Gaussian mean-shift algorithms

Miguel Á. Carreira-Perpiñán

Dept. of Computer Science & Electrical Engineering, OGI/OHSU

http://www.csee.ogi.edu/~miguel

Gaussian mean-shift (GMS) as an EM algorithm

- Gaussian mean-shift (GMS) as an EM algorithm
- Acceleration strategies for GMS image segmentation
- Gaussian blurring mean-shift (GBMS)

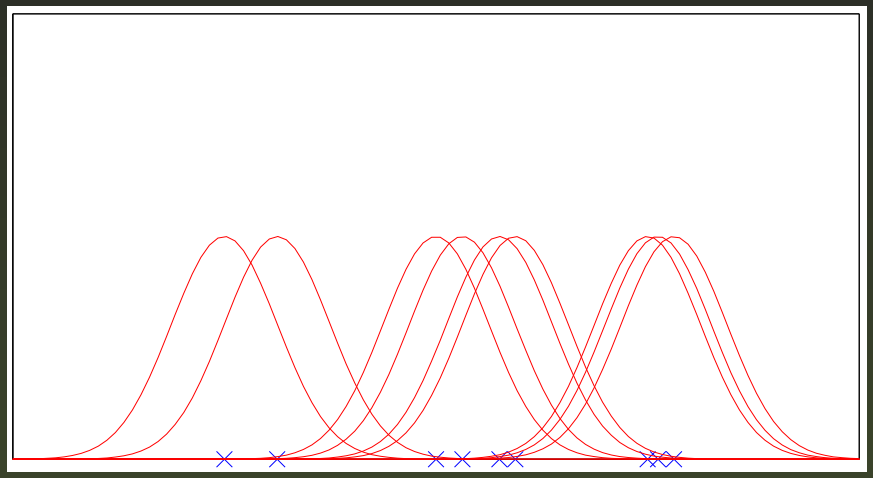
Gaussian mixtures, kernel density estimates

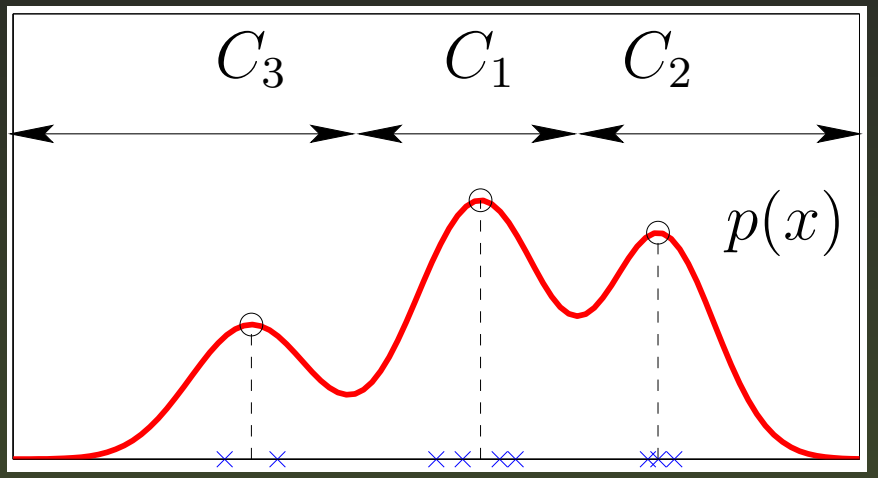
Given a dataset $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N \subset \mathbb{R}^D$, define a Gaussian kernel density estimate with bandwidth σ:

$$p(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} K\left(\left\|\frac{\mathbf{x}-\mathbf{x}_n}{\sigma}\right\|^2\right) \qquad K(t) \propto e^{-t/2}.$$

- Other kernels:
  - Epanechnikov: $K(t) \propto \begin{cases} 1-t, & t \in [0,1) \\ 0, & t \geq 1 \end{cases}$ (finite support).
  - Student's t: $K(t) \propto \left(1+\frac{t}{\alpha}\right)^{-\frac{\alpha+D}{2}}$.
- Useful way to represent the density of a data set.
- Extremely popular in machine learning and statistics.
- Objective here: find modes of $p(\mathbf{x})$.

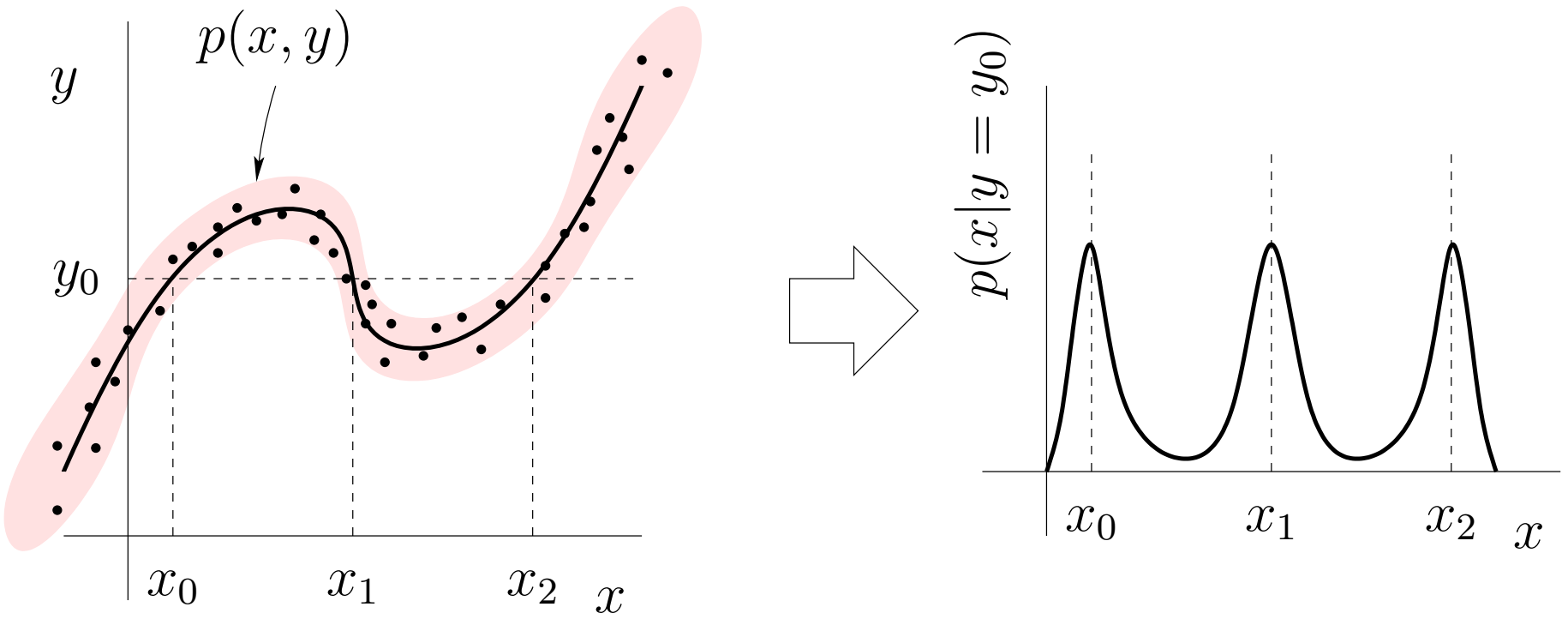
Why find modes? Nonparametric clustering





- Modes $\leftrightarrow$ clusters.
- Assign each point $\mathbf{x}$ to a mode.
- Nonparametric:
  - no model for the clusters' shape
  - no predetermined number of clusters.

Why find modes? Multivalued mappings



- Given a density model for (x, y), represent a multivalued mapping $y \to x$ by modes of the conditional distribution $p(x|y)$. Map y_0 to $\{x_0, x_1, x_2\}$.
- $p(x, y)$ is a Gaussian mixture $\Rightarrow$ $p(x|y)$ is a Gaussian mixture.

Mode-finding algorithms

- Idea: start a hill-climbing algorithm from every centroid. This typically finds all modes in practice (Carreira-Perpiñán '00).
- Hill-climbing algorithms:
  - Gradient ascent
  - Newton's method
  - etc.
  - Fixed-point iteration: the mean-shift algorithm.
    It is based on ideas by Fukunaga & Hostetler '75
    (also Cheng '95, Carreira-Perpiñán '00, Comaniciu & Meer '02, etc.).

The mean-shift algorithm

It is obtained by reorganising the stationary point equation $\nabla p(\mathbf{x}) = \mathbf{0}$ as a fixed-point iteration: $\mathbf{x}^{(\tau+1)} = \mathbf{f}(\mathbf{x}^{(\tau)})$.
For example, for an isotropic kde $p(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} K\left(\left\|\frac{\mathbf{x}-\mathbf{x}_n}{\sigma}\right\|^2\right)$:

$$\nabla p(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} K'\left(\left\|\frac{\mathbf{x}-\mathbf{x}_n}{\sigma}\right\|^2\right)\frac{2}{\sigma^2}(\mathbf{x}-\mathbf{x}_n)$$

$$= \frac{2}{N\sigma^2}\sum_{n=1}^{N} K'\left(\left\|\frac{\mathbf{x}-\mathbf{x}_n}{\sigma}\right\|^2\right)\mathbf{x} -$$

$$\frac{2}{N\sigma^2}\sum_{n=1}^{N} K'\left(\left\|\frac{\mathbf{x}-\mathbf{x}_n}{\sigma}\right\|^2\right)\mathbf{x}_n = \mathbf{0}$$

$$\Rightarrow \mathbf{x} = \sum_{n=1}^{N} \frac{K'\left(\left\|\frac{\mathbf{x}-\mathbf{x}_n}{\sigma}\right\|^2\right)}{\sum_{n'=1}^{N} K'\left(\left\|\frac{\mathbf{x}-\mathbf{x}_{n'}}{\sigma}\right\|^2\right)}\mathbf{x}_n.$$

The mean-shift algorithm (cont.)

Gaussian, full covariance:

$$\mathbf{f}(\mathbf{x}) = \left(\sum_{n=1}^{N} p(n|\mathbf{x}) \boldsymbol{\Sigma}_n^{-1} \right)^{-1} \sum_{n=1}^{N} p(n|\mathbf{x}) \boldsymbol{\Sigma}_n^{-1} \mathbf{x}_n$$

Gaussian, isotropic:

$$\mathbf{f}(\mathbf{x}) = \sum_{n=1}^{N} p(n|\mathbf{x}) \mathbf{x}_n \qquad p(n|\mathbf{x}) = \frac{e^{-\frac{1}{2} \left\| \frac{\mathbf{x}-\mathbf{x}_n}{\sigma} \right\|^2}}{\sum_{n'=1}^{N} e^{-\frac{1}{2} \left\| \frac{\mathbf{x}-\mathbf{x}_{n'}}{\sigma} \right\|^2}}$$

Other kernel, isotropic:

$$\mathbf{f}(\mathbf{x}) = \sum_{n=1}^{N} \frac{K'\left(\left\| \frac{\mathbf{x}-\mathbf{x}_n}{\sigma} \right\|^2 \right)}{\sum_{n'=1}^{N} K'\left(\left\| \frac{\mathbf{x}-\mathbf{x}_{n'}}{\sigma} \right\|^2 \right)} \mathbf{x}_n .$$

- $\mathbf{f}^\infty = \mathbf{f} \circ \mathbf{f} \circ \ldots$ maps points in $\mathbb{R}^D$ to stationary points of p.
- It is a particularly simple algorithm, with no step size.

Gaussian mean-shift (GMS) as a clustering algorithm

- $\mathbf{x}_n$, $\mathbf{x}_m$ in same cluster if they converge to same mode
- nonparametric clustering, able to deal with complex cluster shapes; σ determines the number of clusters
- popular in computer vision (segmentation, tracking)
  (Comaniciu & Meer)

Segmentation example (one data point $\mathbf{x} = (i, j, I)$ per pixel, where $(i, j) =$ location and $I =$ intensity or colour):




Pseudocode: GMS clustering

for $n \in \{1, \ldots, N\}$ — For each data point

$\mathbf{x} \leftarrow \mathbf{x}_n$ — Starting point

repeat — Iteration loop

$$\forall n\text{: } p(n|\mathbf{x}) \leftarrow \frac{\exp\left(-\frac{1}{2}\left\|(\mathbf{x}-\mathbf{x}_n)/\sigma\right\|^2\right)}{\sum_{n'=1}^{N}\exp\left(-\frac{1}{2}\left\|(\mathbf{x}-\mathbf{x}_{n'})/\sigma\right\|^2\right)}$$ — Post. prob. (E step)

$$\mathbf{x} \leftarrow \sum_{n=1}^{N} p(n|\mathbf{x})\mathbf{x}_n$$ — Update $\mathbf{x}$ (M step)

until $\mathbf{x}$'s update $<$ `tol`

$\mathbf{z}_n \leftarrow \mathbf{x}$ — Mode

end

connected-components($\{\mathbf{z}_n\}_{n=1}^{N}$,`min_diff`) — Clusters

GMS is an EM algorithm

Define the following artificial maximum-likelihood problem (Carreira-Perpiñán & Williams '03):

- Model: Gaussian mixture that moves as a rigid body ($\{\pi_n, \mathbf{x}_n, \boldsymbol{\Sigma}_n\}_{n=1}^N$ fixed; only parameter is $\mathbf{x}$)

$$q(\mathbf{y}|\mathbf{x}) = \sum_{n=1}^{N} \pi_n \left|2\pi\boldsymbol{\Sigma}_n\right|^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{y}-(\mathbf{x}_n-\mathbf{x}))^T \boldsymbol{\Sigma}_n^{-1}(\mathbf{y}-(\mathbf{x}_n-\mathbf{x}))}.$$

- Data: just one point at the origin: $\mathbf{y} = \mathbf{0}$.

Then, the resulting expectation-maximisation (EM) algorithm that maximises the log-likelihood wrt the parameter $\mathbf{x}$ (translation of the GM) is formally identical to the GMS step.

- E step: update posterior probabilities $p(n|\mathbf{x})$.
- M step: update parameter $\mathbf{x}$.

GMS is an EM algorithm (cont.)

