

HDCluster: High-Degree Graph Clustering for Robust Analysis of Single Molecule Localization Microscopy

Ismail M. Khater^{1,2*}, Ivan Robert Nabi^{3#}, Ghassan Hamarneh^{4#}

¹ Department of Genetics, Harvard Medical School, Boston, MA, 02115, USA

² Department of Electrical and Computer Engineering, Faculty of Engineering and Technology, Birzeit University, Birzeit P627, Palestine

³ Life Sciences Institute, Department of Cellular and Physiological Sciences, School of Biomedical Engineering, University of British Columbia, Vancouver, BC V6T 1Z3, Canada

⁴ Medical Image Analysis Lab, School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada

Co-senior authors

* Correspondence: ikhater@birzeit.edu; ismail_khater@hms.harvard.edu

Abstract

Clustering is a fundamental task in data analysis: grouping similar objects together into distinguishable subsets. Here, we introduce HDCluster, a novel high-degree graph-based clustering algorithm designed to effectively and rapidly handle various real-world clustering applications, particularly in the context of super-resolution single molecule localization microscopy (SMLM). HDCluster efficiently handles datasets with large and variable numbers of clusters, without requiring prior knowledge of the cluster count, relying on only one parameter. The high speed and efficiency of HDCluster allow it to handle large SMLM datasets with millions of localizations. A comprehensive quantitative comparison against state-of-the-art clustering methods using simulated, public, and real-world datasets demonstrates that HDCluster outperforms other clustering algorithms in terms of time efficiency and clustering performance measures, such as ARI and AMI. HDCluster is particularly robust to noise, making it a promising and effective tool for various clustering tasks in big-data settings, such as SMLM.

INTRODUCTION

Clustering is an unsupervised machine learning task where data samples are grouped according to similarities, ensuring that similar samples fall into the same group while dissimilar ones are separated. Hence, grouping is formed based on intrinsic similarity patterns across the data¹. Data clustering is valuable in domains where ground truth labels are not available or hard to be collected, such as single molecule localization microscopy (SMLM).

SMLM comprises a family of super-resolution imaging techniques, such as Photoactivated Localization Microscopy (PALM)² and Stochastic Optical Reconstruction Microscopy (STORM)³, that circumvent the Abbe diffraction limit of conventional light microscopy via stochastic excitation of fluorophores and fitting of the point spread function (PSF) to localize single fluorophores at precisions of tens of nanometers⁴. DNA points accumulation for imaging in nanoscale topography (DNA-PAINT) utilizes the transient binding of fluorescent DNA strands to achieve both high precision and multiplexing⁵. Minimal photon fluxes (MINFLUX) combines spatially structured illumination with localization analysis⁶ and Resolution Enhancement by Sequential Imaging (RESI) combines DNA-PAINT with sequential imaging of sparse labeling sites⁷ achieving nanometer and sub-nanometer resolution. SMLM data are represented as a point cloud of 2D/3D coordinates of molecular localizations⁸. SMLM data cluster analysis enables quantitative study of nucleic acid organization, protein aggregates, and other cellular structures at the nanoscale^{9–11}. However, SMLM data exhibit challenges such as noise, variations in density and size, large dataset volumes, and the presence of a high and often unknown number of clusters¹².

Several algorithms have been proposed for data clustering in general, and specifically for SMLM data analysis, based on diverse principles, including distance, density, graph/spectral properties, or other similarity-driven approaches¹². Every clustering algorithm has underlying assumptions and parameters that restrict its universality across applications and data types. For example, the k-means algorithm requires the number of clusters to be specified ahead of time and assumes spherical clusters of similar sizes^{1,13}. Other clustering methods are sensitive to noise and outliers (e.g., hierarchical clustering), lack scalability to very large datasets in terms of time and space (i.e., memory), and struggle with clusters of varying shape and density^{1,13}. Here, we introduce HDCluster, a similarity graph-based clustering algorithm, and show that it outperforms competing clustering algorithms and is particularly robust to the challenges of noisy, heterogeneous point cloud-based SMLM data. HDCluster realizes various computational tasks aimed at SMLM data analysis, including reconstructing binding sites, identifying nanoclusters and macromolecular structures, and denoising the data.

RESULTS

1. HDCluster Framework and Computational Tasks

HDCluster is a spatial data clustering method that can be leveraged to find clusters in SMLM data. The HDCluster algorithm has one parameter, merging threshold (*mTh*), used to construct the similarity graph and then get the node degree as a feature to start iterating and merging the nodes to get the clusters. The setting of the *mTh* parameter is related to the physical characteristics of

the clusters, i.e., the size/scale of the sought clusters (Figure 1A). HDCluster finds the clusters after the convergence condition is achieved and returns the per-localization class label, including labels for noisy localizations. β is an optional parameter to control noise removal with a default setting that can be changed if needed. See the Methods Section for more details about the algorithm.

