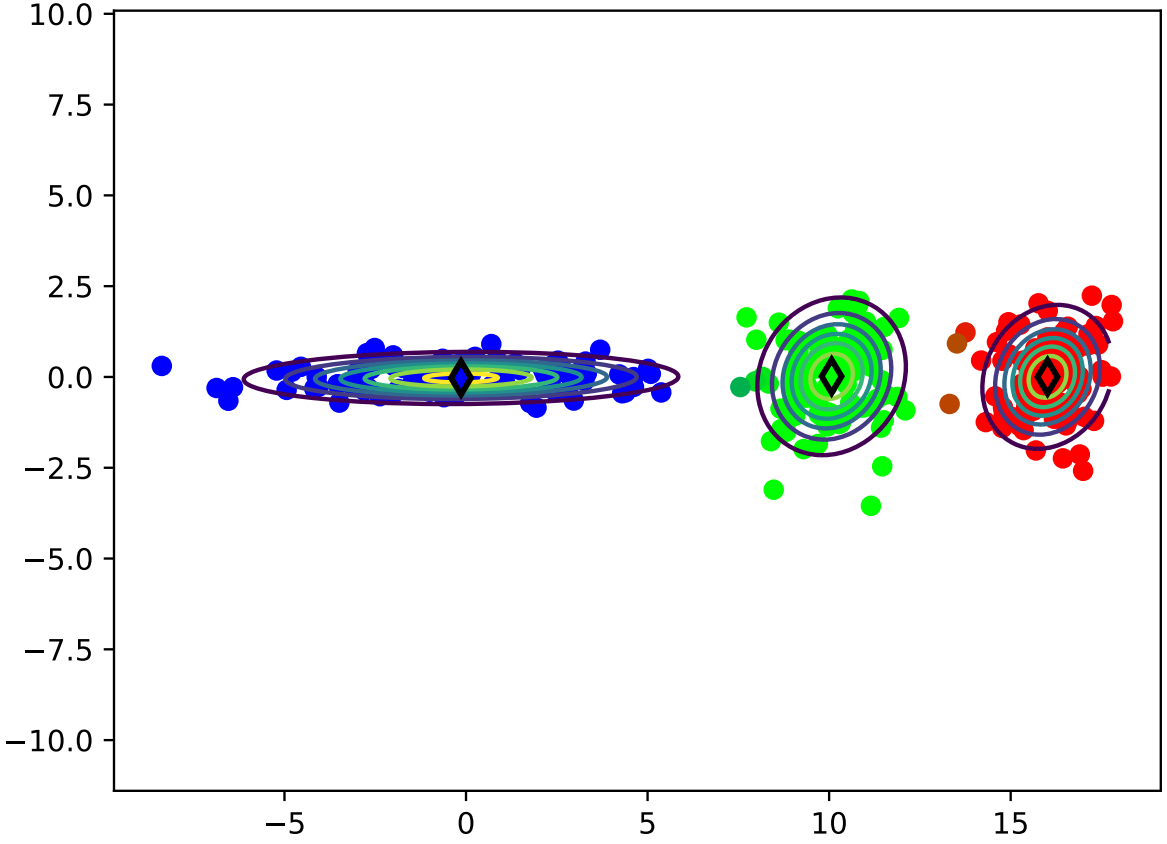
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



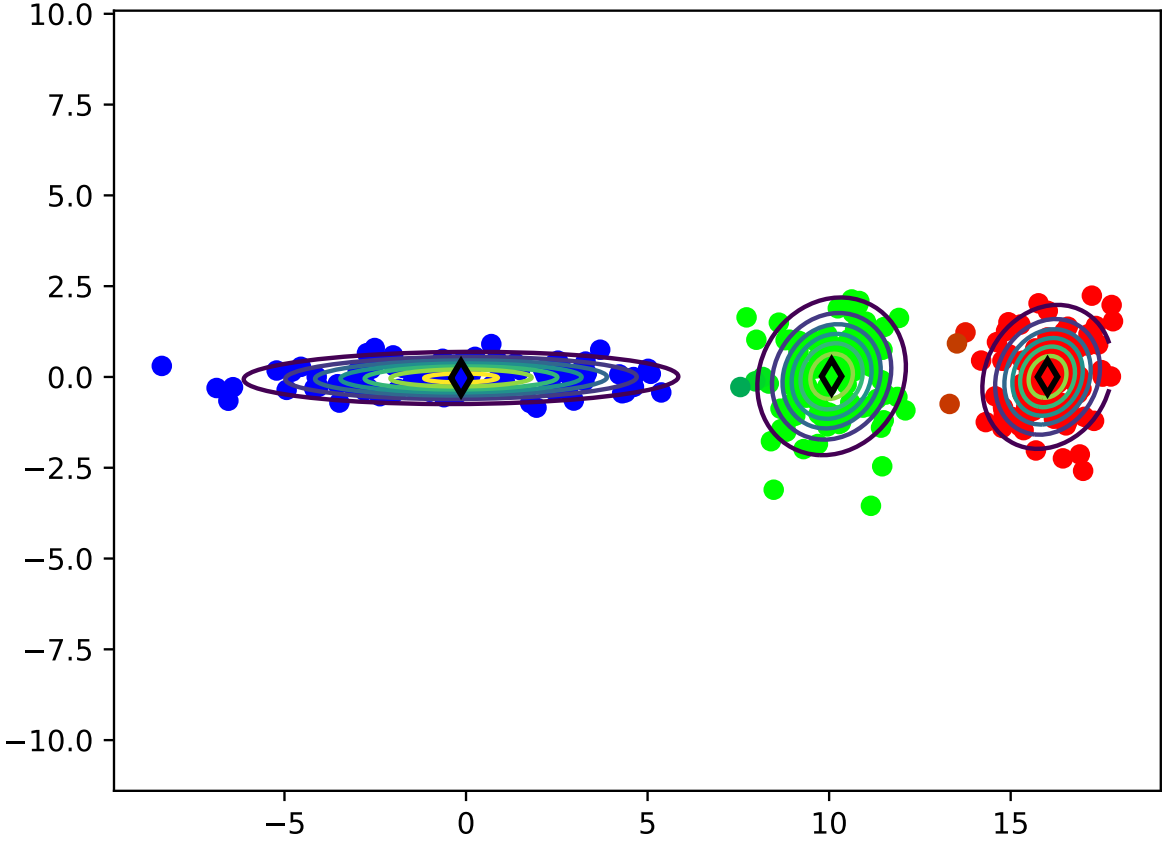
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



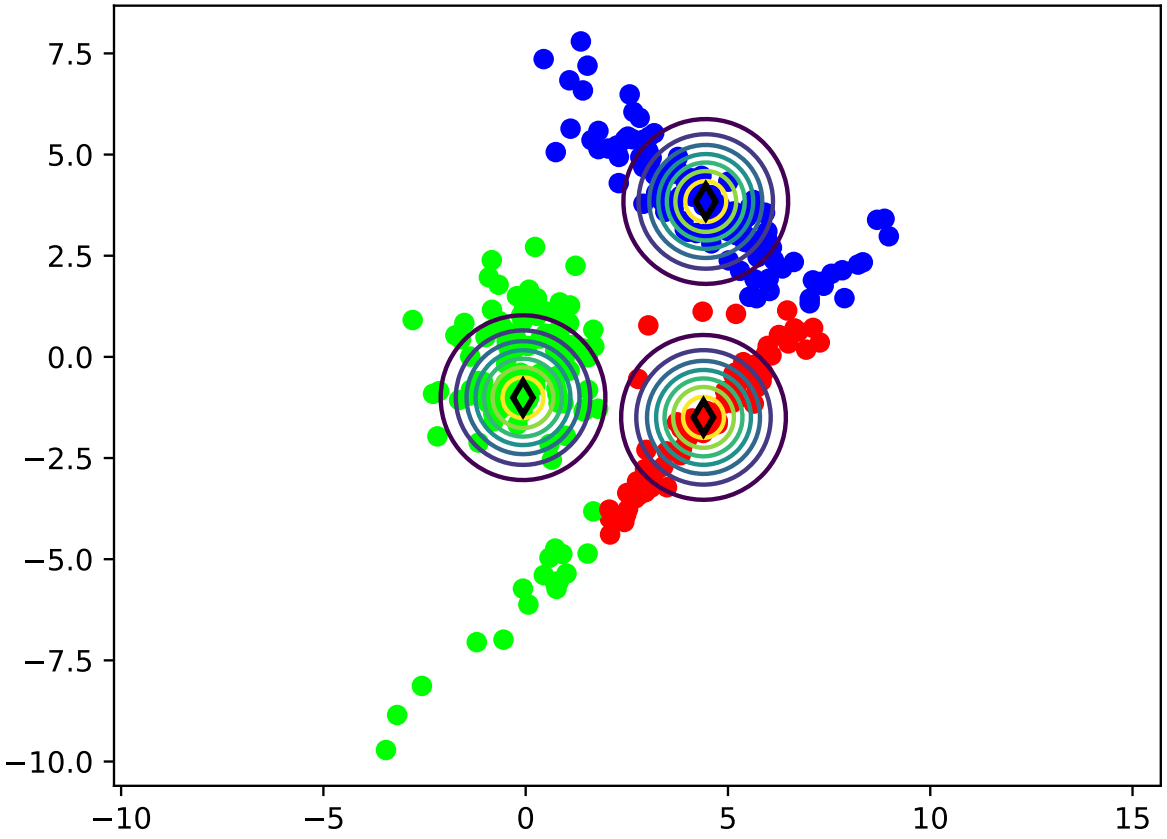
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



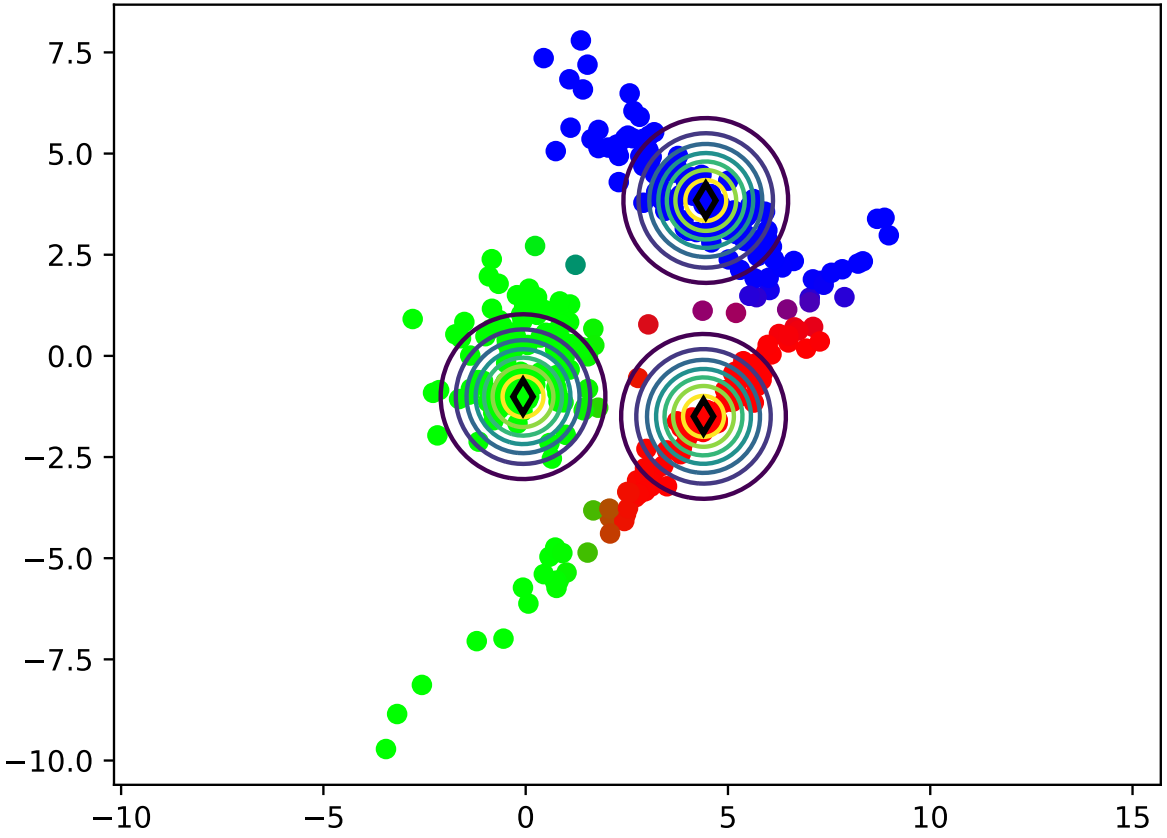
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



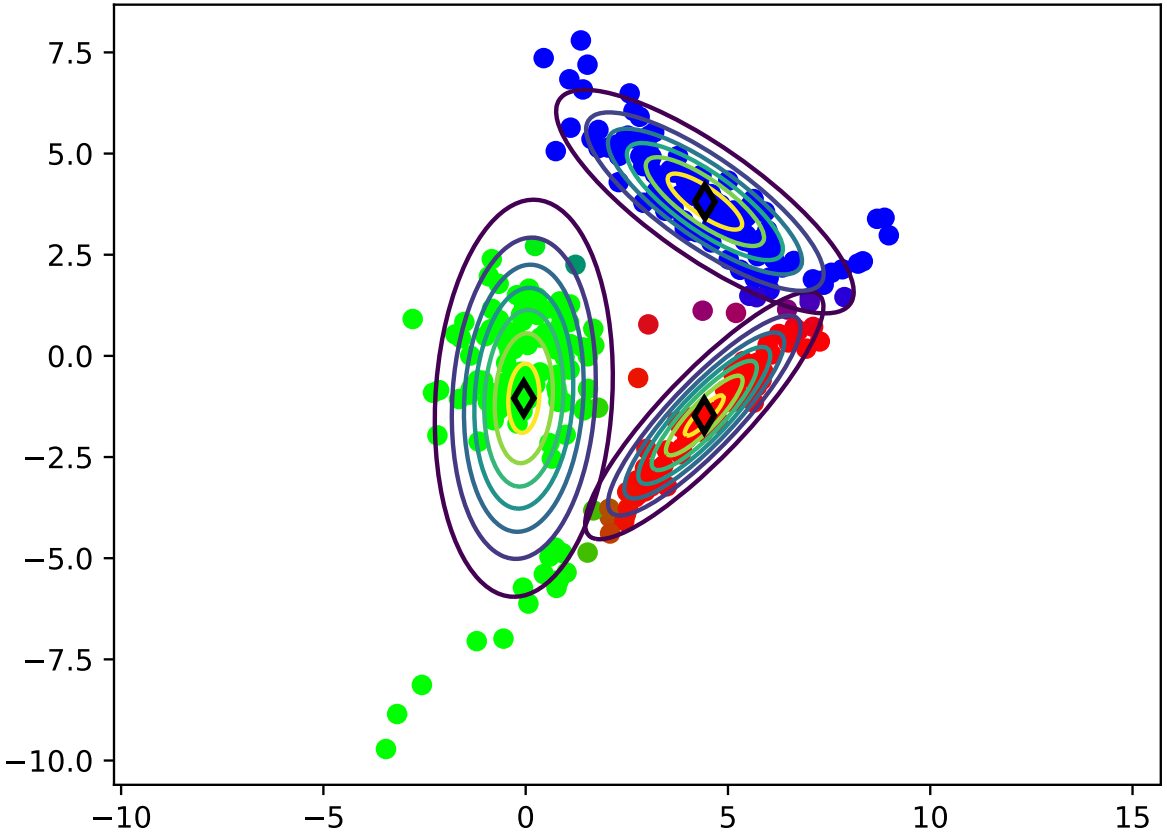
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



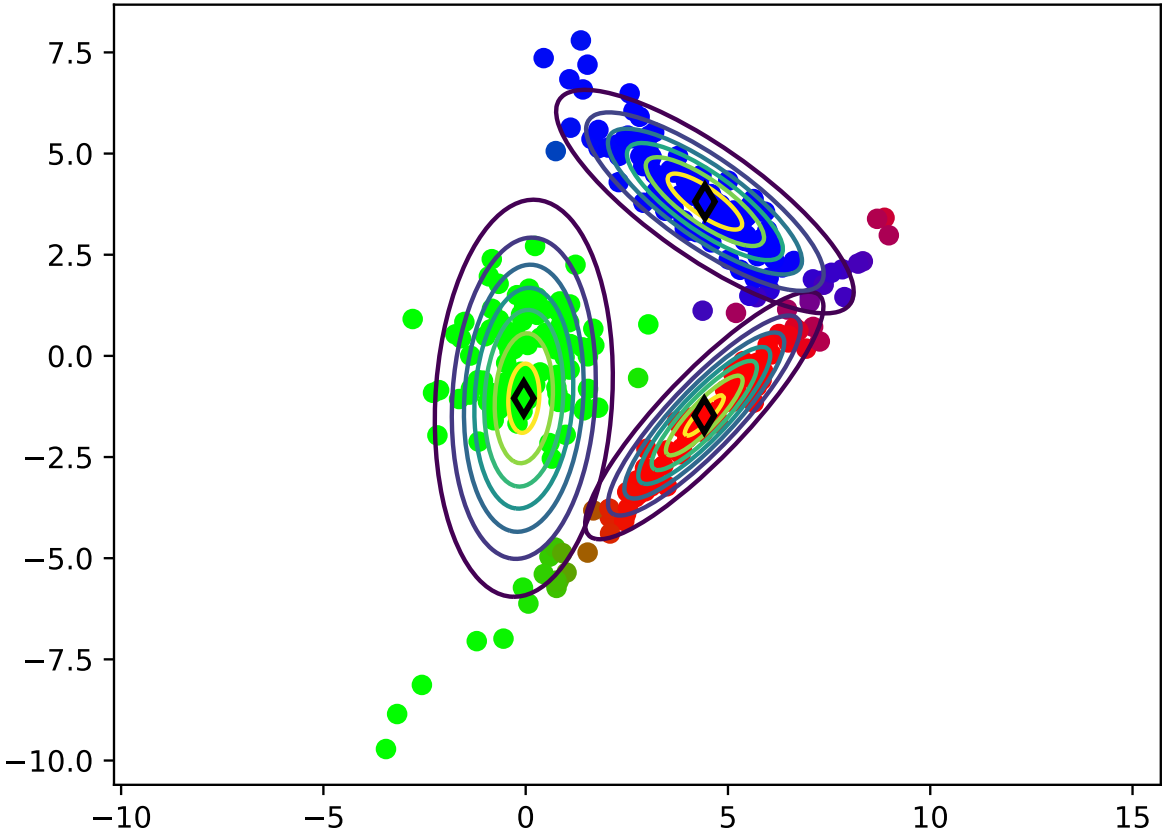
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



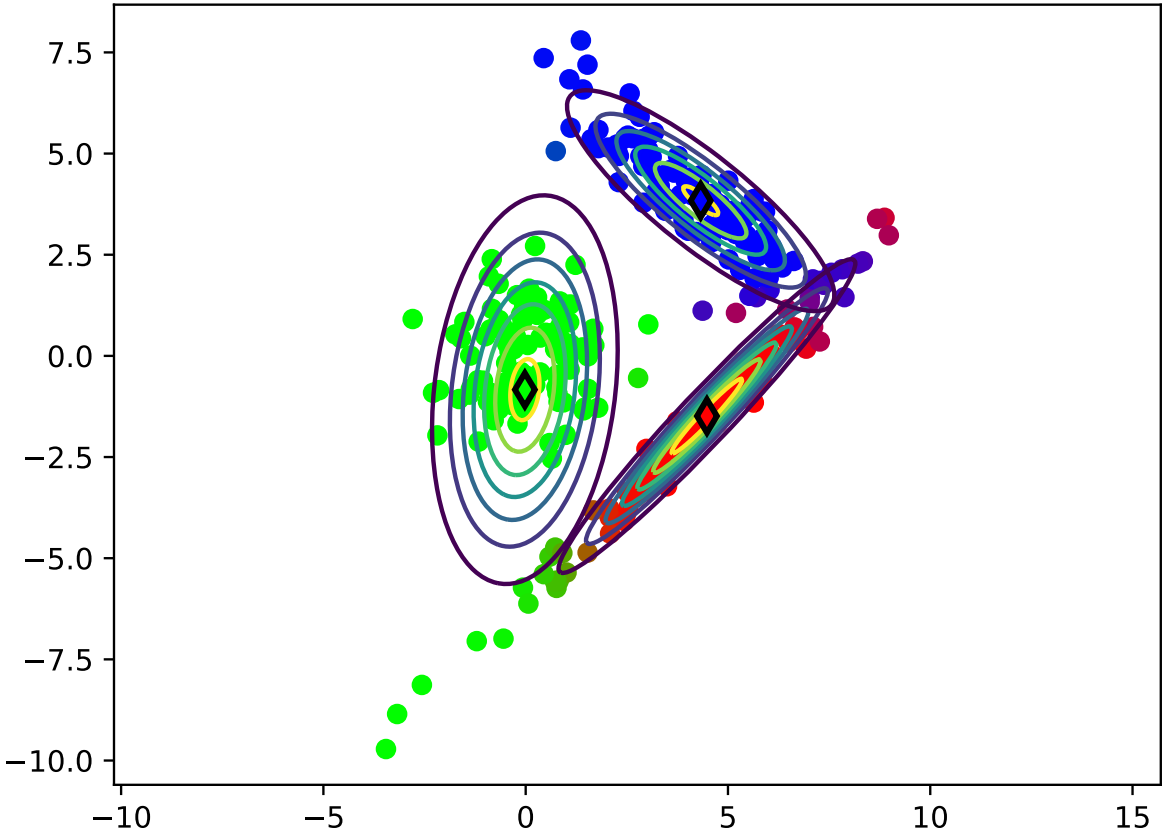
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



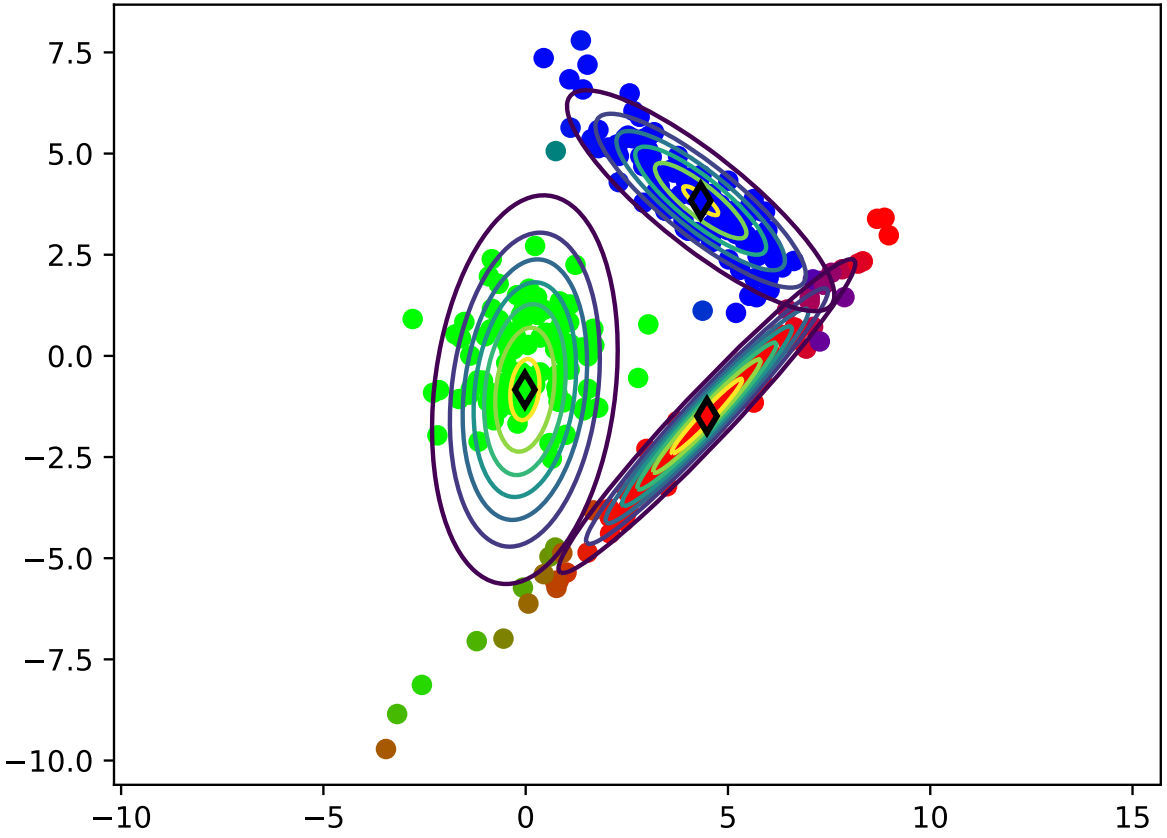
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



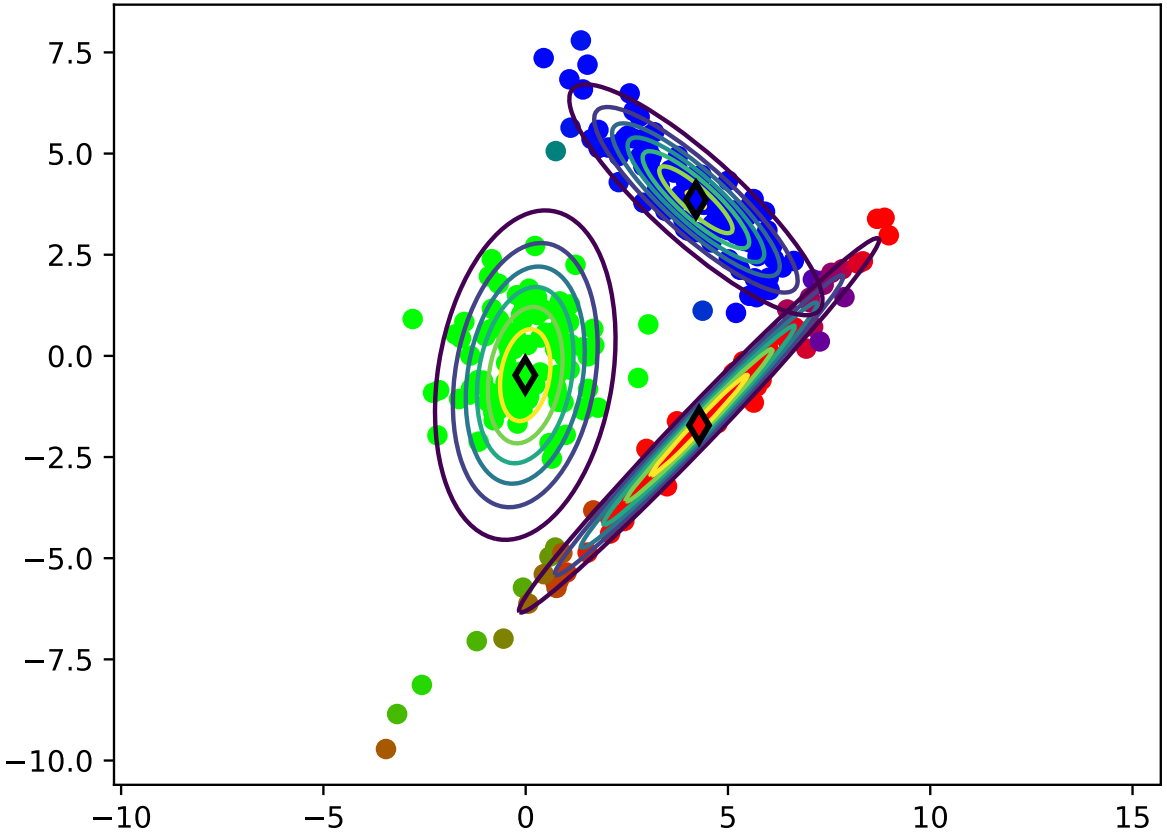
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



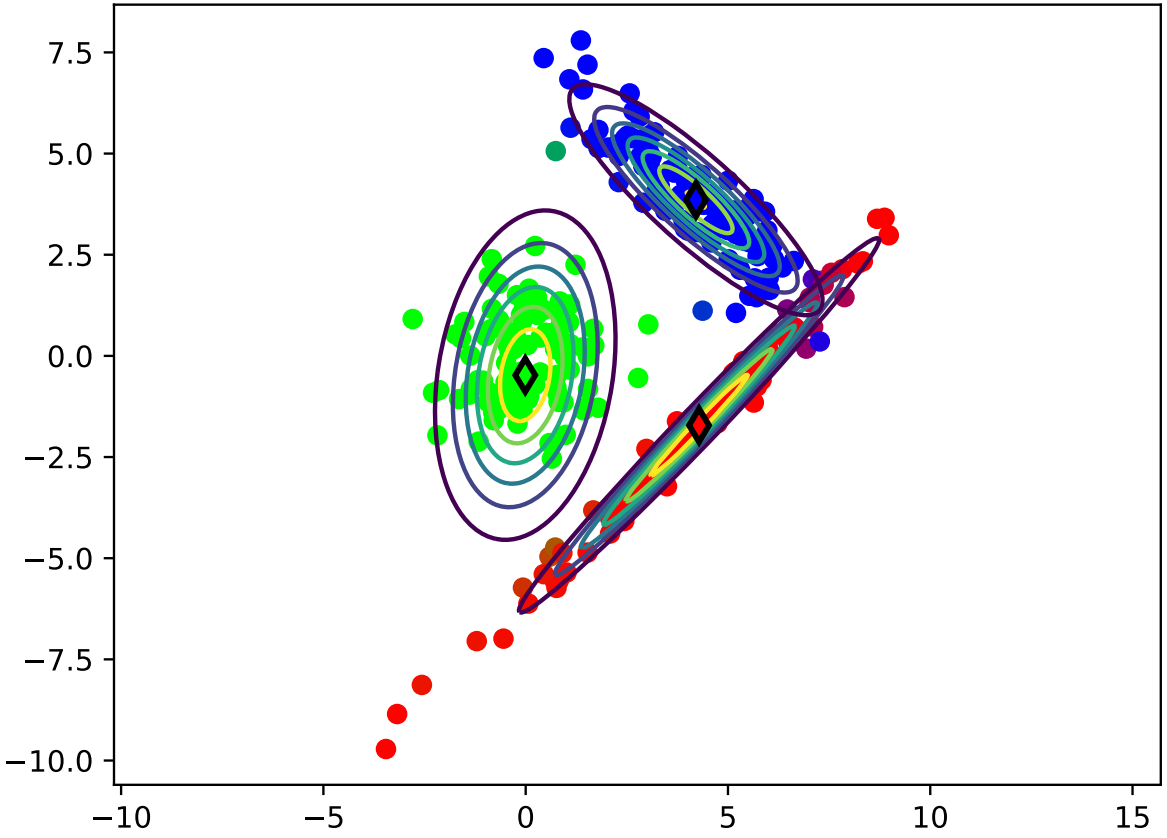
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



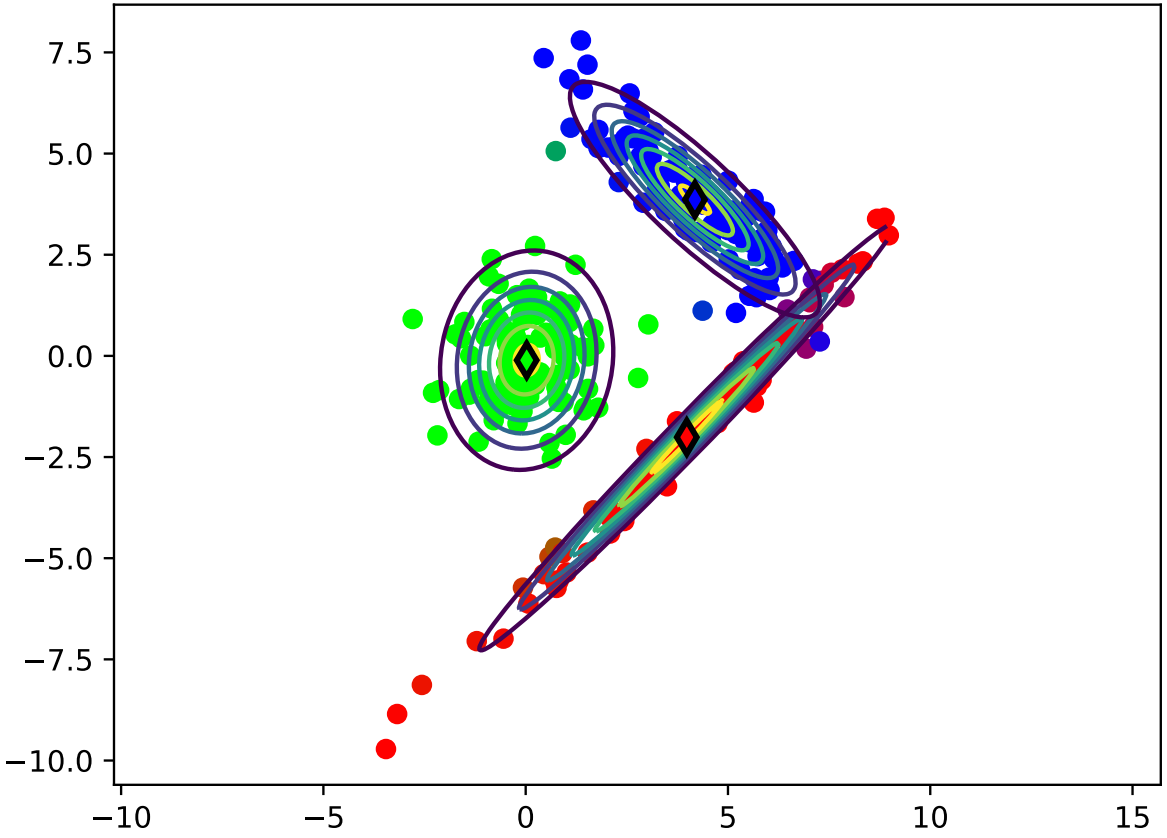
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