The M step maximises the following lower bound on $\log p$:

$$\varrho(\mathbf{x}) = \log p(\mathbf{x}^{(\tau)}) - Q(\mathbf{x}^{(\tau)}|\mathbf{x}^{(\tau)}) + Q(\mathbf{x}|\mathbf{x}^{(\tau)})$$

with

$$\varrho(\mathbf{x}^{(\tau)}) = \log p(\mathbf{x}^{(\tau)}) \text{ and } \varrho(\mathbf{x}) \leq \log p(\mathbf{x}) \; \forall \mathbf{x} \in \mathbb{R}^D$$

where

$$Q(\mathbf{x}|\mathbf{x}^{(\tau)}) = \text{constant} - \frac{1}{2\sigma^2} \sum_{n=1}^{N} p(n|\mathbf{x}^{(\tau)}) \left\|\mathbf{x} - \mathbf{x}_n\right\|^2$$

is the expected complete-data log-likelihood from the E step.

Non-Gaussian mean-shift is a GEM algorithm

- Consider a mixture of non-Gaussian kernels.
- Analogous derivation as maximum-likelihood problem, but now the M step does not have a closed-form solution for $\mathbf{x}$.
- However, it is possible to solve the M step iteratively with a certain fixed-point iteration.
- Taking just one step of this fixed-point iteration provides an inexact M step and so a generalised EM algorithm, which is formally identical to the mean-shift algorithm (Carreira-Perpiñán '06).

Convergence properties of mean-shift

From the properties of (G)EM algorithms (Dempster et al '77) we expect, generally speaking:

- Convergence to a mode from almost any starting point
  Can also converge to saddle points and minima.
- Monotonic increase of p at every step
  (Jensen's inequality).
- Convergence of linear order.

However, particular mean-shift algorithms (for particular kernels) may show different properties.

We analyse in detail GMS (isotropic case: $\mathbf{\Sigma}_n = \sigma^2 \mathbf{I}$).

Convergence properties of GMS

Taylor expansion around a mode $\mathbf{x}^*$ of the fixed-point mapping $\mathbf{f}$:

$$\mathbf{x}^{(\tau+1)} = \mathbf{f}(\mathbf{x}^{(\tau)}) = \mathbf{x}^* + \mathbf{J}(\mathbf{x}^*)(\mathbf{x}^{(\tau)} - \mathbf{x}^*) + \mathcal{O}(\|\mathbf{x}^{(\tau)} - \mathbf{x}^*\|^2)$$
$$\Rightarrow \mathbf{x}^{(\tau+1)} - \mathbf{x}^* \approx \mathbf{J}(\mathbf{x}^*)(\mathbf{x}^{(\tau)} - \mathbf{x}^*)$$

where the Jacobian of $\mathbf{f}$ is

$$\mathbf{J}(\mathbf{x}) = \frac{1}{\sigma^2} \sum_{m=1}^{M} p(m|\mathbf{x})(\boldsymbol{\mu}_m - \mathbf{f}(\mathbf{x}))(\boldsymbol{\mu}_m - \mathbf{f}(\mathbf{x}))^T.$$

At a mode, $\mathbf{J}(\mathbf{x}^*) = \frac{1}{\sigma^2} \times$(local covariance at $\mathbf{x}^*$).

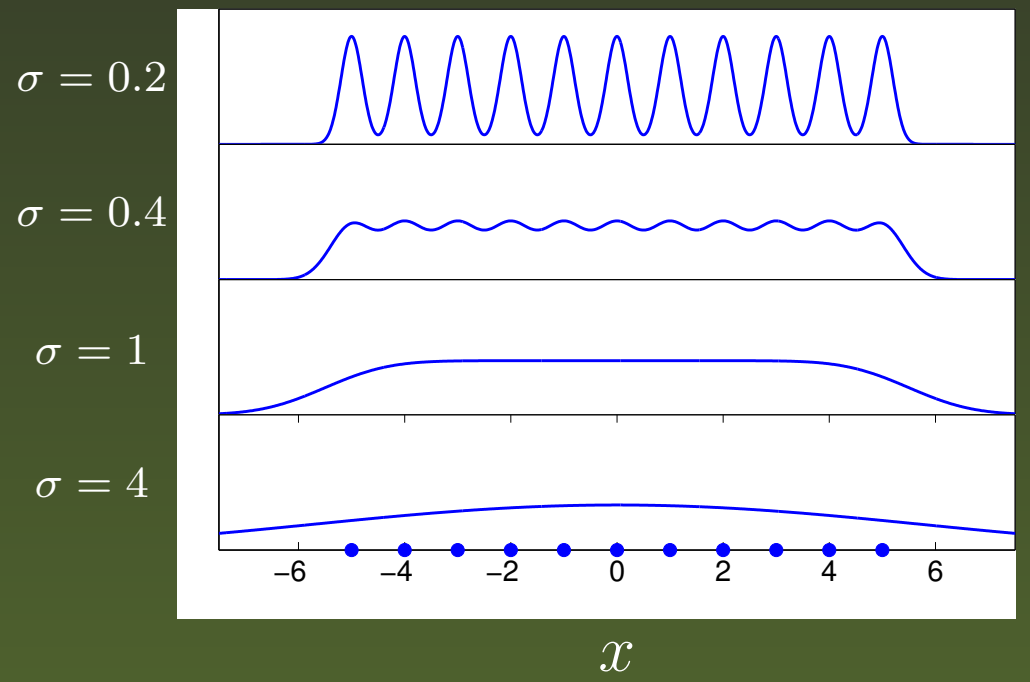
Thus, the convergence is linear with rate

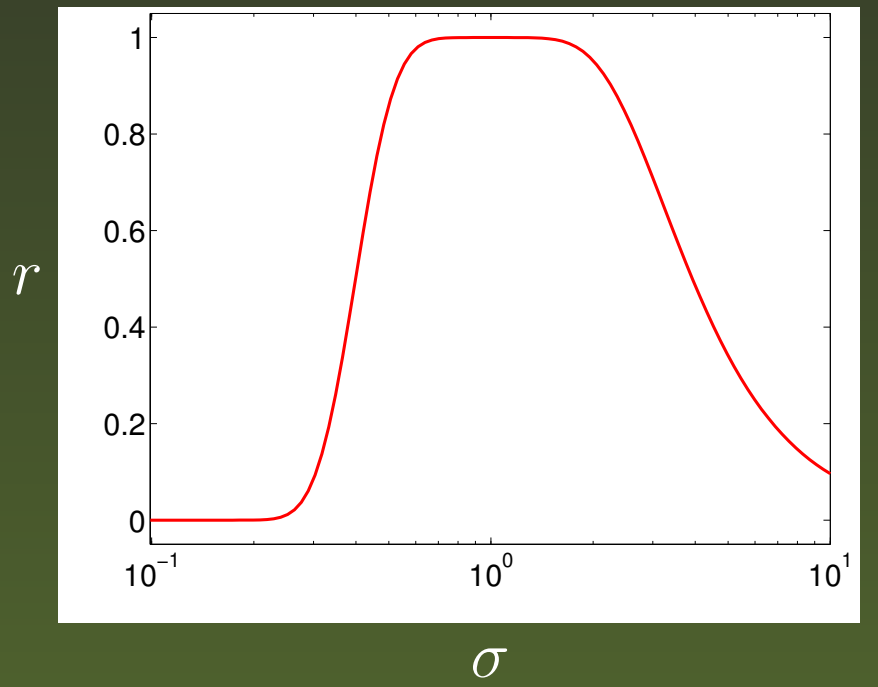
$$r = \lim_{\tau \to \infty} \frac{\|\mathbf{x}^{(\tau+1)} - \mathbf{x}^*\|}{\|\mathbf{x}^{(\tau)} - \mathbf{x}^*\|} = \lambda_{\max}(\mathbf{J}(\mathbf{x}^*)) < 1.$$

Convergence properties of GMS (cont.)

Special cases depending on σ:

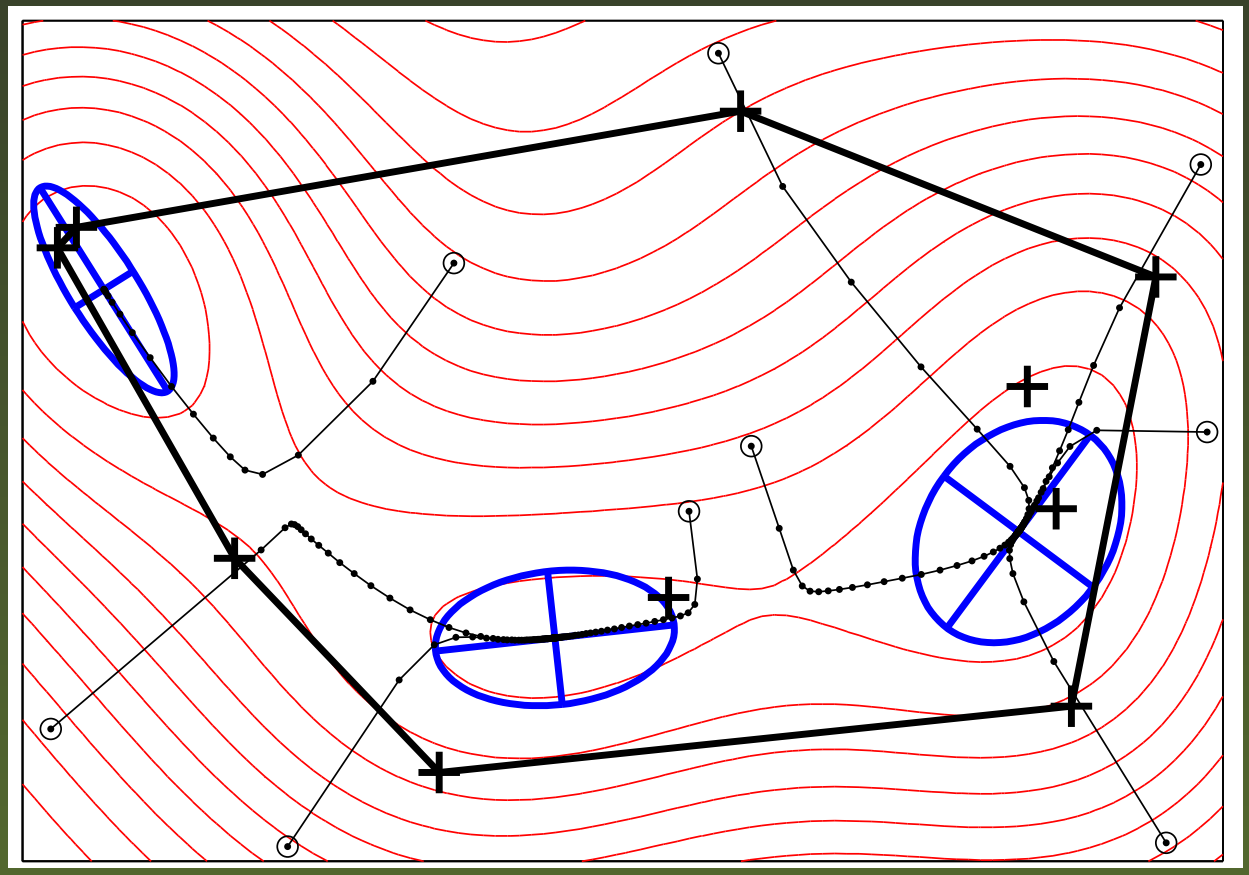
- $\sigma \to 0$, $\sigma \to \infty$ (practically uninteresting): $r \to 0$, superlinear.
- σ at mode merges: $r = 1$, sublinear (awfully slow).
- Intermediate σ (practically useful): r close to 1 (slow).





Convergence properties of GMS (cont.)

- The GMS iterates follow the local principal component at the mode and lie in the interior of the convex hull of the data points.
- Smooth path: angle between consecutive steps in $(-\frac{\pi}{2}, \frac{\pi}{2})$ (Comaniciu & Meer '02).
- GMS step size is not optimal along the GMS search direction.



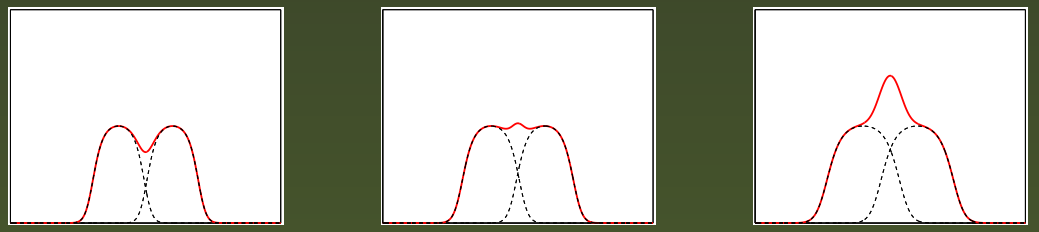
Number of modes of a Gaussian mixture

In principle, one would expect that N points should produce at most N modes (additional modes are an artifact of the kernel). But:

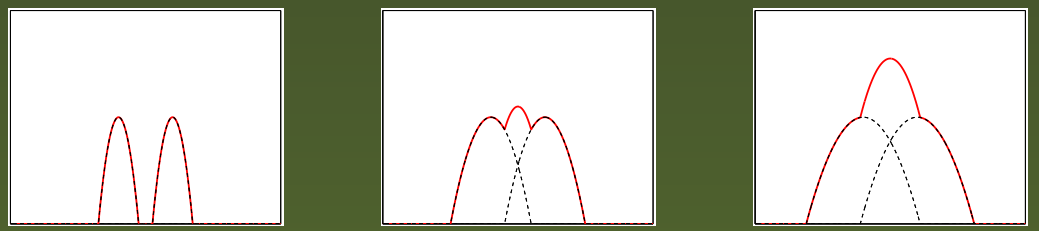
In 1D:

- A Gaussian mixture with N components has at most N modes (Carreira-Perpiñán & Williams '03).
- A mixture of non-Gaussian kernels can have more than N modes:

$K(x) \propto \frac{1}{1+e^x/10}$:

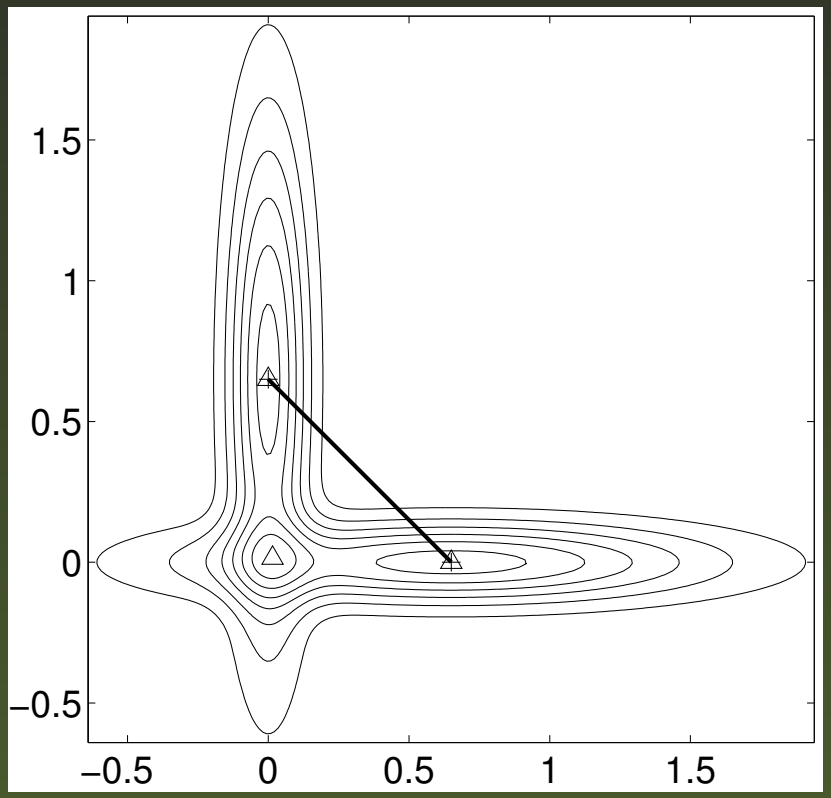


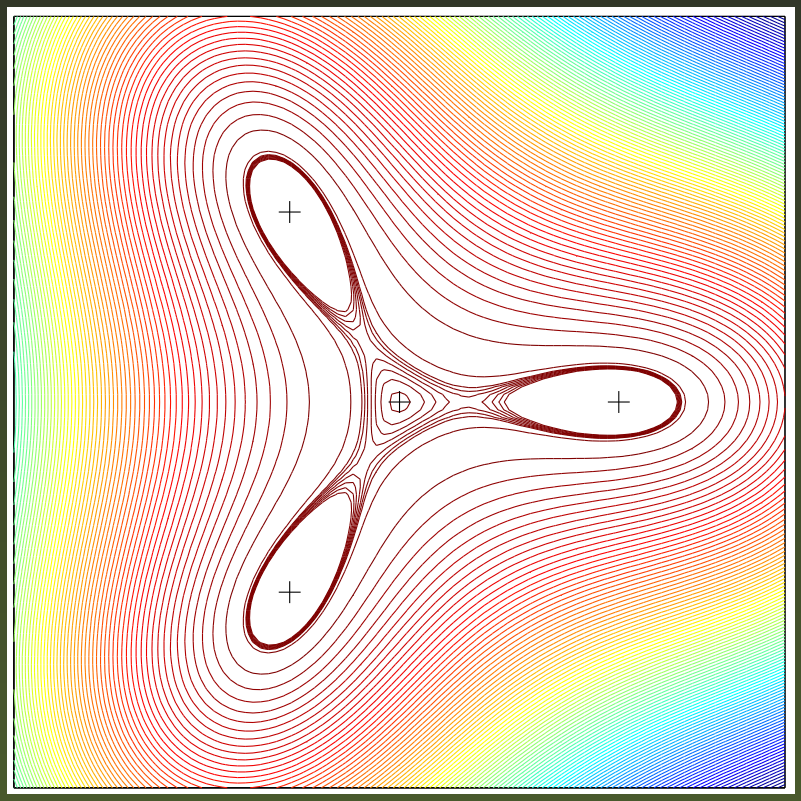
Epanechnikov kernel:



Number of modes of a Gaussian mixture (cont.)

In $D \geq 2$, even a mixture of isotropic Gaussians can have more than N modes (Carreira-Perpiñán & Williams '03):

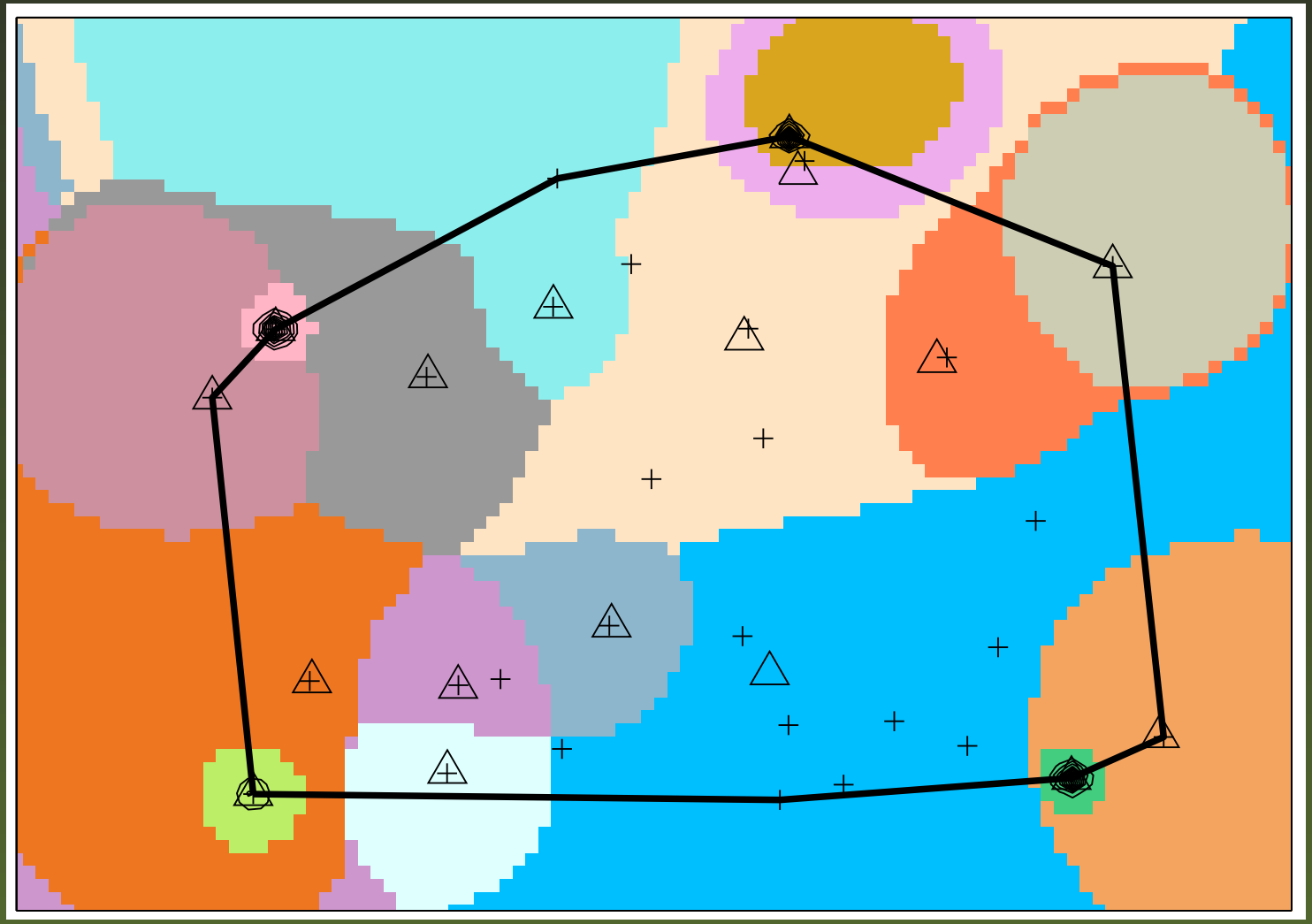




However, in practice the number of modes is much smaller than N because components coalesce. This may be a reason why, in practice, GMS produces better segmentations than other kernels.

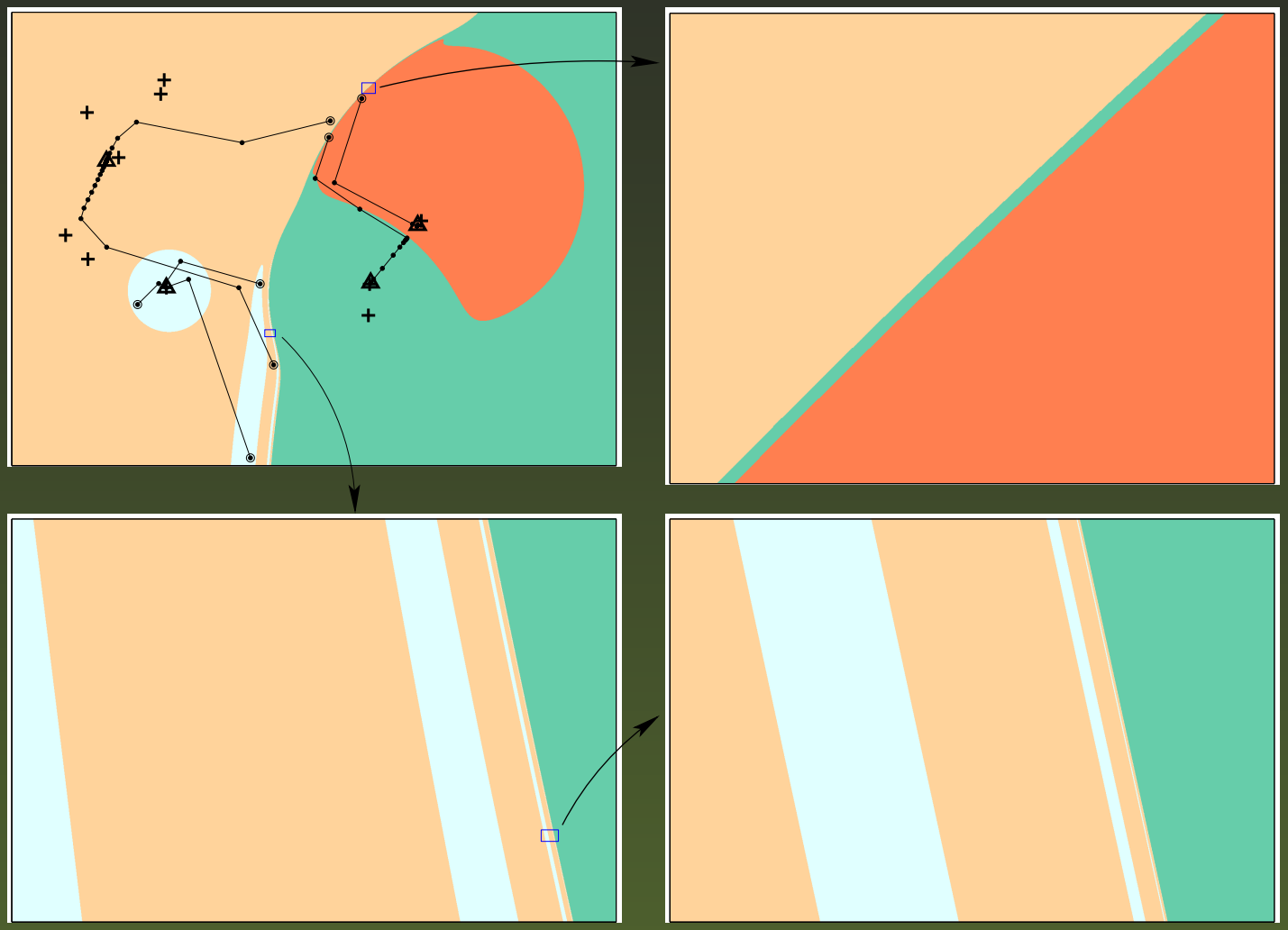
Convergence domains of GMS

In general, the convergence domains (geometric locus of points that converge to each mode) are nonconvex and can be disconnected and take quite fancy shapes:



Convergence domains of GMS (cont.)

The boundary between some domains can be fractal:



These peculiar domains are probably undesirable for clustering; but, small effect (seemingly confined to cluster boundaries).

Gaussian mean-shift as an EM algorithm: summary

- GMS is an EM algorithm
- Non-Gaussian mean-shift is a GEM algorithm
- GMS converges to a mode from almost any starting point
- Convergence is linear (occasionally superlinear or sublinear), slow in practice
- The iterates approach a mode along its local principal component
- Gaussian mixtures and kernel density estimates can have more modes than components (but seems rare)
- The convergence domains for GMS can be fractal

Acceleration strategies for GMS image segmentation

- Gaussian mean-shift (GMS) as an EM algorithm
- Acceleration strategies for GMS image segmentation
- Gaussian blurring mean-shift (GBMS)

Computational bottlenecks of GMS

GMS is slow: $\mathcal{O}(kN^2D)$. Computational bottlenecks:

- B1: large average number of iterations $k \sim 100$ (linear convergence).
- B2: large cost per iteration $\sim 2ND$ multiplications
  - E step: ND to obtain $p(n|\mathbf{x}^{(\tau)})$.
  - M step: ND to obtain $\mathbf{x}^{(\tau+1)}$.

Acceleration techniques must address B1 and/or B2.

We propose four strategies:

- `ms1`: spatial discretisation
- `ms2`: spatial neighbourhood
- `ms3`: sparse EM
- `ms4`: EM–Newton

Evaluation of strategies

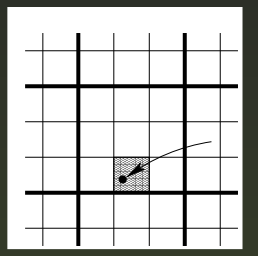
Evaluation of strategies:

- Goal: to achieve the same segmentation as GMS. Visual evaluation of segmentation not enough; we compute the segmentation error wrt GMS segmentation ($=$ no. pixels misclustered as a % over the whole image).
- We report running times in normalised iterations ($=$ 1 iteration of GMS) to ensure independence from implementation details.
- No pre- or postprocessing of clusters (e.g. removal of small clusters).

Why segmentation errors?

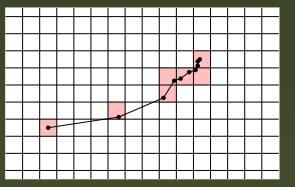
- The accelerated scheme may not converge to a mode of $p(\mathbf{x})$.
- $\mathbf{x}_n$ may converge to a different mode than with exact GMS.

`ms1`: spatial discretisation



Many different pixels converging to the same mode travel almost identical paths. We discretise the spatial domain by subdividing every pixel (i, j) into $n \times n$ cells;

- points projecting to the same cell share the same fate.
- This works because paths in D dimensions are well approximated by their 2D projection on the spatial domain.