Figure 1B illustrates HDCluster capabilities, encompassing various computational tasks for SMLM data analysis, such as (1) emitter reconstruction (or binding sites reconstruction¹⁴), (2) finding true biological clusters and nanoclusters in the SMLM data, and (3) denoising SMLM data.

2. HDCluster Performance Comparison with Other Clustering Methods

To evaluate clustering performance and tune the algorithm's parameters, we used the adjusted Rand index (ARI)¹⁵ and the adjusted mutual information (AMI)¹⁶ as performance metrics. ARI measures similarity between predicted and ground truth labels and AMI captures the amount of shared information between clusters, corrected for chance. AMI is a more robust measure when the number of clusters varies and accounts for bias toward certain cluster size distributions^{17,18}.

We applied HDCluster and other commonly used clustering methods to the shape benchmarking dataset¹⁹, widely used for assessing clustering methods' performance. This data set of 2D points and their corresponding membership as ground truth (GT) labels includes blob-like and spherical clusters similar to SMLM data of biological clusters: Aggregation with 7 clusters, Flame with 2 clusters, R14 with 14 clusters, and D31 with 31 clusters (Supp. Figure 1A). We tested state-of-the-art clustering methods, including Border Peeling (BP)²⁰, Quasi-Cluster Centers (QCC)²¹, Robust Continuous Clustering (RCC)²², Density Peaks (DP)²³, mean-shift (MS)²⁴, and Density-Based Spatial Clustering of Applications with Noise (DBSCAN)²⁵, as well as HDCluster, on these benchmark datasets (Supp. Table 1 shows the number of parameters and other information for the studied methods). We tuned each method's parameters based on AMI. Supp. Figure 1A shows the qualitative results and Figure 1C summarizes the quantitative results when applying the clustering methods to the shape benchmarking dataset. Only HDCluster and DP retrieve the correct number of clusters across all tested datasets. However, DP shows poor performance on the Flame dataset (ARI of 33% and AMI of 41%), where HDCluster achieves much higher accuracy (92% ARI and 86% AMI). Notably, HDCluster outperforms DBSCAN on the D31 and R15 datasets, both of which mimic nanocluster structures observed in SMLM data.

Nieves et al.²⁶ introduced a standard benchmarking framework for comparing clustering algorithms on SMLM point cloud data. Their work includes generating simulated ground truth point patterns with realistic blinking and localization uncertainty. The dataset comprises 10 distinct scenarios/conditions (variations in density, cluster size, blinking behavior, background noise, etc.) with and without multiple blinking artifacts. Each scenario contains 50 different point cloud files. They tested and compared the performance of several clustering methods, which include: DBSCAN²⁵, topological mode analysis tool (ToMATo)²⁷, kernel density estimation (KDE)²⁸, fast optimized clustering algorithm for localizations (FOCAL)²⁹, cluster analysis by machine learning (CAML)³⁰, ClusterViSu³¹, and SR-Tesseler³². The methods' parameter spaces are scanned systematically under each condition. In their results, DBSCAN, ToMATo, and KDE generally

perform well across multiple conditions and are relatively robust, with DBSCAN showing superiority. The other methods, ClusterViSu, SR-Tesseler, and FOCAL, show weaker performance in many settings. All methods suffer performance drops under multiple blinking localization conditions. We assessed HDCluster using the benchmarking SMLM datasets in accordance with the methodology outlined by Nieves et al.²⁶. All experiments were performed with the denoising feature activated. We tuned the key parameters of HDCluster, mTh and the optional filtering parameter β , to maximize the average AMI of the 50 files of each scenario.

For performance comparison, we selected DBSCAN, the most effective method identified by Nieves et al.²⁶, and employed their reported parameters to ensure fairness. The AMI and ARI values for HDCluster and DBSCAN are presented in Supp. Table 2. Across all simulated scenarios, HDCluster outperforms or matches DBSCAN in both ideal (i.e., no added blinking) and multiple blinking scenarios. In the ideal scenarios, both methods achieve comparable results, with HDCluster showing slightly higher ARI and AMI values in most cases (e.g., Scenario 3: 0.90 ± 0.017 vs. 0.84 ± 0.017 for ARI). However, when multiple blinking is introduced, HDCluster demonstrates greater robustness, maintaining high average clustering accuracy (average ARI ≈ 0.57 and AMI ≈ 0.68) compared to DBSCAN, whose performance drops noticeably (average ARI ≈ 0.51 and AMI ≈ 0.62). Under multiple-blinking (the harder, SMLM-like condition), HDCluster shows a clear advantage of about +0.06 absolute difference in both ARI and AMI (≈ 9.7 – 11.8% relative improvement). This highlights HDCluster's ability to efficiently handle emitter counting and multiple blinking artifacts typical in SMLM data.