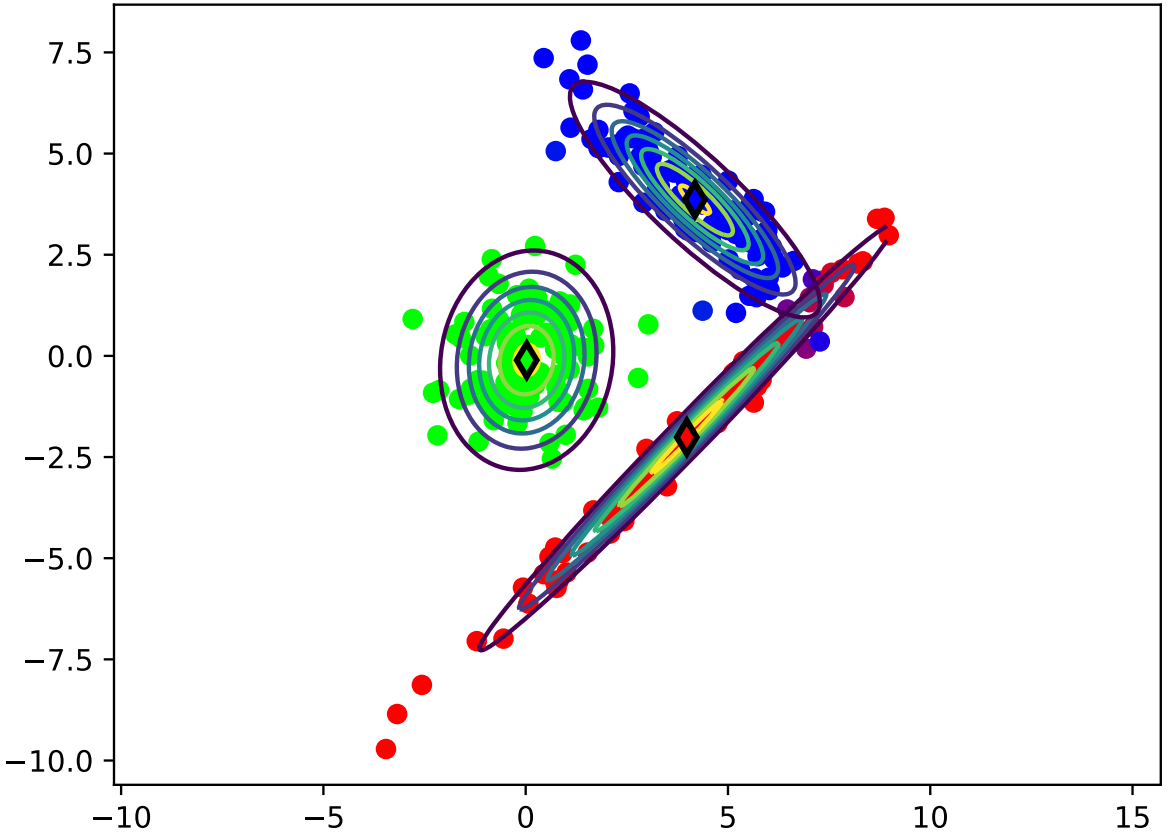
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



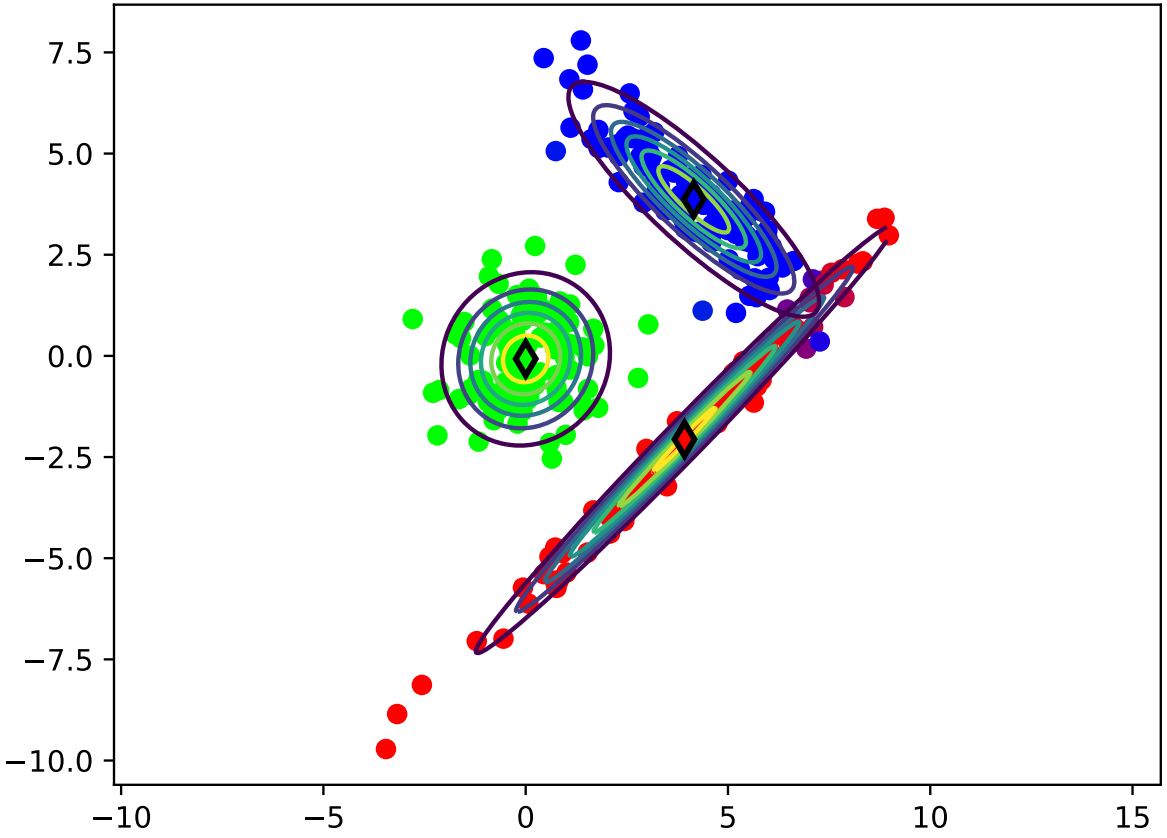
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



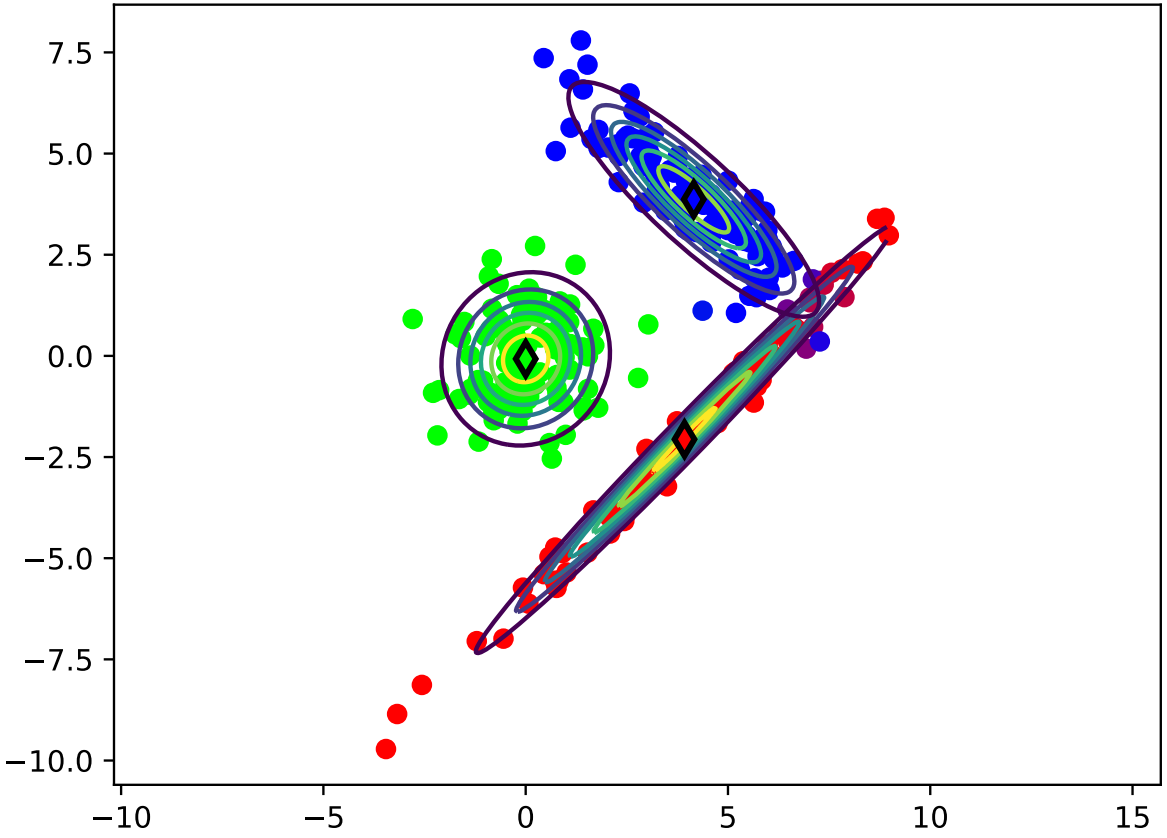
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



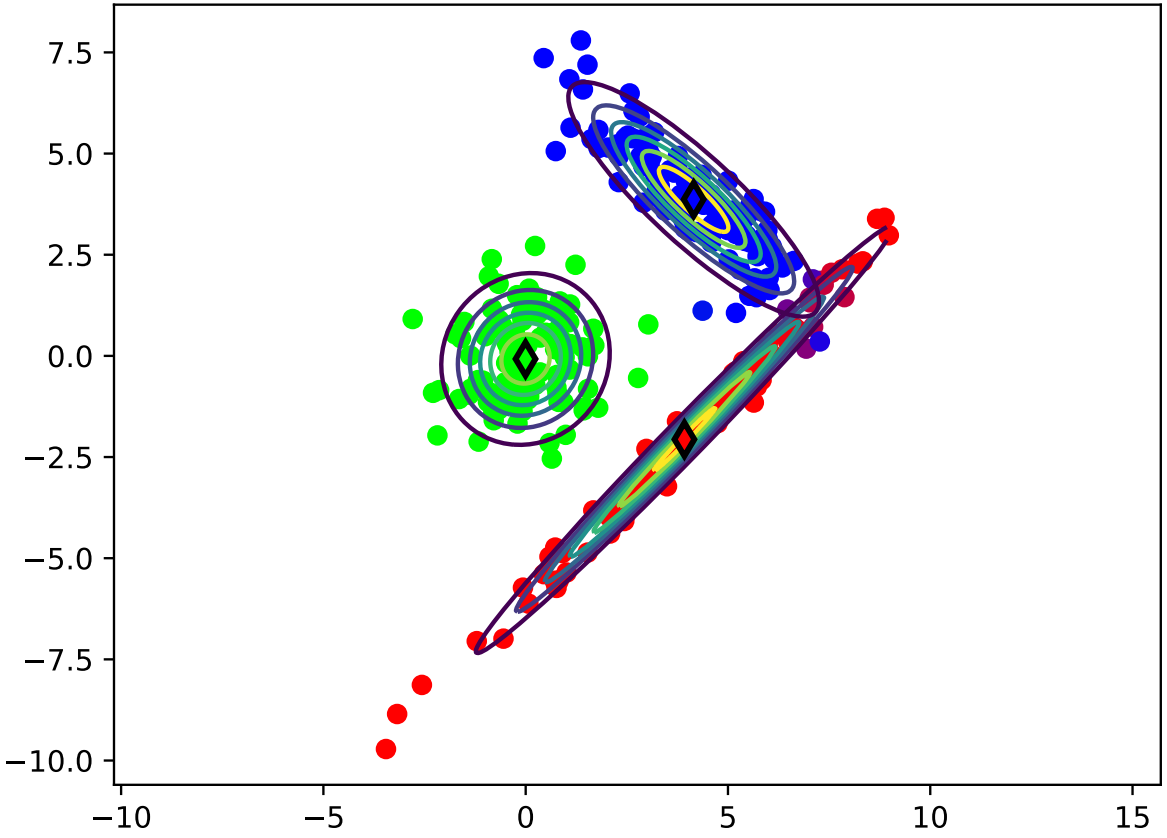
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



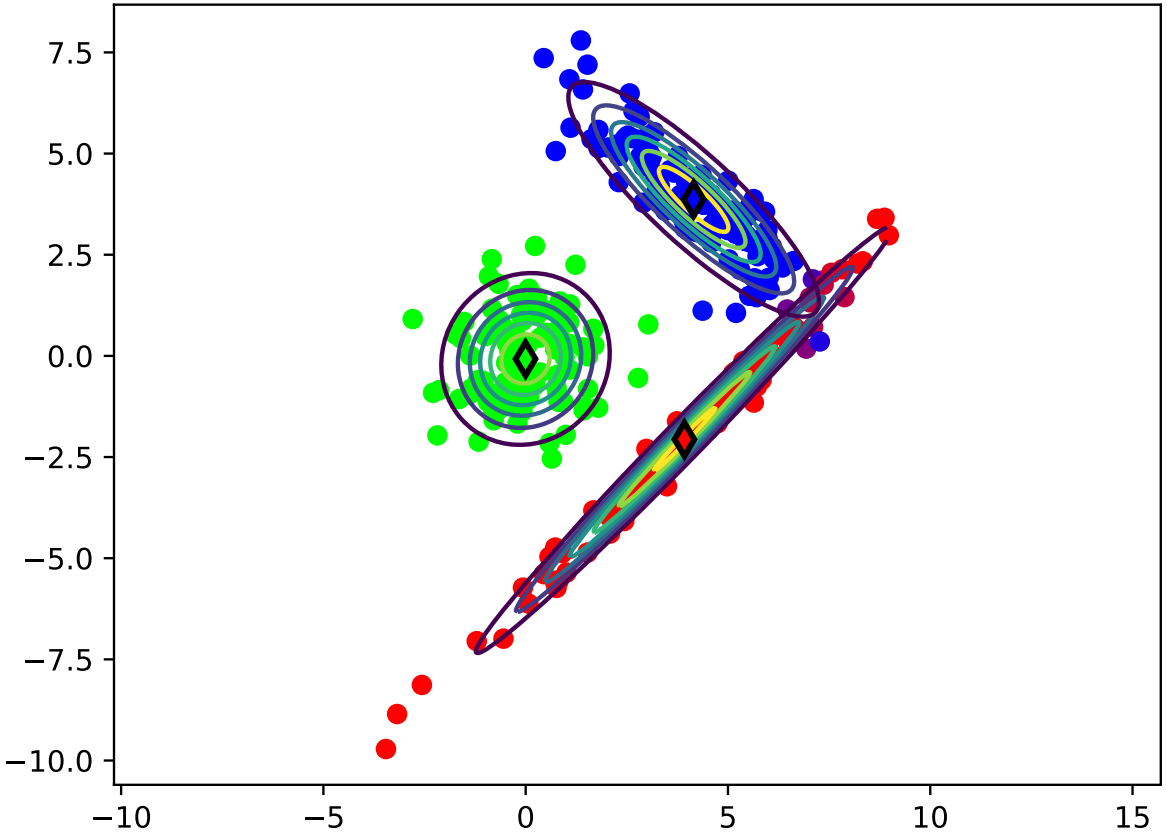
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



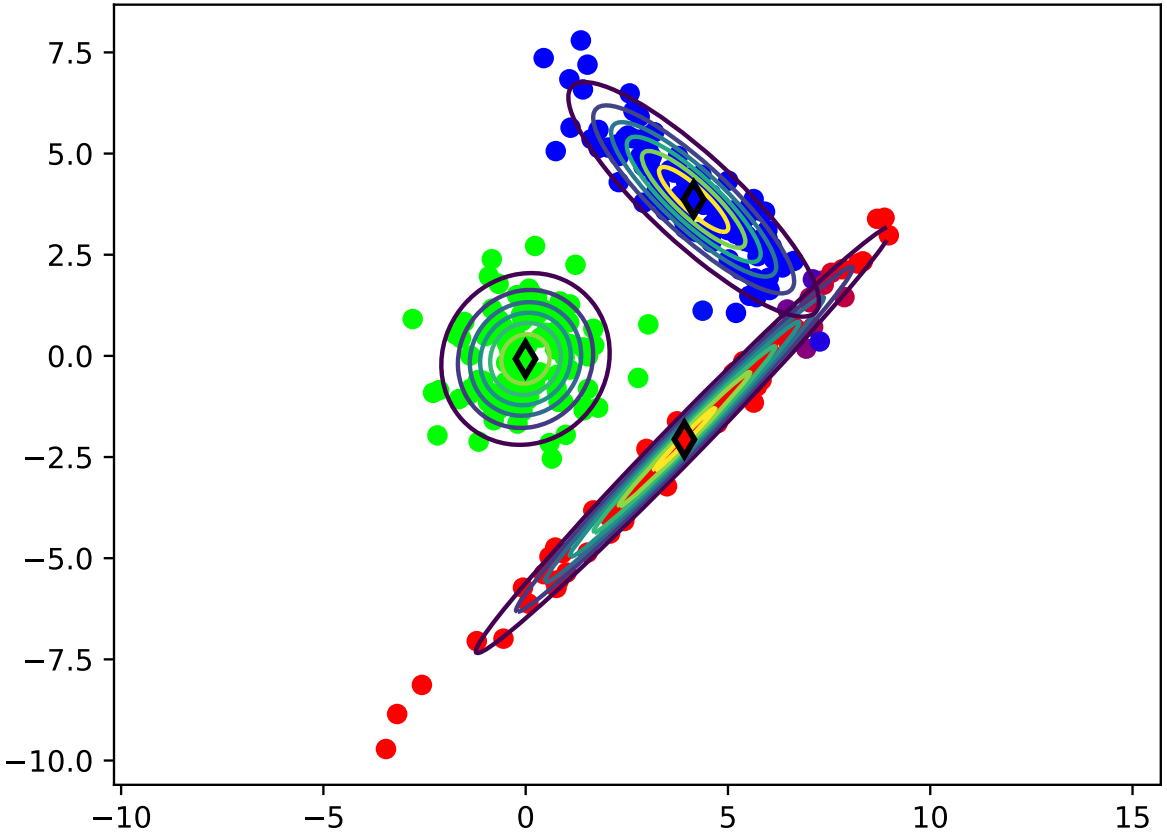
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



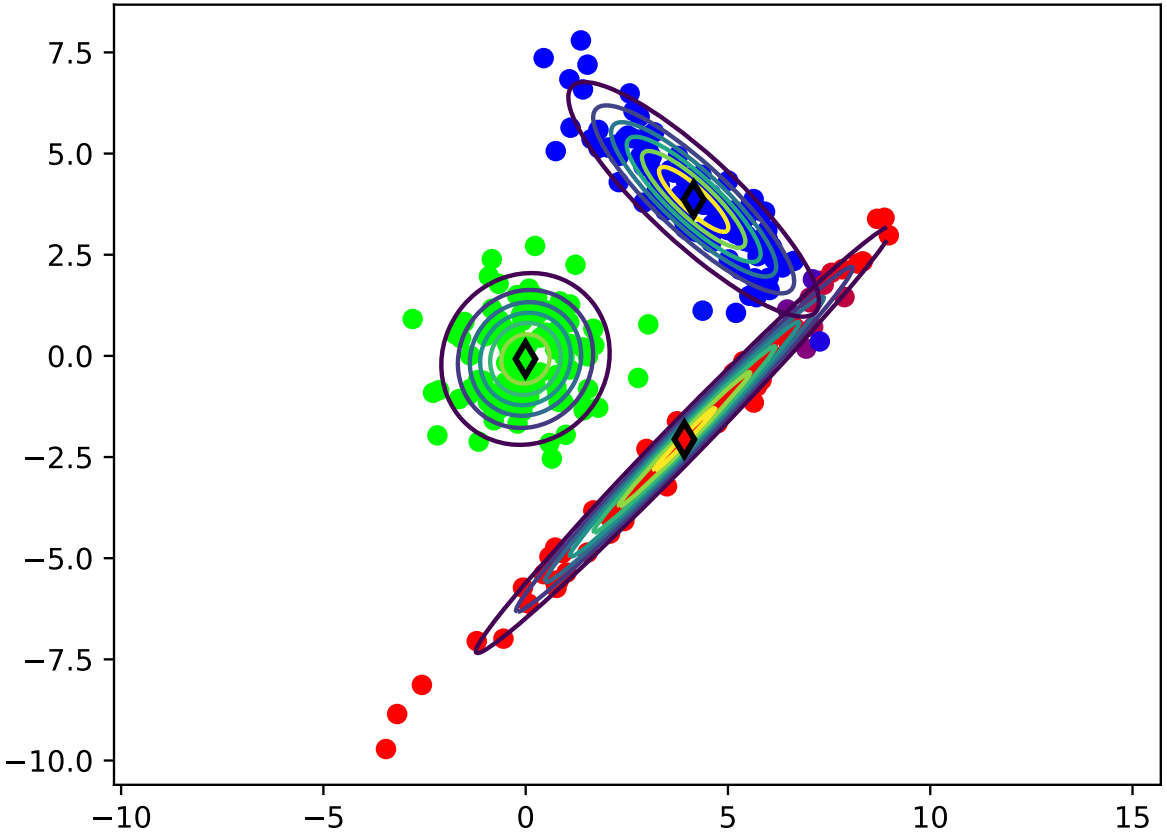
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



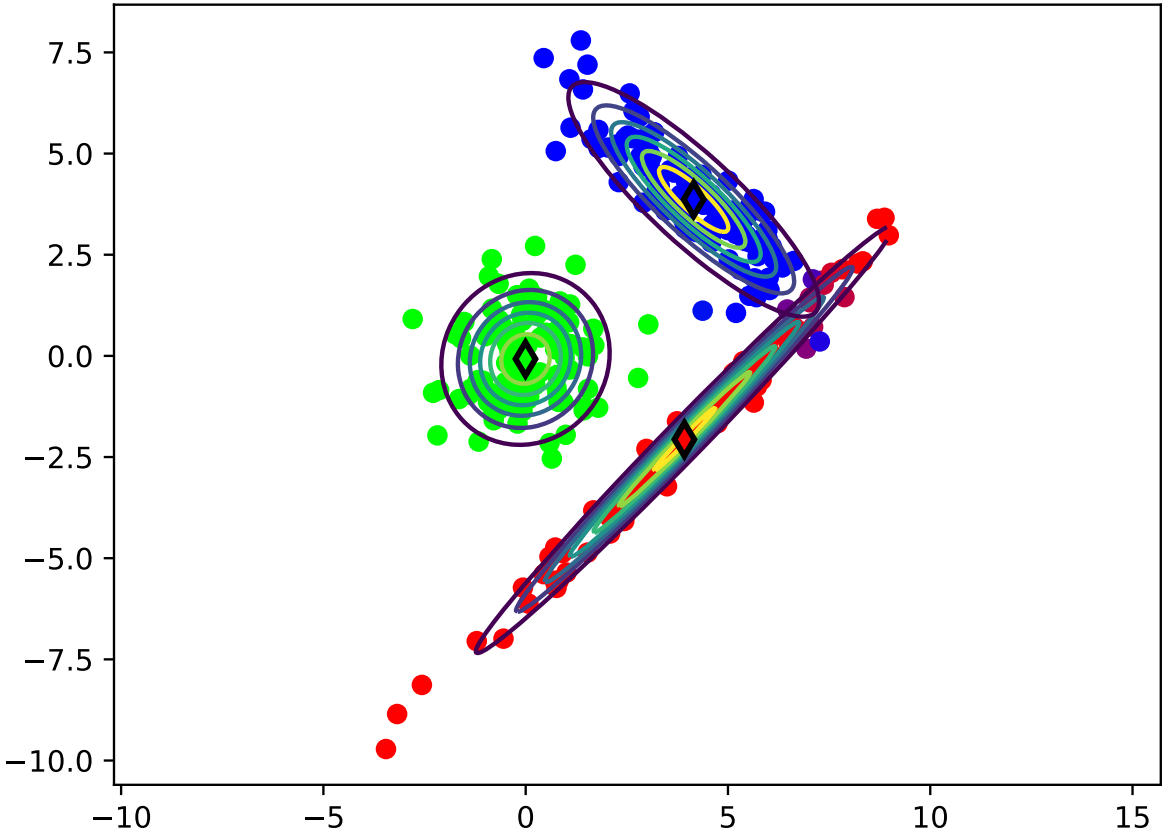
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



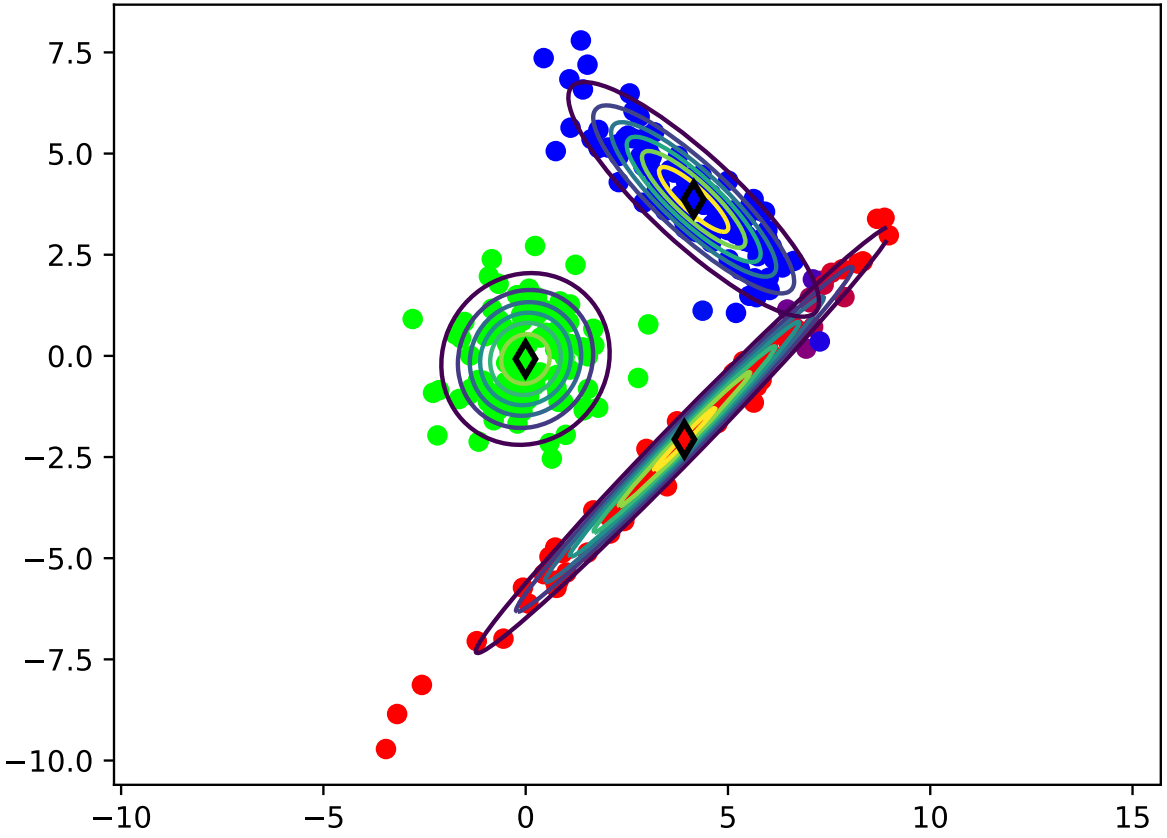
Demo: elliptical clusters

E... M... E... M... E... M... E... M...



Demo: elliptical clusters

E... M... E... M... E... M... E... M...



Theorem.

Suppose $(\boldsymbol{R}_0, \boldsymbol{\theta}_0) \in \mathcal{R}_{n,k} \times \boldsymbol{\Theta}$ arbitrary, thereafter $(\boldsymbol{R}_t, \boldsymbol{\theta}_t)$ given by E-M:

$$(\boldsymbol{R}_t)_{ij} := p_{\boldsymbol{\theta}_{t-1}}(y = j | \boldsymbol{x}_i). \qquad \text{and} \qquad \boldsymbol{\theta}_t := \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R}_t)$$

Then

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_t) &\leq \max_{\boldsymbol{R} \in \mathcal{R}_{n \times k}} \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}) = \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \mathcal{L}(\boldsymbol{\theta}_t) \\ &\leq \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R}_{t+1}). \end{aligned}$$

In particular, $\mathcal{L}(\boldsymbol{\theta}_t) \leq \mathcal{L}(\boldsymbol{\theta}_{t+1})$.

Theorem.
Suppose $(\boldsymbol{R}_0, \boldsymbol{\theta}_0) \in \mathcal{R}_{n,k} \times \boldsymbol{\Theta}$ arbitrary, thereafter $(\boldsymbol{R}_t, \boldsymbol{\theta}_t)$ given by E-M:

$$(\boldsymbol{R}_t)_{ij} := p_{\boldsymbol{\theta}_{t-1}}(y = j | \boldsymbol{x}_i). \qquad \text{and} \qquad \boldsymbol{\theta}_t := \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R}_t)$$

Then

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_t) &\leq \max_{\boldsymbol{R} \in \mathcal{R}_{n \times k}} \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}) = \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \mathcal{L}(\boldsymbol{\theta}_t) \\ &\leq \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R}_{t+1}). \end{aligned}$$

In particular, $\mathcal{L}(\boldsymbol{\theta}_t) \leq \mathcal{L}(\boldsymbol{\theta}_{t+1})$.

Remarks.

- We proved a similar guarantee for k-means, which is also an *alternating minimization* scheme.
- Similarly, MLE for Gaussian mixtures is NP-hard; it is also known to need exponentially many samples in k to information-theoretically recover the parameters.

Proof. We've already shown:

- $\mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \mathcal{L}(\boldsymbol{\theta}_t)$;
- $\mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R}_{t+1}) \leq \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R}_{t+1})$ by definition of $\boldsymbol{\theta}_{t+1}$.

We still need to show: $\mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \max_{\boldsymbol{R} \in \mathcal{R}_{n,k}} \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R})$.
We'll give two proofs.

Proof. We've already shown:

- $\mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \mathcal{L}(\boldsymbol{\theta}_t)$;
- $\mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R}_{t+1}) \leq \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R}_{t+1})$ by definition of $\boldsymbol{\theta}_{t+1}$.

We still need to show: $\mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) = \max_{\boldsymbol{R} \in \mathcal{R}_{n,k}} \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R})$.
We'll give two proofs.

By concavity of $\ln$ ("Jensen's inequality" in convexity lectures), for any $\boldsymbol{R} \in \mathcal{R}_{n,k}$,

$$\begin{aligned}
\mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}) &= \sum_{i=1}^{n} \sum_{j=1}^{k} \boldsymbol{R}_{ij} \ln \frac{p_{\boldsymbol{\theta}_t}(\boldsymbol{x}_i, y_i = j)}{\boldsymbol{R}_{ij}} \\
&\leq \sum_{i=1}^{n} \ln \left(\sum_{j=1}^{k} \boldsymbol{R}_{ij} \frac{p_{\boldsymbol{\theta}_t}(\boldsymbol{x}_i, y_i = j)}{\boldsymbol{R}_{ij}} \right) \\
&= \sum_{i=1}^{n} \ln p_{\boldsymbol{\theta}_t}(\boldsymbol{x}_i) = \mathcal{L}(\boldsymbol{\theta}_t) = \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}).
\end{aligned}$$

Since $\boldsymbol{R}$ was arbitrary, $\max_{\boldsymbol{R} \in \mathcal{R}} \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}) = \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1})$.

Proof (continued). Here's a second proof of that missing fact.
To evaluate $\arg\max_{\boldsymbol{R}\in\mathcal{R}_{n,k}} \mathcal{L}(\boldsymbol{\theta};\boldsymbol{R})$, consider Lagrangian

$$\sum_{i=1}^{n}\left(\sum_{j=1}^{k} R_{ij}\ln p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, y=j) - \sum_{j=1}^{k} R_{ij}\ln R_{ij} + \lambda_i\left(\sum_{j=1}^{k} R_{ij} - 1\right)\right).$$

Fixing i and taking the gradient with respect to R_{ij} for any j,

$$0 = \ln p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, y_i = j) - \ln R_{ij} - 1 + \lambda_i,$$

giving $R_{ij} = p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, y=j)\exp(\lambda_i - 1)$. Since moreover

$$1 = \sum_j R_{ij} = \exp(\lambda_i - 1)\sum_j p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, y=j) = \exp(\lambda_i - 1)p_{\boldsymbol{\theta}}(\boldsymbol{x}_i),$$

it follows that $\exp(\lambda_i - 1) = {}^{1}/_{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)}$,
and the optimal $\boldsymbol{R}$ satisfies $R_{ij} = {}^{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i, y=j)}/_{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)} = p_{\boldsymbol{\theta}}(y=j|\boldsymbol{x}_i)$.

□

Related issues.

Parameter constraints.

E-M for GMMs **still works** if we freeze or constrain some parameters.

Parameter constraints.

E-M for GMMs **still works** if we freeze or constrain some parameters.

Examples:

- **No weights:** initialize $\boldsymbol{\pi} = (1/k, \ldots, 1/k)$ and never update it.
- **Diagonal covariance matrices:** update everything as before, except $\boldsymbol{\Sigma}_j := \mathrm{diag}((\boldsymbol{\sigma}_j)_1^2, \ldots, (\boldsymbol{\sigma}_j)_d^2)$ where

$$(\boldsymbol{\sigma}_j)_l^2 := \frac{\sum_{i=1}^n R_{ij}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)_l^2}{n\pi_j};$$

that is: we use coordinate-wise sample variances weighted by $\boldsymbol{R}$.
Why is this a good idea?

Parameter constraints.

E-M for GMMs **still works** if we freeze or constrain some parameters.