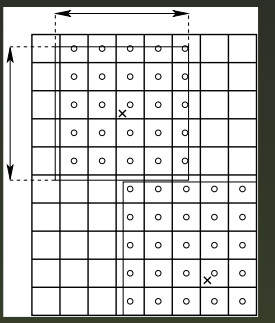


We stop iterating once we hit an already visited cell

- $\Rightarrow$ massively reduces the total number of iterations.
- We start first with a set of pixels uniformly distributed over the image (this finds all modes quickly).
- Converges to a mode
- Segmentation error $\rightarrow 0$ by increasing n
- Addresses B1. Parameter: $n = 1, 2, 3 \ldots$ (discretisation level)

`ms2`: spatial neighbourhood

Approximates E and M steps with a subset of the data points (rather than all N) consisting of a neighbourhood in the spatial domain (not the range domain). Finding neighbours is for free (unlike finding neighbours in full space).



$$p(n|\mathbf{x}^{(\tau)}) = \frac{\exp\left(-\frac{1}{2}\left\|(\mathbf{x}^{(\tau)} - \mathbf{x}_n)/\sigma\right\|^2\right)}{\sum_{n' \in \mathcal{N}(\mathbf{x}^{(\tau)})} \exp\left(-\frac{1}{2}\left\|(\mathbf{x}^{(\tau)} - \mathbf{x}_{n'})/\sigma\right\|^2\right)}$$

$$\mathbf{x}^{(\tau+1)} = \sum_{n \in \mathcal{N}(\mathbf{x}^{(\tau)})} p(n|\mathbf{x}^{(\tau)})\mathbf{x}_n$$

- Does not converge to a mode
- Segmentation error $\to 0$ by increasing e
- Addresses B2. Parameter: $e \in (0, 1]$ (fraction of data set used as neighbours)

ms3: sparse EM

Sparse EM (Neal & Hinton '98): coordinate ascent on the space of $(\mathbf{x}, \tilde{\mathbf{p}})$ where $\tilde{\mathbf{p}}$ are posterior probabilities; this maximises the free energy $F(\tilde{\mathbf{p}}, \mathbf{x}) = \log p(\mathbf{x}) - \mathrm{D}\left(\tilde{\mathbf{p}} \| p(n|\mathbf{x})\right)$ and also $p(\mathbf{x})$.

- Run partial E steps frequently: update $p(n|\mathbf{x})$ only for $n \in S$; $S =$ plausible set (nearest neighbours), kept constant over partial E steps. FAST.
- Run full E steps infrequently: update all $p(n|\mathbf{x})$ and S. SLOW.

We choose S containing as many neighbours as necessary to account for a total probability $\sum_{n \in S} p(n|\mathbf{x}) \geq 1 - \epsilon \in (0, 1]$. Thus, the fraction of data used e varies after each full step.
Using fixed e produced worse results.

- Converges to a mode no matter how S is chosen; computational savings if few full steps
- Segmentation error $\to 0$ by decreasing ϵ
- Addresses B2. Parameter: $\epsilon \in [0, 1)$ (probability not in S)

ms4: EM–Newton

Start with EM steps, which quickly increase p. Switch to Newton steps when EM slows down (reverting to EM if bad Newton step).
Specifically, try Newton step when $\|\mathbf{x}^{(\tau)} - \mathbf{x}^{(\tau-1)}\| < \theta$.
Computing the Newton step $\mathbf{x}_{\mathsf{N}}$ yields the EM step $\mathbf{x}_{\mathsf{EM}}$ for free:

$$\text{Gradient: } \mathbf{g}(\mathbf{x}) = \frac{p(\mathbf{x})}{\sigma^2} \sum_{n=1}^{N} p(n|\mathbf{x})(\mathbf{x}_n - \mathbf{x}) = \frac{p(\mathbf{x})}{\sigma^2}(\mathbf{x}_{\mathsf{EM}} - \mathbf{x})$$

$$\text{Hessian: } \mathbf{H}(\mathbf{x}) = \frac{p(\mathbf{x})}{\sigma^2}\left(-\mathbf{I} + \frac{1}{\sigma^2} \sum_{n=1}^{N} p(n|\mathbf{x})(\mathbf{x}_n - \mathbf{x})(\mathbf{x}_n - \mathbf{x})^T\right)$$

Newton step: $\mathbf{x}_{\mathsf{N}} = \mathbf{x} - \mathbf{H}^{-1}(\mathbf{x})\mathbf{g}(\mathbf{x})$.

- Converges to a mode with quadratic order
- Segmentation error $\to 0$ by decreasing θ
- Addresses B1. Parameter: $\theta > 0$ (minimum EM step length) relative to σ

Computational cost

Strategy	Cost per iteration relative to exact GMS
`ms1`: spatial discretisation	1
`ms2`: spatial neighbourhood	e
`ms3`: sparse EM	2 if full step, e if partial step
`ms4`: EM–Newton	1 if EM step, $\left(1 + \frac{D+1}{4}\right)$ if Newton step, $\left(\frac{3}{2} + \frac{D+1}{4}\right)$ if EM step after failed N. step

$e \in (0, 1]$ is the fraction of the data set used (neighbours for `ms2`, plausible set for `ms3`).

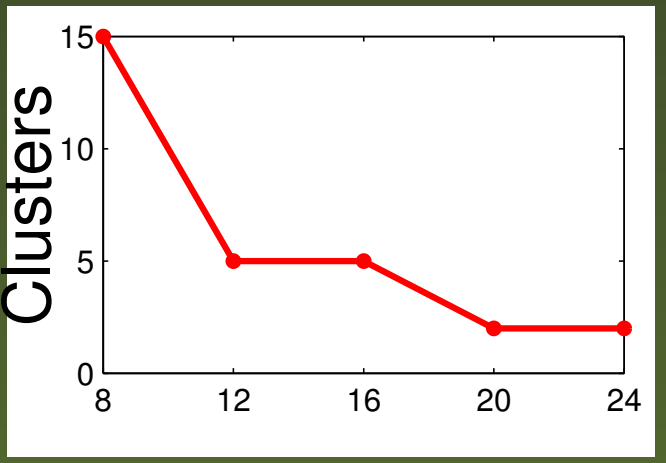
Using iterations normalised wrt GMS allows direct comparison of the numbers of iterations in the experiments.

Experimental results with image segmentation

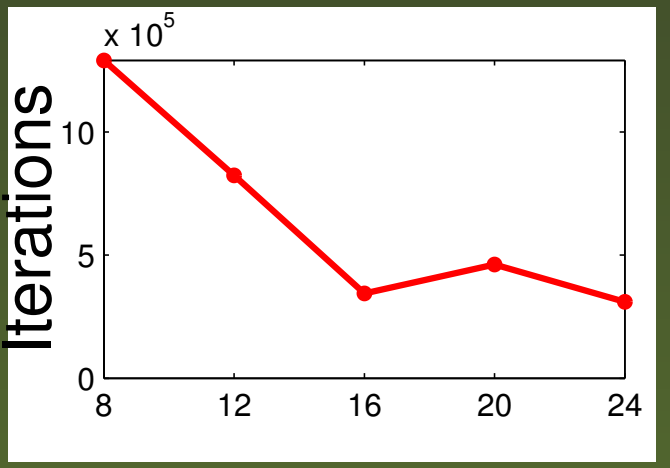
- Dataset: $\mathbf{x}_n = (i_n, j_n, I_n)$ (greyscale) or $\mathbf{x}_n = (i_n, j_n, L^*_n, u^*_n, v^*_n)$ (colour) where (i, j) is the pixel's spatial position. Prescale I or (L^*, u^*, v^*) so same σ for all dimensions (in pixels).
- Best segmentations for large bandwidths: $\sigma \approx \frac{1}{5} \times$(image side).
- We study all strategies with different images over a range of σ.
- Convergence tolerance $\texttt{tol} = 10^{-3}$ pixels.

`cameraman`
100×100

Number of clusters

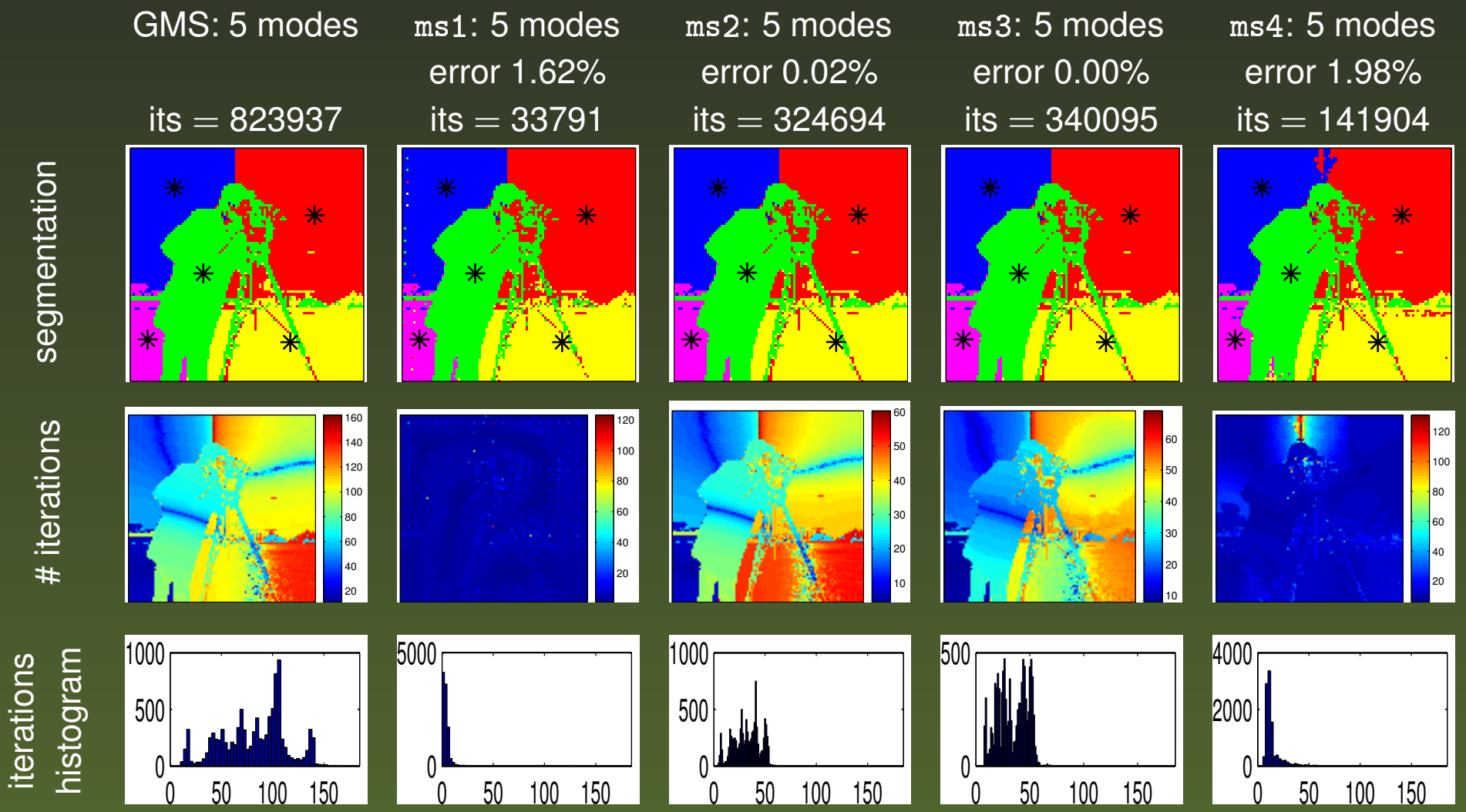


Total number of iterations



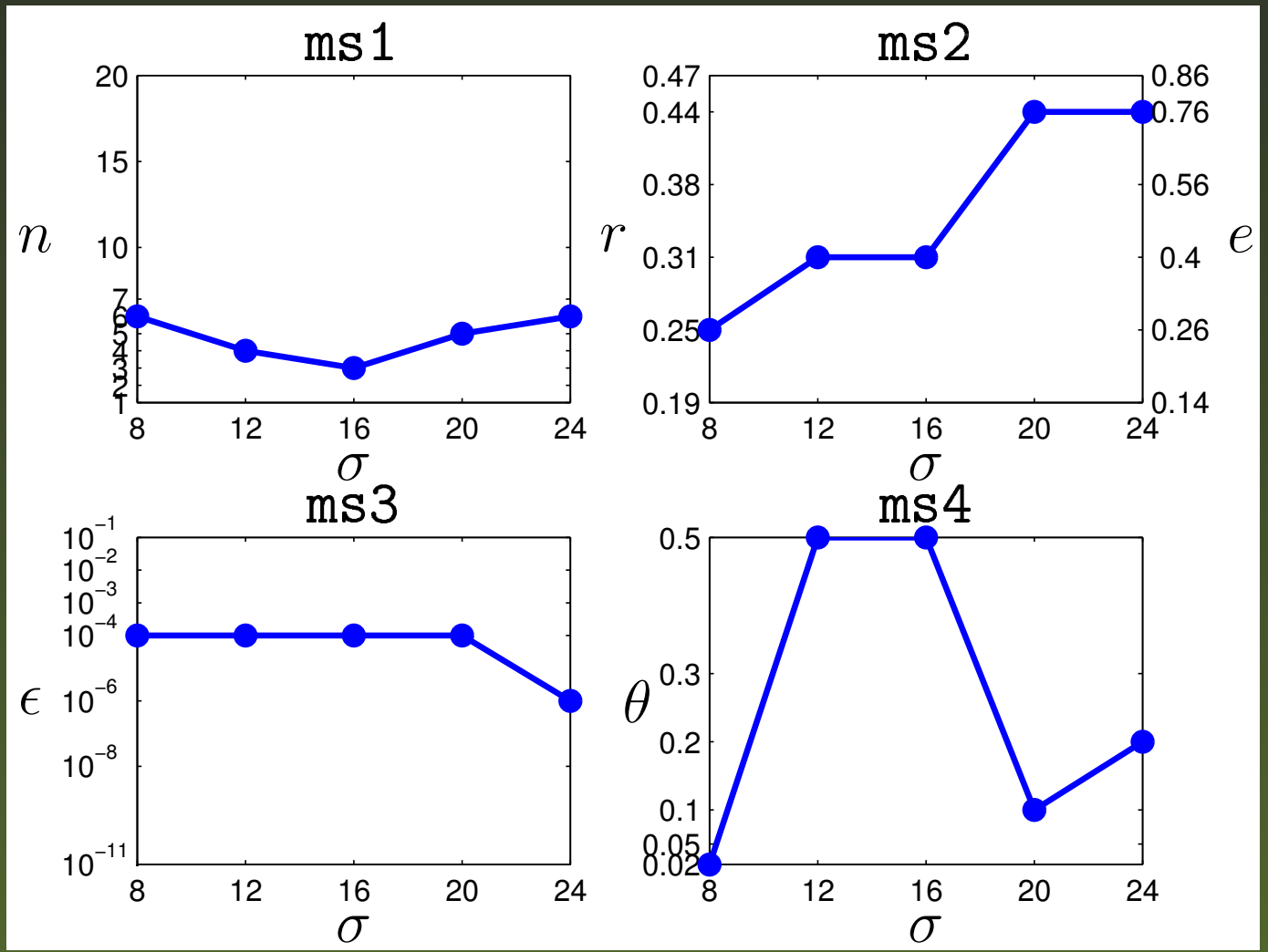
Experimental results with image segmentation (cont.)