3. HDCluster Robustness to Noise and Scalability

To test the ability of the different clustering methods to handle noisy data, we generated datasets that comprise clusters with various shapes and densities with additive noise points as described in Methods Section 4. Figure 2A shows one experiment for the generated clusters concatenated with a noise level of 20% of the total clustered points drawn from a uniform distribution. DBSCAN and HDCluster efficiently handle the noisy data and extract clusters, while QCC, DP, RCC and MS are unable to accurately segment noise from clusters. We varied the noise level from 0% (clusters only) to 100% with a step size of 10% and compared DBSCAN, BP and HDCluster at each noise level (Figure 2B). The parameter(s) for DBSCAN and HDCluster are selected based on ground truth data and the AMI score, as shown in Figure 2C, which shows a single case where the noise level is 100%. DBSCAN and HDCluster are robust to high noise levels; for BP, AMI and ARI are progressively reduced as the noise level exceeds 40% (Figure 2B).

To assess the speed of the clustering algorithm for SMLM data, we applied the clustering methods to the same datasets and on the same hardware (Intel(R) Core(TM) i7-4790 CPU @ 3.6GHz. Memory 32 GB, without other processes/applications running during testing). We first calibrated the running time for each algorithm on the small dataset from Figure 2A. Some of the algorithms are implemented in Python, while others are implemented in MATLAB. In our comparison, we used the latest optimized implementation of DBSCAN in MATLAB 2024b. HDCluster is the fastest algorithm, retrieving clusters in ~ 0.046 seconds, with DBSCAN second followed by MS, DP, RCC, and QCC, and finally the BP algorithm (Figure 2D). We then applied the algorithms to a real SMLM dataset of Caveolin-1 labeled HeLa cells of more than 1.5 million localizations³³ (Supp. Figure 2).

Only HDCluster, DBSCAN and MS were able to handle the data on our testing workstation (Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz. Memory 94 GB) with the other algorithms crashing the computer and failing to complete. We calculated the runtime for MS, DBSCAN, and HDCluster starting with 10% of total localizations, progressively increasing the sample size to the total number of localizations (Figure 2E). The runtime for MS grew exponentially with dataset size, and we stopped the analysis after reaching 30% of the 1.5 million localizations. For DBSCAN and HDCluster runtimes were calculated for increments up to the full data size (i.e., 100%, ~1.5 million localizations). At 100% of the total data, HDCluster is >55 times faster than DBSCAN, and we expect the trend to continue as the dataset becomes larger (Figure 2E).

4. HDCluster Binding Site Reconstruction Outperforms DBSCAN

Binding site and emitter reconstruction and protein counting are important tasks in SMLM data analysis^{14,34}. To test whether the representative (or consensus) localizations of HDCluster iterative merging approximate true emitter positions, we analyzed SMLM data from the Gatta-PAINT 80nm nanoruler that has three binding sites (emitters) organized linearly with a spacing of 80nm (Supp. Figure 3A). Calculation of the 1st–5th nearest neighbor (NN) distances calculated for HDCluster reconstructed emitters, shows that the majority of emitters are within ~80nm. HDCluster was then applied to the DNA origami nanostructure dataset published recently for the SPINNA method that compares experimental single-protein NN distances with simulated data generated from user-defined protein oligomerization models³⁴. HDCluster was used to reconstruct the binding sites for linear and triangular DNA origami nanostructures with three binding sites spaced ~15nm (Supp. Figure 3B). We used $mTh = 0.04 \times pixelSize$ (i.e., $mTh = 5.2nm$), where the pixel size is 130nm. For linear origami, HDCluster effectively reconstructs the binding sites for the two origami structures and calculated NN distances, where the first peak is at ~15nm and a second peak at ~30nm for linear origami and ~15nm for triangular nanostructures.

We then compared the binding site reconstruction performance of HDCluster and DBSCAN using a dataset of 3×4 origami grids arranged at 20nm spacing (Figure 3A)³⁵. HDCluster was applied with $mTh = 0.06 \times pixelSize$ (i.e., $mTh = 7.8nm$ for a pixel size of 130nm) and evaluated against the results reported by Stein et al.³⁵, who analyzed NN of DBSCAN cluster centers using the same parameters reported in their study: $\epsilon = 2 \times \sigma_{NeNA} = 2 \times 4.2 = 8.4 nm$ and $MinPts = 6$. Calculations of the 1st–5th NN distance show that the peaks are sharper and clearer for HDCluster relative to DBSCAN, with the highest peak at 20nm (peak (a) in Figure 3B), the second highest peak at 28nm (peak (b)) and so on (Figure 3B). The zoomed-in regions show superior clustering performance when using HDCluster over DBSCAN. With DBSCAN, some of the binding sites are filtered out, some merged and some averaged, affecting counting and the subsequent analysis of NN values. Indeed, the frequency of occurrences is more representative for HDCluster than DBSCAN; the expected 17 occurrences of peak (a) vs 8 occurrences of peak (d) in the 3×4 origami grids are accurately reported by HDCluster but not DBSCAN, which shows a similar frequency of events for peaks (a) and (d). Overall, HDCluster outperforms DBSCAN at various benchmarking clustering tasks and is significantly faster.