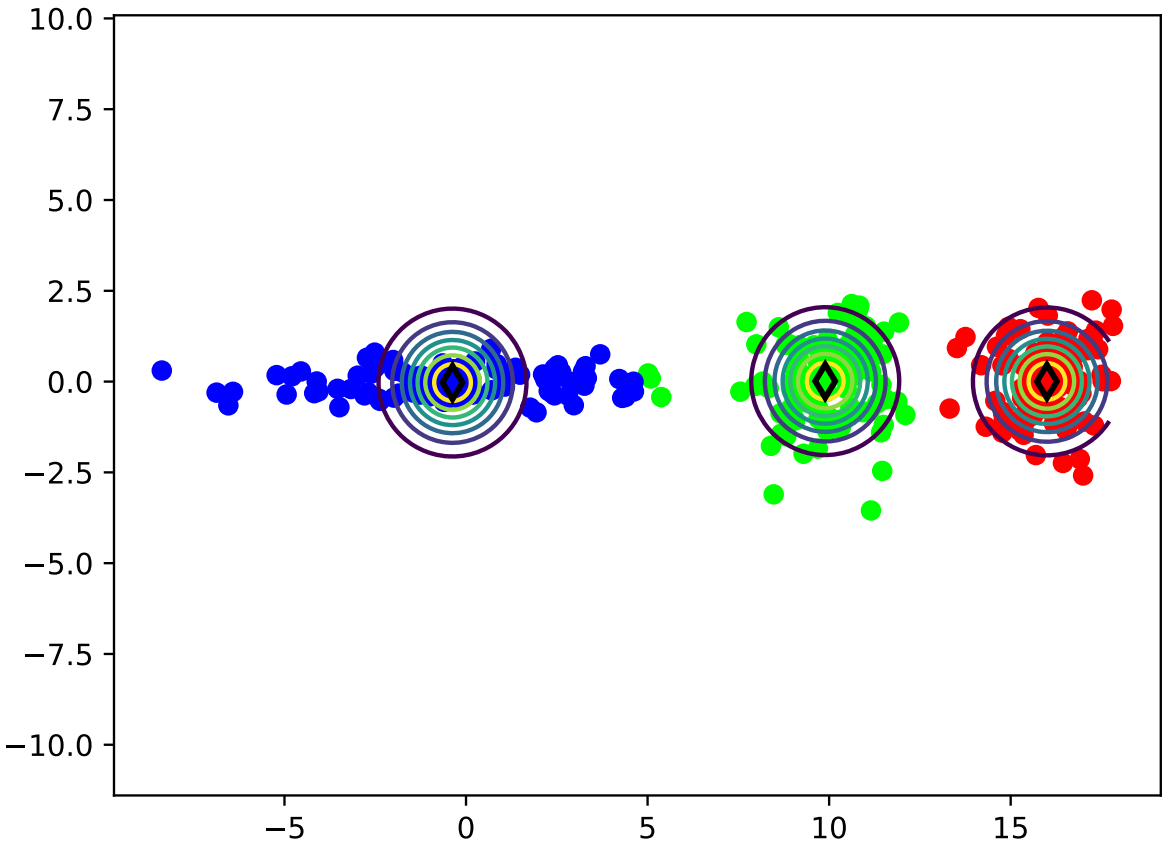
Examples:

- **No weights:** initialize $\boldsymbol{\pi} = (1/k, \ldots, 1/k)$ and never update it.
- **Diagonal covariance matrices:** update everything as before, except $\boldsymbol{\Sigma}_j := \mathrm{diag}((\boldsymbol{\sigma}_j)_1^2, \ldots, (\boldsymbol{\sigma}_j)_d^2)$ where

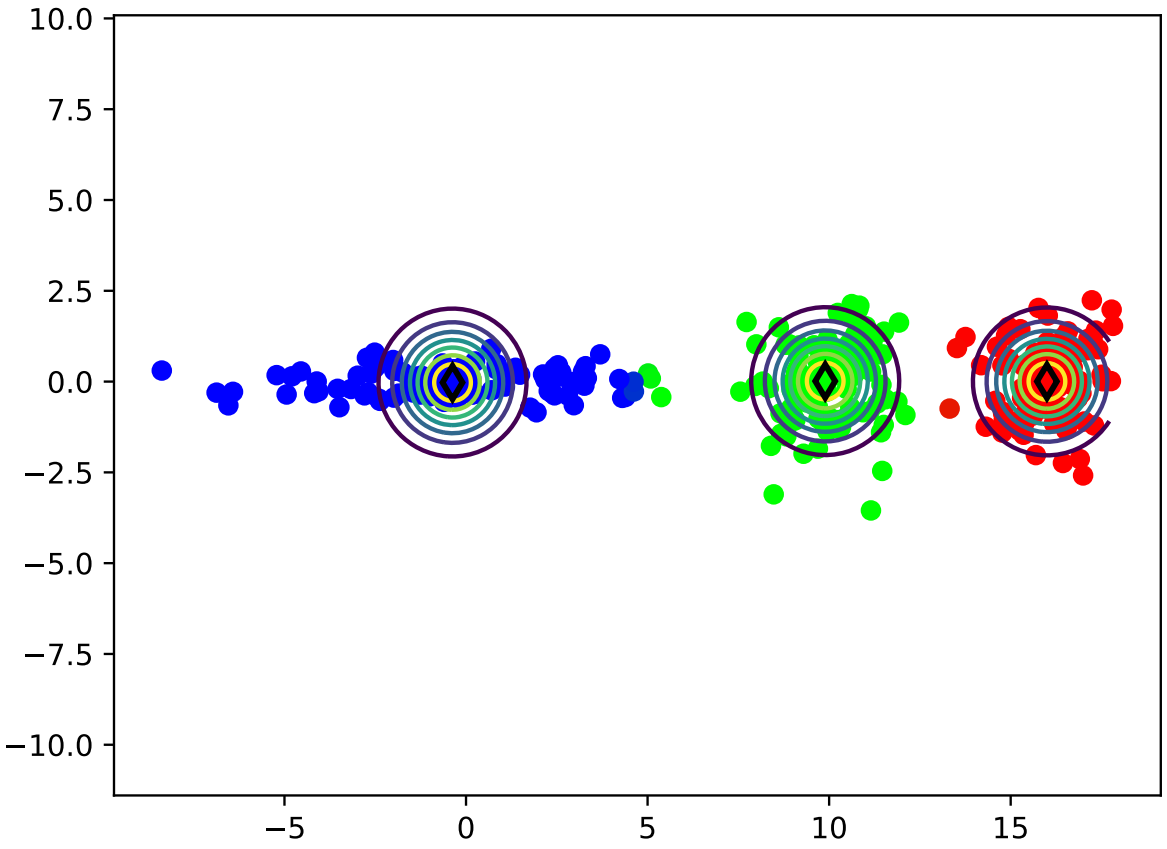
$$(\boldsymbol{\sigma}_j)_l^2 := \frac{\sum_{i=1}^n R_{ij}(\boldsymbol{x}_i - \boldsymbol{\mu}_j)_l^2}{n\pi_j};$$

that is: we use coordinate-wise sample variances weighted by $\boldsymbol{R}$.
Why is this a good idea?
Computation (of inverse), sample complexity, ...

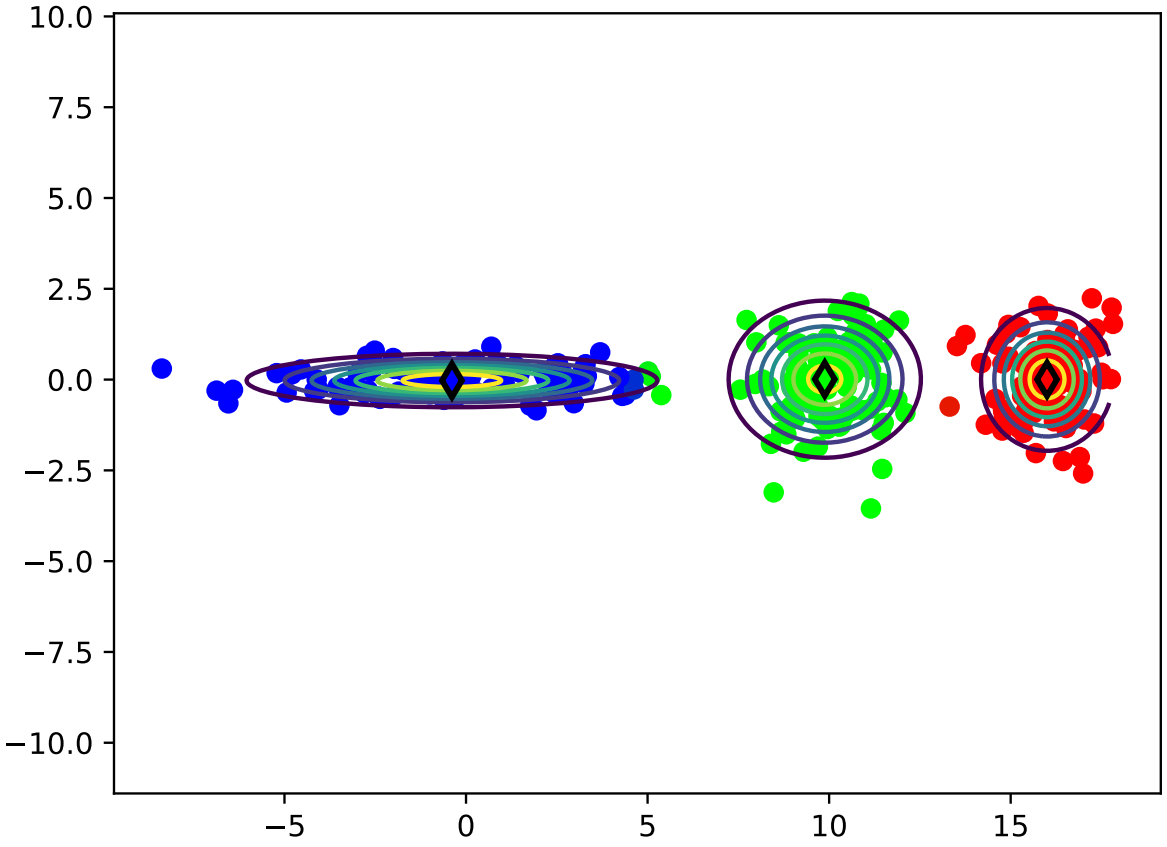
Gaussian Mixture Model with diagonal covariances.



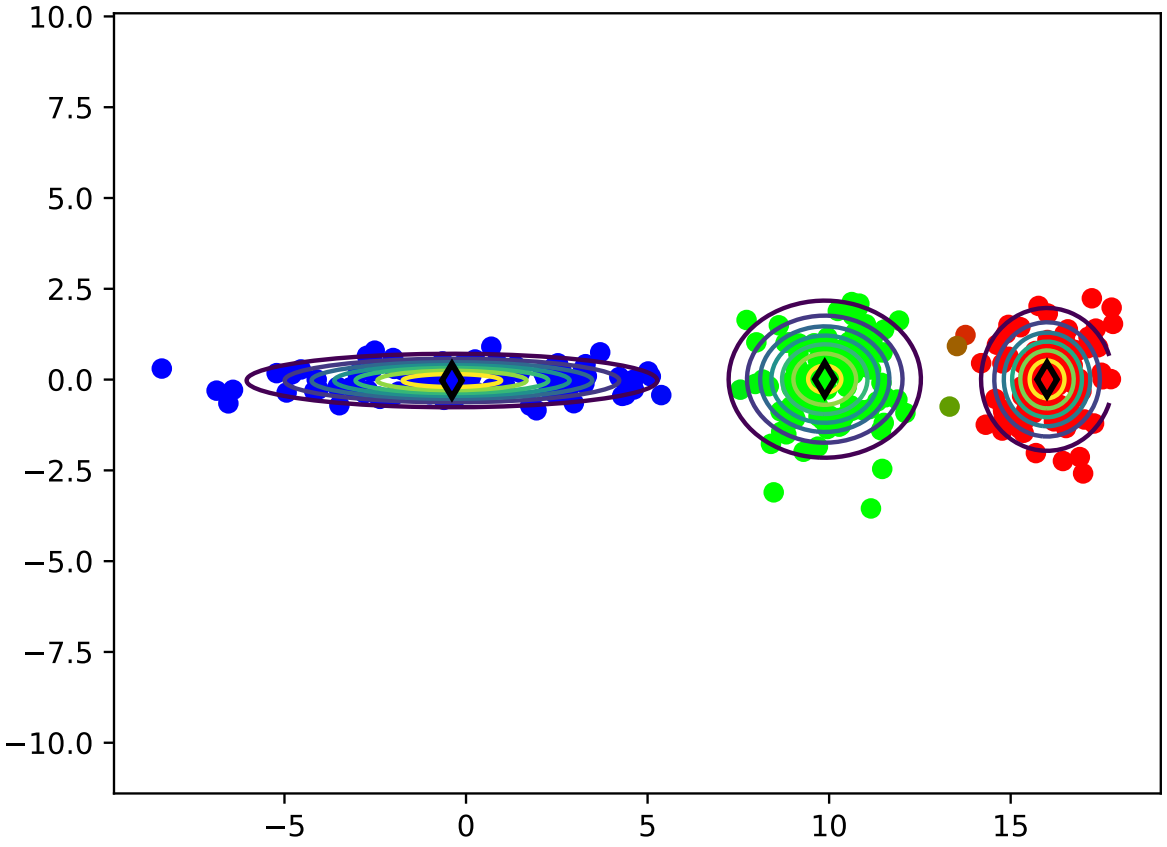
Gaussian Mixture Model with diagonal covariances.



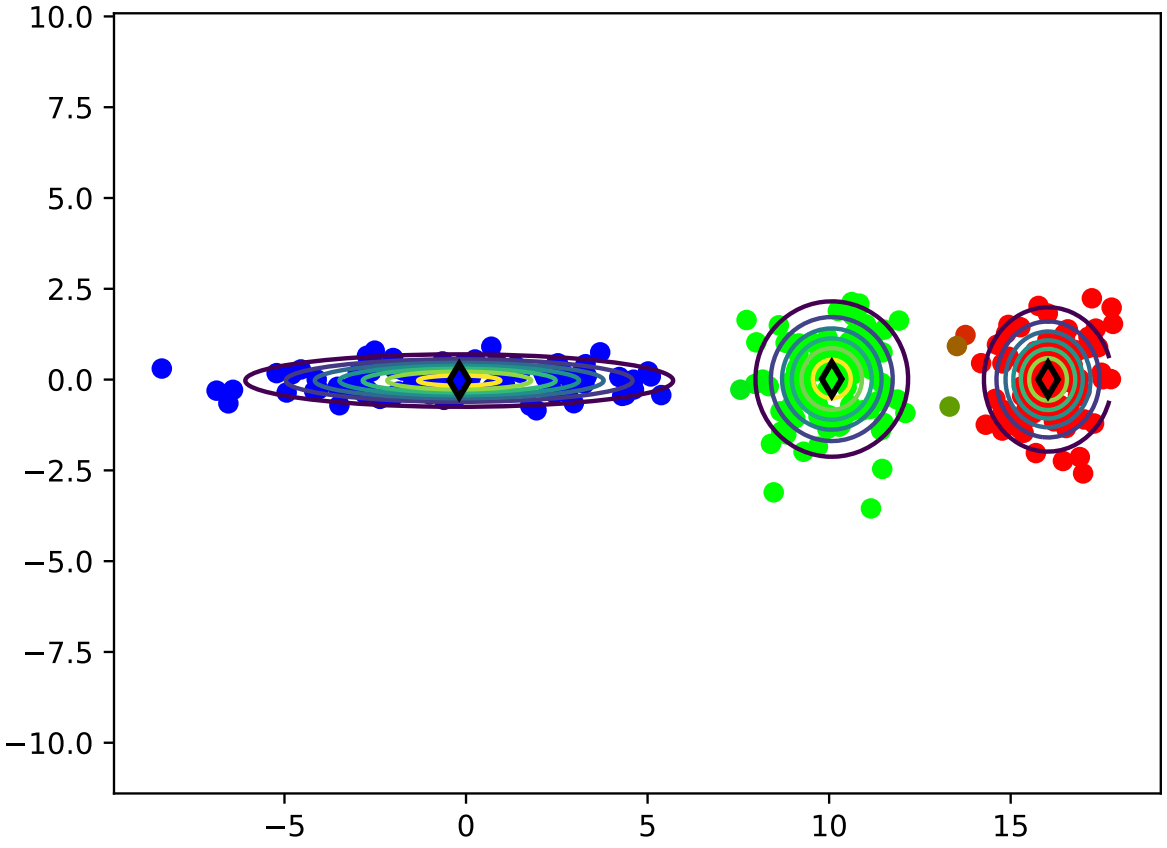
Gaussian Mixture Model with diagonal covariances.



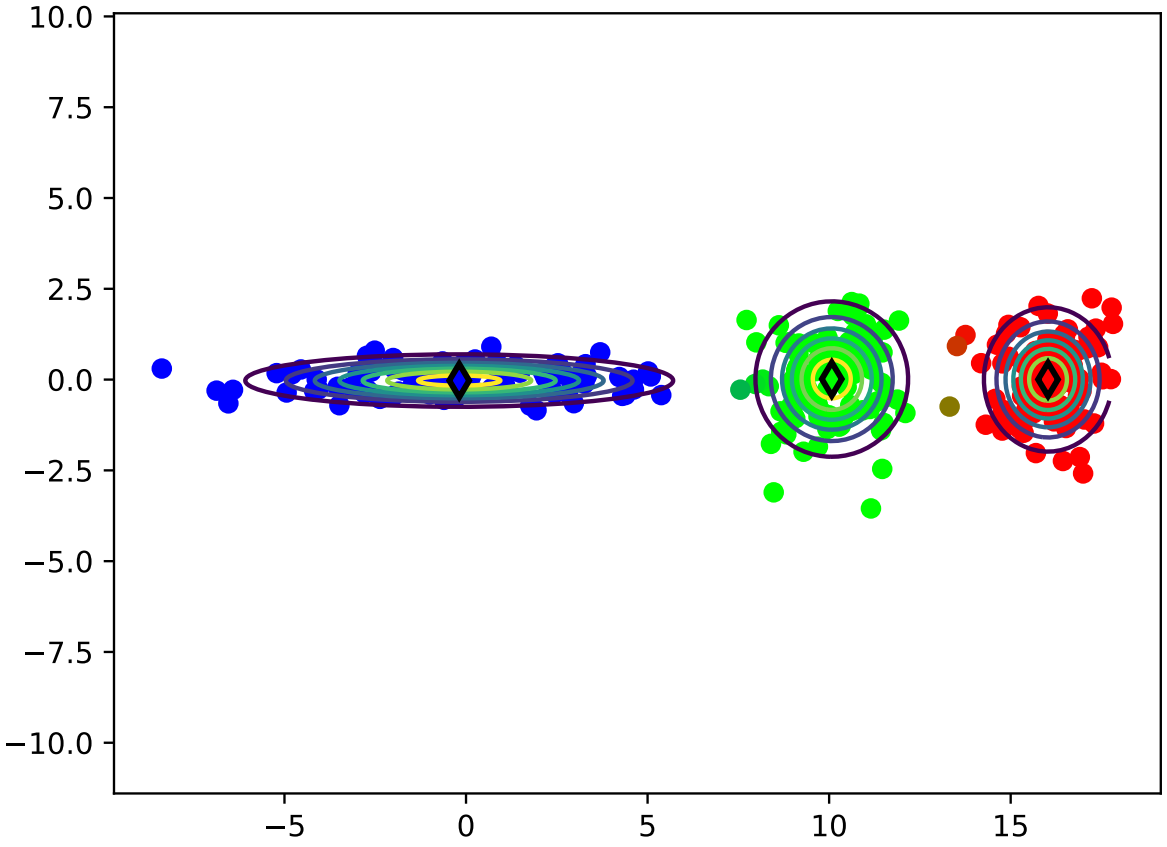
Gaussian Mixture Model with diagonal covariances.



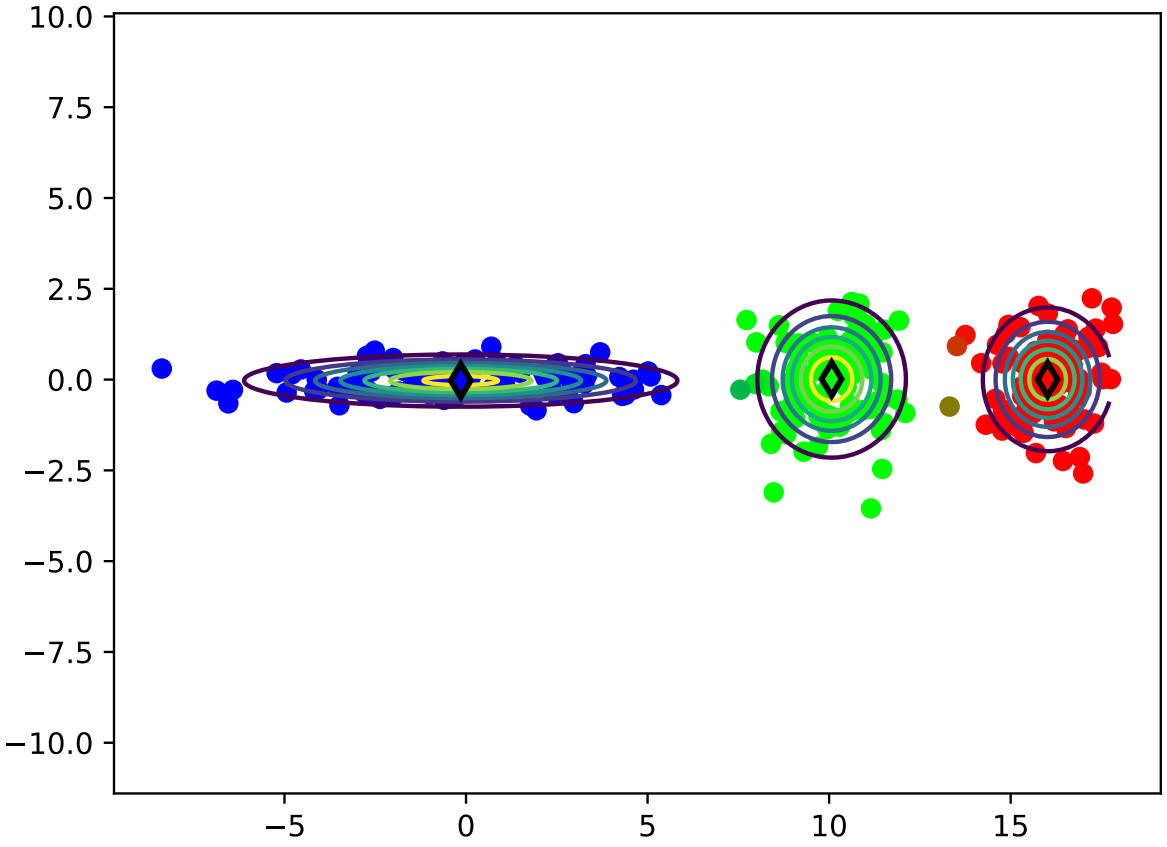
Gaussian Mixture Model with diagonal covariances.



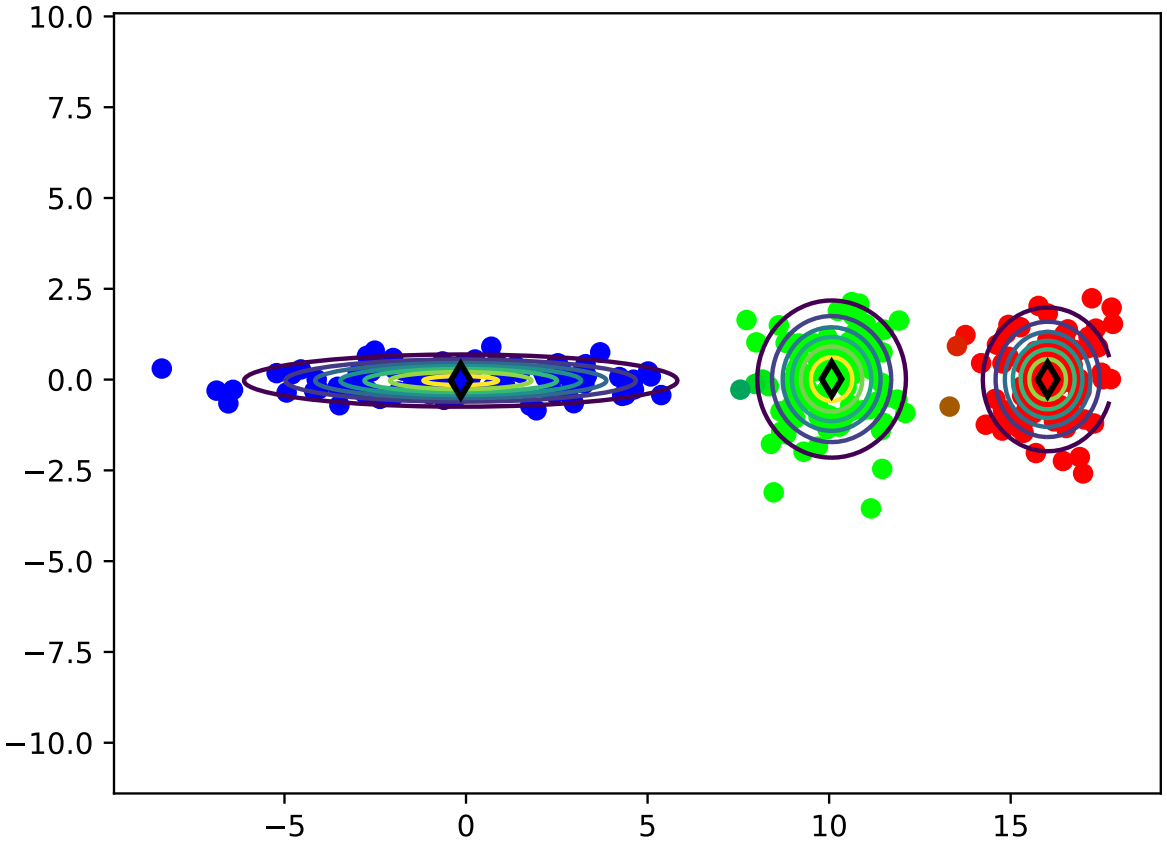
Gaussian Mixture Model with diagonal covariances.



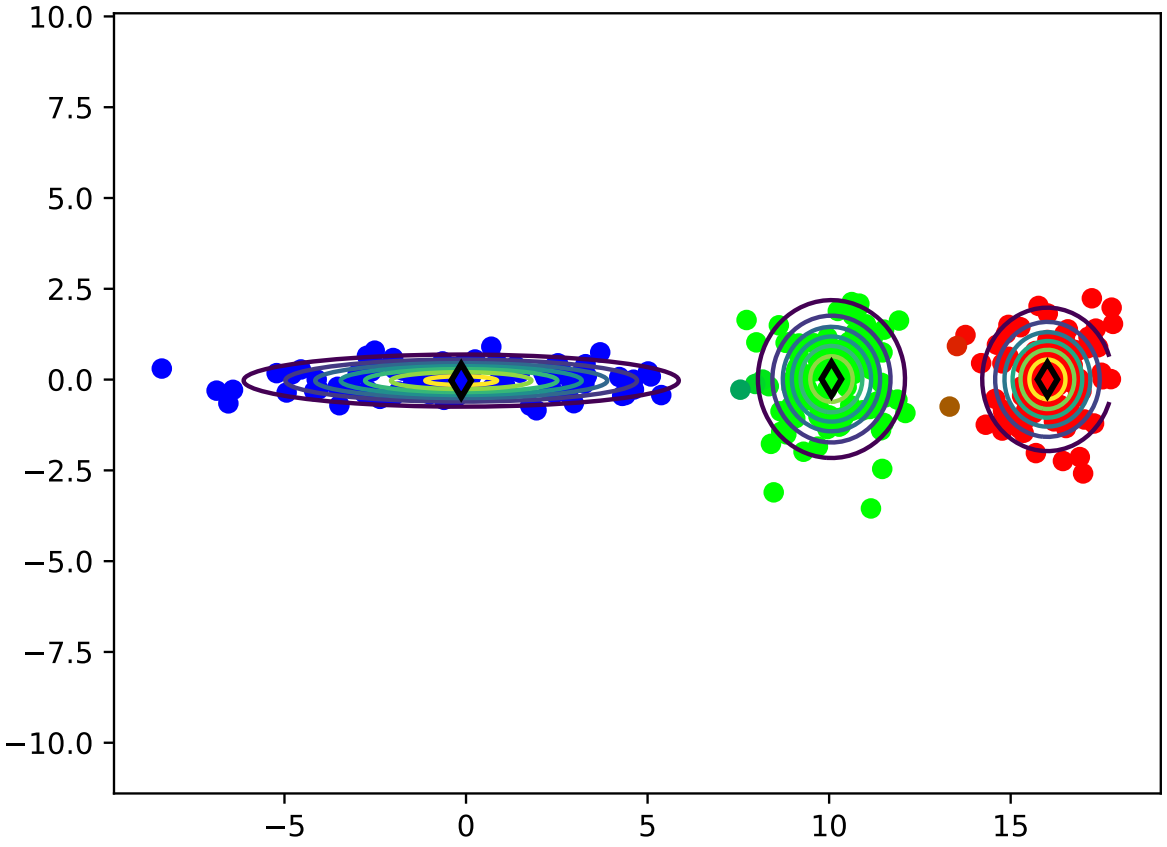
Gaussian Mixture Model with diagonal covariances.



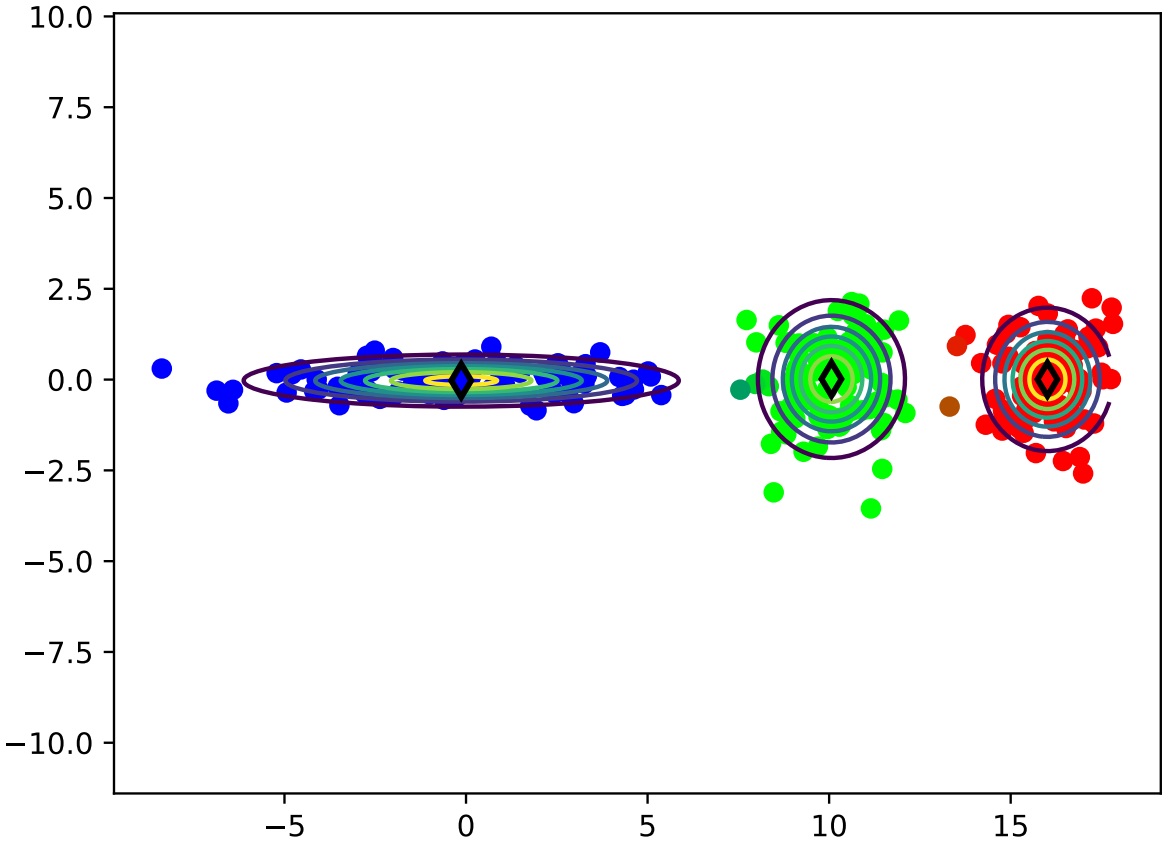
Gaussian Mixture Model with diagonal covariances.



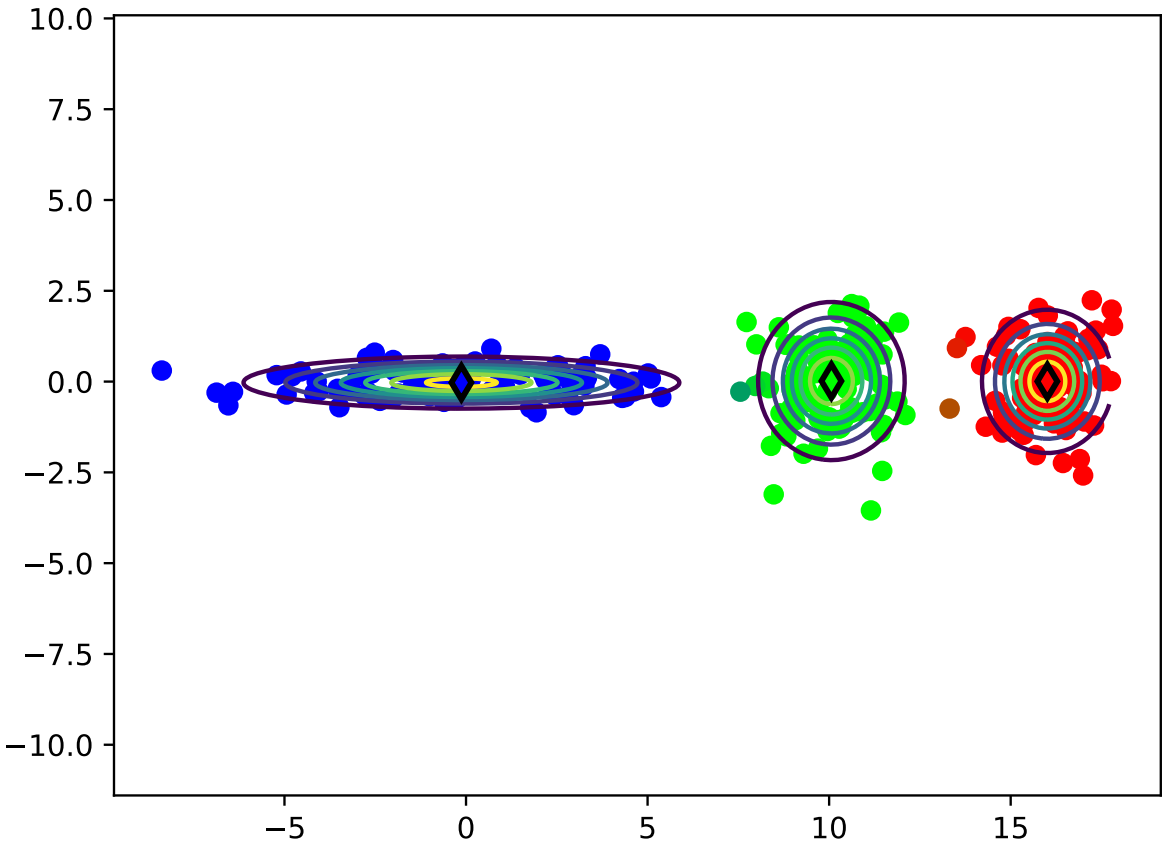
Gaussian Mixture Model with diagonal covariances.



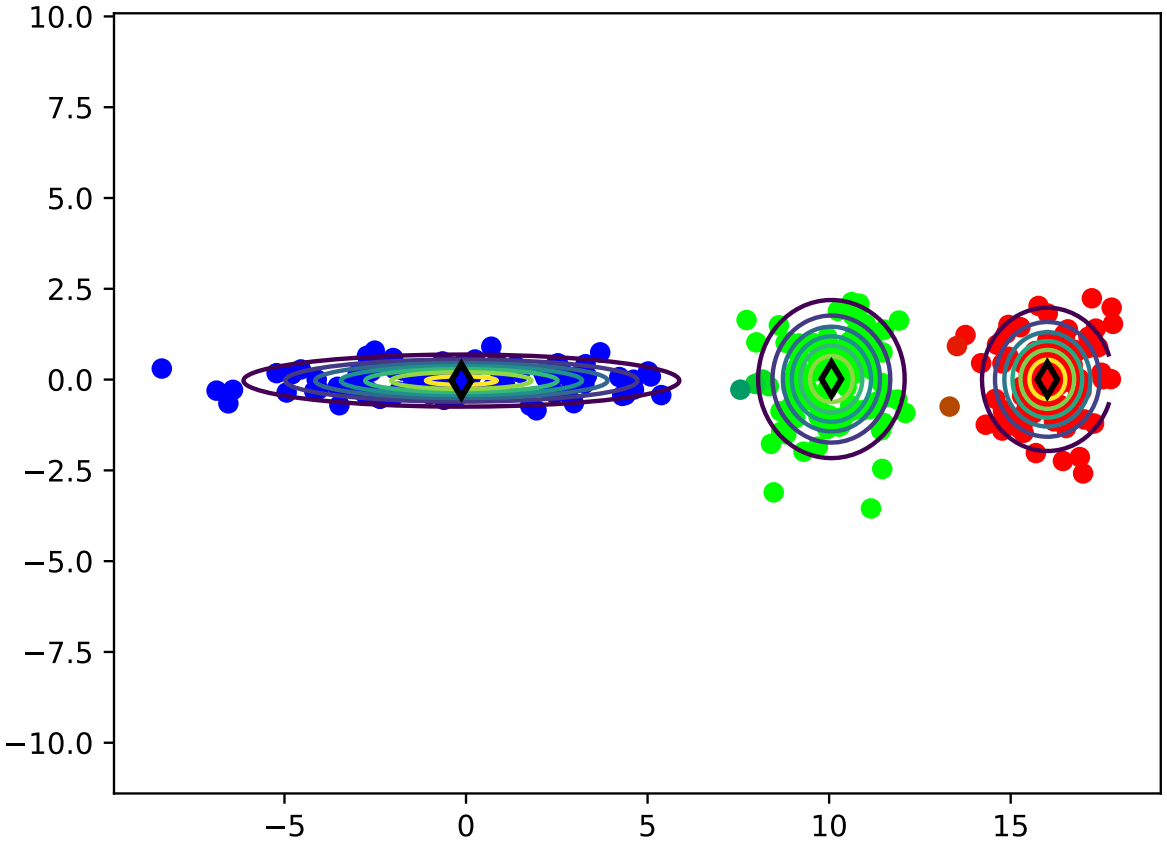
Gaussian Mixture Model with diagonal covariances.