Segmentation results for each method under its optimal parameter value for $\sigma = 12$:



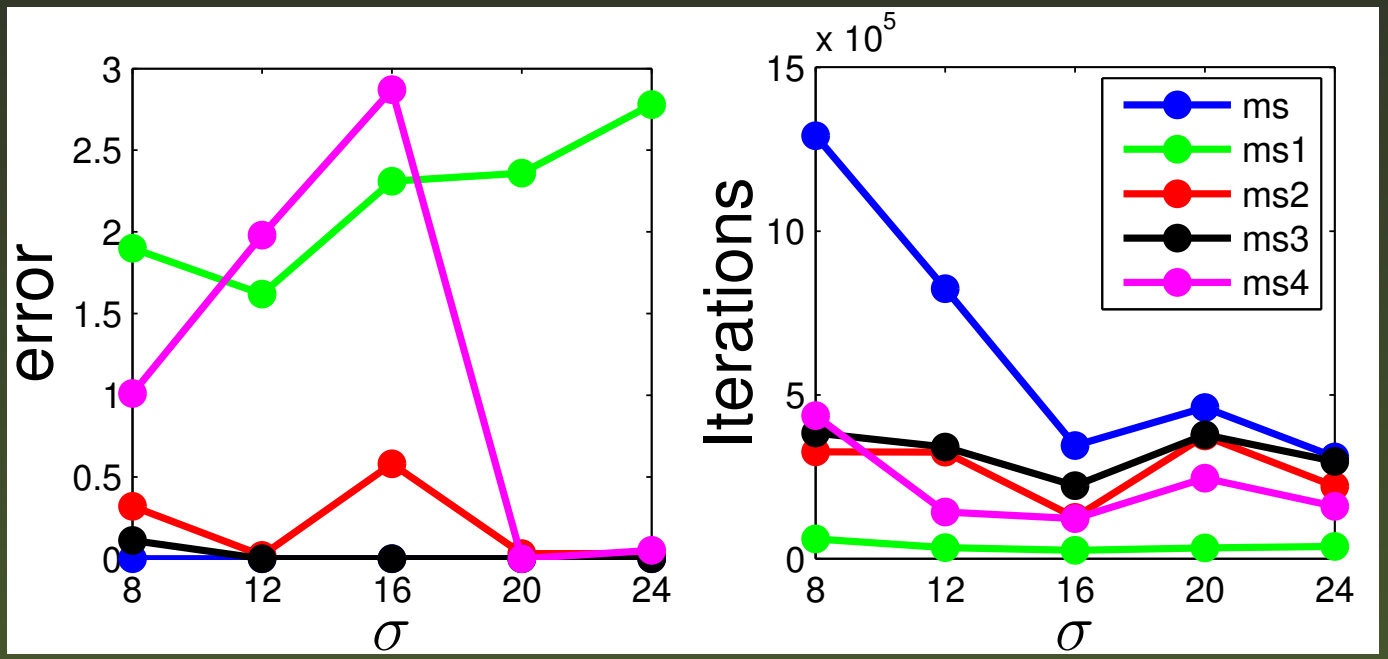
Experimental results with image segmentation (cont.)

Optimal parameter value for each method (= that which minimises the iterations ratio s. t. achieving an error $< 3\%$) as a function of σ.



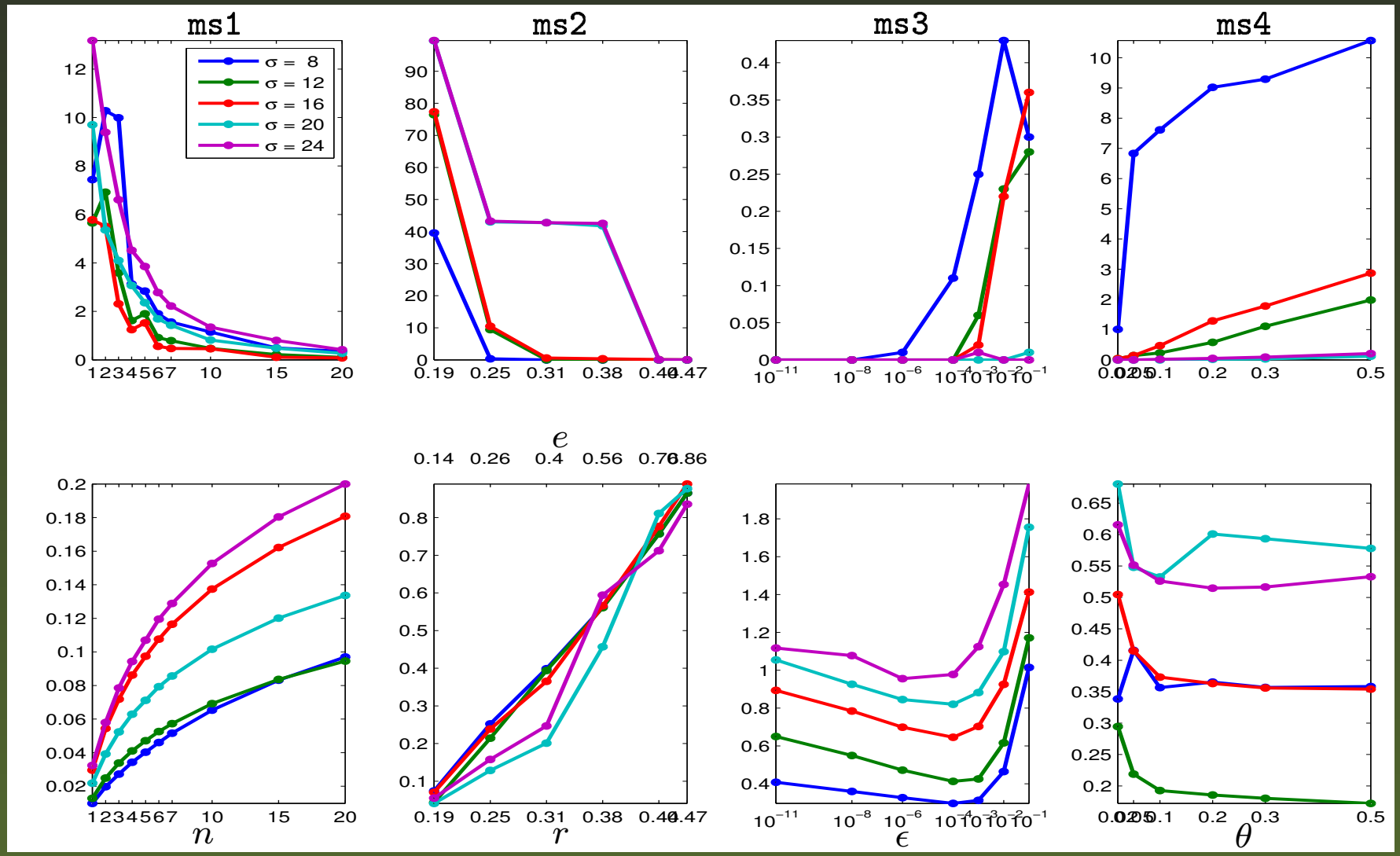
Experimental results with image segmentation (cont.)

Clustering error (percent) and number of iterations for each method under its optimal parameter value as a function of σ.



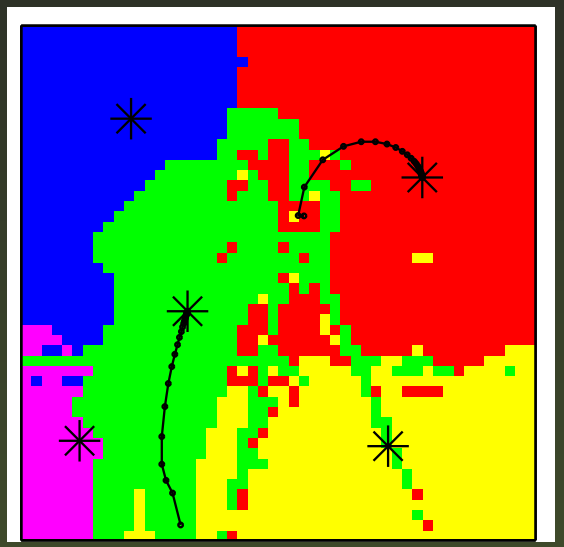
Experimental results with image segmentation (cont.)

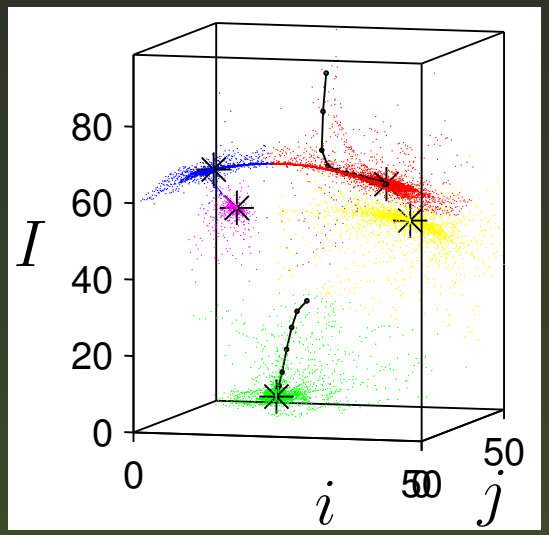
Clustering error (percent) and computational cost for each method as a function of its parameter.

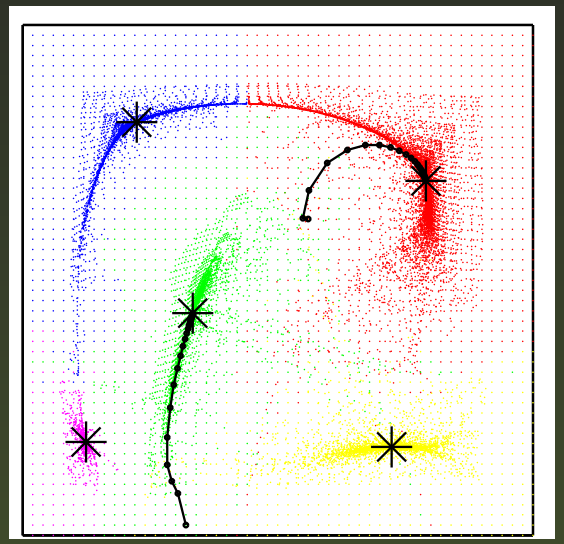


Experimental results with image segmentation (cont.)

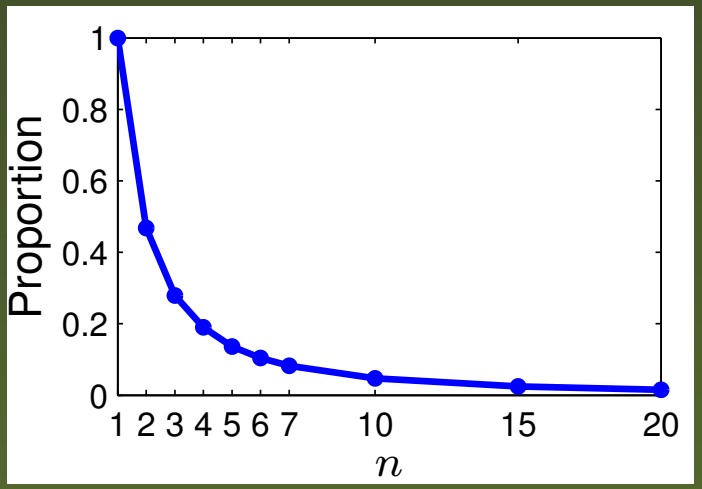
Plot of all iterates for all starting pixels: most cells in `ms1` are empty.



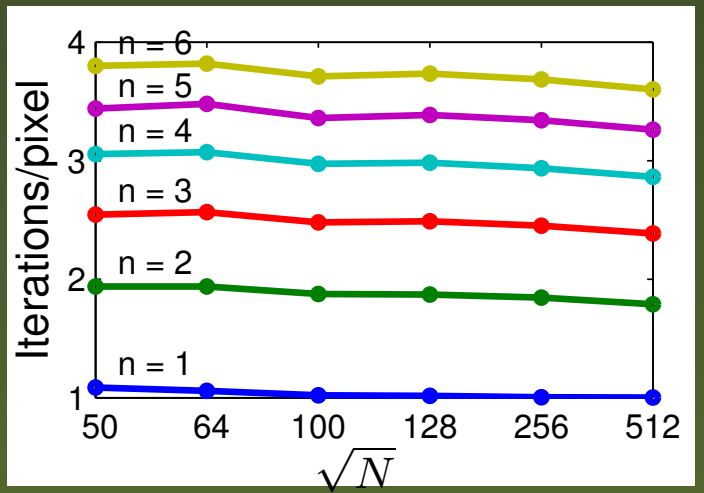




Proportion of cells visited by `ms1` (less than nN out of n^2N)



Average number of iterations per pixel for `ms1`



Acceleration strategies: summary (cont.)