5. Conclusion

HDCluster is a simple, scalable, one-parameter clustering method that is very fast and robust to noise. By combining density-based clustering with graph-based techniques, HDCluster provides robust and accurate cluster identification in SMLM datasets, even with very large datasets of hundreds of thousands or millions of DNA-PAINT localizations, which allows for precise emitter reconstruction.

METHODS

1. HDCluster Algorithm

HDCluster is an unsupervised algorithm for spatial data clustering. It is an iterative similarity graph-based clustering method that discovers clusters and estimates their centroids, which are referred to as consensus localizations that approximate the true emitter positions. Figure 1A illustrates the algorithm based on 2D data points and details the underlying methodology. However, the algorithm works for 3D spatial data as well. The one parameter that the user needs to choose to run the algorithm is the “merging threshold (mTh)”. The algorithm begins by constructing a similarity graph based on the spatial proximity parameter mTh and calculating the node degree. Nodes are sorted based on the degree in descending order, so the algorithm iteratively merges each high-degree node with its connected neighbors, computing their mean spatial coordinates. The algorithm keeps iterating until all nodes are visited and produces a set of mean nodes that represent preliminary cluster centers. In the next stage, HDCluster reconstructs a new graph using the mean nodes and repeats the merging process across multiple iterations until the convergence condition is reached, where no edges remain between mean nodes, which is an indication that clusters are fully separated. Each top-level mean node is then assigned as a unique cluster ID, and labels are backtracked from these centroids to the original nodes such that every localization/point belongs to one cluster only. The resulting top-level mean nodes represent the emitters of the extracted nanocluster/cluster. Additionally, HDCluster can incorporate a denoising step to filter out low-degree nodes, effectively removing noisy/background localizations (see Methods Section 3 for a more extensive discussion of filtering options).

The mTh parameter selection is based on the physical characteristics of the underlying clusters, and one of the heuristics to set this parameter is based on Ripley’s H-function (see Methods Section 2 for a more extensive discussion of mTh). However, it can be selected based on the application and the spread of the clusters, as illustrated in Supp. Figure 4 and Equation (2). β is an optional parameter to control the noise removal with a default setting that can be changed if needed (see Methods Section 3 for a more extensive discussion of β).

2. HDCluster Merging threshold (mTh) and Evaluation Measures

Merging threshold (mTh) in HDCluster is a spatial proximity parameter that governs the similarity graph construction and guides the iterative merging of nodes toward the convergence of cluster means of consensus localization. Selection of mTh depends on the intrinsic properties of the dataset and the physical characteristics of the clusters, such as their spatial extent or point

dispersion. Also, it can be determined by H-Ripley's function or empirically through a data-driven heuristic to balance cluster separation and cohesion.

To demonstrate how we can set the mTh parameter, we simulated various scenarios of neighboring emitters (Supp. Figure 4A) with varying separation distance, defined as the space between the centers of the clusters. σ is the standard deviation of the Gaussian that quantifies the spread of the localizations/points around each cluster center. We derived the overlapping distance as a function of σ and separation distance to quantify the extent of overlap between the clusters, with negative values reflecting the separation between the boundaries of non-overlapping clusters (Equation (1)). The parameter mTh can be set according to Equation (2). We also define three measures of error: *counting error* ($error_C$), *localization error* ($error_L1$), and *localization error* ($error_L2$) (Equations (3–5)).

To assess the applicability of the HDCluster in reconstructing the emitters, we simulate various scenarios that may be similar to real-world situations, where the molecules can be sparse or very dense. We simulated two Gaussian emitters and change two parameters, (1) the separation distance (i.e., the distance between the centers/means of the two Gaussians or the emitters, which we use to simulate the density of the molecule in larger grids) and (2) the standard deviation of the Gaussians σ , that simulates the localization precision, i.e., the uncertainty in detecting the position of that emitter across repeated localizations, as shown in Supp. Figure 4A. The overlapping distance between the two clusters is defined as the distance between the borders of the clusters. The *overlap* between the localizations of the emitters is calculated according to Equation (1). The overlapping distance is a function of σ and *separation* distance. In the simulation, we can change the *separation* distance and σ to compute the amount of *overlap* between the neighboring clusters. The *overlap* is negative when the localizations of the neighboring emitters are not overlapping, and the *overlap* is positive when the localizations of the neighboring emitters are overlapping. The *overlap* equals zero when the borders of the two clusters are touching each other (Supp. Figure 4A).

$$Overlap = -(Separation - 2\sigma) \quad Eq(1)$$

Where σ is the standard deviation of the Gaussian-generated data.