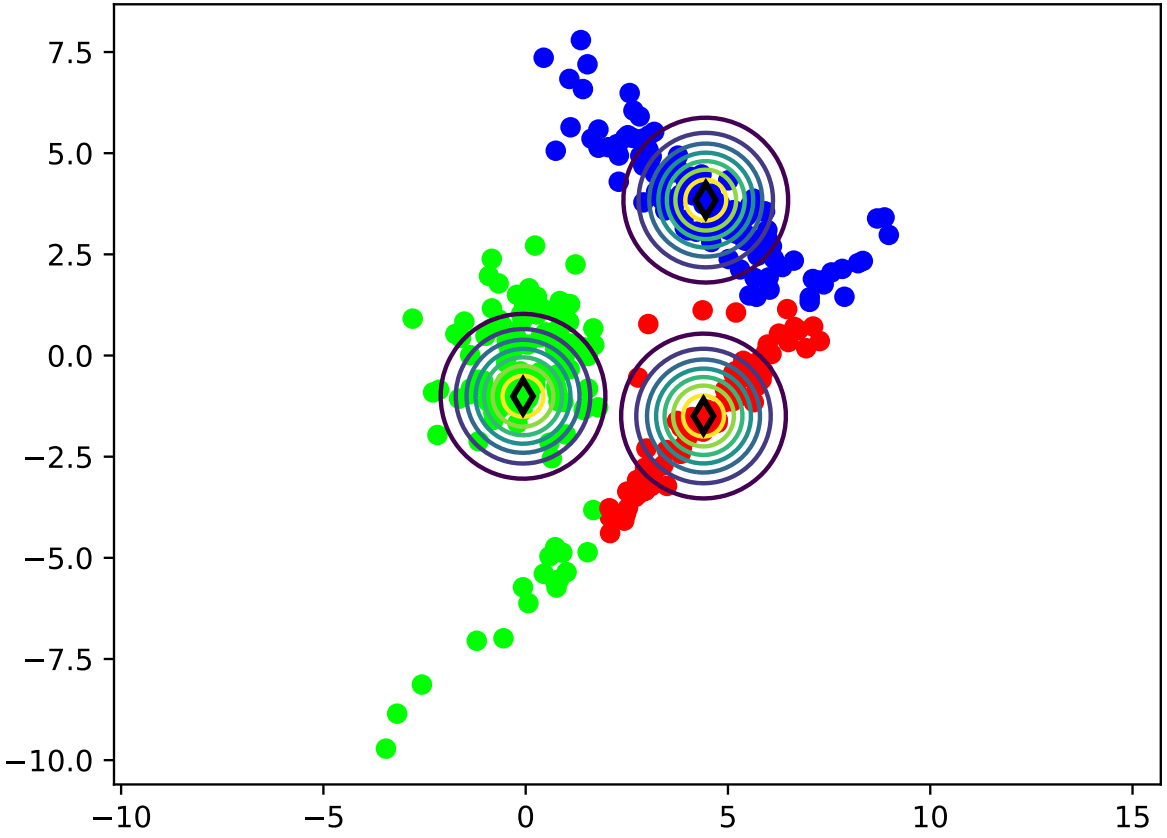
Gaussian Mixture Model with diagonal covariances.



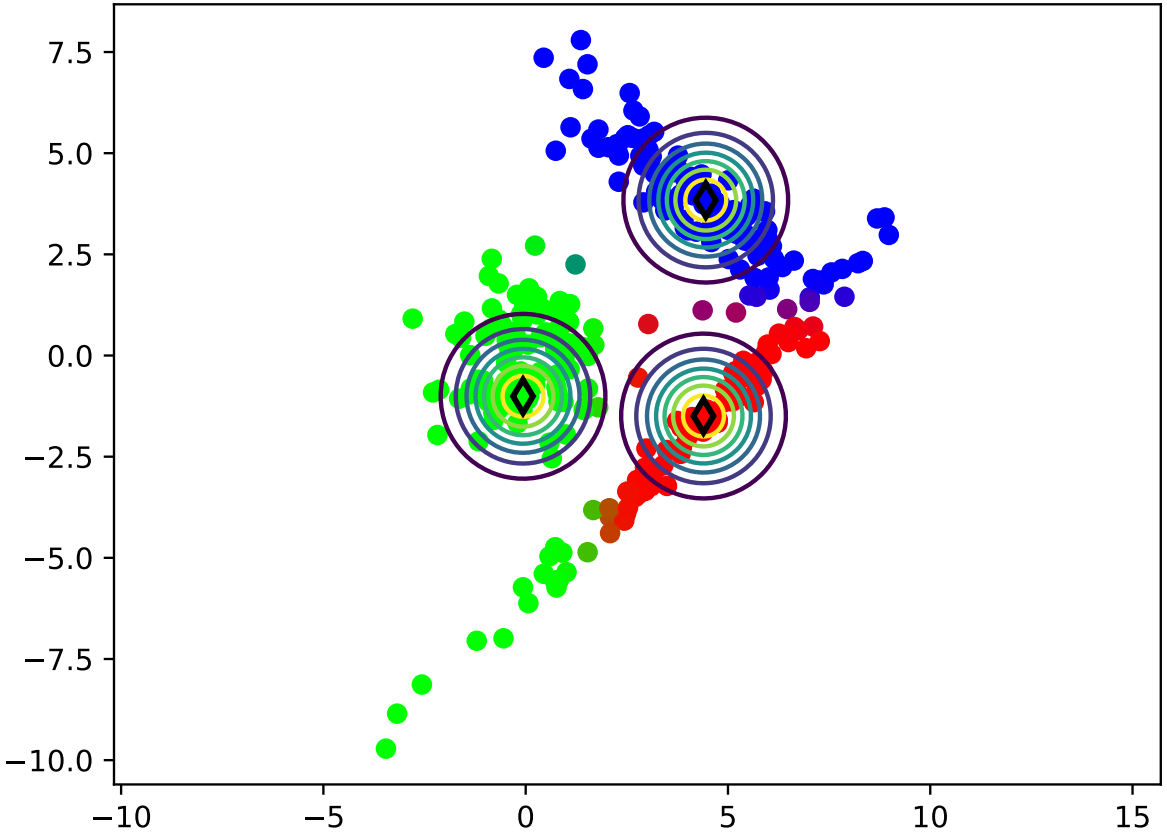
Gaussian Mixture Model with diagonal covariances.



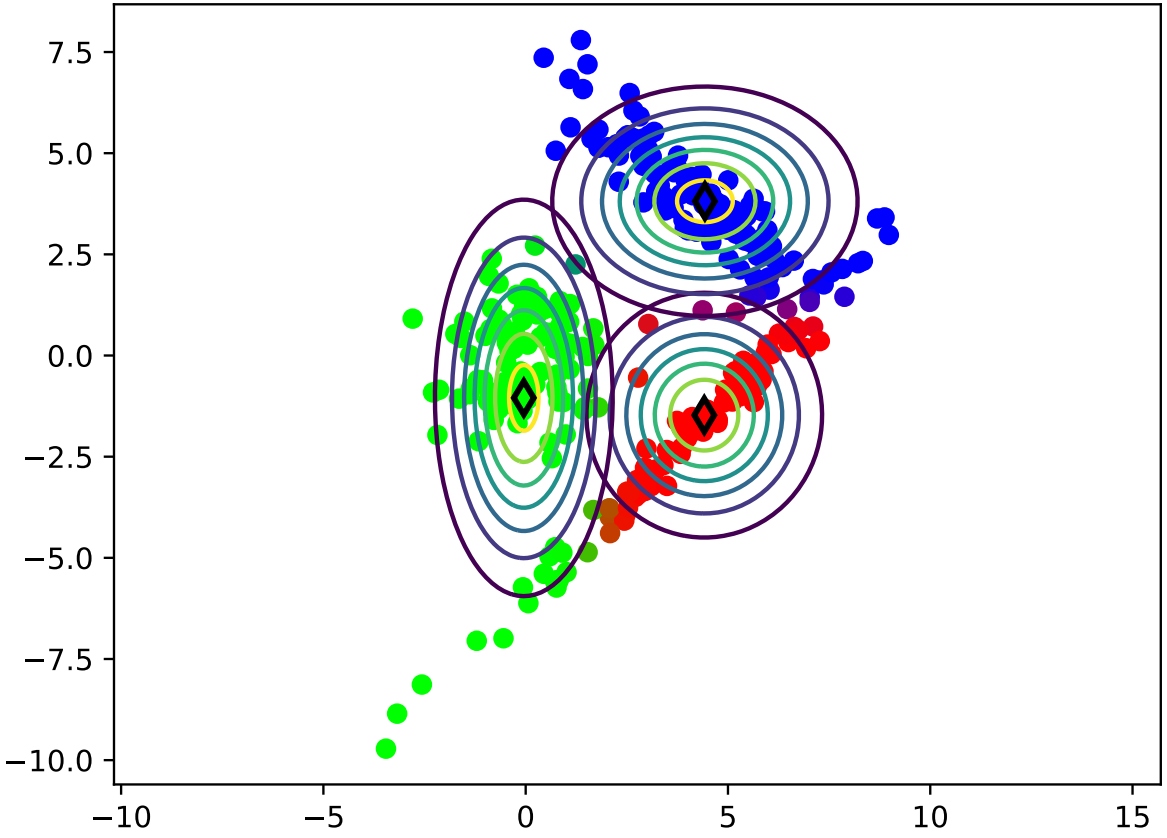
Gaussian Mixture Model with diagonal covariances.



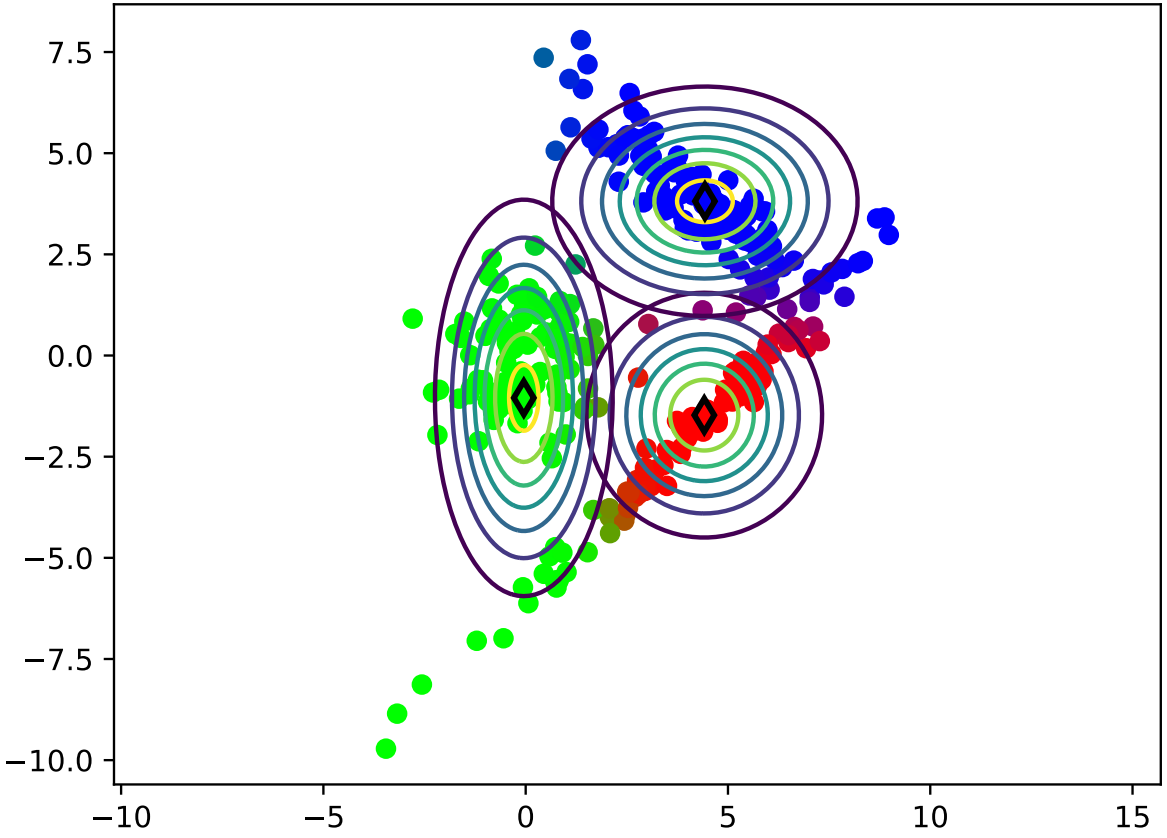
Gaussian Mixture Model with diagonal covariances.



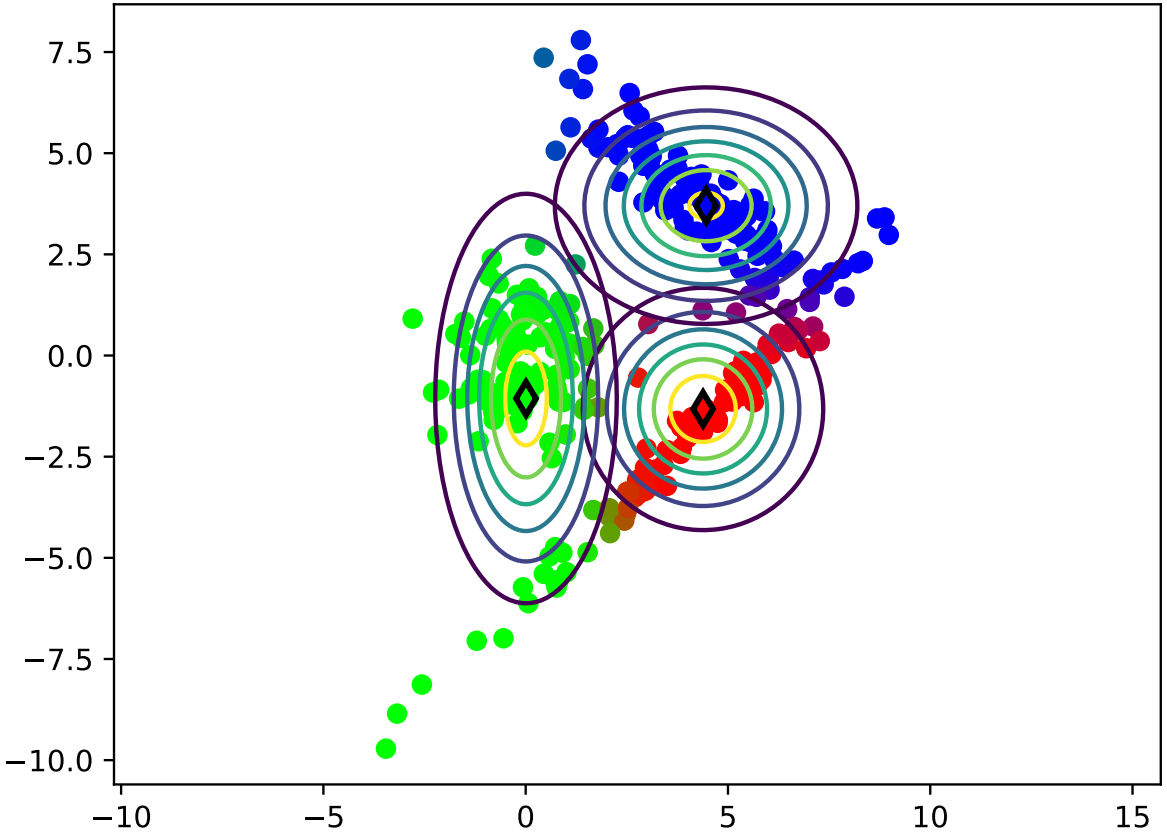
Gaussian Mixture Model with diagonal covariances.



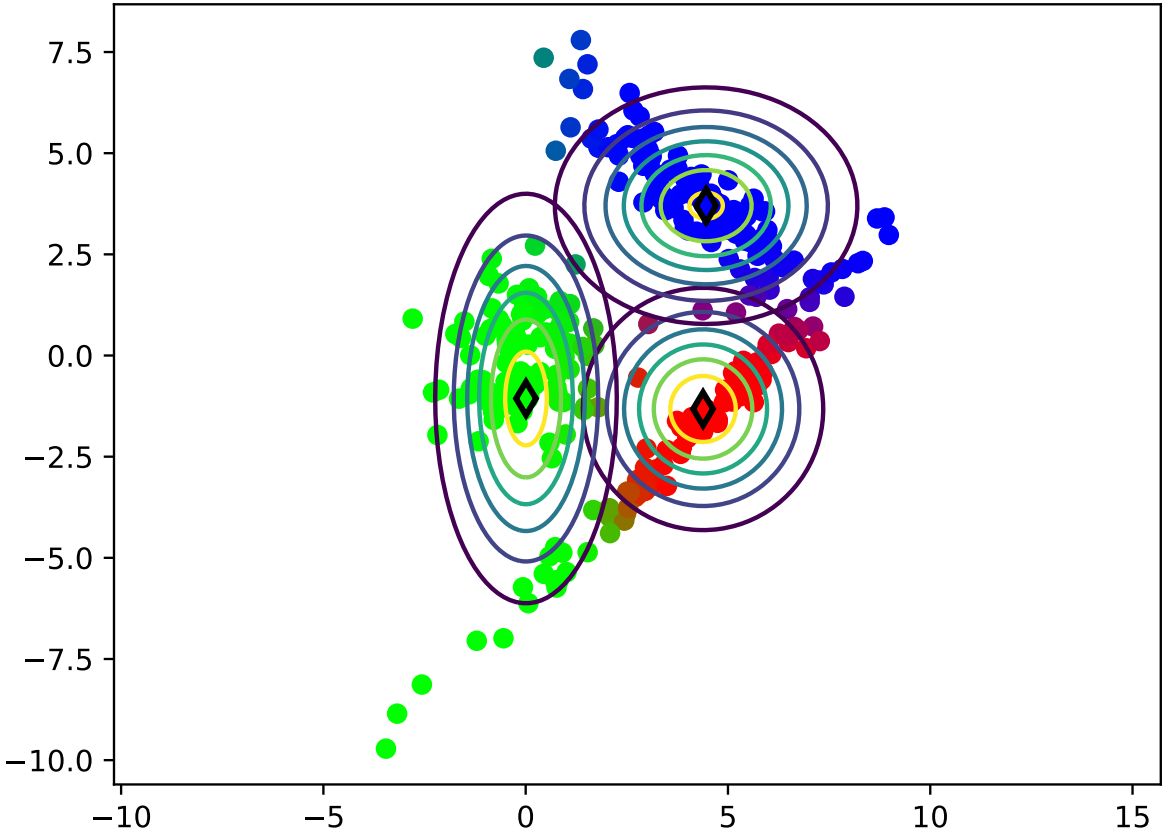
Gaussian Mixture Model with diagonal covariances.



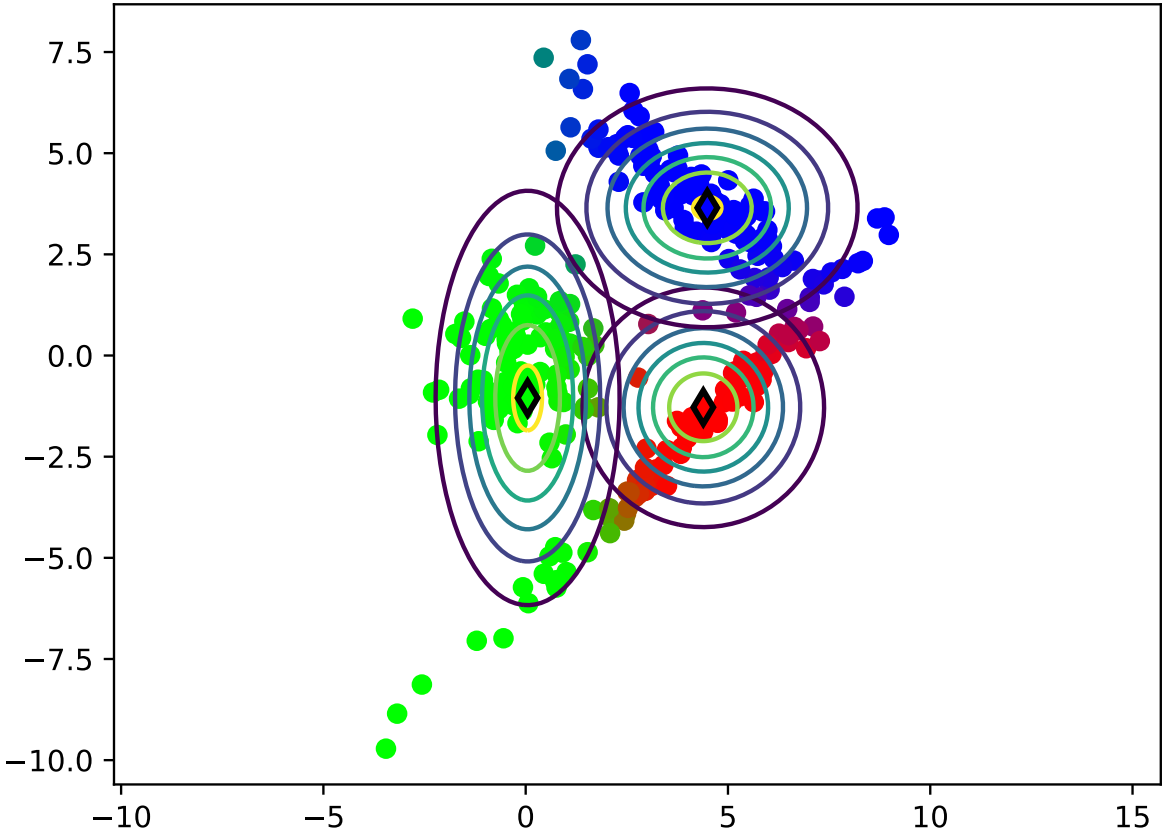
Gaussian Mixture Model with diagonal covariances.



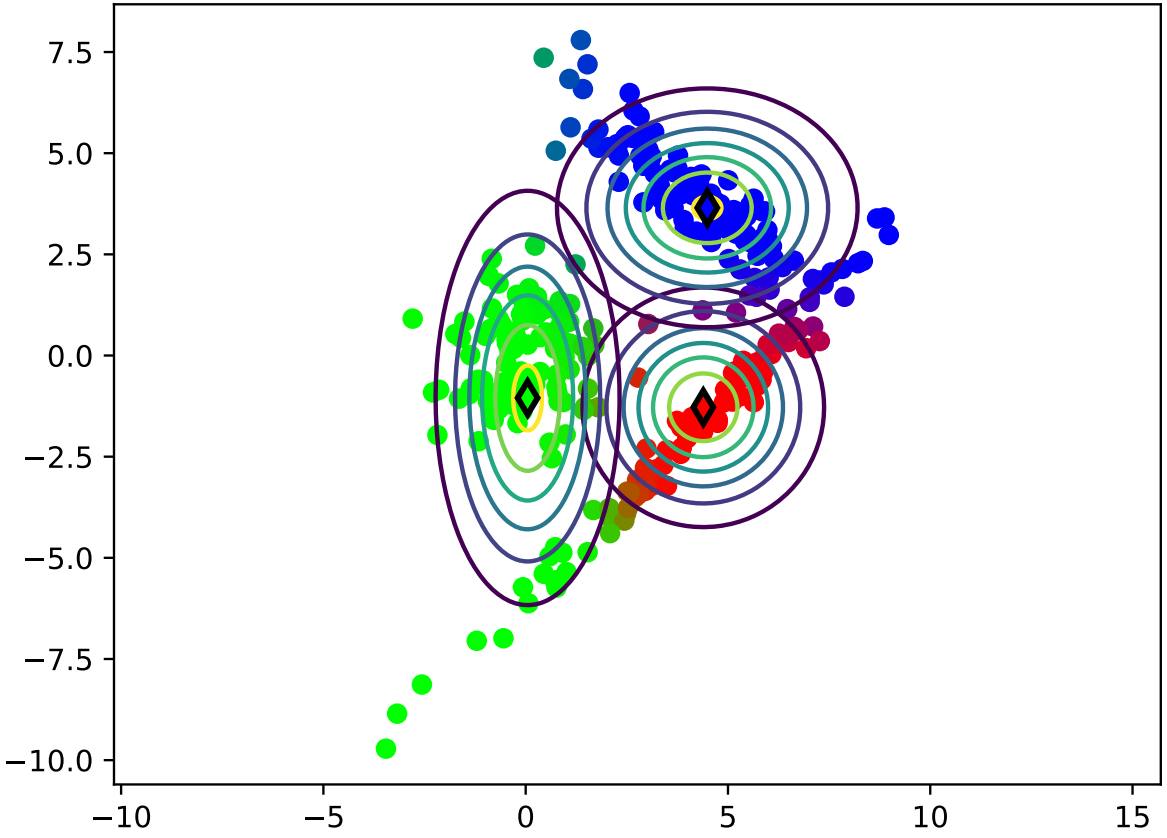
Gaussian Mixture Model with diagonal covariances.



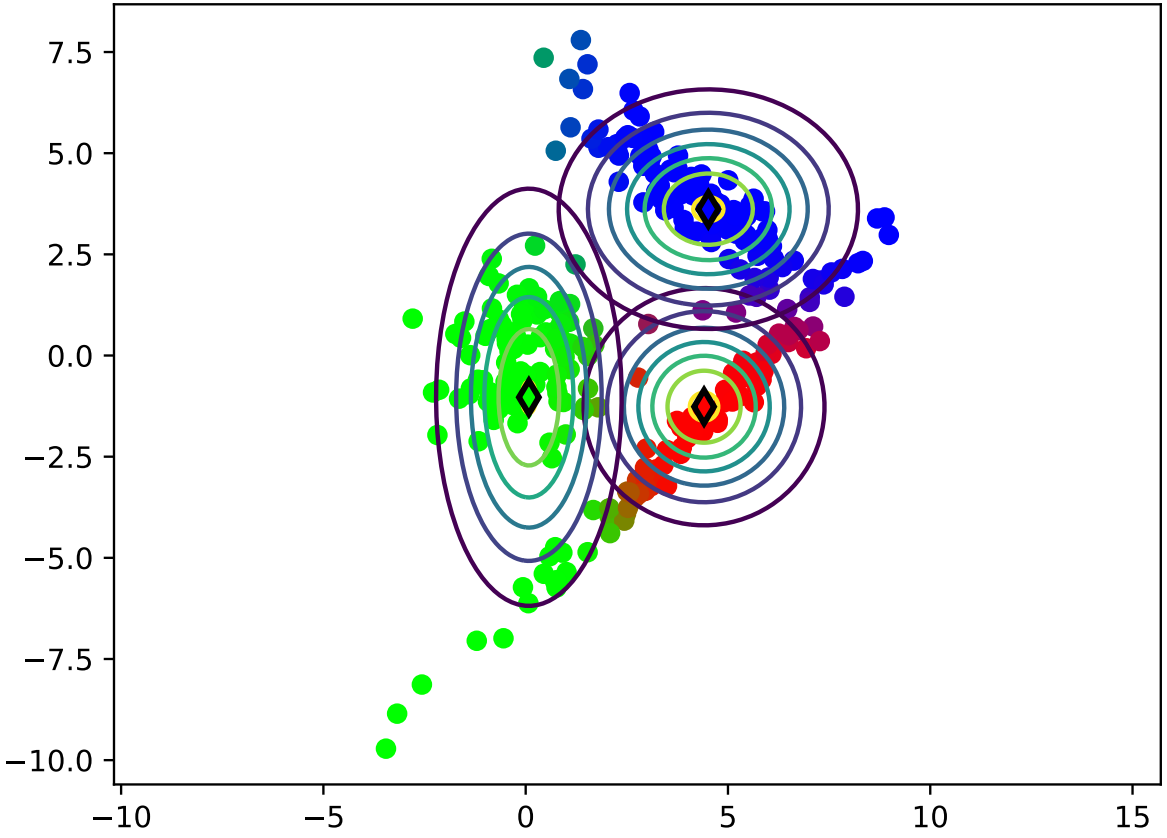
Gaussian Mixture Model with diagonal covariances.



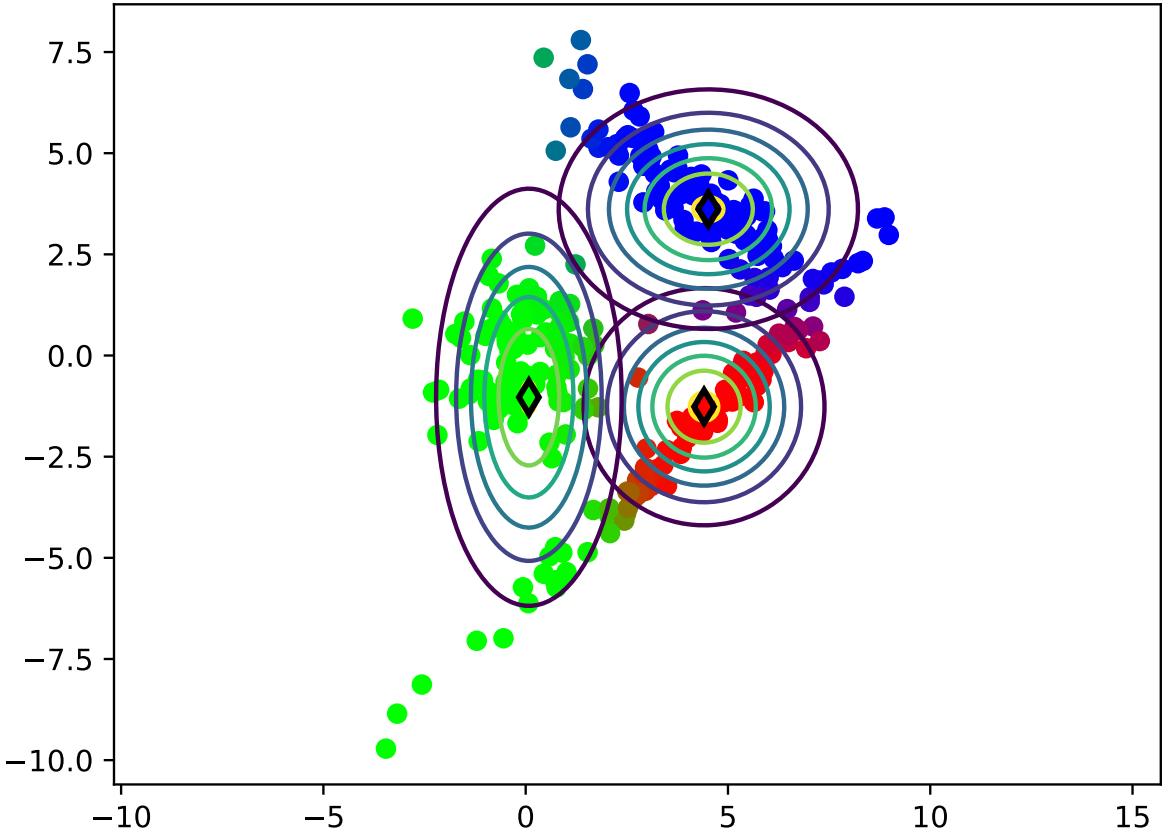
Gaussian Mixture Model with diagonal covariances.



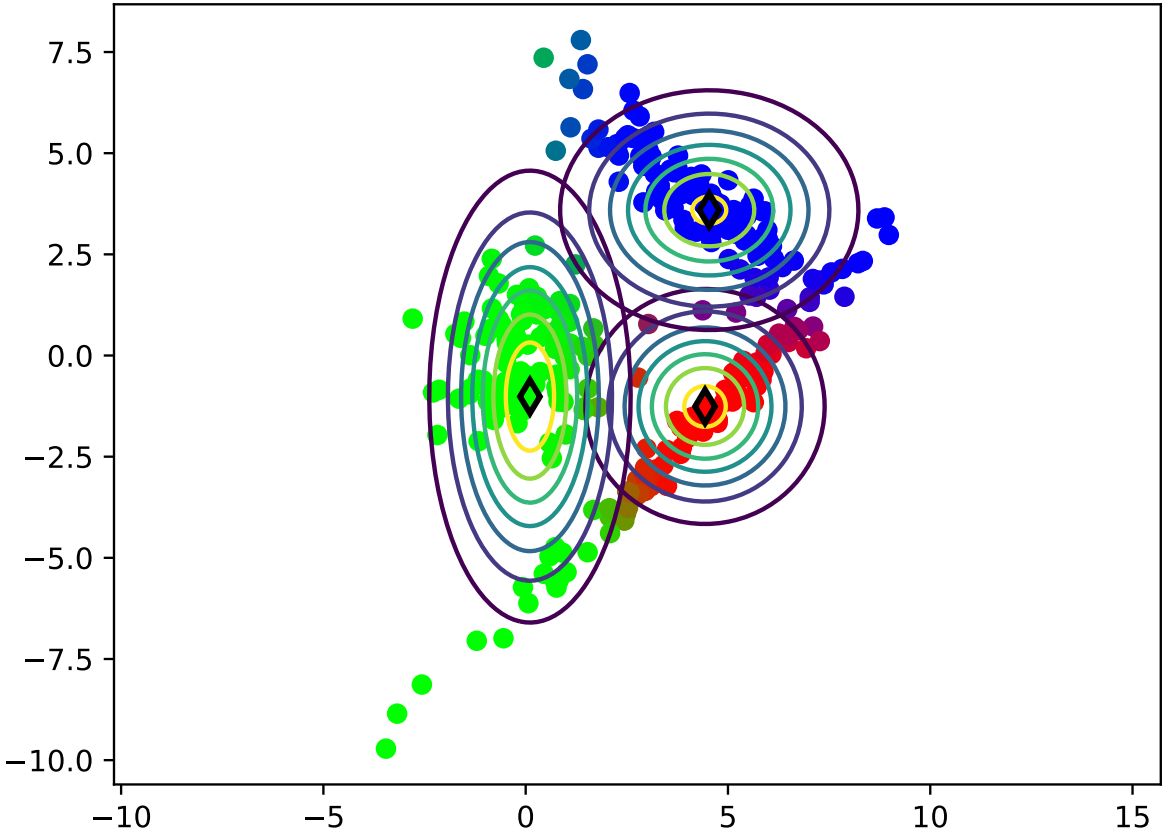
Gaussian Mixture Model with diagonal covariances.