- With parameters set optimally, all methods can obtain nearly the same segmentation as GMS with significant speedups. Near-optimal parameter values can easily be set in advance for `ms1`, `ms3` and `ms4`, and somewhat more heuristically for `ms2`.
- `ms1` (spatial discretisation): best strategy; 10×–100× speedup (average number of iterations per pixel $k = 2$–4 only) with very small error controlled by the discretisation level.
- `ms4` (EM–Newton): second best; 1.5×–6× speedup.
- `ms2`, `ms3` (neighbourhood methods): 1×–3× speedup; `ms3` attains the lowest (near-zero) error, while `ms2` can result in unacceptably large errors for a suboptimal setting of its parameter (neighbourhood size).
  The neighbourhood methods (`ms2`, `ms3`) are less effective because GMS needs very large neighbourhoods (comparable to the data set size for the larger bandwidths).

Acceleration strategies: summary (cont.)

Other methods:

- Approximate algorithms for neighbour search (e.g. kd-trees): less effective because GMS needs large neighbourhoods.
- Other optimisation algorithms (e.g. quasi-Newton): need to ensure first steps are EM.
- Fast Gauss transform: can be combined with our methods.

Extensions:

- Adaptive and non-isotropic bandwidths: all 4 methods.
- Non-image data: `ms3`, `ms4` readily applicable.
  Is it possible to extend `ms1` to higher dimensions?

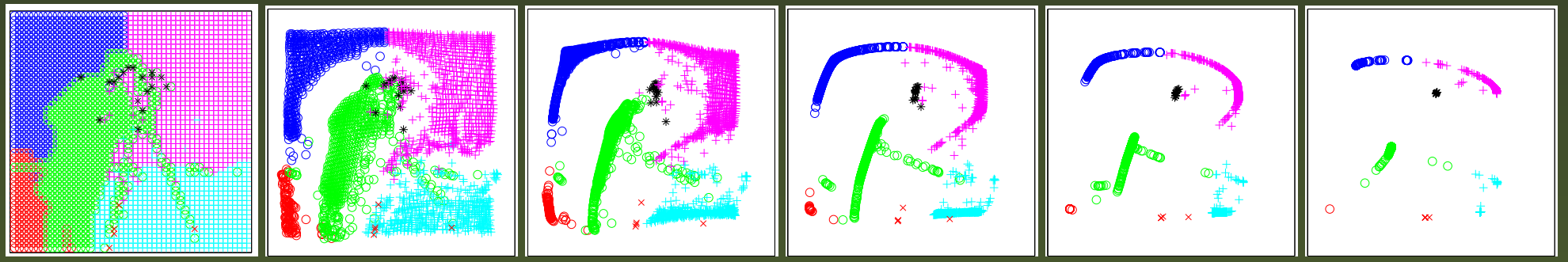
Gaussian blurring mean-shift (GBMS)

- Gaussian mean-shift (GMS) as an EM algorithm
- Acceleration strategies for GMS image segmentation
- Gaussian blurring mean-shift (GBMS)

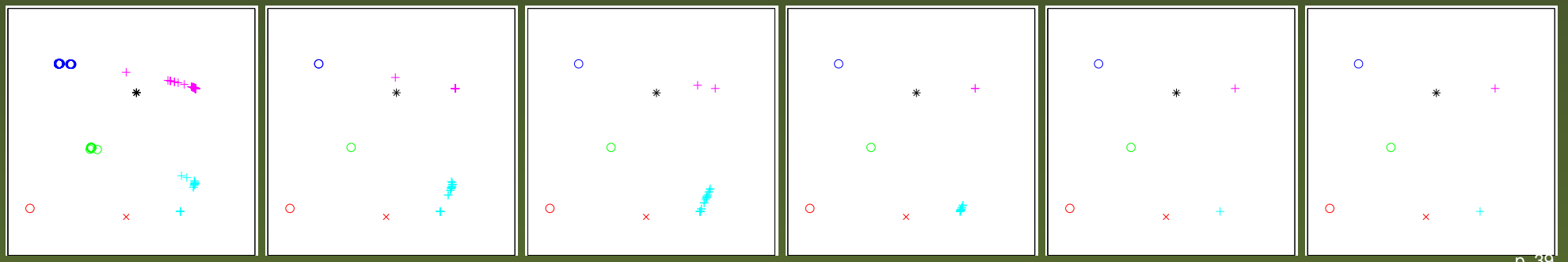
Gaussian blurring mean-shift (GBMS)

- This is the algorithm that Fukunaga & Hostetler really proposed
- It has gone largely unnoticed.
- Same iterative scheme, but now the data points move at each iteration. Result: sequence of progressively shrunk datasets $\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \ldots$ converging to single, all-points-coincident cluster.

$\tau = 0$ $\tau = 1$ $\tau = 2$ $\tau = 3$ $\tau = 4$ $\tau = 5$



$\tau = 6$ $\tau = 7$ $\tau = 8$ $\tau = 9$ $\tau = 10$ $\tau = 11$



Pseudocode: GBMS clustering

repeat — Iteration loop

  for $m \in \{1, \ldots, N\}$ — For each data point

$$\forall n\text{: } p(n|\mathbf{x}_m) \leftarrow \frac{\exp\left(-\frac{1}{2}\left\|(\mathbf{x}_m - \mathbf{x}_n)/\sigma\right\|^2\right)}{\sum_{n'=1}^{N} \exp\left(-\frac{1}{2}\left\|(\mathbf{x}_m - \mathbf{x}_{n'})/\sigma\right\|^2\right)}$$

$$\mathbf{y}_m \leftarrow \sum_{n=1}^{N} p(n|\mathbf{x}_m)\mathbf{x}_n$$ — One GMS step

  end

  $\forall m\text{: } \mathbf{x}_m \leftarrow \mathbf{y}_m$ — Update whole dataset

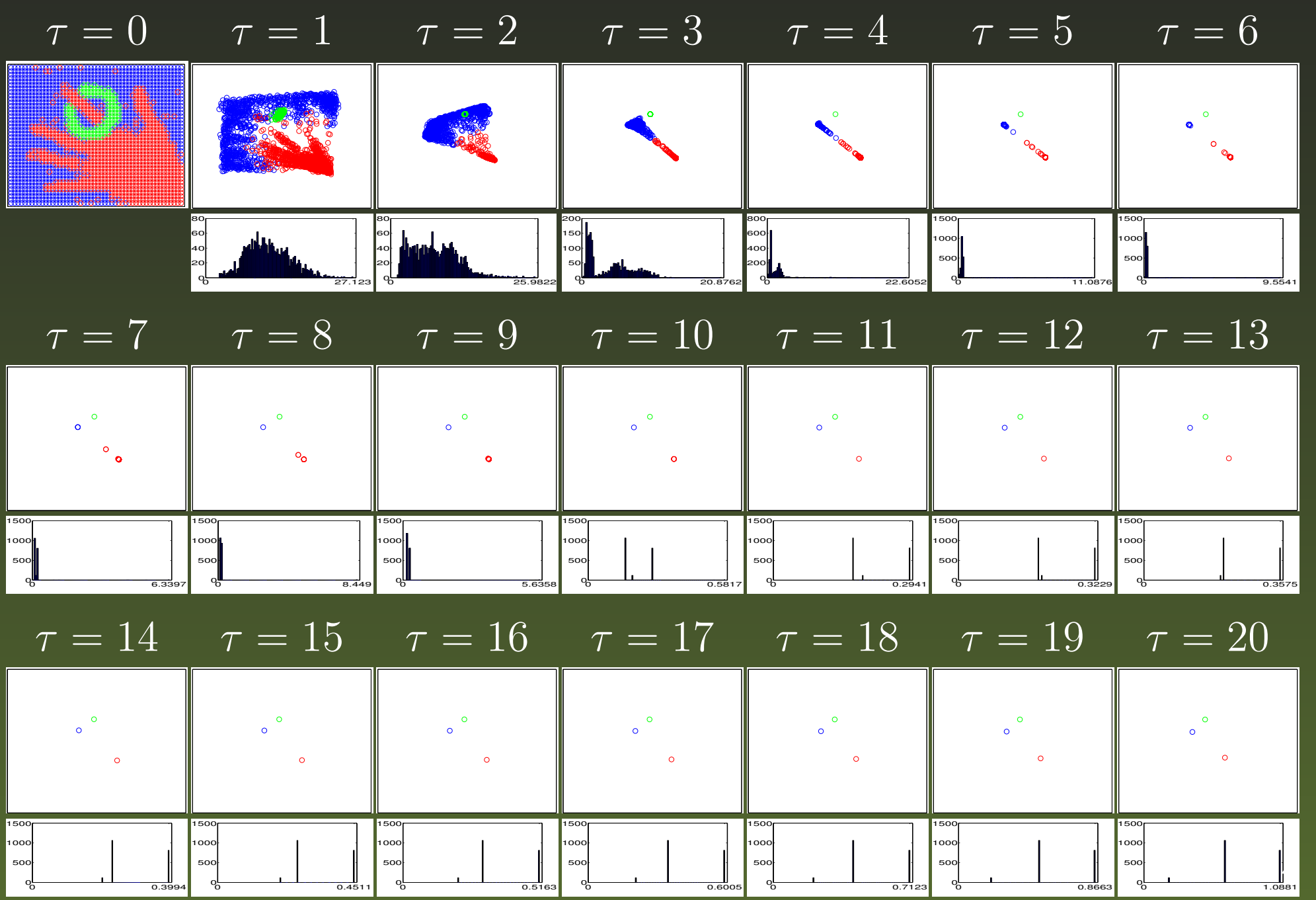
until stop — Stopping criterion

connected-components($\{\mathbf{x}_n\}_{n=1}^{N}$,`min_diff`) — Clusters

Stopping criterion for GBMS

- GBMS converges to an all-points-coincident cluster (Cheng '95). Proof: diameter of data set decreases at least geometrically.
- However, GBMS shows two phases:
  - Phase 1: points merge into clusters of coincident points (a few iterations); we want to stop here.
  - Phase 2: clusters keep approaching and merging (a few to hundreds of iterations); slowly erases clustering structure.
- We need a stopping criterion (as opposed to a convergence criterion) to stop just after phase 1.
- Simply checking $\|\mathbf{X}^{(\tau)} - \mathbf{X}^{(\tau-1)}\| < \texttt{tol}$ does not work because the points are always moving.
- Instead, consider the histogram of updates $\{e_n^{(\tau)}\}_{n=1}^N$, $e_n^{(\tau)} = \|\mathbf{x}_n^{(\tau)} - \mathbf{x}_n^{(\tau-1)}\|$. Though the histograms change as points move, in phase 2 the entropy H does not change (the histogram bins change their position but not their values).

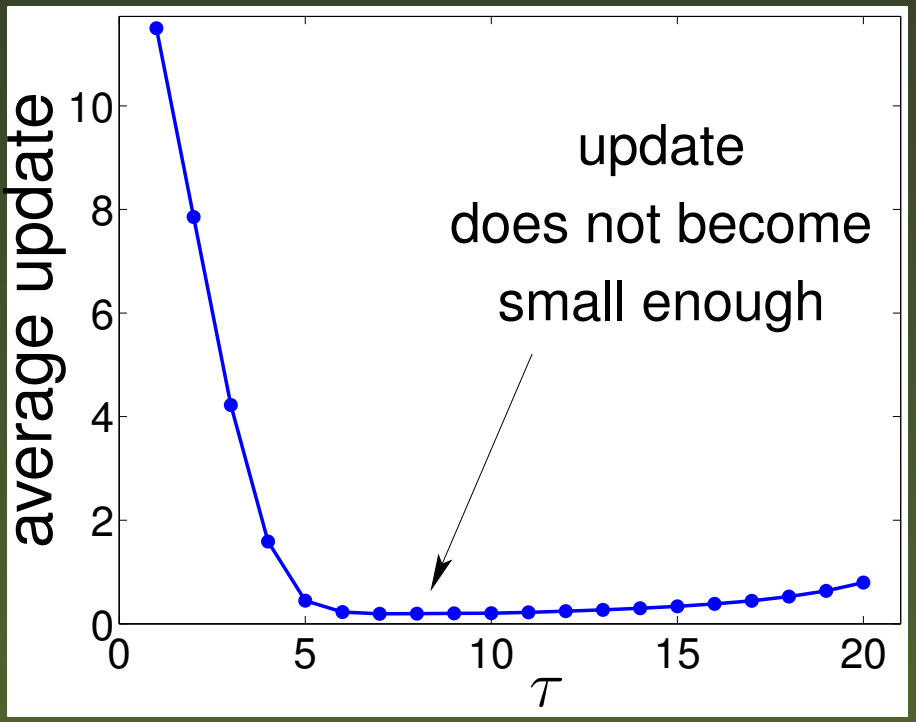
Stopping criterion for GBMS (cont.)

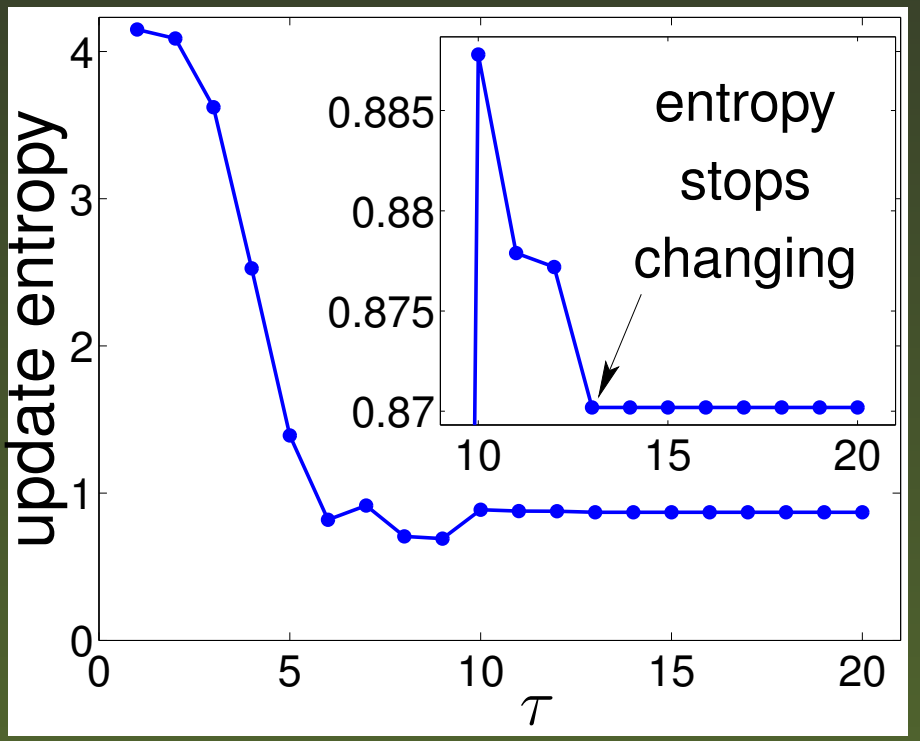


Stopping criterion for GBMS (cont.)