The HDCluster parameter, *merging threshold* (mTh), can be selected to reconstruct the emitters' locations based on Equation (2).

$$mTh = \begin{cases} 2\sigma & : Overlap \leq 0 \\ |2\sigma - |\frac{Overlap}{2}|| & : Overlap > 0 \end{cases} \quad Eq(2)$$

In Equation (2), when the neighboring clusters are well separated or their borders touch each other (i.e., $Overlap \leq 0$), the parameter $mTh = 2\sigma$ ensures appropriate merging within each cluster. However, as clusters become closer and their borders overlap (i.e., $Overlap > 0$), mTh should be reduced to prevent unintended merging of adjacent overlapping clusters (i.e., mTh can be safely adjusted to $mTh = |2\sigma - |\frac{Overlap}{2}||$). This approach aligns with adaptive thresholding

strategies used in hierarchical and graph-based clustering methods, where thresholds decrease as clusters become denser or closer.

More formally, we simulated 3D grids of size $3 \times 3 \times 3$ and we changed the *separation* distance and the standard deviation of the Gaussians σ . Moreover, we also changed the density of the localizations around everyone of the simulated emitters as shown in Supp. Figure 4B. HDCluster is used to reconstruct the emitters. The parameter *mTh* is selected according to Equation (2). To assess the quality of emitter location reconstruction, we introduced three measures, as we have the ground truth location for the emitters. We quantitatively calculated three types of errors: *error_C* Equation (3), *error_L1* Equation (4), and *error_L2* Equation (5).

$$Error_C = \frac{1}{G} \times (R - G) \times 100\% \quad Eq(3)$$

where G is the number of ground truth (GT) emitters and R is the reconstructed emitters or binding sites. In the *error_C*, positive values mean overcounting, negative values mean undercounting, and zero means no counting error.

$$Error_L1 (nm) = \frac{1}{G} \sum_{i=1} D1_i \quad Eq(4)$$

where $D1_i$ is the distance between the GT emitter i and the location of the nearest reconstructed emitter. *Error_L1* is normalized by the number of ground truth emitters G .

$$Error_L2 (nm) = \frac{1}{R} \sum_{i=1} D2_i \quad Eq(5)$$

where $D2_i$ is the distance between the reconstructed emitter i and the location of the nearest GT emitter. *Error_L2* is normalized by the number of reconstructed emitters R .

We generated a larger 3D $3 \times 3 \times 3$ grid of simulated emitters and localizations, where the localizations' membership and the emitters' location are considered as a ground truth (GT) for evaluation via the error measures (Supp. Figure 4B). We generated 20 grids for each pair of σ and separation distance and computed the three types of counting (i.e., *error_C*) and localization errors (i.e., *error_L1* and *error_L2*). We started with a low density of 10 localizations per emitter and increased the density to 35 localizations per emitter. We notice that the counting error (*error_C*) is improved (Supp Figure 4B, left vs right), indicating emitter reconstruction is much better for a higher density of localizations. As σ decreases (i.e., more precise localizations), the 3 types of error become smaller (Supp. Figure 4B). For example, when using $\sigma = 2$, for any separation distance ≥ 10 nm, and for a number of localizations 35 per emitter, the *error_C* is zero, and both the *error_L1* and *error_L2* are almost zeros (i.e., the location of the reconstructed emitters is almost identical to the ground truth location). The error graphs show that the reconstruction errors remain relatively low, even under challenging conditions where nearby emitters exhibit substantial overlap. In the worst cases, *error_L1* and *error_L2* stay below ~ 30 nm and 14nm, respectively. *Error_L1* and *error_L2* fluctuate due to overcounting or undercounting, specifically in challenging cases of highly overlapping or low-density nanoclusters.

3. HDCluster Denosing Parameter - Spatial Filtering Option

HDCluster incorporates a denoising option to filter out noisy localizations (background, outlier points, etc.). The denoising capability/function in HDCluster can be enabled or disabled by the user. If enabled, the user can set a β value or rely on the default β value. The embedded filtering works by calculating the degree for all the nodes, then labeling all the nodes with a degree below a *low degree* with a 0 label (i.e., noisy localizations). The *low_degree* value is calculated as shown in Equation (6).

$$\text{Low_degree} = \beta \times \text{mean}(\text{degree}), \quad \text{Eq(6)}$$

where $\text{mean}(\text{degree})$ is the average node degree that is calculated when the graph is constructed with the mTh parameter, β which is a scalar that controls the denoising process. As a rule-of-thumb, the default value $\beta = 0.2$ is selected to label the non-clustered localizations as noisy. However, the user can select other β values that might control the noise removal according to the data, where larger β values employ harsh filtering. β is a scaling factor that controls the minimum node degree relative to the average connectivity of the graph, which determines how aggressively HDCluster removes low-connectivity nodes (potential noise). Setting $\beta = 0.2$ ensures that only nodes whose connectivity <20% of the mean graph degree are labeled as noise and generalizes well across different datasets without prior tuning. Setting β balances between retaining true cluster points at the periphery and filtering isolated/background points that do not contribute meaningfully to the cluster structure. We provided the user with the ability to change β . Higher values (i.e., >0.2) can lead to harsher filtering, which can fragment clusters and remove localizations within low-density regions.