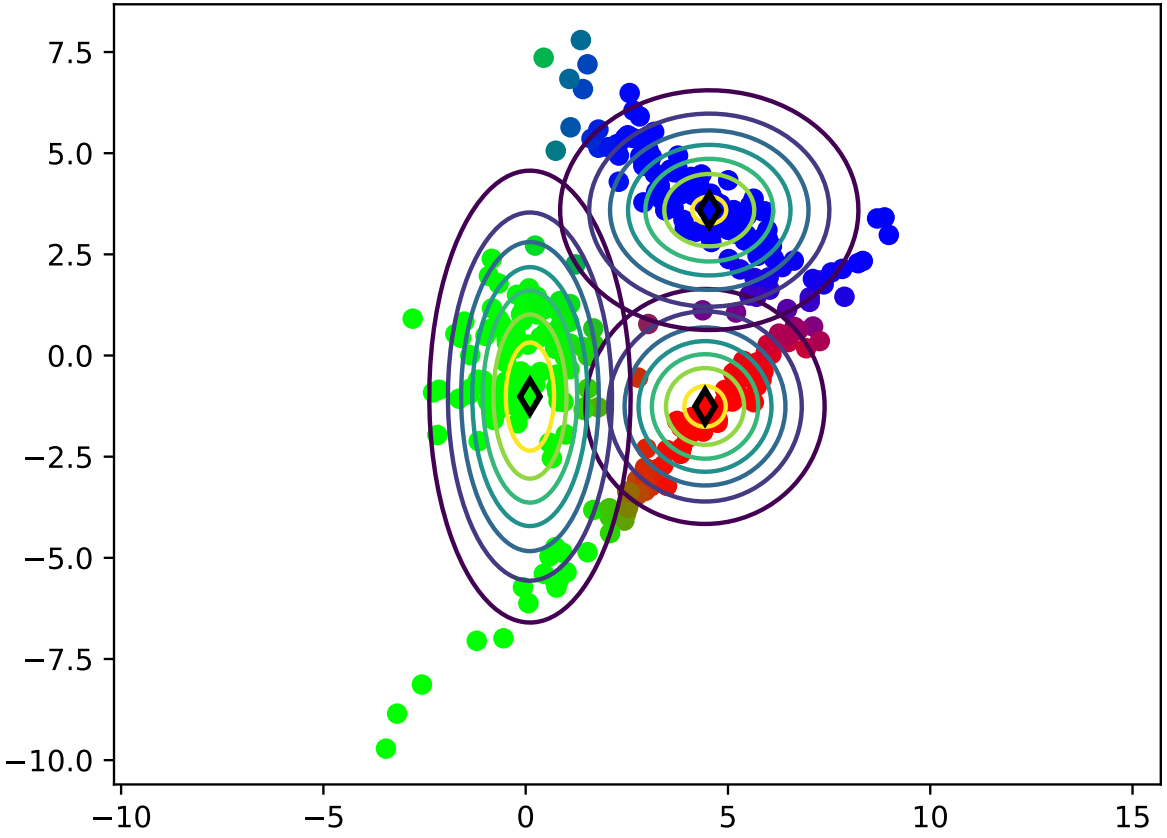
Gaussian Mixture Model with diagonal covariances.



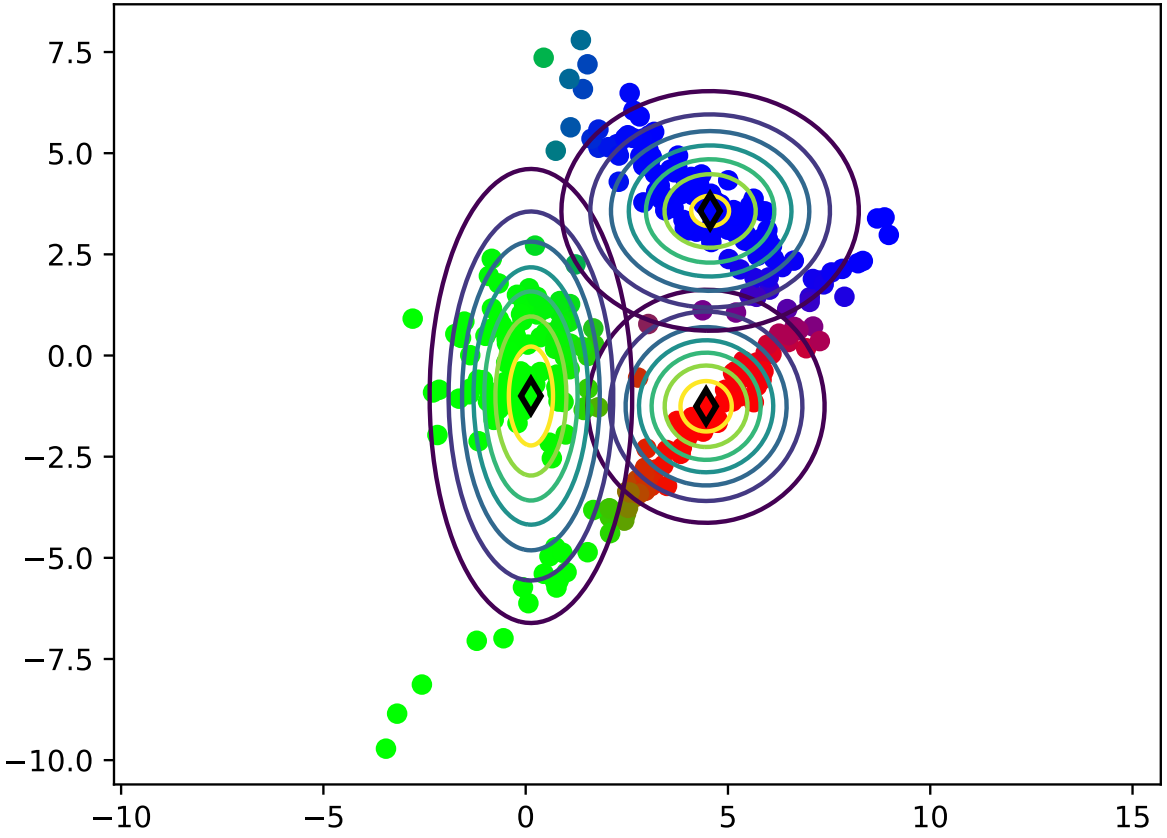
Gaussian Mixture Model with diagonal covariances.



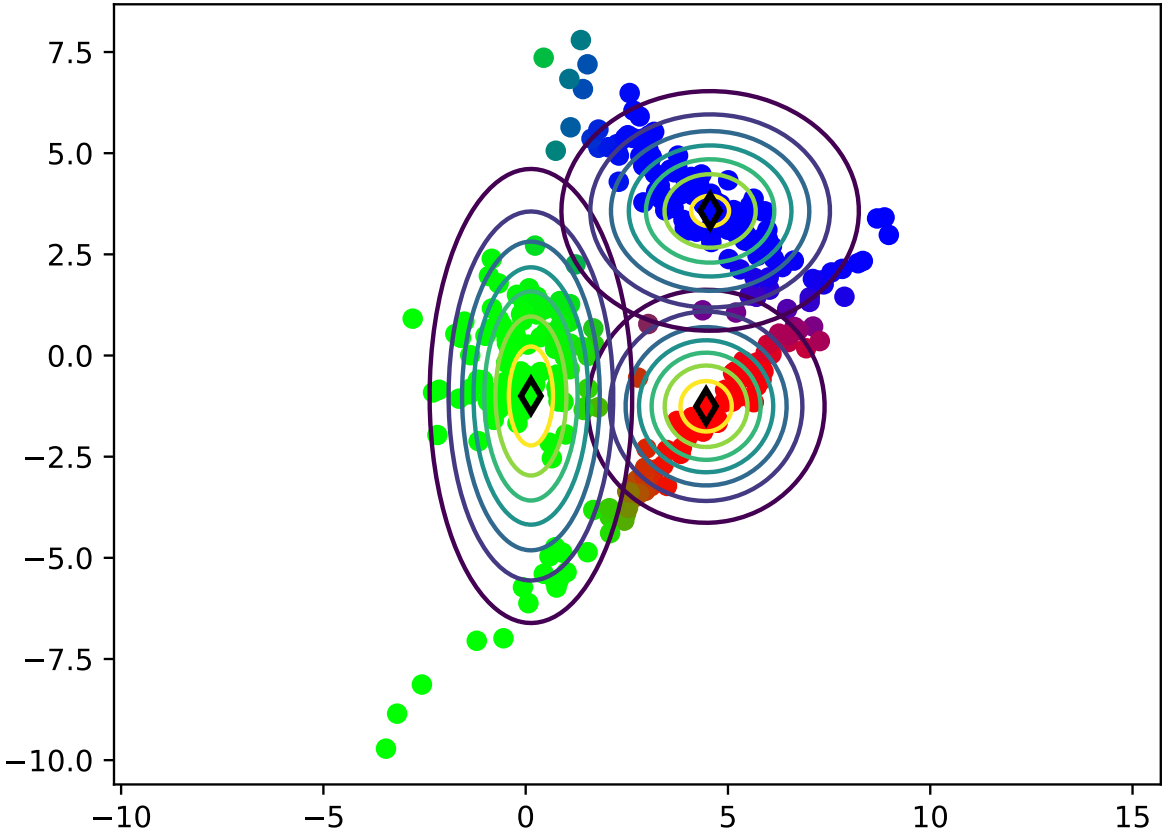
Gaussian Mixture Model with diagonal covariances.



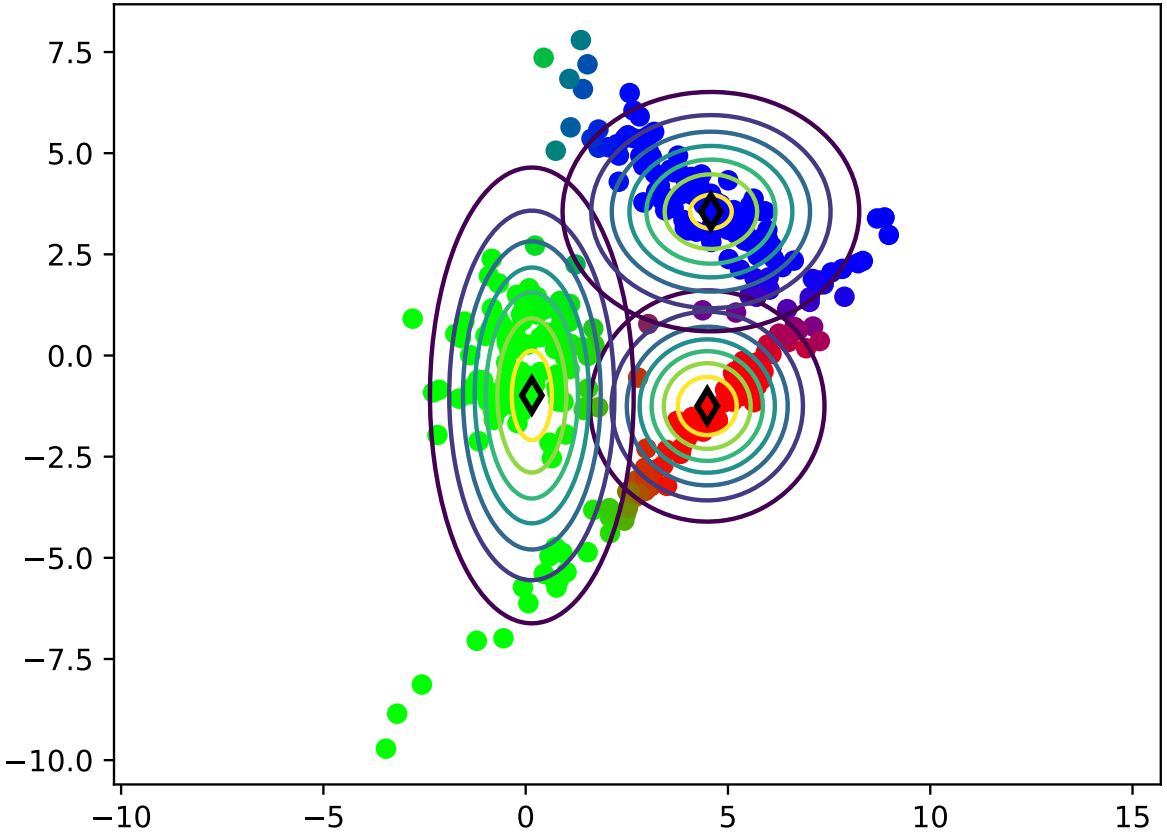
Gaussian Mixture Model with diagonal covariances.



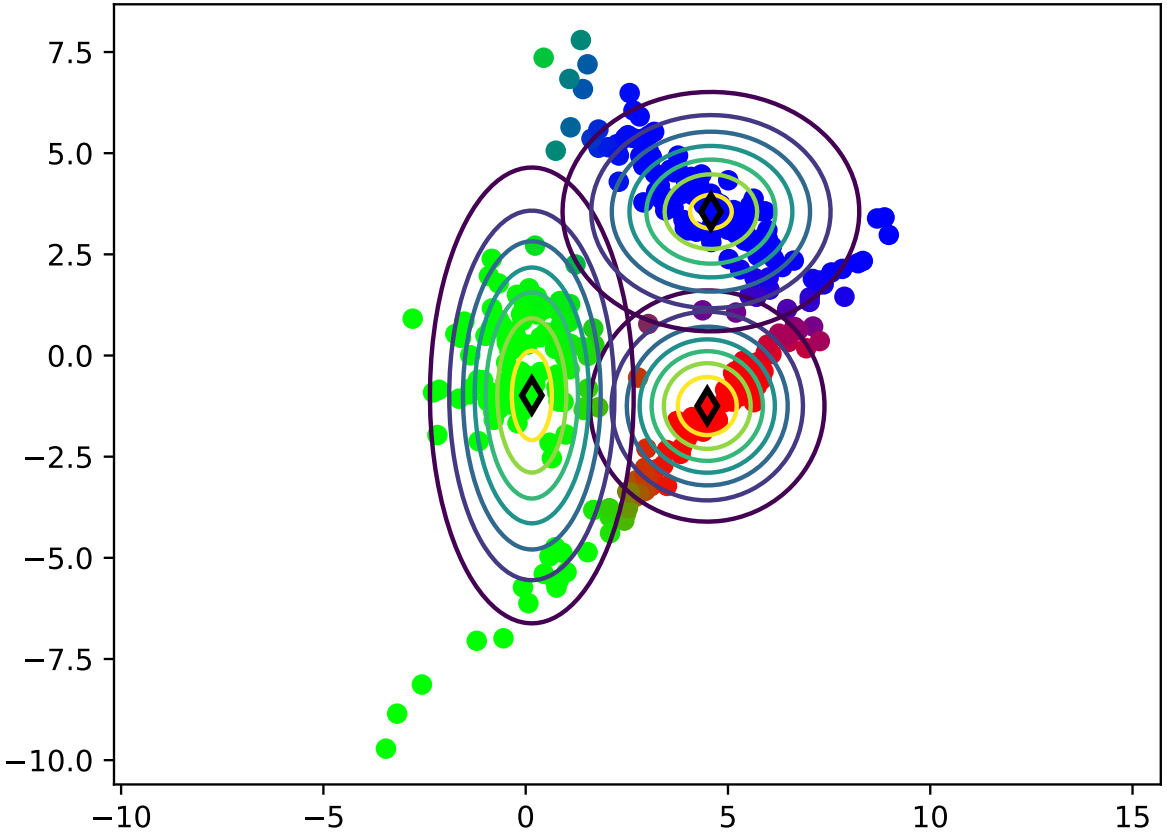
Gaussian Mixture Model with diagonal covariances.



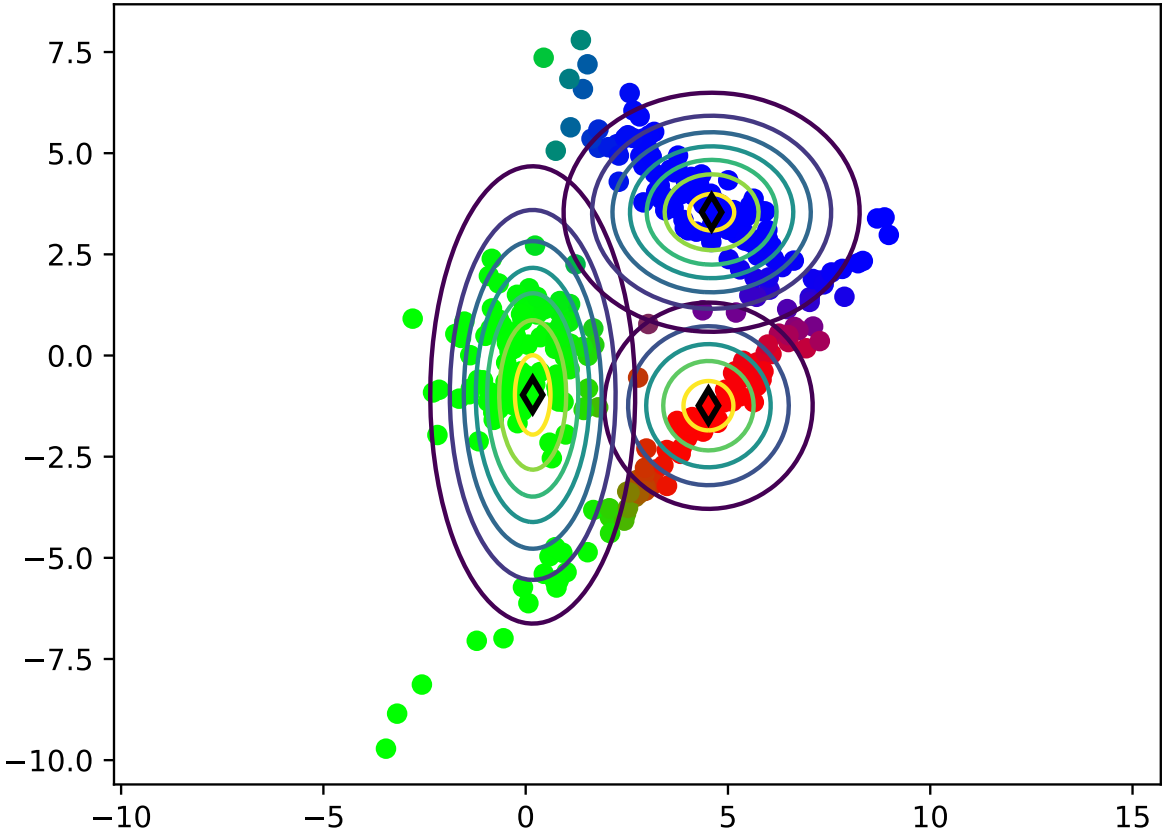
Gaussian Mixture Model with diagonal covariances.



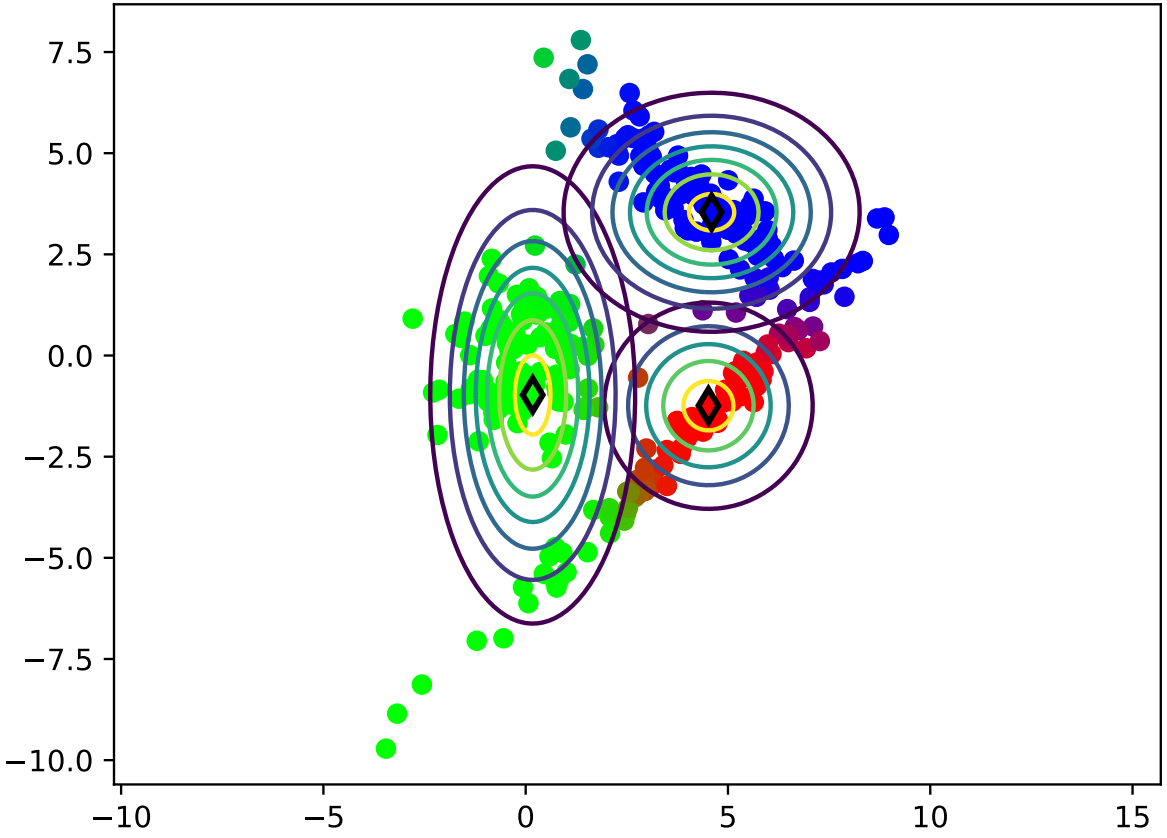
Gaussian Mixture Model with diagonal covariances.



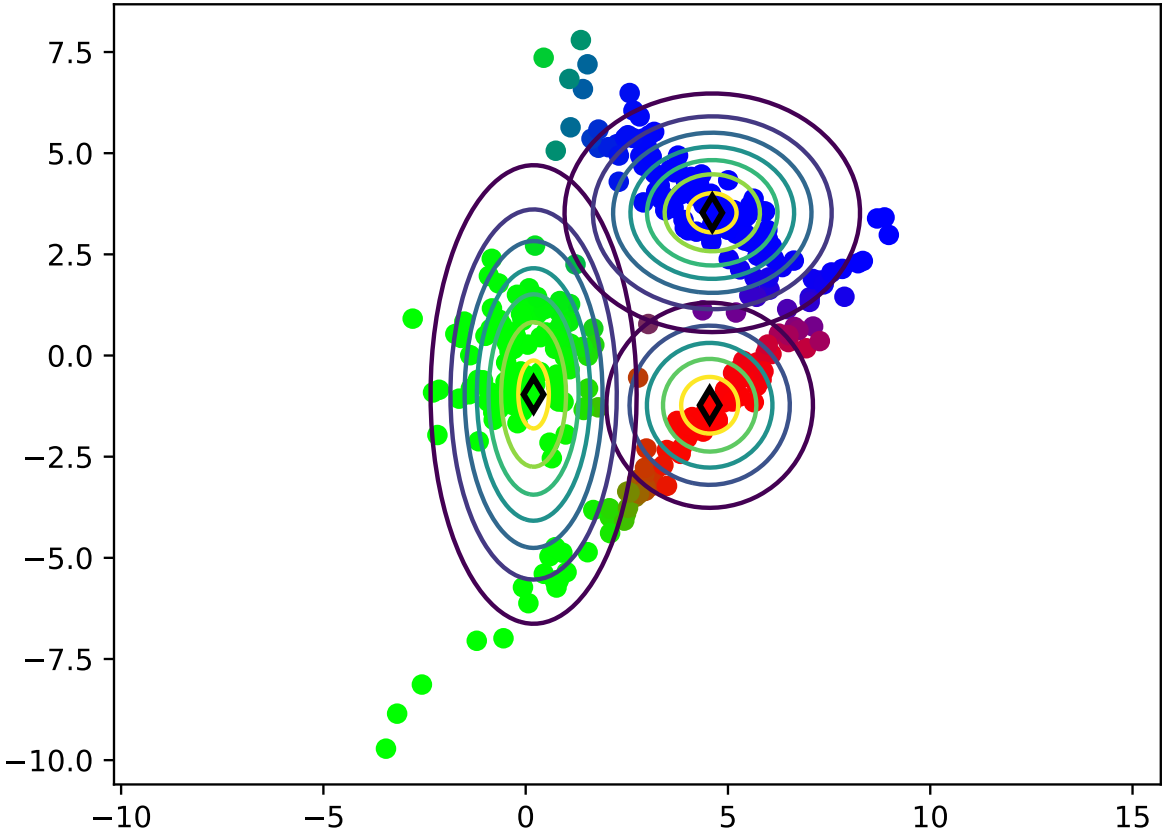
Gaussian Mixture Model with diagonal covariances.



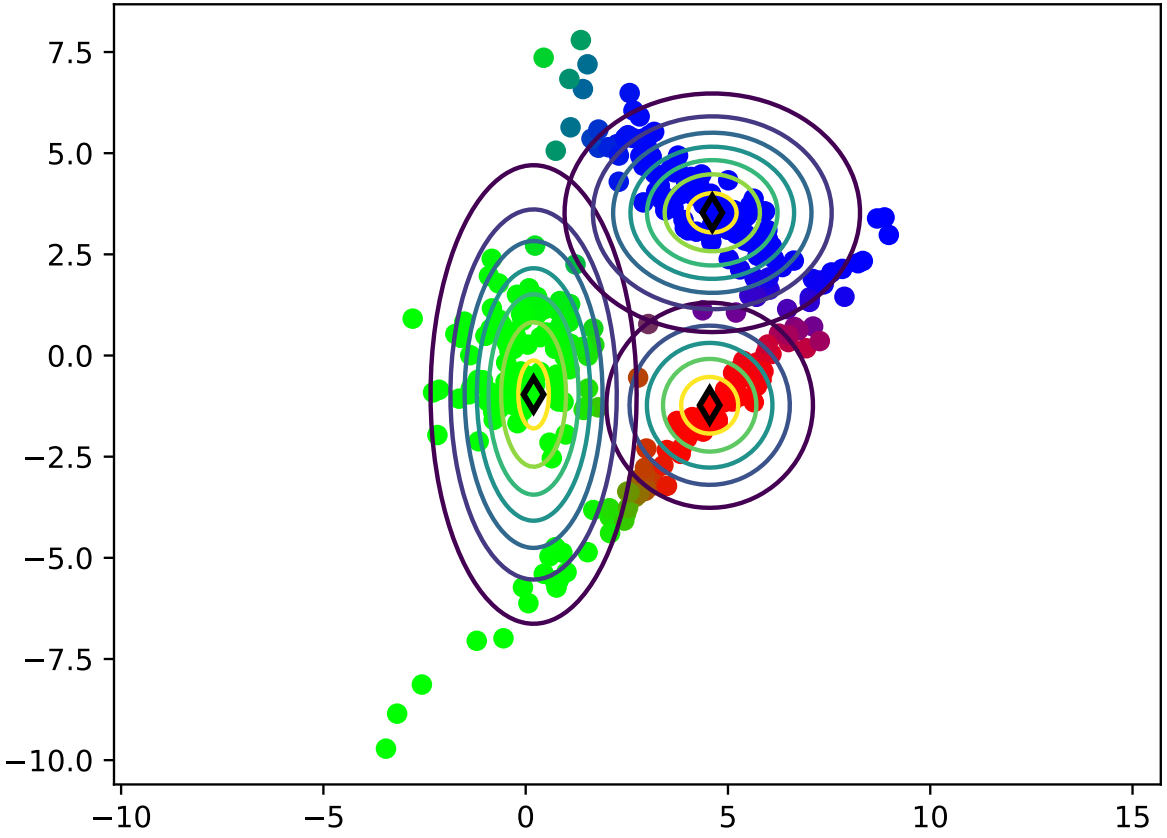
Gaussian Mixture Model with diagonal covariances.



Gaussian Mixture Model with diagonal covariances.



Gaussian Mixture Model with diagonal covariances.



Singularities

E-M with GMMs suffers from **singularities**:
trivial situations where the likelihood goes to ∞ but the solution is bad.

- Suppose:
  $d = 1$, $k = 2$, $\pi_j = 1/2$,
  $n = 3$ with $\boldsymbol{x}_1 = -1$ and $\boldsymbol{x}_2 = +1$ and $\boldsymbol{x}_3 = +3$.
  Initialize with $\boldsymbol{\mu}_1 = 0$ and $\boldsymbol{\sigma}_1 = 1$,
  but $\boldsymbol{\mu}_2 = +3 = \boldsymbol{x}_3$ and $\boldsymbol{\sigma}_2 = 1/100$.
  Then $\boldsymbol{\sigma}_2 \to 0$ and $\mathcal{L} \uparrow \infty$.

Interpolating between k-means and GMM E-M

Same M-step: fix $\boldsymbol{\pi} = (1/k, \ldots, 1/k)$ and $\boldsymbol{\Sigma}_j = c\boldsymbol{I}$ for a fixed $c > 0$.

Interpolating between k-means and GMM E-M

Same M-step: fix $\boldsymbol{\pi} = (1/k, \ldots, 1/k)$ and $\boldsymbol{\Sigma}_j = c\boldsymbol{I}$ for a fixed $c > 0$.

Same E-step: define $q_{ij} := \frac{1}{2}\|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$; the E-step chooses

$$R_{ij} := p_{\boldsymbol{\theta}}(y_i = j|\boldsymbol{x}_i) = \frac{p_{\boldsymbol{\theta}}(y_i = j, \boldsymbol{x}_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)} = \frac{p_{\boldsymbol{\theta}}(y_i = j, \boldsymbol{x}_i)}{\sum_{l=1}^k p_{\boldsymbol{\theta}}(y_i = l, \boldsymbol{x}_i)}$$

$$= \frac{\pi_j p_{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j}(\boldsymbol{x}_i)}{\sum_{l=1}^k \pi_l p_{\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l}(\boldsymbol{x}_i)} = \frac{\exp(-q_{ij}/c)}{\sum_{l=1}^k \exp(-q_{il}/c)}$$

Fix $i \in \{1, \ldots, n\}$ and suppose minimum $q_i := \min_j q_{ij}$ is unique:

Interpolating between k-means and GMM E-M

Same M-step: fix $\boldsymbol{\pi} = (1/k, \ldots, 1/k)$ and $\boldsymbol{\Sigma}_j = c\boldsymbol{I}$ for a fixed $c > 0$.

Same E-step: define $q_{ij} := \frac{1}{2}\|\boldsymbol{x}_i - \boldsymbol{\mu}_j\|^2$; the E-step chooses

$$R_{ij} := p_{\boldsymbol{\theta}}(y_i = j|\boldsymbol{x}_i) = \frac{p_{\boldsymbol{\theta}}(y_i = j, \boldsymbol{x}_i)}{p_{\boldsymbol{\theta}}(\boldsymbol{x}_i)} = \frac{p_{\boldsymbol{\theta}}(y_i = j, \boldsymbol{x}_i)}{\sum_{l=1}^k p_{\boldsymbol{\theta}}(y_i = l, \boldsymbol{x}_i)}$$

$$= \frac{\pi_j p_{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j}(\boldsymbol{x}_i)}{\sum_{l=1}^k \pi_l p_{\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l}(\boldsymbol{x}_i)} = \frac{\exp(-q_{ij}/c)}{\sum_{l=1}^k \exp(-q_{il}/c)}$$

Fix $i \in \{1, \ldots, n\}$ and suppose minimum $q_i := \min_j q_{ij}$ is unique:

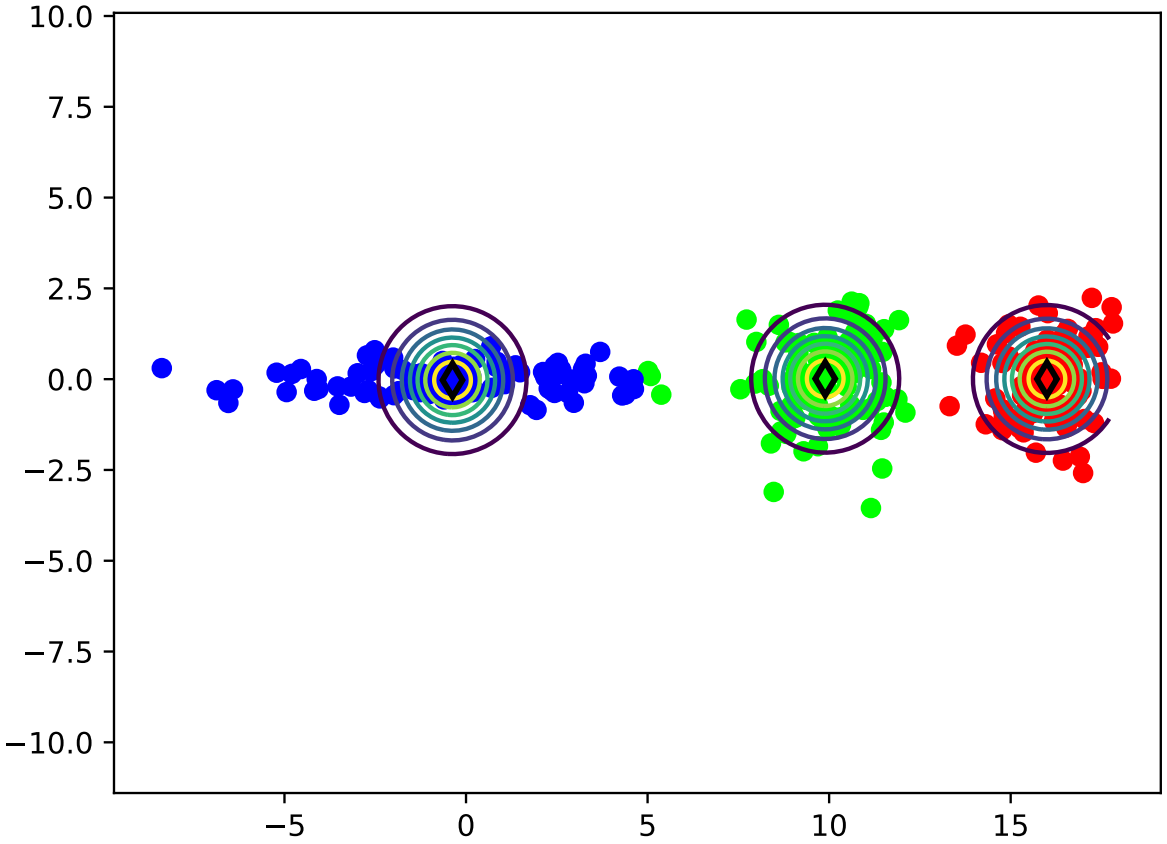
$$\lim_{c \downarrow 0} R_{ij} = \lim_{c \downarrow 0} \frac{\exp(-q_{ij}/c)}{\sum_{l=1}^k \exp(-q_{il}/c)} = \lim_{c \downarrow 0} \frac{\exp(q_i - q_{ij}/c)}{\sum_{l=1}^k \exp(q_i - q_{il}/c)}$$

$$= \begin{cases} 1 & q_{ij} = q_i, \\ 0 & q_{ij} \neq q_i. \end{cases}$$

That is, R becomes hard assignment A as $c \downarrow 0$.