Stopping criterion:

$$\left(|H(\mathbf{e}^{(\tau+1)}) - H(\mathbf{e}^{(\tau)})| < 10^{-8}\right) \text{ OR } \left(\frac{1}{N}\sum_{n=1}^{N} e_n^{(\tau+1)} < \texttt{tol}\right)$$





Convergence rate of GBMS

GBMS shrinks a Gaussian cluster towards its mean with cubic convergence rate. Proof: work with continuous pdf

$$p(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} K\left(\left\|\frac{\mathbf{x}-\mathbf{x}_n}{\sigma}\right\|^2\right) \Rightarrow p(\mathbf{x}) = \int_{\mathbb{R}^D} q(\mathbf{y})K(\mathbf{x}-\mathbf{y})\,d\mathbf{y}$$

New dataset:

$$\tilde{\mathbf{x}} = \int_{\mathbb{R}^D} p(\mathbf{y}|\mathbf{x})\mathbf{y}\,d\mathbf{y} = \mathrm{E}\left\{\mathbf{y}|\mathbf{x}\right\}\ \forall \mathbf{x} \in \mathbb{R}^D \text{ with } p(\mathbf{y}|\mathbf{x}) = \frac{K(\mathbf{x}-\mathbf{y})q(\mathbf{y})}{p(\mathbf{x})}$$

Considering 1D w.l.o.g. (dimensions decouple):

$$q(x) = \mathcal{N}(x; 0, s^2) \Rightarrow p(x) = \mathcal{N}(x; 0, s^2+\sigma^2),\ q(\tilde{x}) = \mathcal{N}(\tilde{x}; 0, (rs)^2)$$

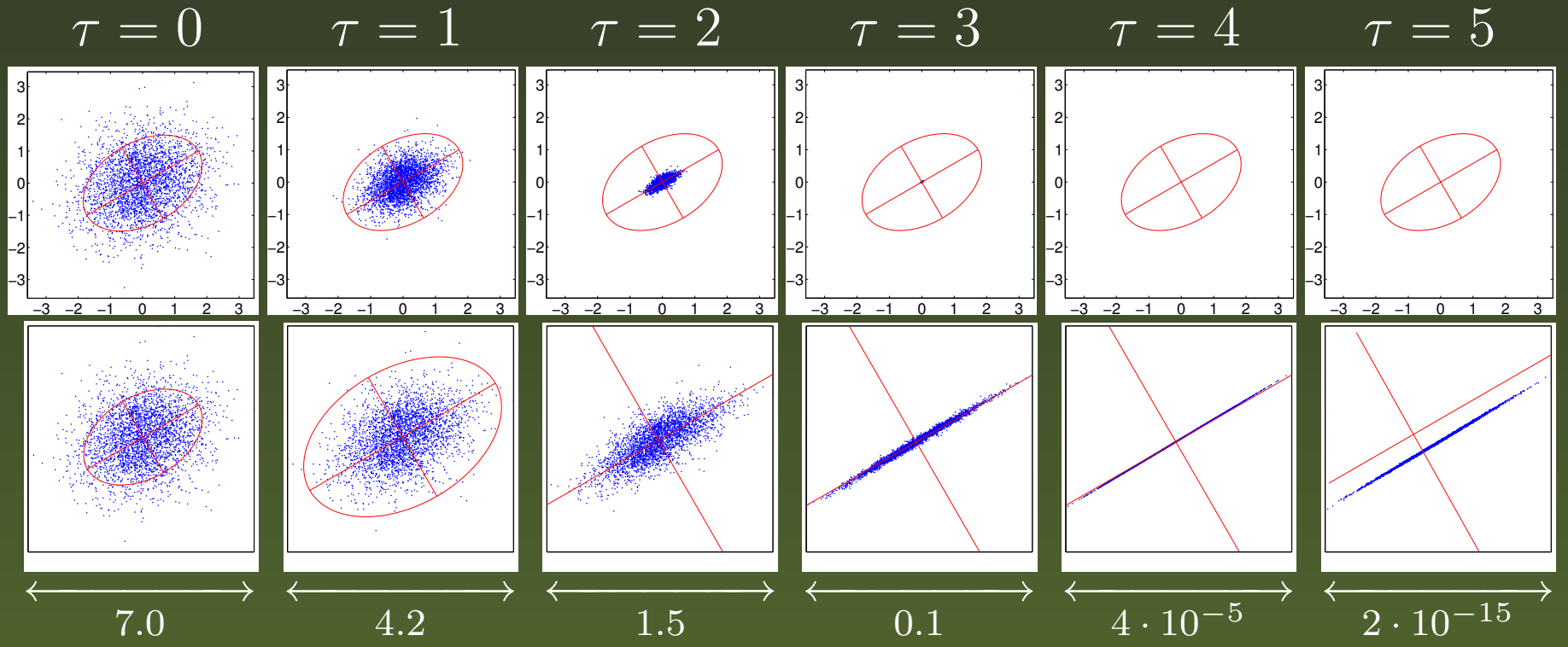
with $r = \frac{1}{1+(\sigma/s)^2} \in (0,1)$.

Convergence rate of GBMS (cont.)

Recurrence relation for the standard deviation $s^{(\tau)}$:

$$s^{(\tau+1)} = \frac{1}{1 + (\sigma / s^{(\tau)})^2} s^{(\tau)} \qquad \text{for } s^{(0)} > 0$$

which converges to 0 with cubic order, i.e., $\lim_{\tau \to \infty} \frac{\left| s^{(\tau+1)} - s^{(\infty)} \right|}{\left| s^{(\tau)} - s^{(\infty)} \right|^3} < \infty$. Effectively, σ increases at every step.



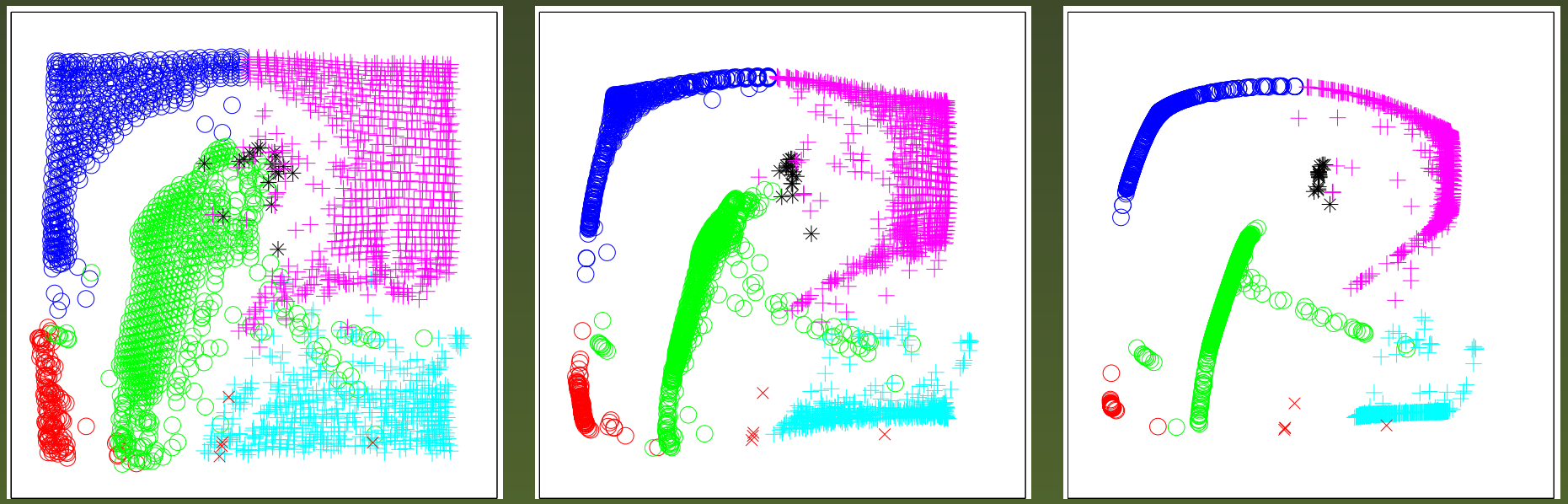
The cluster shrinks without changing orientation or mean. The principal direction shrinks more slowly relative to other directions.

Convergence rate of GBMS (cont.)

In summary, a Gaussian distribution remains Gaussian with the same mean but each principal axis decreases cubically. This explains the practical behaviour shown by GBMS:

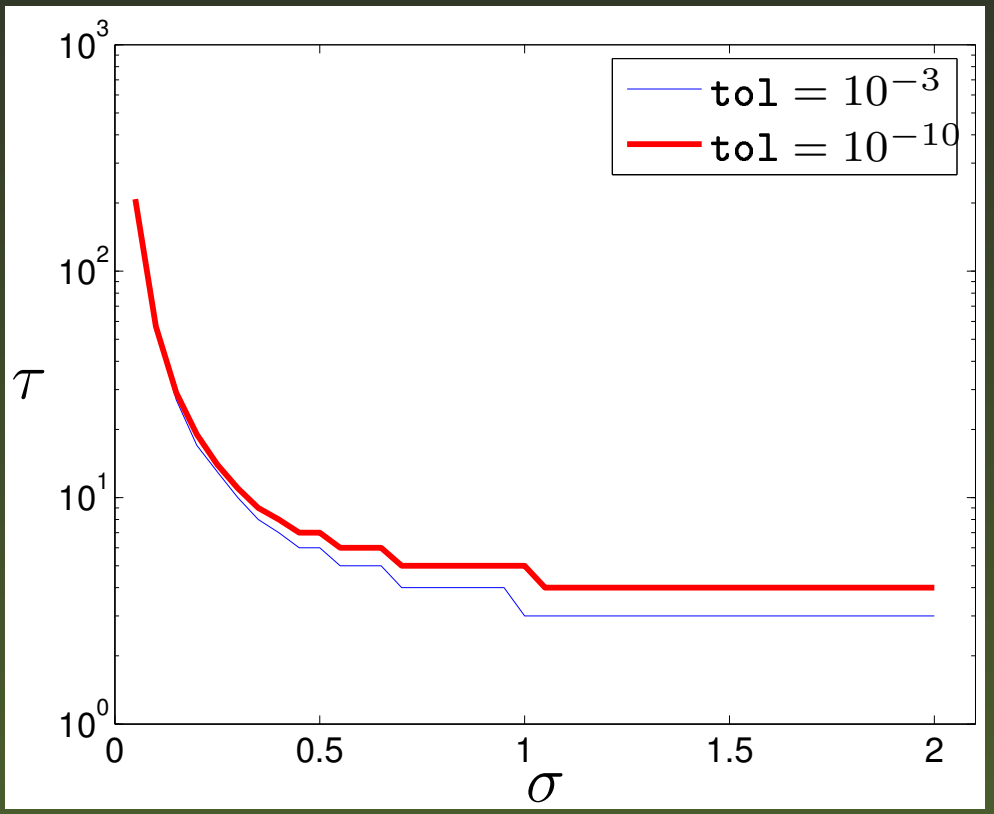
1. Clusters collapse extremely fast (clustering).
2. After a few iterations only the local principal component survives $\Rightarrow$ temporary linearly-shaped clusters (denoising).

$\tau = 1$ $\tau = 2$ $\tau = 3$



Convergence rate of GBMS (cont.)

Number of GBMS iterations τ necessary to achieve $s^{(\tau)} < \texttt{tol}$ as a function of the bandwidth σ for $s^{(0)} = 1$:



Note that GMS converges with linear order ($p = 1$), thus requiring many iterations to converge to a mode.

Connection with spectral clustering

GBMS pseudocode (inner loop)

for $m \in \{1, \ldots, N\}$ — For each data point

$$\forall n\text{: } p(n|\mathbf{x}_m) \leftarrow \frac{\exp\left(-\frac{1}{2}\left\|(\mathbf{x}_m - \mathbf{x}_n)/\sigma\right\|^2\right)}{\sum_{n'=1}^{N} \exp\left(-\frac{1}{2}\left\|(\mathbf{x}_m - \mathbf{x}_{n'})/\sigma\right\|^2\right)}$$

$$\mathbf{y}_m \leftarrow \sum_{n=1}^{N} p(n|\mathbf{x}_m)\mathbf{x}_n$$ — One GMS step

end

$\forall m\text{: } \mathbf{x}_m \leftarrow \mathbf{y}_m$ — Update whole dataset

Same pseudocode in matrix form

$\mathbf{W} = \left(\exp\left(-\frac{1}{2}\left\|(\mathbf{x}_m - \mathbf{x}_n)/\sigma\right\|^2\right)\right)_{nm}$ — Gaussian affinity matrix

$\mathbf{D} = \operatorname{diag}\left(\sum_{n=1}^{N} w_{nm}\right)$ — Degree (normalising) matrix