4. Generating Data with Various Shapes and Densities with Additive Noise

To test the robustness of the clustering methods to noise, we generated a dataset with $K = 80$ Gaussian clusters that varied in density and were randomly positioned in 3D space. Each cluster has a minimum of 10 points and a maximum of 100 (for various-density clusters), and of various shapes, varying the covariance matrix diagonal of the Gaussians as given in Equation (7).

Let each cluster $i \in \{1, 2, \dots, K\}$ be defined by:

- Mean $\mu_i = [\mu_{ix}, \mu_{iy}, \mu_{iz}]^T \in R^3$

$$\Sigma_i = \begin{bmatrix} \sigma_{ix}^2 & 0 & 0 \\ 0 & \sigma_{iy}^2 & 0 \\ 0 & 0 & \sigma_{iz}^2 \end{bmatrix}$$

- Covariance matrix (diagonal):

$$\text{each cluster } i \text{ contains } n_i \text{ points: } x_{ij} \sim N(\mu_i, \Sigma_i), \quad j = 1, \dots, n_i \quad \text{Eq(7)}$$

Sampling procedure:

(i) For each cluster $i \in \{1, 2, \dots, K\}$, randomly sample the mean:

$\mu_i \sim \text{Uniform}([x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}] \times [z_{\min}, z_{\max}]);$ then

(ii) randomly sample diagonal variances:

$$\sigma_{ix}^2, \sigma_{iy}^2, \sigma_{iz}^2 \sim \text{Uniform}(\sigma_{\min}^2, \sigma_{\max}^2), \quad \text{where } \sigma_{\min} = 1 \text{ and } \sigma_{\max} = 21$$

from the diagonal covariance matrix Σ_i as shown above. Sample n_i points from the 3D Gaussian distribution $x_{ij} \sim N(\mu_i, \Sigma_i)$. Notice that the off-diagonal elements are all zero, so the clusters are axis-aligned ellipsoids. The variance values control the spread along each axis independently to get different shapes. We simulated the irregular cluster sizes by drawing n_i randomly (i.e., uniform distribution) to vary the clusters' densities. We then varied the noise by controlling the noise level, where the noise level can be selected for every experiment and drawn from a uniform distribution.

Acknowledgements

The authors gratefully acknowledge Dr. Johannes Stein (Harvard Medical School; currently, Max Planck Institute for Molecular Genetics) for his valuable discussions and for providing the DNA-PAINT dataset used in this study. We thank Dr. Antonios Lioutas for his valuable discussions and for providing infrastructure and experimental support for testing HDCluster. We also thank Dr. Ricardo Nunes Bastos (ONI) for kindly providing the Gatta-PAINT 80nm nanoruler dataset acquired using the ONI Nanoimager. Funding for this project was provided by the Canadian Institutes of Health Research (CIHR: PJT-175112) and the Natural Sciences and Engineering Research Council of Canada (NSERC: RGPIN-2019-05179, RGPIN-2020-06752). Antonios Lioutas and Johannes Stein were supported by awards to Dr. C.-ting Wu from the National Institutes of Health (NHGRI; RM1HG011016 and UM1HG011593) and the Genome Imaging Fund of Harvard Medical School. I.M.K. gratefully acknowledges the support of Birzeit University and the Fulbright Visiting Scholar Program for funding this research, and Dr. C.-ting Wu and the Genome Imaging Fund of Harvard Medical School for hosting and supporting the completion of this work.

References

1. Jain, A. K., Murty, M. N. & Flynn, P. J. Data clustering: a review. *ACM Comput. Surv.* **31**, 264–323 (1999).
2. Betzig, E. *et al.* Imaging intracellular fluorescent proteins at nanometer resolution. *Science* **313**, 1642–1645 (2006).
3. Rust, M. J., Bates, M. & Zhuang, X. Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM). *Nat. Methods* **3**, 793–795 (2006).
4. Shashkova, S. & Leake, M. C. Single-molecule fluorescence microscopy review: shedding new light on old problems. *Biosci. Rep.* **37**, (2017).
5. Jungmann, R. *et al.* Multiplexed 3D cellular super-resolution imaging with DNA-PAINT and Exchange-PAINT. *Nat. Methods* **11**, 313–318 (2014).
6. Balzarotti, F. *et al.* Nanometer resolution imaging and tracking of fluorescent molecules with minimal photon fluxes. *Science* **355**, 606–612 (2017).
7. Reinhardt, S. C. M. *et al.* Ångström-resolution fluorescence microscopy. *Nature* **617**, 711–716 (2023).
8. Sauer, M. & Heilemann, M. Single-molecule localization microscopy in eukaryotes. *Chem. Rev.* **117**, 7478–7509 (2017).
9. Owen, D. M. *et al.* PALM imaging and cluster analysis of protein heterogeneity at the cell surface. *J. Biophotonics* **3**, 446–454 (2010).
10. Ricci, M. A., Manzo, C., García-Parajo, M. F., Lakadamyali, M. & Cosma, M. P. Chromatin fibers are formed by heterogeneous groups of nucleosomes in vivo. *Cell* **160**, 1145–1158 (2015).
11. Bzdok, D., Altman, N. & Krzywinski, M. Statistics versus machine learning. *Nat. Methods* **15**, 233–234 (2018).
12. Khater, I. M., Nabi, I. R. & Hamarneh, G. A Review of Super-Resolution Single-Molecule