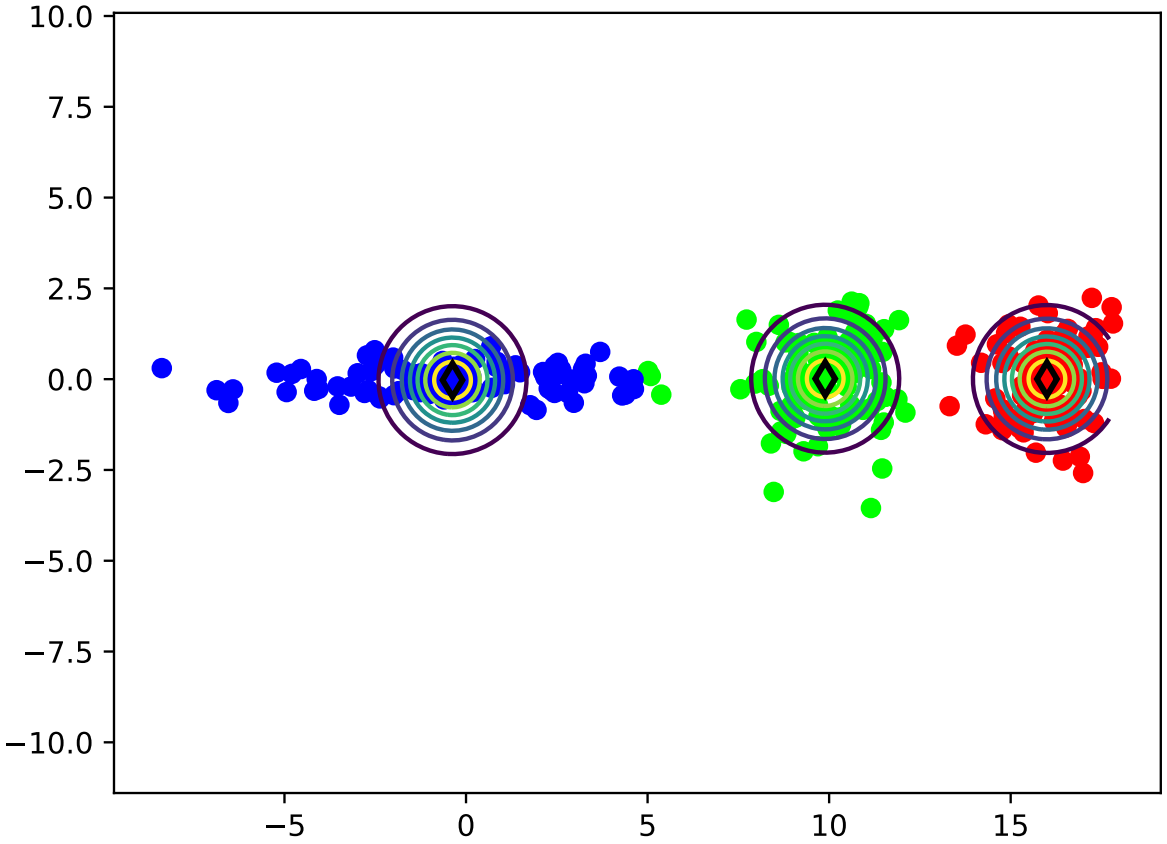
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



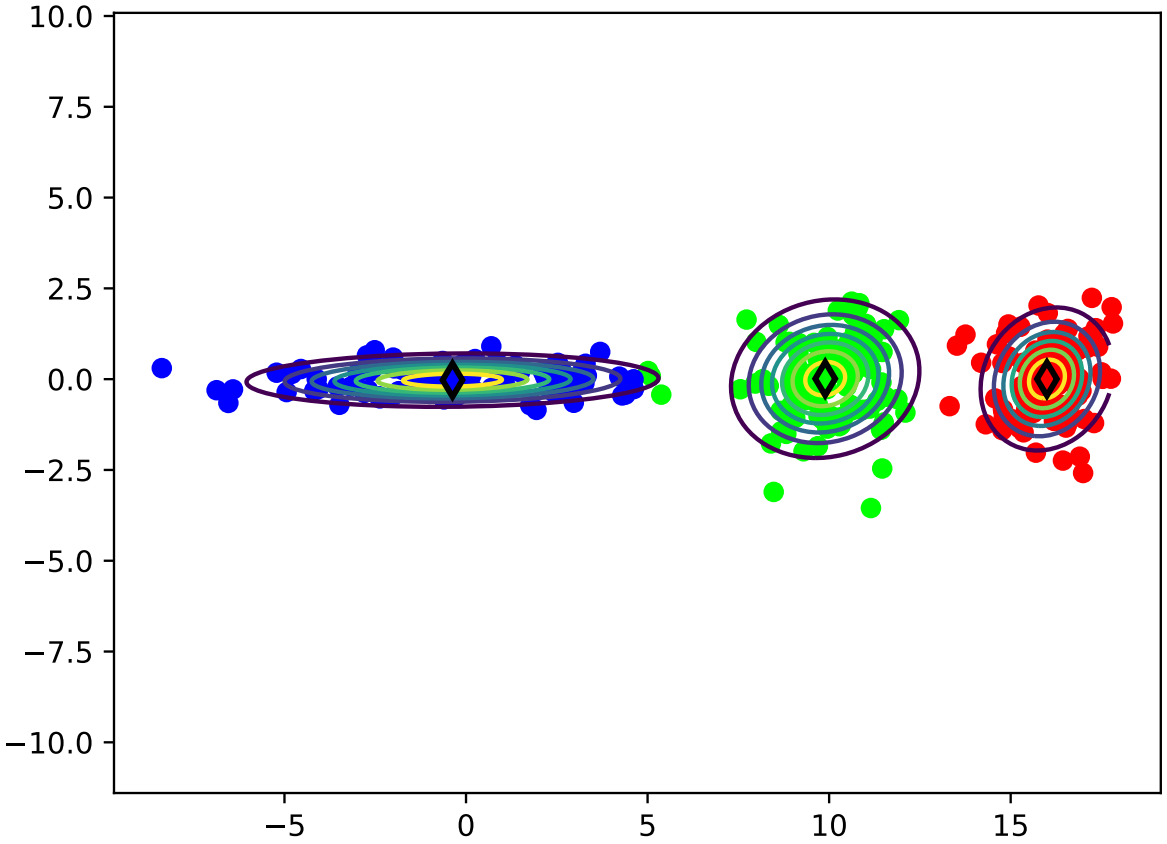
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



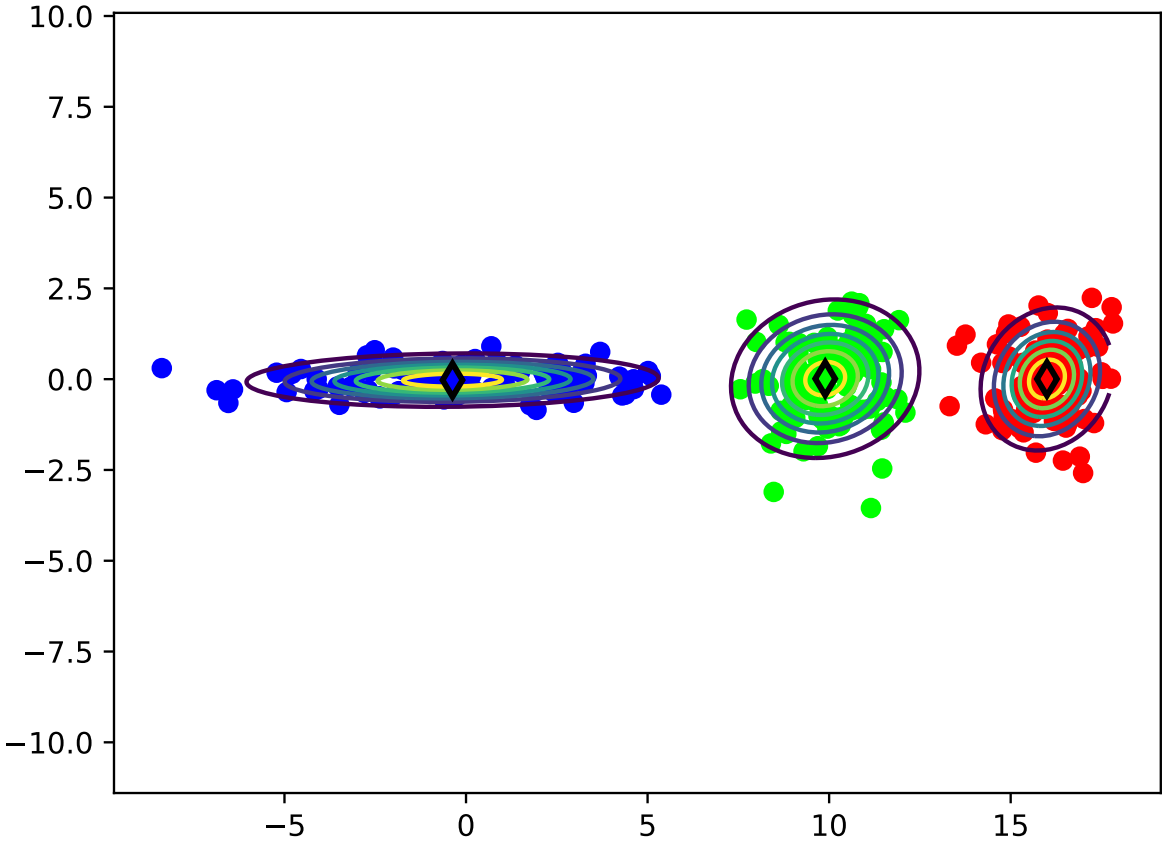
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



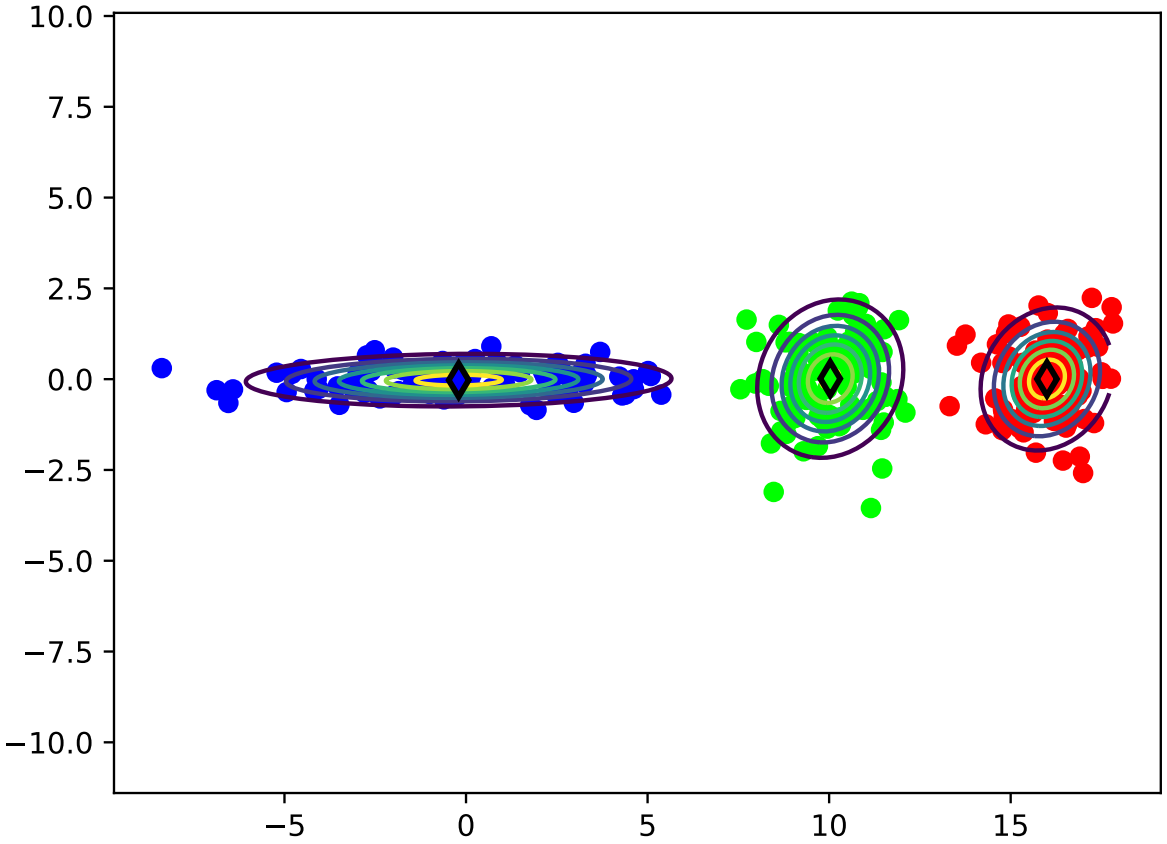
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



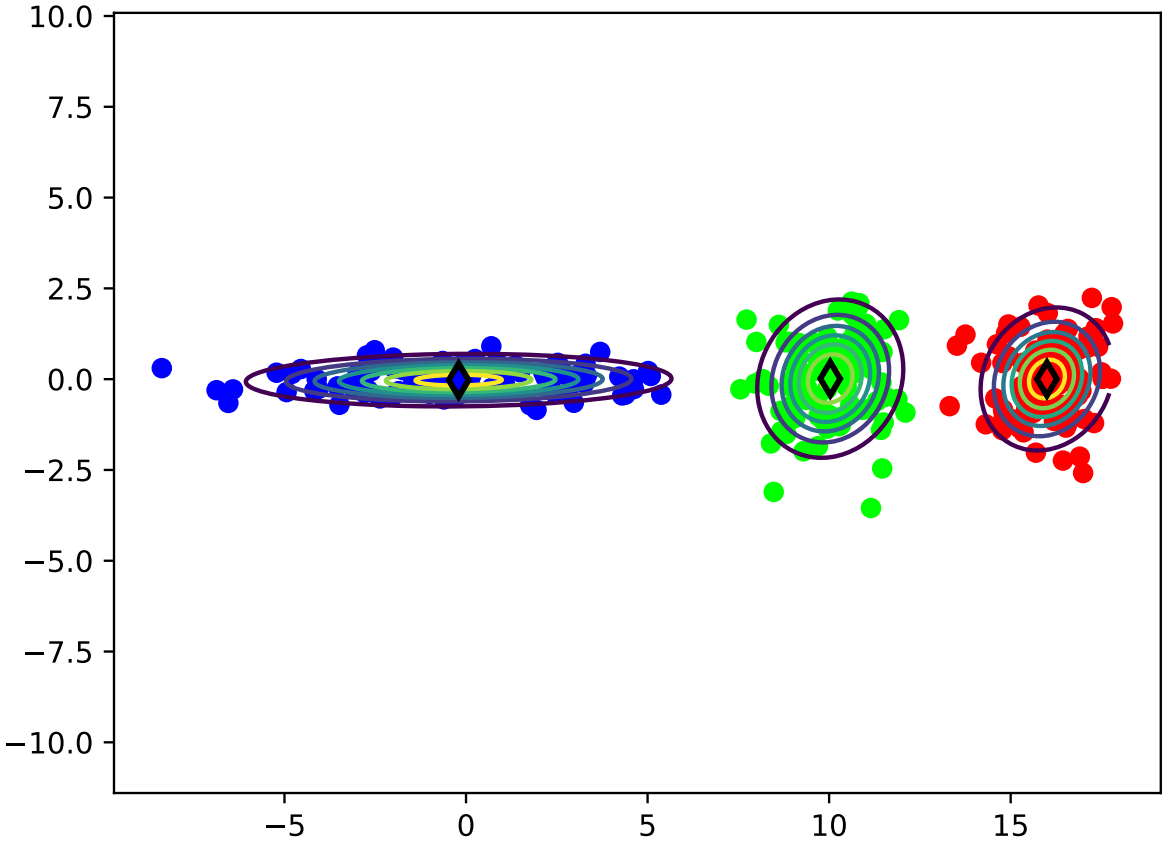
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



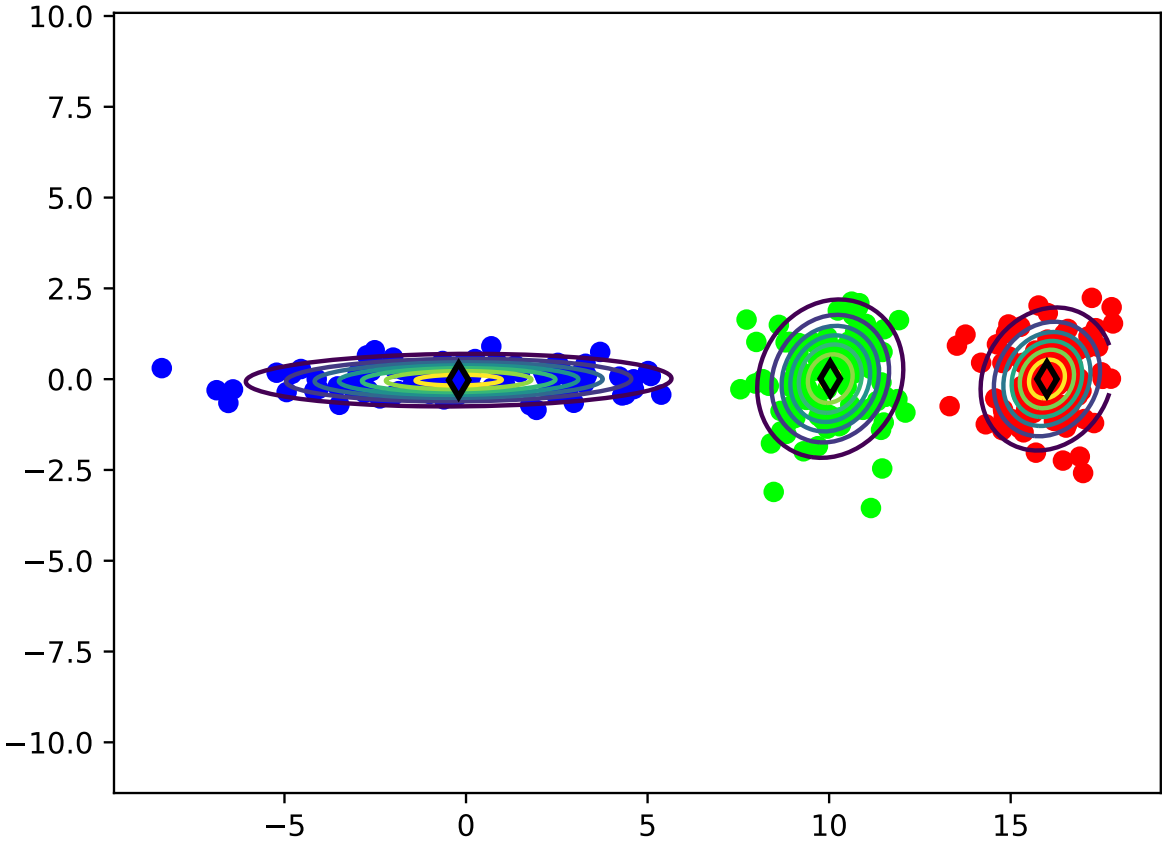
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



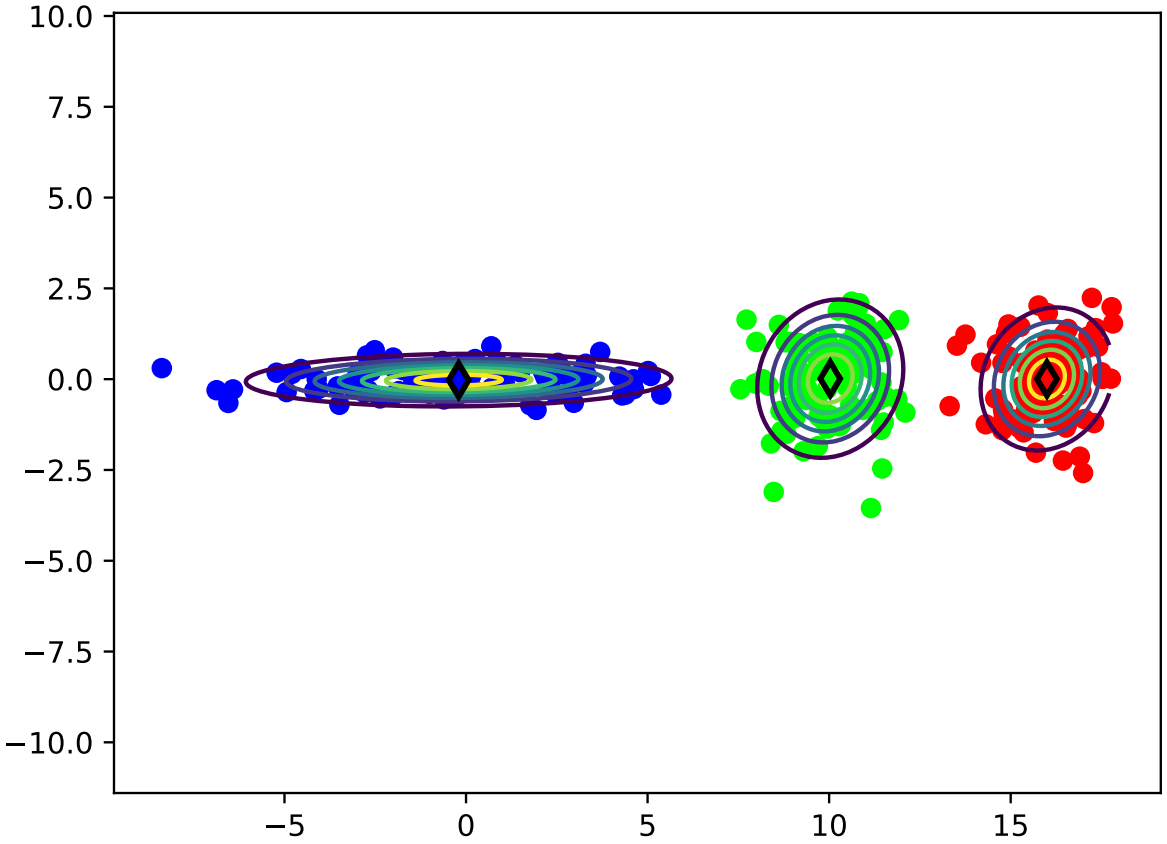
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



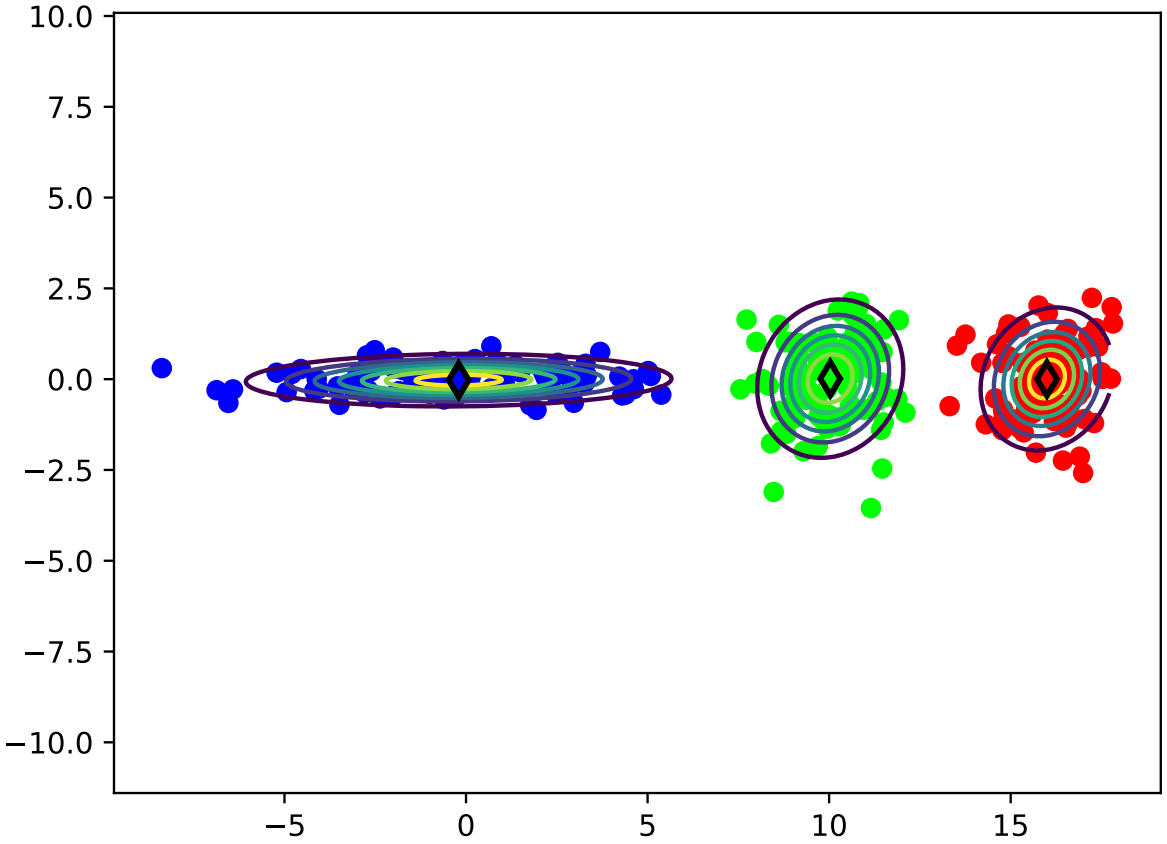
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



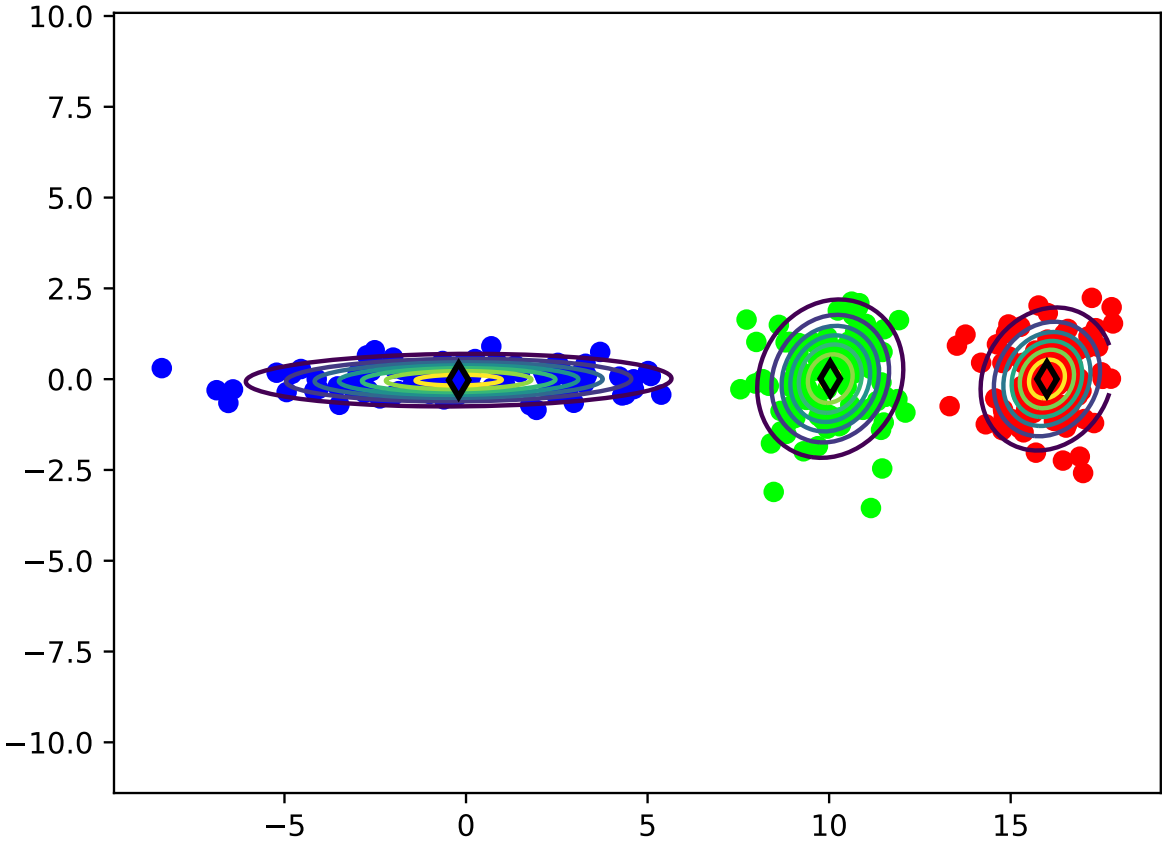
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



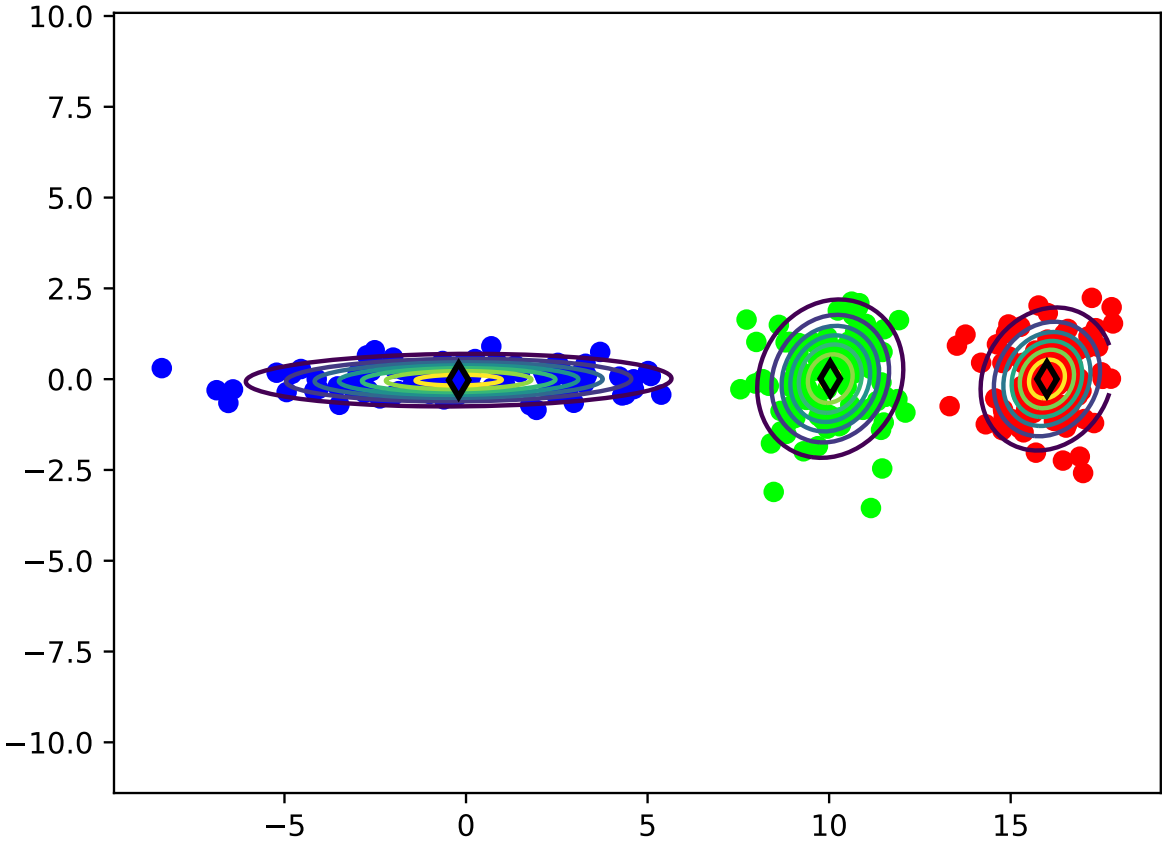
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



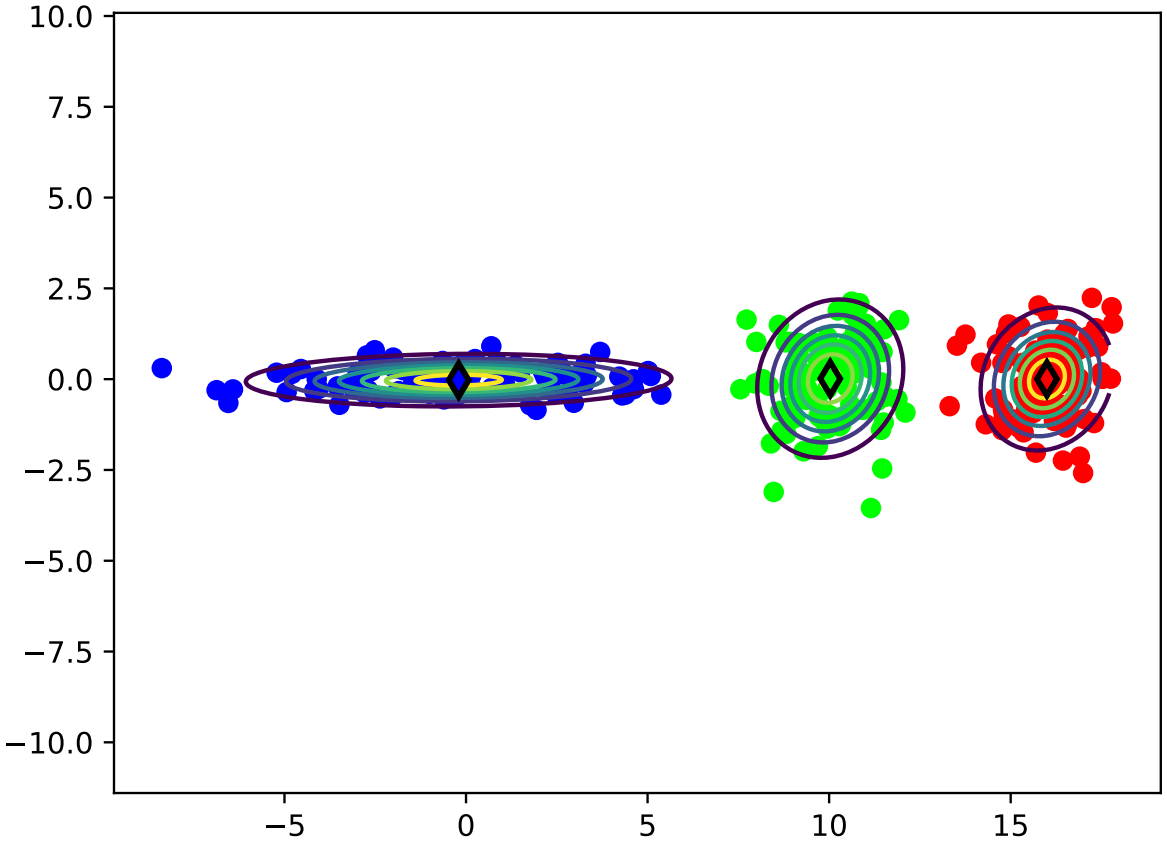
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