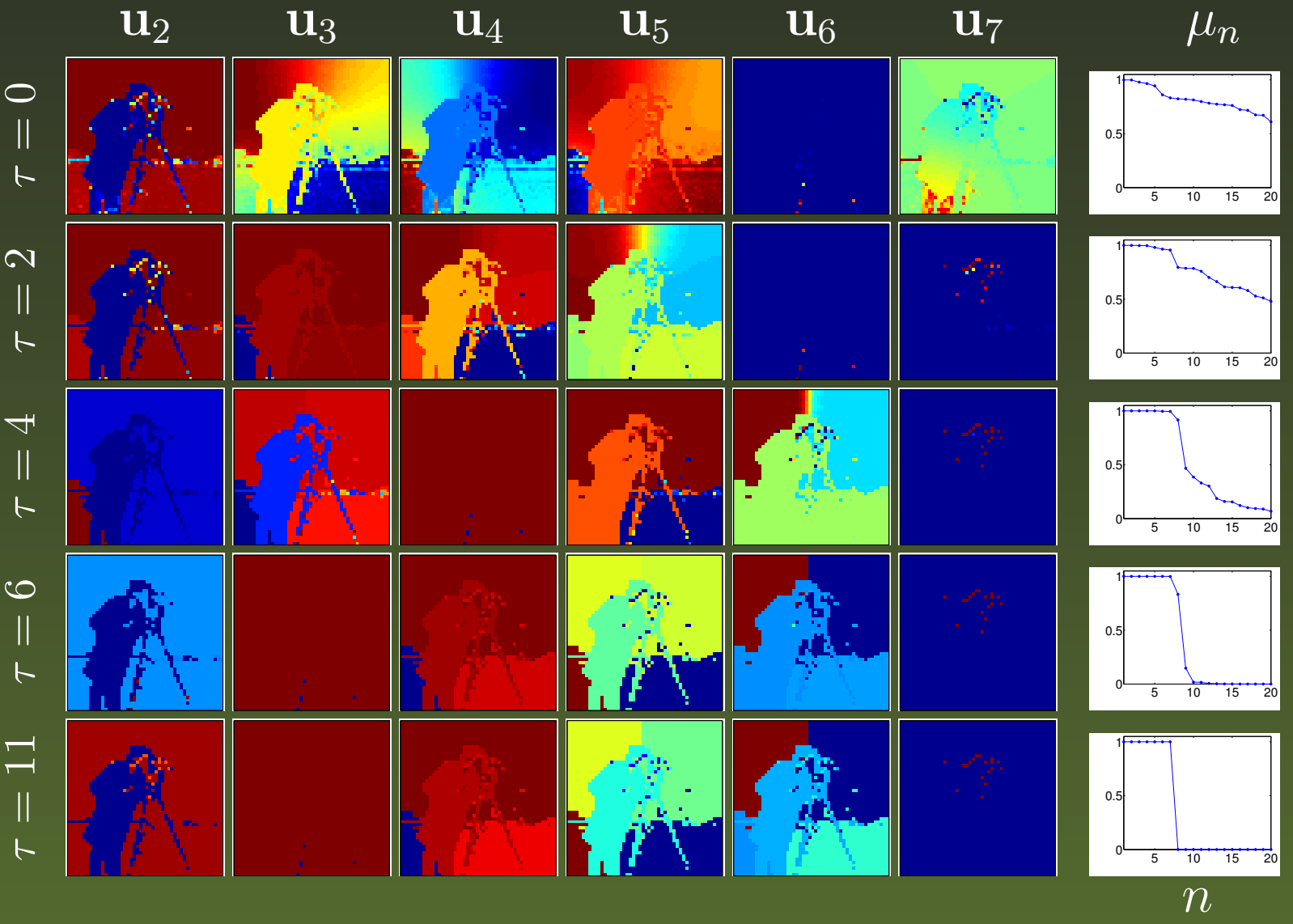
$\mathbf{X} = \mathbf{X}\mathbf{W}\mathbf{D}^{-1}$ — Update whole dataset

Connection with spectral clustering (cont.)

- GBMS can be written as repeated products $\mathbf{X} \leftarrow \mathbf{XP}$ with the random-walk matrix $\mathbf{P} = \mathbf{WD}^{-1}$ equivalent to the graph Laplacian ($\mathbf{W}$: Gaussian affinities, $\mathbf{D}$: degree matrix, $\mathbf{P}$: posterior probabilities $p(n|\mathbf{x}_m)$).
- In phase 1 $\mathbf{P}$ is quickly changing as points cluster.
- In phase 2 $\mathbf{P}$ is almost constant (and perfectly blocky) so GBMS implicitly extracts the leading eigenvectors (power method) like spectral clustering.
- Thus GBMS is much faster than computing eigenvectors (about 5 matrix-vector products are enough).
- Actually, since $\mathbf{P}$ is a positive matrix it has a single leading eigenvector (Perron-Frobenius th.) with constant components, so eventually all points collapse.

Connection with spectral clustering (cont.)

Leading 7 eigenvectors $\mathbf{u}_1, \ldots, \mathbf{u}_7 \in \mathbb{R}^N$ and leading 20 eigenvalues $\mu_1, \ldots, \mu_{20} \in (0, 1]$ of $\mathbf{P} = (p(n|\mathbf{x}_m))_{nm}$:



Accelerated GBMS algorithm

- At each iteration, replace clusters already formed with a single point of mass $=$ no. points in cluster.
- The algorithm is now an alternation between GMS blurring steps and connected-component reduction steps.
- Equivalent to the original GBMS but faster.
- The effective number of points $N^{(\tau)}$ (thus the computation) decreases very quickly.
- Computational cost:
  - k_1: average number of iterations per point for GMS
  - k_2: number of iterations for GBMS and accelerated GBMS

GMS	GBMS	Accelerated GBMS
$2N^2Dk_1$	$\frac{3}{2}N^2Dk_2$	$\frac{3}{2}D\sum_{\tau=1}^{k_2}(N^{(\tau-1)})^2$

Experiments with image segmentation

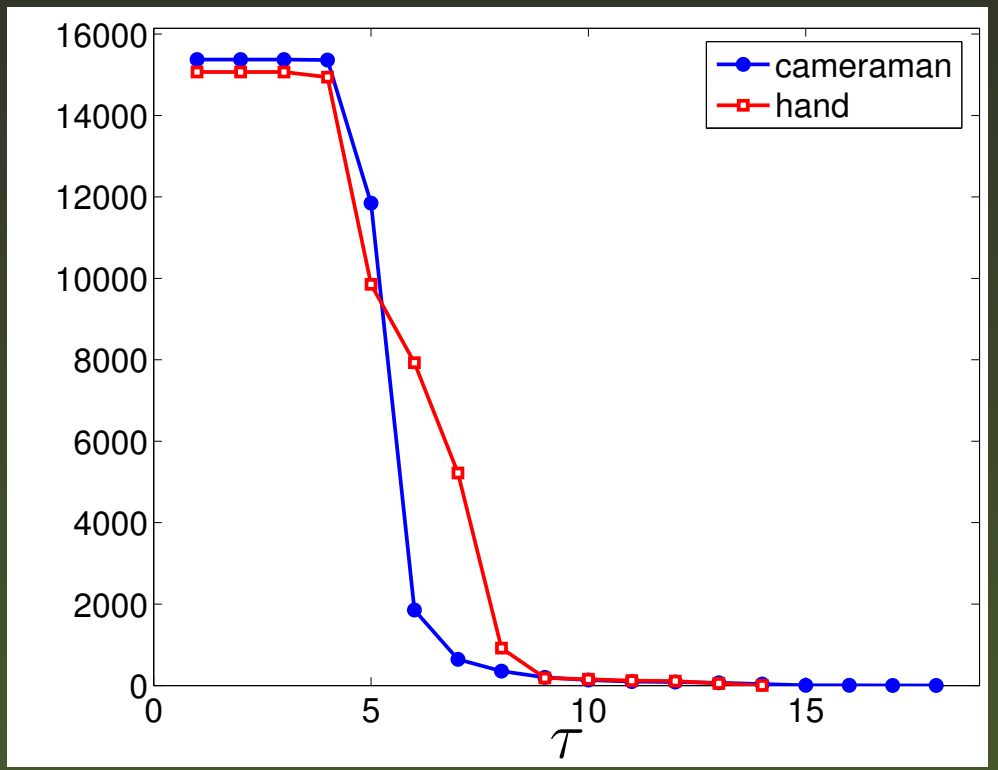
Original image | GMS | GBMS

	Number of iterations		
Image	GMS	GBMS	Accelerated GBMS
`cameraman` 124×124	71.5 ($\sigma = 24.2$)	18 ($\sigma = 20.3$)	4.6 ($\sigma = 20.3$)
`hand` 137×110	36.4 ($\sigma = 33$)	14 ($\sigma = 24$)	4.8 ($\sigma = 24$)

Experiments with image segmentation (cont.)

Effective number of points $N^{(\tau)}$ in accelerated GBMS:

$N^{(\tau)}$

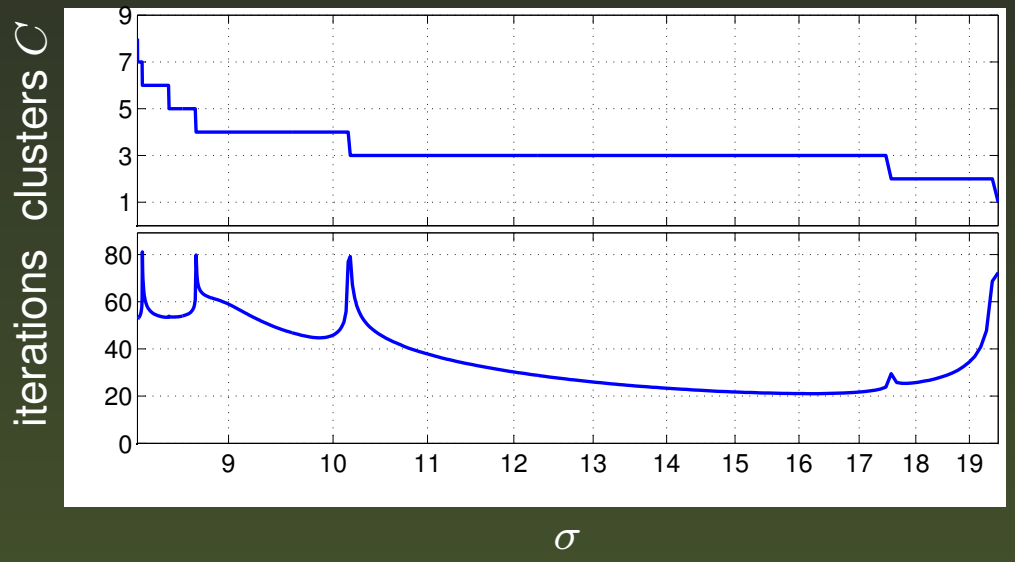


Accelerated GBMS is

- $2\times$–$4\times$ faster than GBMS
- $5\times$–$60\times$ faster than GMS

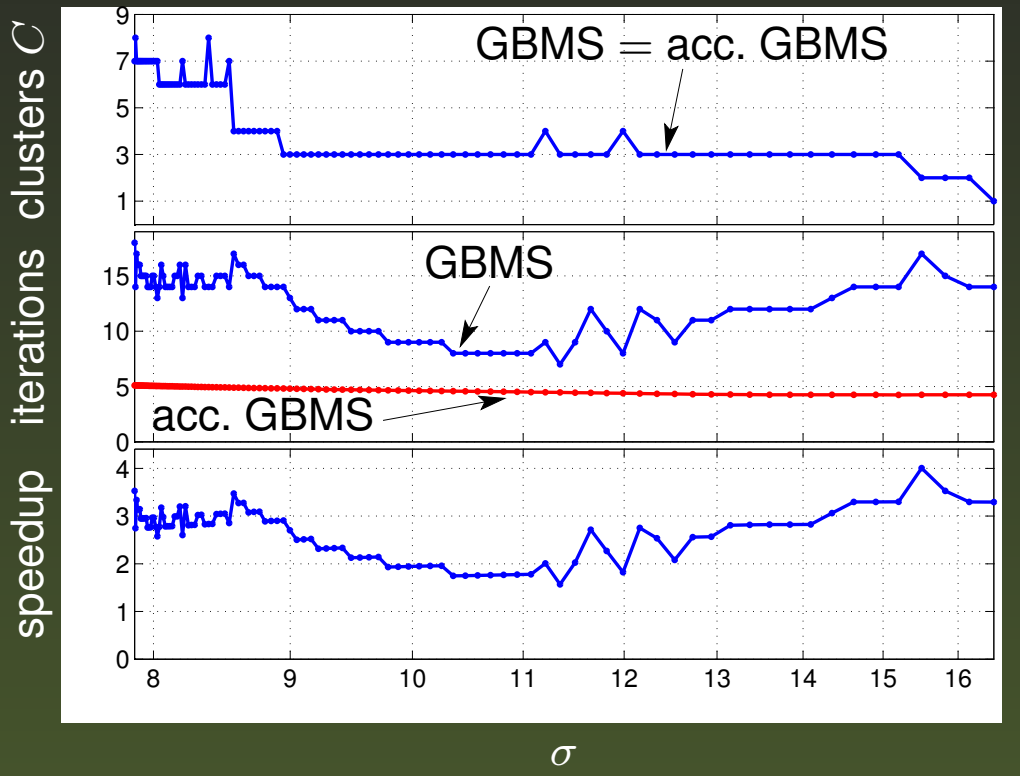
Experiments with image segmentation (cont.)

GMS

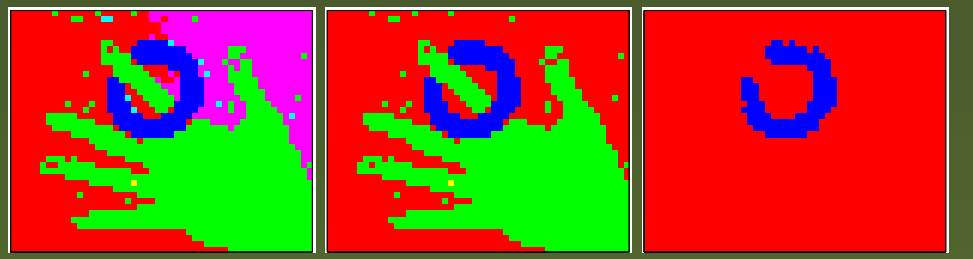


$\sigma = 8.4$ $C = 6$ | $\sigma = 10.1$ $C = 4$ | $\sigma = 18.3$ $C = 2$

GBMS and accelerated GBMS



$\sigma = 8.1$ $C = 6$ | $\sigma = 8.8$ $C = 4$ | $\sigma = 15.8$ $C = 2$



Experiments with image segmentation (cont.)

- Segmentations of similar quality to those of GMS but faster.
- Computational cost $\mathcal{O}(kN^2)$. Possible further accelerations:
  - Fast Gauss transform
  - Sparse affinities $\mathbf{W}$:
    - Truncated Gaussian kernel
    - Proximity graph (k-nearest-neighbours, etc.)
- Bandwidth σ set by the user; good values of σ:
  - GMS: $\sigma \approx \frac{1}{5} \times$(image-side)
  - GBMS: somewhat smaller

  Can also try values of σ in a scaled-down version of the image (fast) and then scale up σ.

GBMS: summary

- We have turned a neglected algorithm originally due to Fukunaga & Hostetler into a practical algorithm by introducing a reliable stopping criterion and an acceleration.
- Fast convergence (cubic order for Gaussian clusters).
- Connection with spectral clustering (GBMS interleaves refining the affinities with power iterations).
- Excellent segmentation results, faster than GMS and spectral clustering; only parameter is σ (which controls the granularity).
- Very simple to implement.
- We hope to see it more widely applied.