- Localization Microscopy Cluster Analysis and Quantification Methods. *Patterns* vol. 1
100038 Preprint at <https://doi.org/10.1016/j.patter.2020.100038> (2020).
13. Xu, R. & Wunsch, D., 2nd. Survey of clustering algorithms. *IEEE Trans. Neural Netw.* **16**, 645–678 (2005).
 14. Jungmann, R. *et al.* Quantitative super-resolution imaging with qPAINT. *Nat. Methods* **13**, 439–442 (2016).
 15. Hubert, L. & Arabie, P. Comparing partitions. *J. Classif.* **2**, 193–218 (1985).
 16. Nguyen, X., Epps, J. & Bailey, J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.* **11**, 2837–2854 (2010).
 17. Xu, D. & Tian, Y. A comprehensive survey of clustering algorithms. *Ann. Data Sci.* **2**, 165–193 (2015).
 18. Pineda, J. *et al.* Spatial clustering of molecular localizations with graph neural networks. *ArXiv* **abs/2412.00173**, (2024).
 19. Fränti, P. & Sieranoja, S. K-means properties on six clustering benchmark datasets. *Appl. Intell.* **48**, 4743–4759 (2018).
 20. Averbuch-Elor, H., Bar, N. & Cohen-Or, D. Border-peeling clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **42**, 1791–1797 (2020).
 21. Huang, J., Zhu, Q., Yang, L., Cheng, D. & Wu, Q. QCC: a novel clustering algorithm based on Quasi-Cluster Centers. *Mach. Learn.* **106**, 337–357 (2017).
 22. Shah, S. A. & Koltun, V. Robust continuous clustering. *Proc. Natl. Acad. Sci. U. S. A.* **114**, 9814–9819 (2017).
 23. Rodriguez, A. & Laio, A. Machine learning. Clustering by fast search and find of density peaks. *Science* **344**, 1492–1496 (2014).
 24. Comaniciu, D. & Meer, P. Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**, 603–619 (2002).

25. Ester, M., Kriegel, H.-P., Sander, J., Xu, X. & Others. A density-based algorithm for discovering clusters in large spatial databases with noise. in *Kdd* vol. 96 226–231 (aaai.org, 1996).
26. Nieves, D. J. *et al.* A framework for evaluating the performance of SMLM cluster analysis algorithms. *Nat. Methods* **20**, 259–267 (2023).
27. Pike, J. A. *et al.* Topological data analysis quantifies biological nano-structure from single molecule localization microscopy. *Bioinformatics* **36**, 1614–1621 (2020).
28. Silverman, B. W. *Density Estimation for Statistics and Data Analysis*. (Routledge, 2018).
29. Mazouchi, A. & Milstein, J. N. Fast Optimized Cluster Algorithm for Localizations (FOCAL): a spatial cluster analysis for super-resolved microscopy. *Bioinformatics* **32**, 747–754 (2016).
30. Williamson, D. J. *et al.* Machine learning for cluster analysis of localization microscopy data. *Nat. Commun.* **11**, 1493 (2020).
31. Andronov, L., Orlov, I., Lutz, Y., Vonesch, J.-L. & Klaholz, B. P. ClusterViSu, a method for clustering of protein complexes by Voronoi tessellation in super-resolution microscopy. *Sci. Rep.* **6**, 24084 (2016).
32. Levet, F. *et al.* SR-Tesseler: a method to segment and quantify localization-based super-resolution microscopy data. *Nat. Methods* **12**, 1065–1071 (2015).
33. Khater, I. M., Liu, Q., Chou, K. C., Hamarneh, G. & Nabi, I. R. Super-resolution modularity analysis shows polyhedral caveolin-1 oligomers combine to form scaffolds and caveolae. *Sci. Rep.* **9**, 9888 (2019).
34. Masullo, L. A. *et al.* Spatial and stoichiometric in situ analysis of biomolecular oligomerization at single-protein resolution. *Nat. Commun.* **16**, 4202 (2025).
35. Stein, J., Magni, L. & Church, G. M. DNA-PAINT imaging with hydrogel imprinting and clearing. *ACS Sens.* **10**, 3340–3346 (2025).