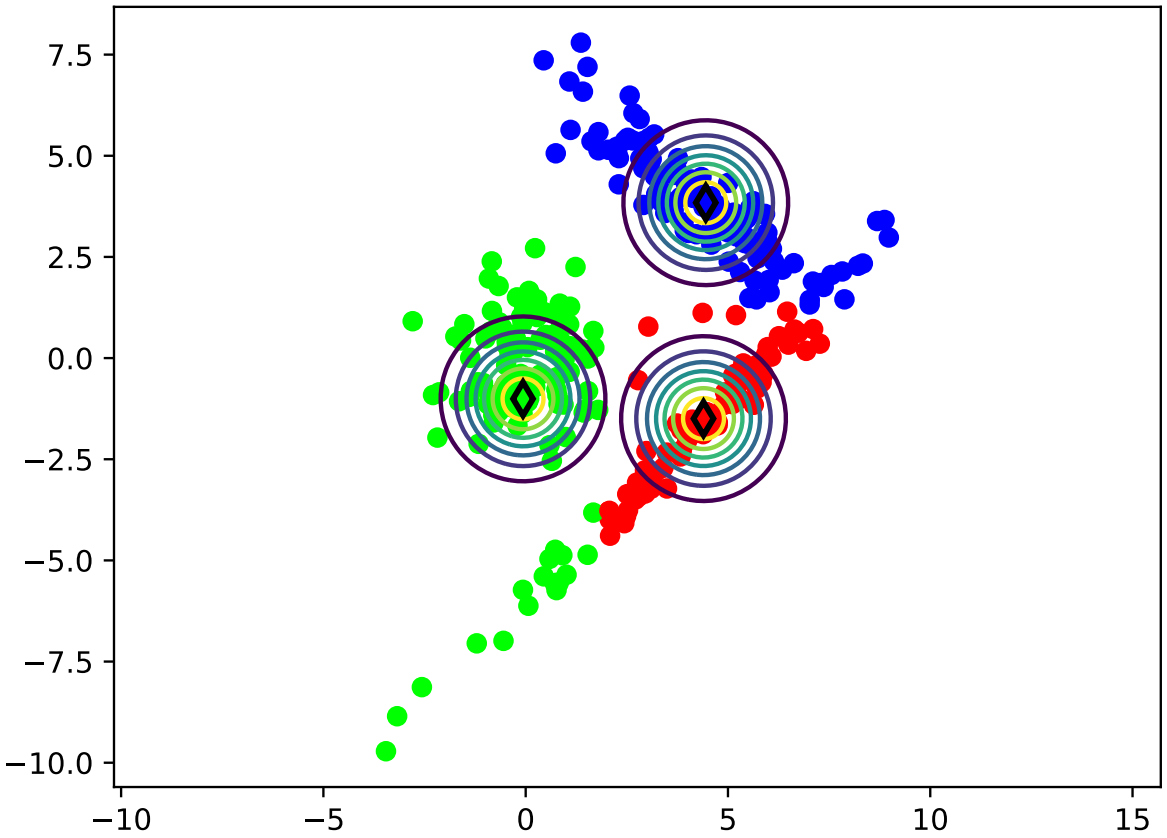
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



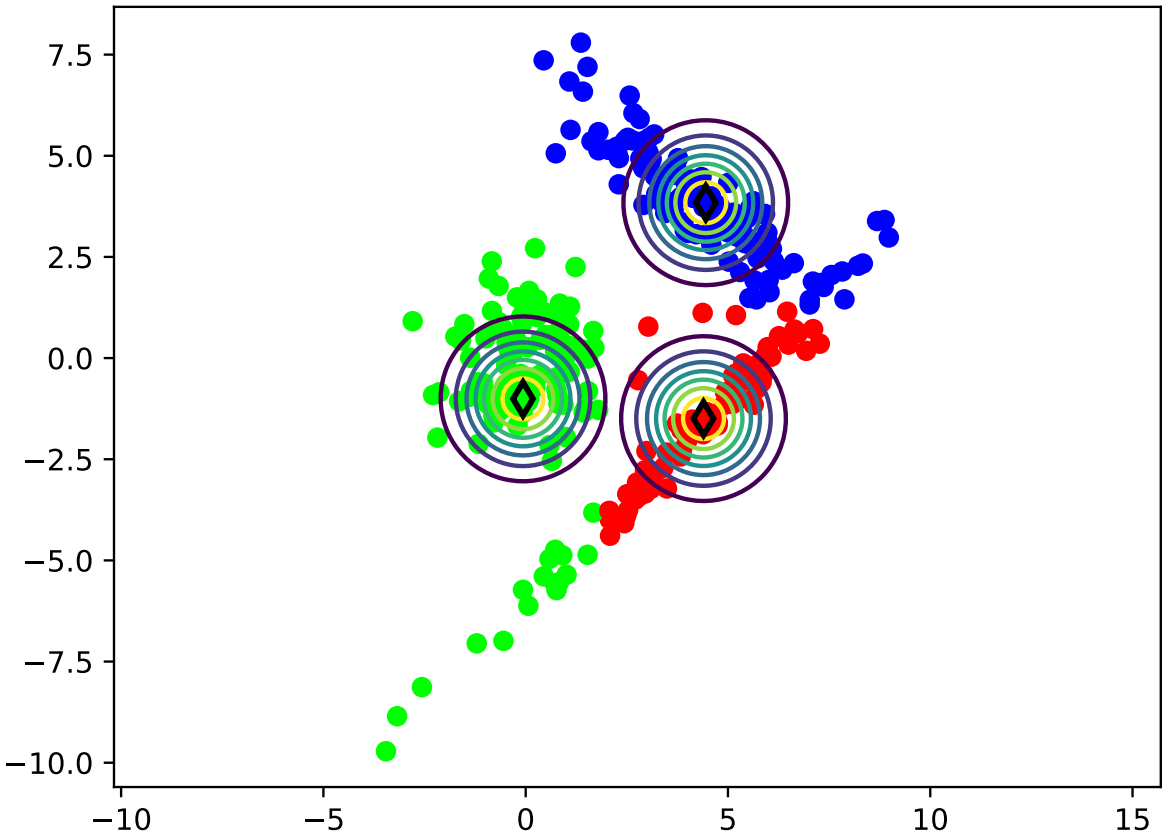
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



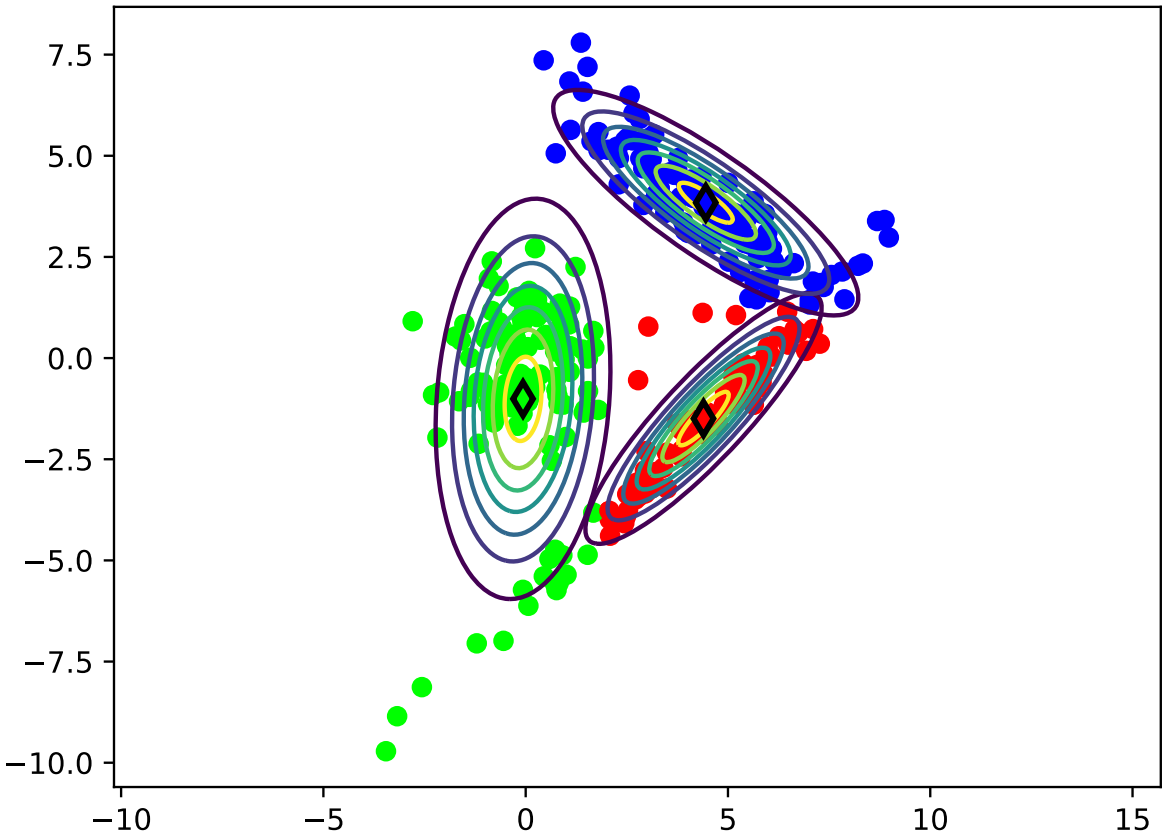
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



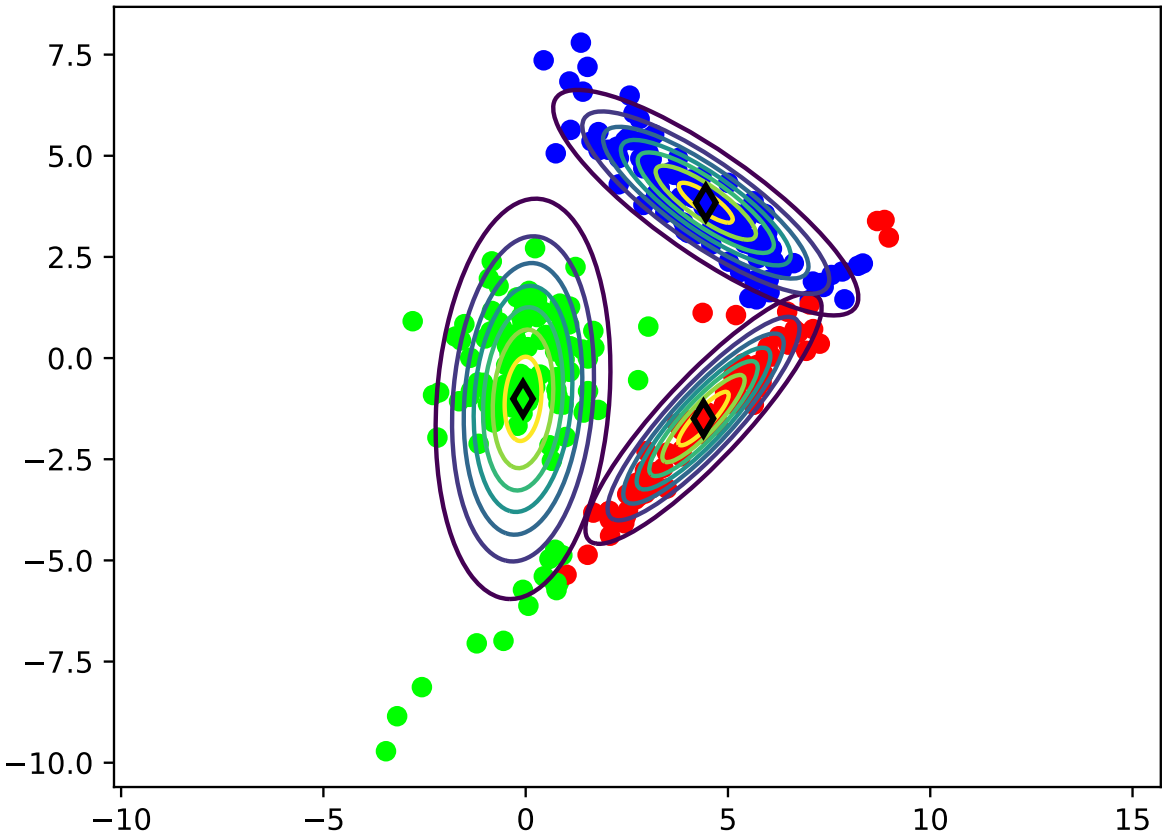
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



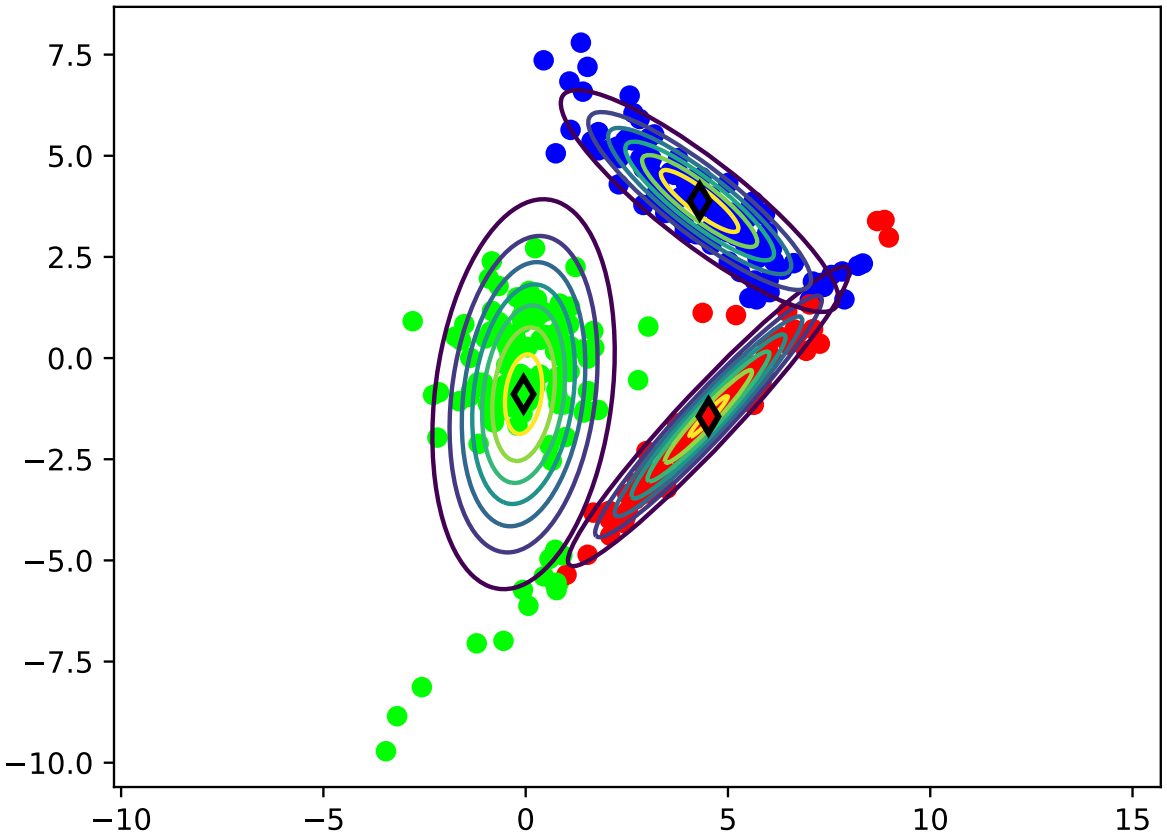
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



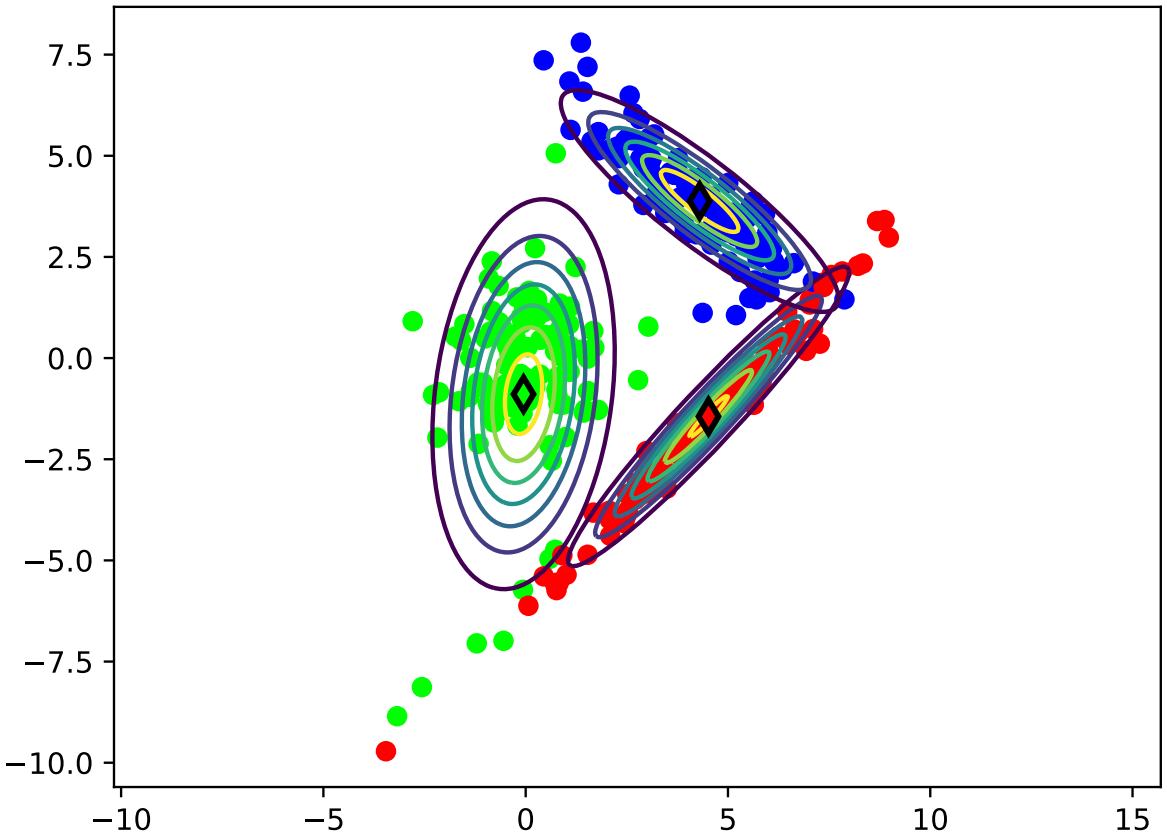
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



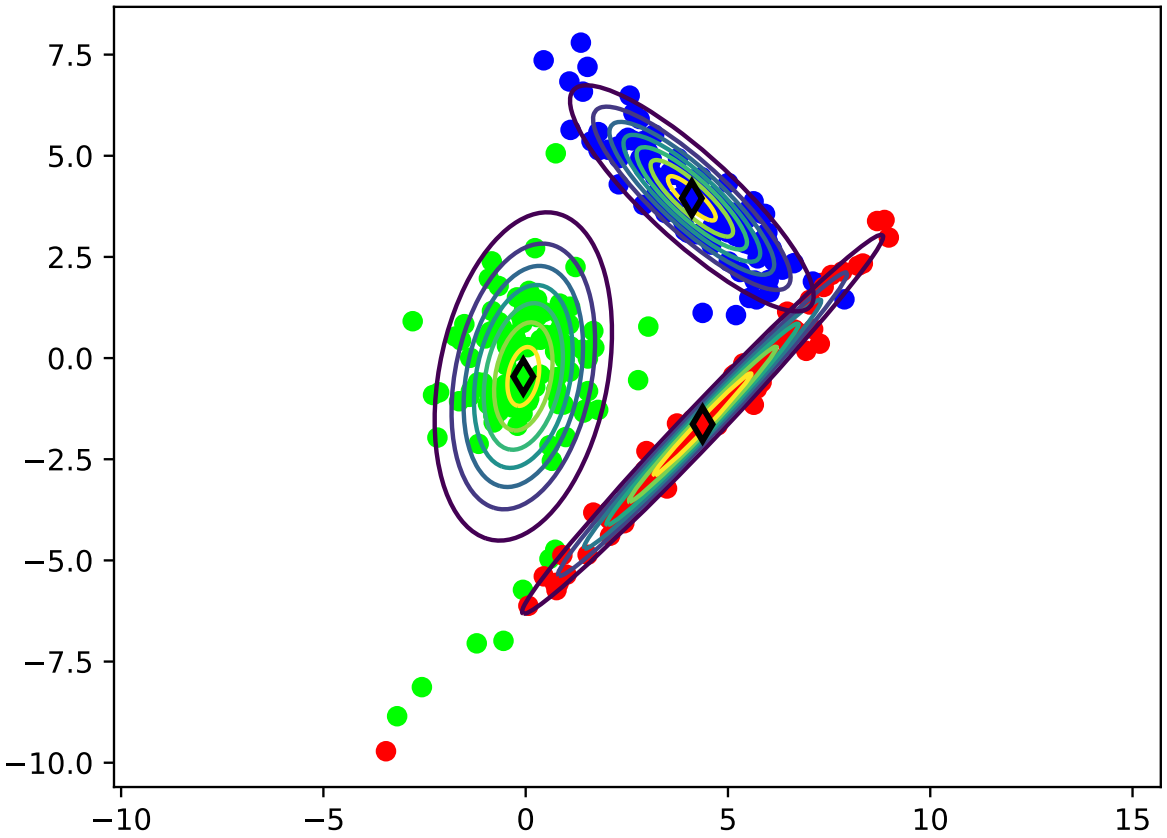
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



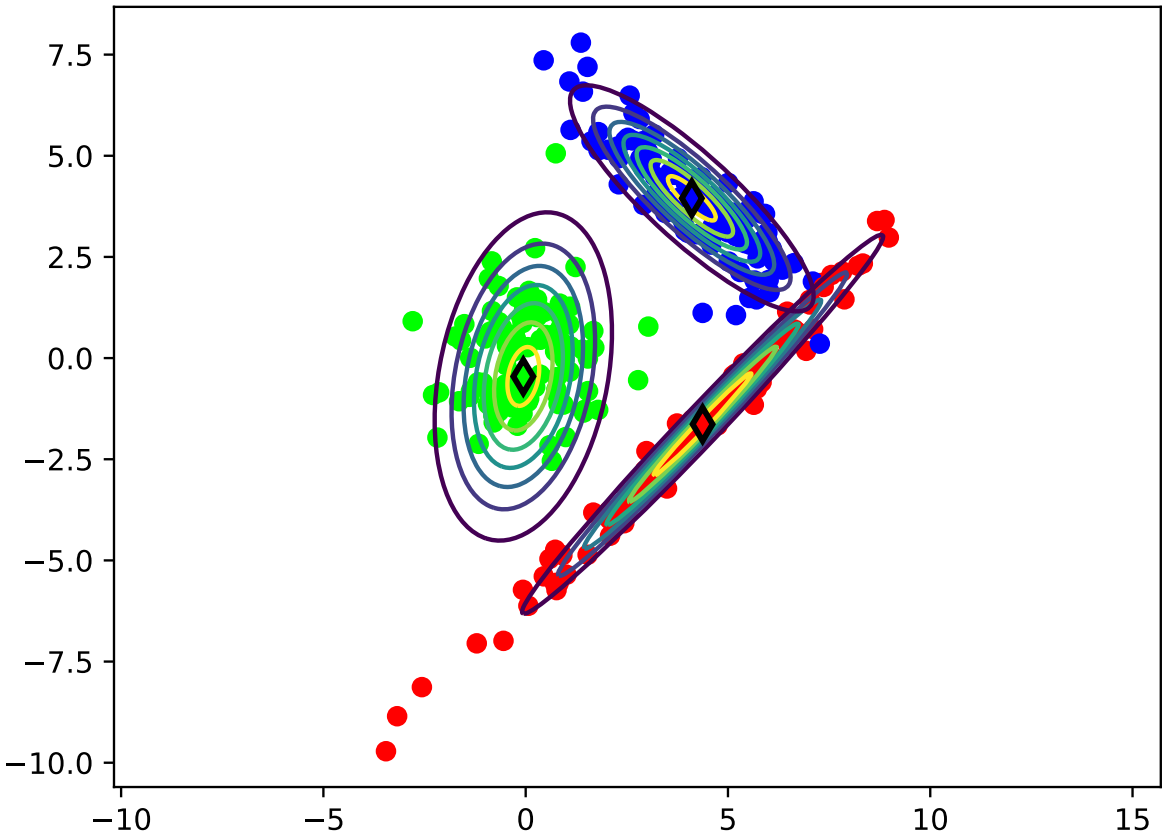
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



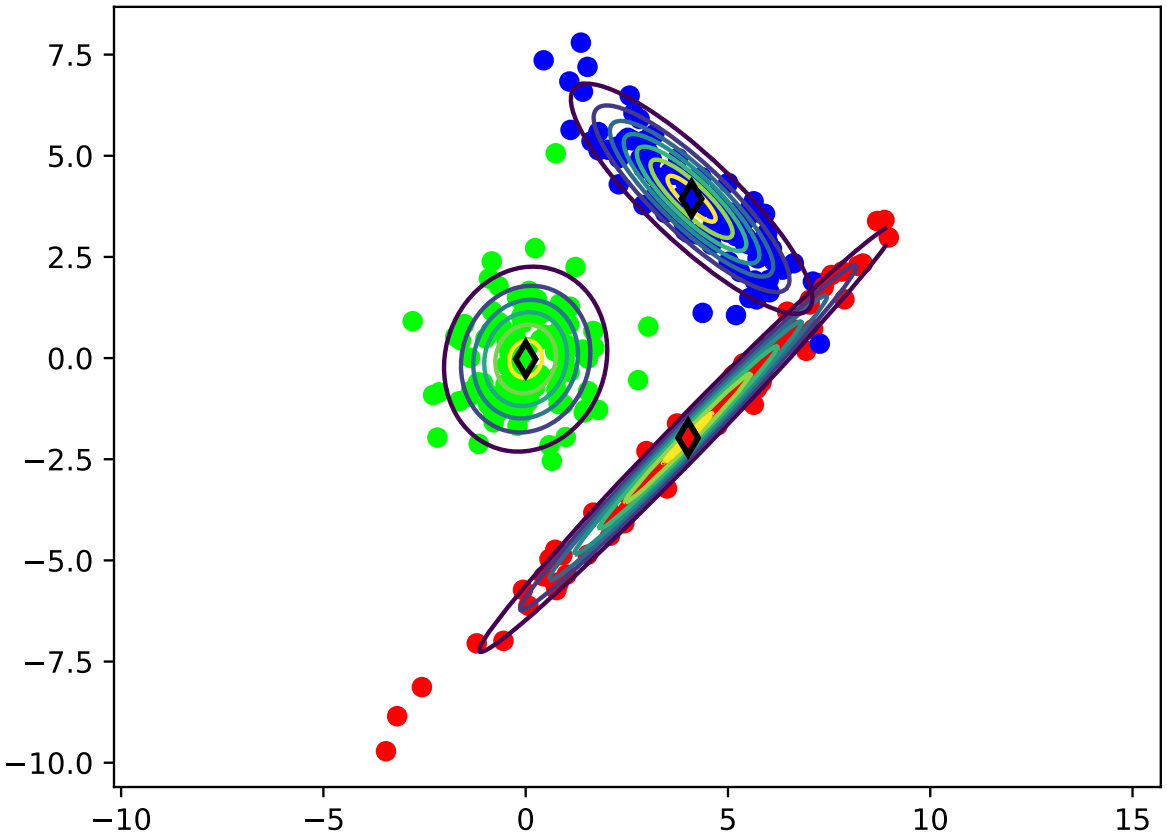
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



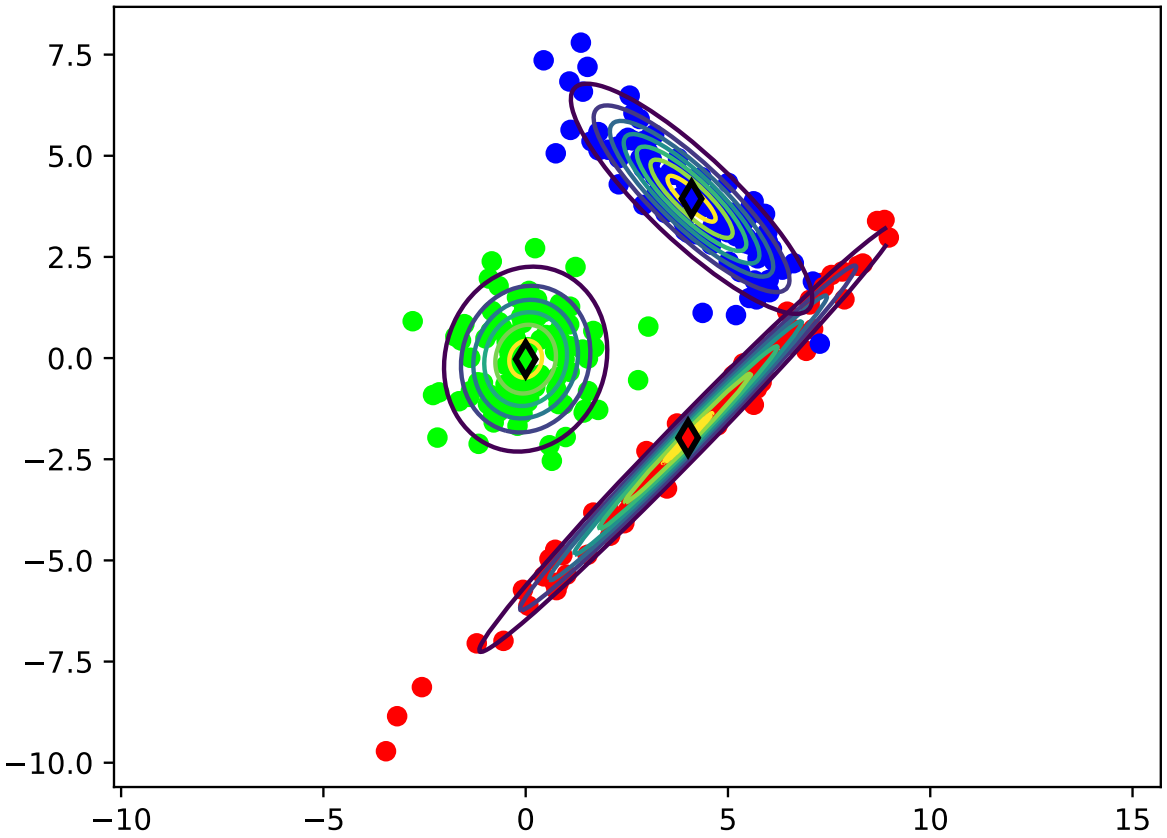
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



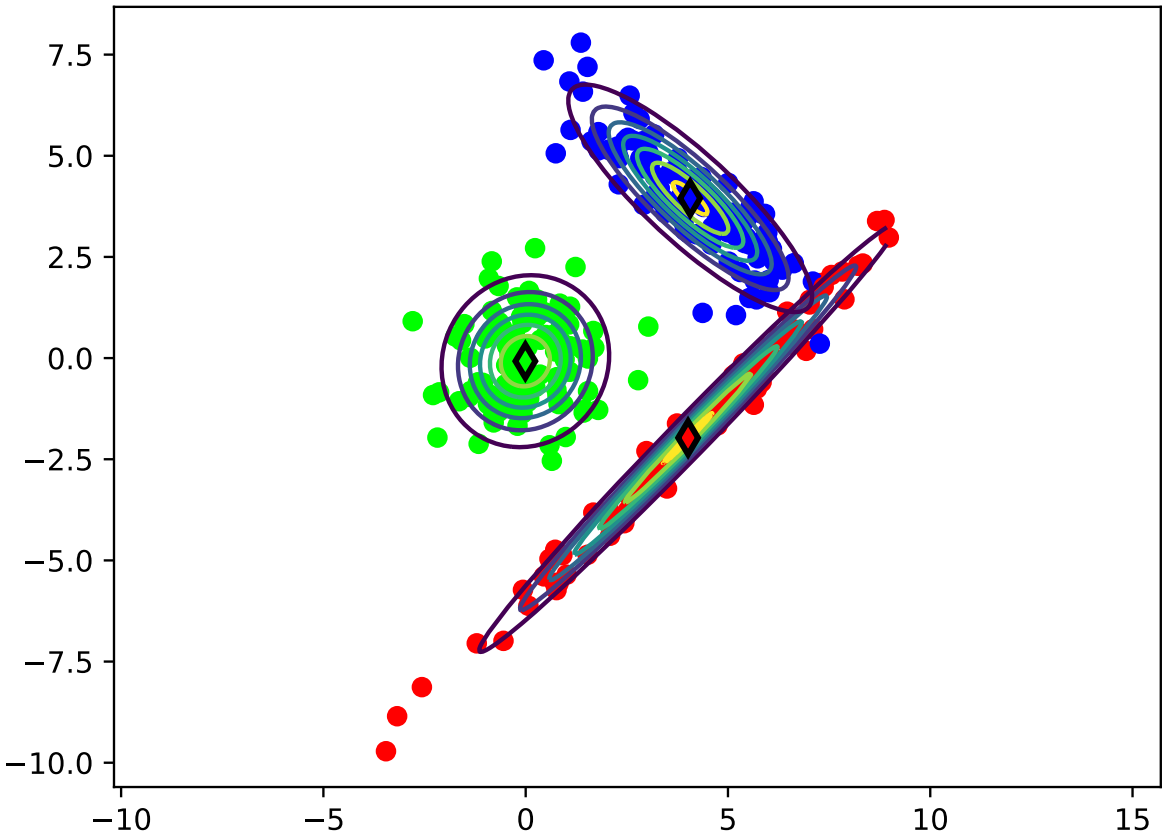
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



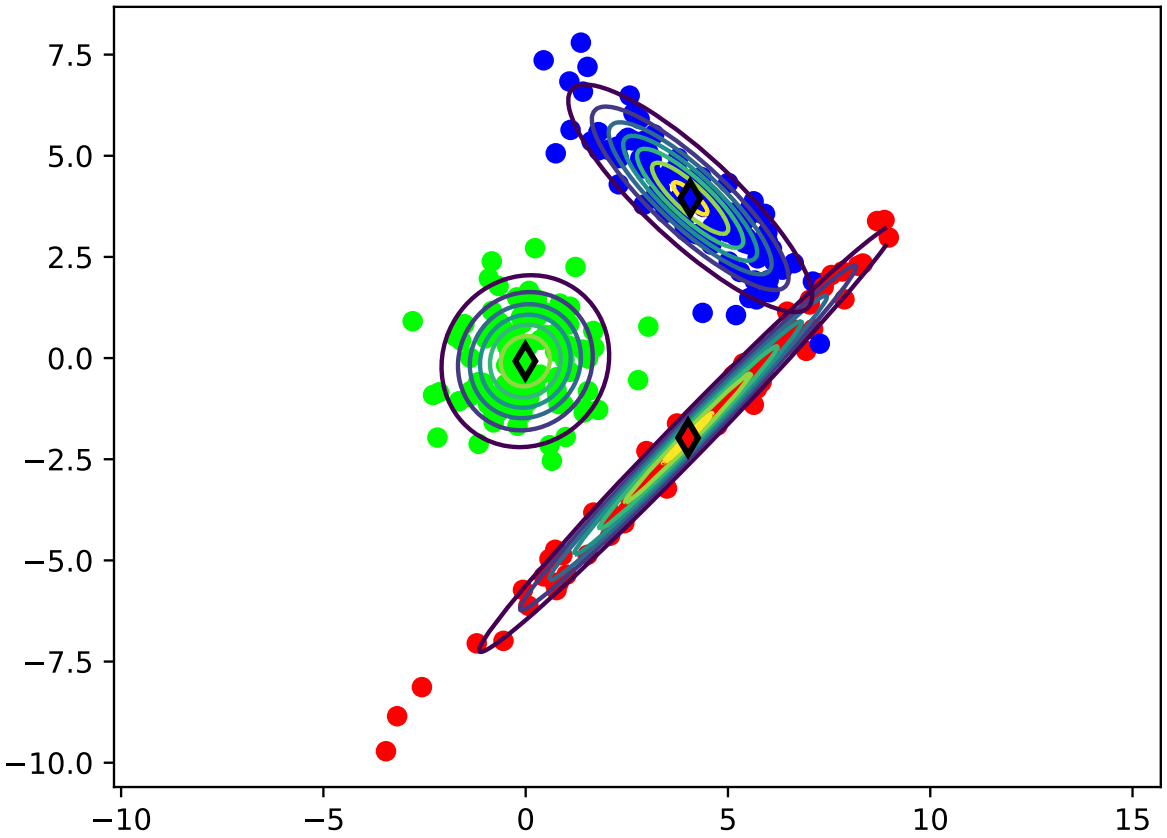
Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



Interpolating between k-means and GMM E-M (part 2)

We can interpolate algorithmically,
meaning we can create algorithms that have elements of both.
Here's something like k-means but with weights and covariances.



Summary of MLE part 2

Summary of MLE part 2

- Generative definition ("sampling story") of Gaussian mixture models (GMMs).
- PDF of GMMs.
- E-M in general, and what it guarantees.
- E-M for GMMs.
- Diagonal covariance GMM E-M.

MLE part 3

Graphical models

Recall the **generative story** for GMMs:

$$\begin{aligned} Y &\sim \text{Discrete}(\pi_1, \ldots, \pi_k) && \text{pick a Gaussian;} \\ \boldsymbol{X}|Y = j &\sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) && \text{pick a point.} \end{aligned}$$

Y is **latent/hidden/unobserved**, X is **observed**.

Graphical models

Recall the **generative story** for GMMs:

$$\begin{aligned} Y &\sim \text{Discrete}(\pi_1, \ldots, \pi_k) && \text{pick a Gaussian;} \\ \boldsymbol{X}|Y = j &\sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) && \text{pick a point.} \end{aligned}$$

Y is **latent/hidden/unobserved**, X is **observed**.

This model consists of random variables $(\boldsymbol{X}, Y)$
with a specific **conditional dependence structure**.

Graphical models

Recall the **generative story** for GMMs:

$$\begin{aligned} Y &\sim \text{Discrete}(\pi_1, \ldots, \pi_k) && \text{pick a Gaussian;} \\ \boldsymbol{X}|Y = j &\sim \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) && \text{pick a point.} \end{aligned}$$

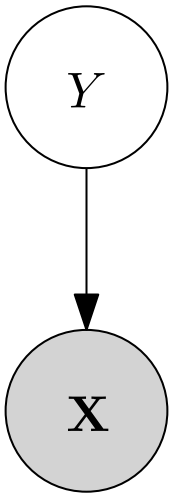
Y is **latent/hidden/unobserved**, X is **observed**.

This model consists of random variables $(\boldsymbol{X}, Y)$
with a specific **conditional dependence structure**.

A **graphical model** is a compact way of representing a family of r.v.'s, most notably their conditional dependencies.
Typically, the graphical model gives us

- a way to write down the (joint) probability distribution,
- guidance on how to sample.

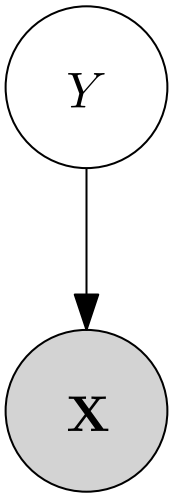
Graphical model for GMMs



Basic rules (there are many more):

- Nodes denote random variables. (Here we have $(\boldsymbol{X}, Y)$.)
- Edges denote conditional dependence. (Here $\boldsymbol{X}$ depends on Y.)
- Shaded nodes (e.g., $\boldsymbol{X}$) are observed; unshaded (Y) are unobserved.

Graphical model for GMMs



Basic rules (there are many more):

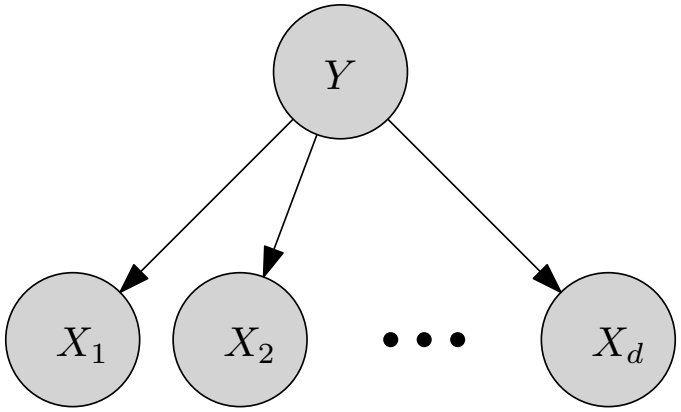
- Nodes denote random variables. (Here we have $(\boldsymbol{X}, Y)$.)
- Edges denote conditional dependence. (Here $\boldsymbol{X}$ depends on Y.)
- Shaded nodes (e.g., $\boldsymbol{X}$) are observed; unshaded (Y) are unobserved.

Likelihood of observations $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n)$ drawn from GMM:

$$p(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n) = \sum_{j_1 \in \{1,\ldots,k\}, \ldots, j_n \in \{1,\ldots,k\}} p(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n, Y_1 = j_1, \ldots, Y_n = j_n)$$

$$= \sum_{\substack{j_1 \in \{1,\ldots,k\} \\ \ldots \\ j_n \in \{1,\ldots,k\}}} \prod_{i=1}^{n} p(Y_i = j_i) p(\boldsymbol{X}_i | Y_i = j_i) = \prod_{i=1}^{n} \sum_{j_i=1}^{k} p(Y_i = j_i) p(\boldsymbol{X}_i | Y_i = j_i).$$

Graphical model for Naive Bayes

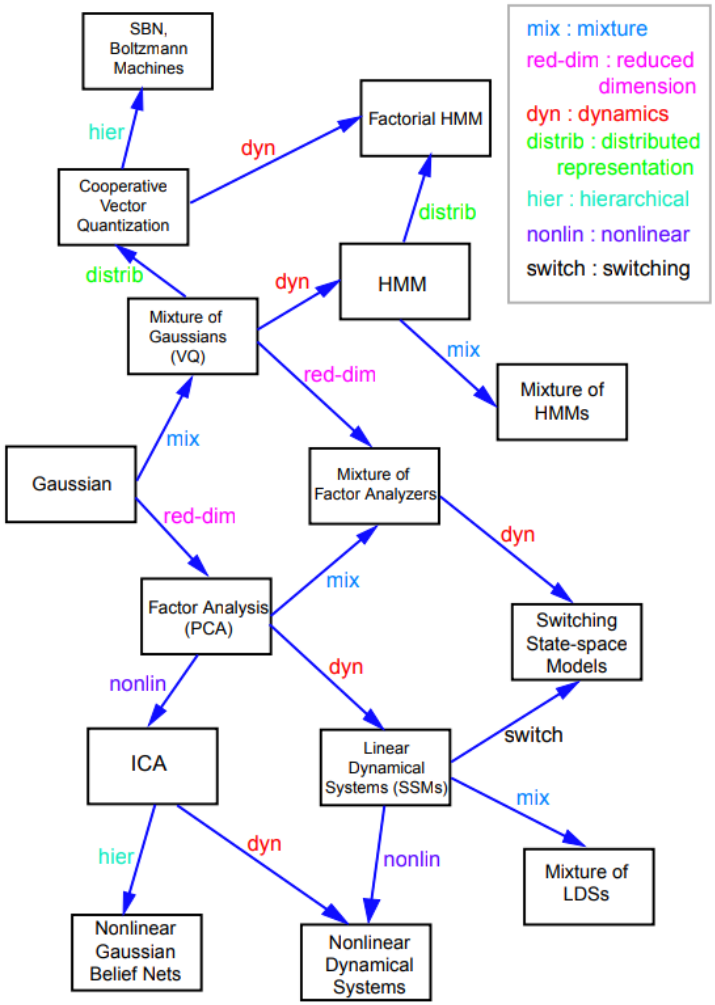


Recall the **Naive Bayes model:**
both inputs and outputs $(\boldsymbol{X}, Y)$ are observed (both are shaded!);
coordinates $(X_1, \ldots, X_d)$ are **conditionally independent given Y**
(as indicated by the arrows!).

Why do people use graphical models?

- Easy to interpret how data inter-depends, flows.
- Easy to add nodes and edges based on observations and beliefs.
- MLE, E-M, and others provide a well-weathered toolbox to fit them to data, sample, etc.
- Were very popular in the natural sciences (easy to encode domain knowledge); not yet clear how deep networks are displacing them (how to encode prior knowledge with deep networks?).

A Generative Model for Generative Models



(Figure by Zoubin Ghahramani.)

Hidden Markov Models

HMM basics.

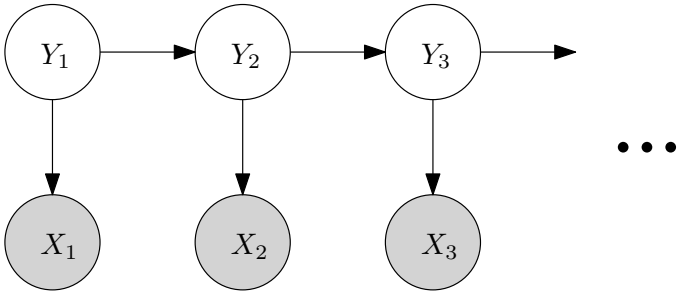
- As with GMM:
  - **Observed** random variables $(X_1, \ldots, X_n)$.
  - **Latent variables** $(Y_1, \ldots, Y_n)$.
  - Conditional independence of observations given latent variables: e.g., $p(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_t, Y_1, \ldots Y_t) = p(X_i|Y_i)$.

HMM basics.

- As with GMM:
  - **Observed** random variables $(X_1, \ldots, X_n)$.
  - **Latent variables** $(Y_1, \ldots, Y_n)$.
  - Conditional independence of observations given latent variables: e.g., $p(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_t, Y_1, \ldots Y_t) = p(X_i|Y_i)$.
- **Unlike GMMs:** $(Y_1, \ldots, Y_t)$ have dependencies!
  - **Markov assumption:** Y_{i+1} depends *only* on Y_i.

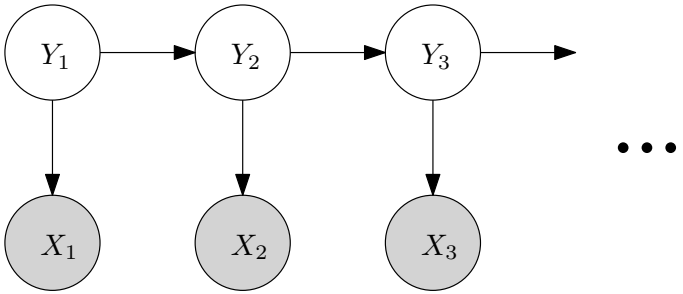
HMM basics.

- As with GMM:
  - **Observed** random variables $(X_1, \ldots, X_n)$.
  - **Latent variables** $(Y_1, \ldots, Y_n)$.
  - Conditional independence of observations given latent variables: e.g., $p(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_t, Y_1, \ldots Y_t) = p(X_i|Y_i)$.
- **Unlike GMMs:** $(Y_1, \ldots, Y_t)$ have dependencies!
  - **Markov assumption:** Y_{i+1} depends *only* on Y_i.
- Graphical model:



HMM likelihood

- Graphical model.



- Likelihood.

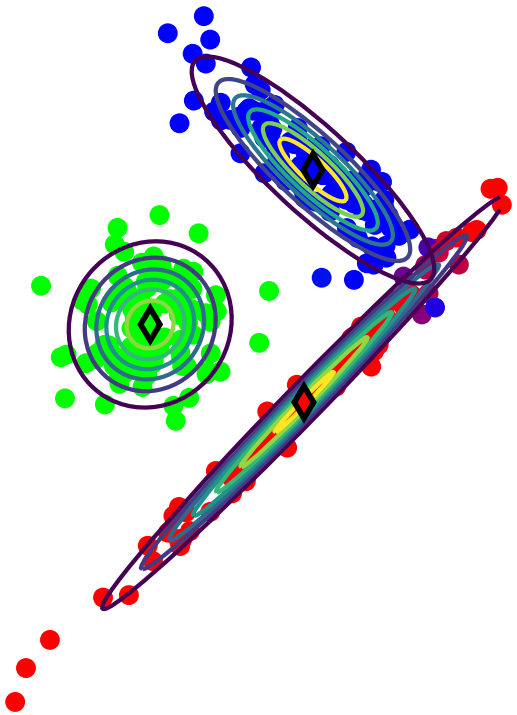
$$
\begin{aligned}
&p(X_1, \ldots, X_n, Y_1, \ldots, Y_n) \\
&= p(Y_1)p(X_1, \ldots, X_n, Y_2, \ldots, Y_n | Y_1) \\
&= p(Y_1)p(X_1|Y_1)p(X_2, \ldots, X_n, Y_2, \ldots, Y_n | Y_1) \\
&= p(Y_1)p(X_1|Y_1)p(X_2|Y_2)p(Y_2|Y_1)p(X_3, \ldots, X_n, Y_2, \ldots, Y_n | Y_2, Y_1) \\
&= p(Y_1)p(X_1|Y_1)p(X_2|Y_2)p(Y_2|Y_1)p(X_3, \ldots, X_n, Y_2, \ldots, Y_n | Y_2) \\
&= p(Y_1) \left(\prod_{i=1}^{n} p(X_i|Y_i) \right) \left(\prod_{i=2}^{n} p(Y_i|Y_{i-1}) \right).
\end{aligned}
$$

Hidden Markov Models: parameters.

- Still have parameters for $p(X_i|Y_i = j)$.
  E.g., if this is Gaussian, have parameters (μ_j, Σ_j).
- Still have parameters $(\pi_1, \ldots, \pi_k)$ for Y_1.
- For $(Y_2, \ldots, Y_k)$ have **transition probabilities** $p(Y_{i+1} = j'|Y_i = j)$.
  These are assumed **homogeneous/time-invariant:** e.g.,
  $p(Y_{i+1} = j'|Y_i = j) = p(Y_{i+2} = j'|Y_{i+1} = j)$.
  Write these as a matrix $A \in [0,1]^{k \times k}$: $p(Y_{i+1} = l|Y_i = k) = A_{jl}$.
- Depiction: like GMM, but have time series over hidden states!

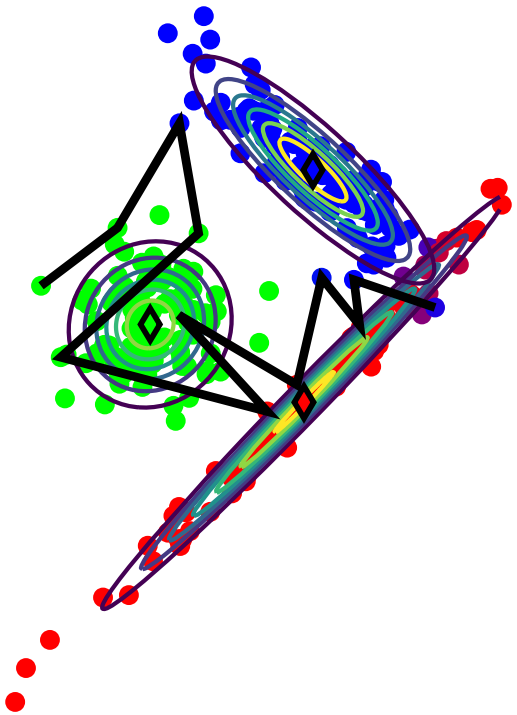
Hidden Markov Models: parameters.

- Still have parameters for $p(X_i|Y_i = j)$.
  E.g., if this is Gaussian, have parameters (μ_j, Σ_j).
- Still have parameters $(\pi_1, \ldots, \pi_k)$ for Y_1.
- For $(Y_2, \ldots, Y_k)$ have **transition probabilities** $p(Y_{i+1} = j'|Y_i = j)$.
  These are assumed **homogeneous/time-invariant:** e.g.,
  $p(Y_{i+1} = j'|Y_i = j) = p(Y_{i+2} = j'|Y_{i+1} = j)$.
  Write these as a matrix $A \in [0,1]^{k \times k}$: $p(Y_{i+1} = l|Y_i = k) = A_{jl}$.
- Depiction: like GMM, but have time series over hidden states!



Hidden Markov Models: parameters.

- Still have parameters for $p(X_i|Y_i = j)$.
  E.g., if this is Gaussian, have parameters (μ_j, Σ_j).
- Still have parameters $(\pi_1, \ldots, \pi_k)$ for Y_1.
- For $(Y_2, \ldots, Y_k)$ have **transition probabilities** $p(Y_{i+1} = j'|Y_i = j)$.
  These are assumed **homogeneous/time-invariant:** e.g.,
  $p(Y_{i+1} = j'|Y_i = j) = p(Y_{i+2} = j'|Y_{i+1} = j)$.
  Write these as a matrix $A \in [0,1]^{k \times k}$: $p(Y_{i+1} = l|Y_i = k) = A_{jl}$.
- Depiction: like GMM, but have time series over hidden states!



Hidden Markov Models: applications.

- Speech modeling; e.g., phonemes (/A/, /a/, /b/, ...).
  - Multivariate Gaussian is over amplitude or frequency window.
  - Transition matrix A allows self-loops!
    Very useful: imagine saying a word slowly.
- Sequence alignment in biology.
- See Murphy Chapter 17 for more applications.
  (Many are being replaced with DNNs, RNNs, ...!)

More E-M: learning Hidden Markov Models.

More E-M: learning Hidden Markov Models.

Another interpretation of E-M:
E-M maximizes the **expected complete log-likelihood**

$$\mathbb{E}_\theta \left[\ln p_\theta(X_1, \ldots, X_n, Y_1, \ldots, Y_n) | x_1, \ldots, x_n \right],$$

where "$\mathbb{E}_\theta$" means the distribution uses the learned parameters θ, and "$|x_1, \ldots, x_n$" means we condition on the observed data.

More E-M: learning Hidden Markov Models.

Another interpretation of E-M:
E-M maximizes the **expected complete log-likelihood**

$$\mathbb{E}_\theta \left[\ln p_\theta(X_1, \ldots, X_n, Y_1, \ldots, Y_n) | x_1, \ldots, x_n \right],$$

where "$\mathbb{E}_\theta$" means the distribution uses the learned parameters θ, and "$|x_1, \ldots, x_n$" means we condition on the observed data.

For GMMs, this becomes

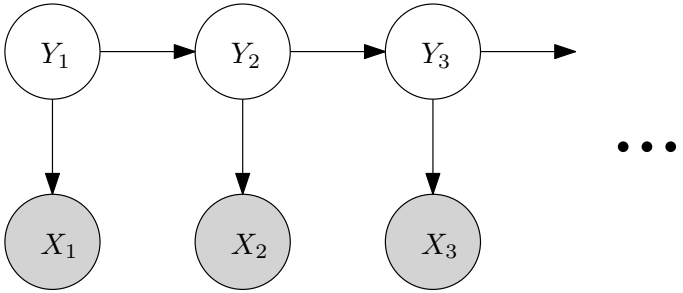
$$\begin{aligned} &= \mathbb{E}_\theta \left[\sum_{i=1}^{n} \ln p_\theta(X_1 = x_1, Y_1) \right] \\ &= \sum_{i=1}^{n} \sum_{j=1}^{k} p_\theta(Y_i = j | X_1 = x) \ln p_\theta(X_1 = x_1, Y_1 = j), \end{aligned}$$

which matches what we optimized before; specifically,

$$p_\theta(Y = j | X = x) = \frac{\pi_j p_\theta(X = x | Y = j)}{\sum_{l=1}^{k} \pi_l p_\theta(X = x | Y = l)}.$$

Expected complete log-likelihood for HMMs.

Graphical model:



Expected complete log-likelihood:

$$\mathbb{E}_\theta \left[\ln p_\theta(X_1, \ldots, X_n, Y_1, \ldots, Y_n) | x_1, \ldots, x_n \right]$$

$$= \mathbb{E}_\theta \left[\ln \left(p_\theta(Y_1) \left(\prod_{i=1}^{n} p_\theta(X_i|Y_i) \right) \left(\prod_{i=2}^{n} p_\theta(Y_i|Y_{i-1}) \right) \right) \right]$$

$$= \sum_{j=1}^{k} p_\theta(Y_1|x_1, \ldots, x_k) \ln \pi_j + \sum_{i=1}^{n} \sum_{j=1}^{k} p_\theta(Y_i = j|x_1, \ldots, x_n) \ln p_\theta(X_i|Y_i)$$

$$+ \sum_{i \geq 2} \sum_{j,j'=1}^{k} p_\theta(Y_i = j, Y_{i-1} = j'|x_1, \ldots, x_n) \ln p_\theta(Y_i = j|Y_{i-1} = j')$$

Expected complete log-likelihood for HMMs.

Expected complete log-likelihood:

$$\sum_{j=1}^{k} p_\theta(Y_1|x_1,\ldots,x_k)\ln\pi_j + \sum_{i=1}^{n}\sum_{j=1}^{k} p_\theta(Y_i=j|x_1,\ldots,x_n)\ln p_\theta(X_i|Y_i=j)$$
$$+\sum_{i\geq 2}\sum_{j,j'=1}^{k} p_\theta(Y_i=j,Y_{i-1}=j'|x_1,\ldots,x_n)\ln p_\theta(Y_i=j|Y_{i-1}=j').$$

- M step for observable $p_\theta(X_i|Y_i)$ similar to mixture case; replace old $\sum_i A'_{ij}$ with new $\sum_i p_\theta(Y_i|x_1,\ldots,x_n)$.
- M step for π and $A_{jj'} = p_\theta(Y_i=j|Y_{i-1}=j')$ also easy.
- Real annoyance is computing the conditional probabilities (E step)!

E-step for HMMs.

- Need to compute $p_\theta(Y_1|x_1,\ldots,x_k)$, $p_\theta(Y_i=j|x_1,\ldots,x_n)$, $p_\theta(Y_i=j,Y_{i-1}=j'|x_1,\ldots,x_n)\ln p_\theta(Y_i=j|Y_{i-1}=j')$.
- Boils down to a bunch of games with conditioning.
  Kindof cool but I decided to skip.
  See Murphy book (Chapter 17) for details.

Summary of HMMs.

- Graphical models give a succinct way to specify conditional dependencies of random variables.
- Expected complete log likelihood is another way to reason about E-M.
- HMMs allow dependence amongst latent variables.

Variational methods

Why isn't E-M enough?

Recall the E-M updates:

- **Update responsibilities** $R_{ij} := p_{\boldsymbol{\theta}}(y_i = j | \boldsymbol{x}_i)$;
- **Update parameters** $\boldsymbol{\theta} := \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R})$.

Guarantee: $\mathcal{L}(\boldsymbol{\theta}_t) = \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) \leq \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R}_{t+2}) = \mathcal{L}(\boldsymbol{\theta}_{t+1})$.

Why isn't E-M enough?

Recall the E-M updates:

- **Update responsibilities** $R_{ij} := p_{\boldsymbol{\theta}}(y_i = j | \boldsymbol{x}_i)$;
- **Update parameters** $\boldsymbol{\theta} := \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R})$.

Guarantee: $\mathcal{L}(\boldsymbol{\theta}_t) = \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) \leq \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R}_{t+2}) = \mathcal{L}(\boldsymbol{\theta}_{t+1})$.

What if we can't efficiently compute $p_{\boldsymbol{\theta}}(y_i = j | \boldsymbol{x}_i)$?
(Example: "Latent Dirichlet Allocation", Blei-Jordan-Ng.)

Why isn't E-M enough?

Recall the E-M updates:

- **Update responsibilities** $R_{ij} := p_{\boldsymbol{\theta}}(y_i = j | \boldsymbol{x}_i)$;
- **Update parameters** $\boldsymbol{\theta} := \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathcal{L}(\boldsymbol{\theta}; \boldsymbol{R})$.

Guarantee: $\mathcal{L}(\boldsymbol{\theta}_t) = \mathcal{L}(\boldsymbol{\theta}_t; \boldsymbol{R}_{t+1}) \leq \mathcal{L}(\boldsymbol{\theta}_{t+1}; \boldsymbol{R}_{t+2}) = \mathcal{L}(\boldsymbol{\theta}_{t+1})$.

What if we can't efficiently compute $p_{\boldsymbol{\theta}}(y_i = j | \boldsymbol{x}_i)$?
(Example: "Latent Dirichlet Allocation", Blei-Jordan-Ng.)

A standard approach is **variational approximation**:

- Replace $p(\boldsymbol{y}|\boldsymbol{x})$ with some simpler $q(\boldsymbol{y})$;
  a common way is to cut edges in p's graphical model.
- q is still chosen with a good deal of flexibility so that it well-approximates p.

Core of variational bayes method

Let's replace $p(\boldsymbol{y}|\boldsymbol{x})$ with a model family $\{q_\phi : \phi \in \Phi\}$:

$$\begin{aligned}
\ln p(\boldsymbol{x}) &= \int q_\phi(\boldsymbol{y}) \ln p(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{y} = \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}) p(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y}|\boldsymbol{x})} \right) \mathrm{d}\boldsymbol{y} \\
&= \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}, \boldsymbol{y}) q_\phi(\boldsymbol{y})}{p(\boldsymbol{y}|\boldsymbol{x}) q_\phi(\boldsymbol{y})} \right) \mathrm{d}\boldsymbol{y} \\
&= \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}, \boldsymbol{y})}{q_\phi(\boldsymbol{y})} \right) \mathrm{d}\boldsymbol{y} + \int q_\phi(\boldsymbol{y}) \ln \left(\frac{q_\phi(\boldsymbol{y})}{p(\boldsymbol{y}|\boldsymbol{x})} \right) \mathrm{d}\boldsymbol{y} \\
&= \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}, \boldsymbol{y})}{q_\phi(\boldsymbol{y})} \right) \mathrm{d}\boldsymbol{y} + K(q_\phi, p(\boldsymbol{y}|\boldsymbol{x})),
\end{aligned}$$

which is ≥ 0 since KL divergence $K \geq 0$;

Core of variational bayes method

Let's replace $p(\boldsymbol{y}|\boldsymbol{x})$ with a model family $\{q_\phi : \phi \in \Phi\}$:

$$\begin{aligned}
\ln p(\boldsymbol{x}) &= \int q_\phi(\boldsymbol{y}) \ln p(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{y} = \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}) p(\boldsymbol{y}|\boldsymbol{x})}{p(\boldsymbol{y}|\boldsymbol{x})} \right) \mathrm{d}\boldsymbol{y} \\
&= \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}, \boldsymbol{y}) q_\phi(\boldsymbol{y})}{p(\boldsymbol{y}|\boldsymbol{x}) q_\phi(\boldsymbol{y})} \right) \mathrm{d}\boldsymbol{y} \\
&= \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}, \boldsymbol{y})}{q_\phi(\boldsymbol{y})} \right) \mathrm{d}\boldsymbol{y} + \int q_\phi(\boldsymbol{y}) \ln \left(\frac{q_\phi(\boldsymbol{y})}{p(\boldsymbol{y}|\boldsymbol{x})} \right) \mathrm{d}\boldsymbol{y} \\
&= \int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{x}, \boldsymbol{y})}{q_\phi(\boldsymbol{y})} \right) \mathrm{d}\boldsymbol{y} + K(q_\phi, p(\boldsymbol{y}|\boldsymbol{x})),
\end{aligned}$$

which is ≥ 0 since KL divergence $K \geq 0$; indeed,

$$\begin{aligned}
K(q_\phi, p(\boldsymbol{y}|\boldsymbol{x})) &= -\int q_\phi(\boldsymbol{y}) \ln \left(\frac{p(\boldsymbol{y}|\boldsymbol{x})}{q_\phi(\boldsymbol{y})} \right) \mathrm{d}\boldsymbol{y} \\
&\geq -\ln \left(\int q_\phi(\boldsymbol{y}) \frac{p(\boldsymbol{y}|\boldsymbol{x})}{q_\phi(\boldsymbol{y})} \, \mathrm{d}\boldsymbol{y} \right) = \ln 1 = 0.
\end{aligned}$$

E-M from a variational perspective

We've derived

$$\sum_{i=1}^{n} p_\theta(\boldsymbol{x}_i) = \sum_{i=1}^{n} \left(\int q_\phi(\boldsymbol{y}_i) \ln \left(\frac{p_\theta(\boldsymbol{x}_i)}{q_\phi(\boldsymbol{y}_i)} \right) \mathrm{d}\boldsymbol{y}_i + K\left(q_\phi(\boldsymbol{y}_i), p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)\right) \right),$$

where KL divergence ≥ 0, and $= 0$ when $q_\phi(\boldsymbol{y}_i) = p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)$.

E-M from a variational perspective

We've derived

$$\sum_{i=1}^{n} p_\theta(\boldsymbol{x}_i) = \sum_{i=1}^{n} \left(\int q_\phi(\boldsymbol{y}_i) \ln \left(\frac{p_\theta(\boldsymbol{x}_i)}{q_\phi(\boldsymbol{y}_i)} \right) \mathrm{d}\boldsymbol{y}_i + K\left(q_\phi(\boldsymbol{y}_i), p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)\right) \right),$$

where KL divergence ≥ 0, and $= 0$ when $q_\phi(\boldsymbol{y}_i) = p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)$.

Approach:

- Choose $\{q_\phi : \phi \in \Phi\}$ so that can pick $q_\phi \approx p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)$ (can pick a q_{ϕ_i} for each $p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)$!).
- Simultaneously, finding this optimal q_ϕ should be easier than computing p_θ.
- The resulting algorithm can be very simple and clean; see "Latent Dirichlet Allocation" (Blei-Jordan-Ng, 2003).

E-M from a variational perspective

We've derived

$$\sum_{i=1}^{n} p_\theta(\boldsymbol{x}_i) = \sum_{i=1}^{n} \left(\int q_\phi(\boldsymbol{y}_i) \ln \left(\frac{p_\theta(\boldsymbol{x}_i)}{q_\phi(\boldsymbol{y}_i)} \right) \mathrm{d}\boldsymbol{y}_i + K\left(q_\phi(\boldsymbol{y}_i), p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)\right) \right),$$

where KL divergence ≥ 0, and $= 0$ when $q_\phi(\boldsymbol{y}_i) = p_\theta(\boldsymbol{y}_i|\boldsymbol{x}_i)$.

Further remarks:

- This is a major topic in graphical models (and statistics), but we won't go deeper into it;
  we will use a version of it to motivate VAEs, but that's it.
- One standard approach takes $q_\phi(\boldsymbol{y}) = \prod_j q_{\phi,j}(\boldsymbol{y}_j)$
  (coordinate-weise factorization; called "mean-field").

Kernel Density Estimates (KDE / “Parzen windows”)

Kernel density estimates ("Parzen windows")

Let's cover one more standard distribution modeling tool.

Kernel density estimates ("Parzen windows")

Let's cover one more standard distribution modeling tool.

- Let random draw $(\boldsymbol{x}_i)_{i=1}^n$ from some density be given.

Kernel density estimates ("Parzen windows")

Let's cover one more standard distribution modeling tool.

- Let random draw $(\boldsymbol{x}_i)_{i=1}^n$ from some density be given.
- Define $\hat{p}(\boldsymbol{x}) := \frac{1}{n} \sum_{i=1}^n k\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)$,
  where k is a **kernel function** (not the SVM one!),
  h is the "bandwidth"; for example:

Kernel density estimates ("Parzen windows")

Let's cover one more standard distribution modeling tool.

- Let random draw $(\boldsymbol{x}_i)_{i=1}^n$ from some density be given.
- Define $\hat{p}(\boldsymbol{x}) := \frac{1}{n}\sum_{i=1}^n k\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)$,
  where k is a **kernel function** (not the SVM one!),
  h is the "bandwidth"; for example:
  - Gaussian: $k(\boldsymbol{z}) \propto \exp\left(-\|\boldsymbol{z}\|^2/2\right)$;
  - Epanechnikov: $k(\boldsymbol{z}) \propto \max\{0, 1-\|\boldsymbol{z}\|^2\}$.

Kernel density estimates: illustration

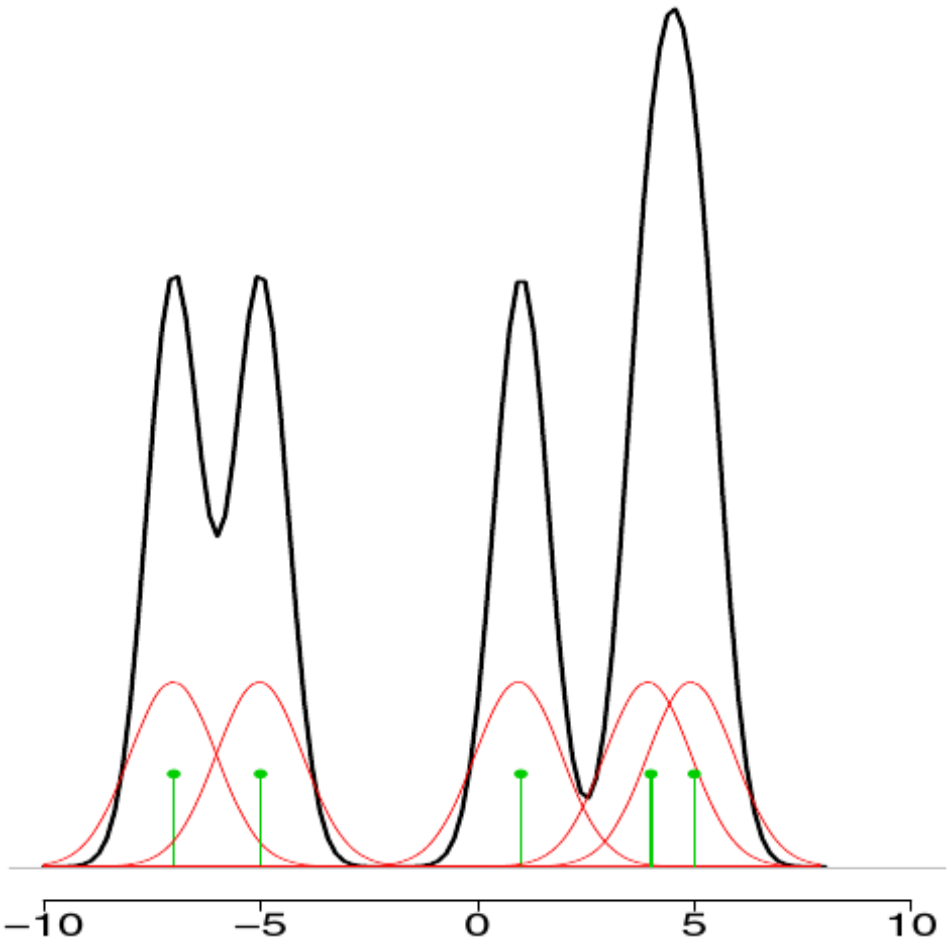


FIGURE 6.4. A kernel density estimator $\widehat{f}_n$. At each point x, $\widehat{f}_n(x)$ is the average of the kernels centered over the data points X_i. The data points are indicated by short vertical bars. The kernels are not drawn to scale.

(From Larry Wasserman's "All of nonparametric statistics".)

Kernel density estimates (KDE) vs GMM.

- KDE fits (basically) any density as $n \to \infty$
  (and variance ("bandwidth") on kernel is tuned).
- GMM fits any density as $k \to \infty$
  (and variance is tuned).
- GMM can succinctly fit some densities
  for which KDE needs many samples.
- KDE is computationally trivial;
  GMM a mess.

Summary of MLE part 3

Summary (of MLE part 3)

Main things to know:

- Graphical model concept.
- Variational Bayes concept.
- Kernel density estimates.

(I will not test you on HMMs, but many ML classes make a big deal out of them.)