Supplementary Information and Figure Legends

Figure 1 Overview of the HDCluster framework to cluster, denoise, and reconstruct the emitters/binding sites in simulated and DNA-PAINT data. (A) Illustration of the HDCluster algorithm to obtain data clusters as well as the centroids (the consensus localizations that approximate the true emitter positions by computing the mean of means) of the extracted clusters. HDCluster is an iterative algorithm that starts from the similarity graph constructed based on the mTh parameter. The degrees of the graph nodes are calculated. The nodes are sorted in descending order of node degree. The algorithm starts iterating from the node with the highest degree. All the nodes that share edges with this high-degree node are merged; the mean (in terms of average coordinates for the nodes' spatial locations) of these merged nodes is calculated and all these merged nodes are marked as visited nodes. After that, the algorithm moves to the next highest degree node in the list, with at least one node that is marked as not visited, and so on until all the nodes are marked as visited. After concluding this first round of iterations, HDCluster enters another round by constructing an updated graph based on the mean nodes that were produced from the node merging of the first round of iterations. The algorithm will start iterating again from the mean node with the highest degree, as it did in the first round, and keep iterating and producing the mean of means until the convergence condition is reached. The convergence condition is reached when the mean nodes have no edges (i.e., the degree of the top-level mean nodes equals zero). Every one of the high-level mean nodes will be considered as a top-level cluster ID and will be used to backtrack the cluster labels to their means and so on down to the nodes that the means come from. The algorithm keeps propagating/backtracking the labels from the top-level mean node down to the nodes, and all the nodes will get an ID or label based on the top-level mean node. The algorithm does so for the rest of the top-level mean nodes. At the end, all the nodes get labeled and have membership in one and only one cluster with a unique ID. The top-level mean nodes are the emitters (consensus localizations that approximate the true emitter positions) of the nanoclusters in SMLM data. HDCluster is capable of filtering out the noisy localizations in addition to data clustering. (B) A general framework of SMLM data analysis and the computational tasks that HDCluster can perform for SMLM data analysis. (C) Quantitative analysis (ARI, AMI, #clusters) of clustering methods evaluated on the benchmark shape dataset¹⁹ presented in Supp. Fig. 1. The best parameter(s) for each method are selected based on the AMI measure.

Figure 2 Noise robustness analysis of various clustering methods at different noise levels. Runtime comparison of the clustering methods based on the simulated and real SMLM dataset. (A) Simulated 3D data of 80 Gaussian clusters with various densities and ellipsoidal shapes is generated with various noise levels. In this simulation, we show the data with a 20% noise level of the total number of localizations. The various clustering methods have been applied to find the clusters and label the noisy localizations. For the parametric methods, the method's parameter(s) are tuned based on the AMI. (B) Comparison of the clustering methods with the denoising capability for various noise levels. Some methods can handle noise but are not robust to high noise values in the data. (C) Parameter selection for the DBSCAN and HDCluster methods when applied to data with a 100% noise level. Both ARI and AMI measures were plotted for both

methods at various parameters. The parameters with high AMI value were selected to report the results presented in (B). (D) Calibrate the runtime for the various clustering methods when applied to the simulated dataset presented in (A). All the methods were run on the same computer and the same dataset for fair comparison. Not all the methods can be used on large datasets. (E) Runtime analysis for the methods that can be run on large datasets, such as SMLM data. HDCluster, DBSCAN, and mean-shift (MS) algorithms were run on a dataset of Caveolin-1 localizations³³ presented in the Supp. Figure 2. We start running the clustering methods on 10% of the whole data and calibrate the runtime, then we increase the localizations by 10% until we reach 100% of the total localizations. Not all the methods scale with the dataset size. HDCluster* refers to the version of the algorithm with the noise-handling option enabled.

Figure 3 Qualitative and quantitative comparison of binding sites reconstruction and quality of clustering when using DBSCAN and HDCluster when applied to 3×4 DNA origami dataset³⁵. (A) The quality of clusters impacts the binding sites' subsequent analysis, as in the nearest neighbor (NN) analysis. We adopt the same DBSCAN parameters used in Stein et al.³⁵ to find the clusters. DBSCAN can find the clusters, but when the clusters overlap, it might get them wrong. On the other hand, HDCluster can easily be adopted with one parameter to find more accurate clusters and consequently the binding sites. (B) Quantitative analysis of the reconstructed binding sites from both DBSCAN and HDCluster methods. The NN analysis for the binding sites from the 3×4 DNA 20 nm origami grids shows that the histogram peaks correspond to the geometrical grid of sites that distant 20 nm apart for both DBSCAN and HDCluster methods. However, the HDCluster is more realistic, while DBSCAN has limitations for the overlapped clusters and is sensitive to noisy points.

Supplementary Figure 1 Various clustering methods applied to benchmarking shape datasets of a known number of clusters and points' memberships to which cluster. The qualitative results comparison of the various clustering methods shows how the methods classify the points in comparison to the provided ground truth. We depict the results of seven clustering methods that do not require the number of clusters to be known ahead of time.

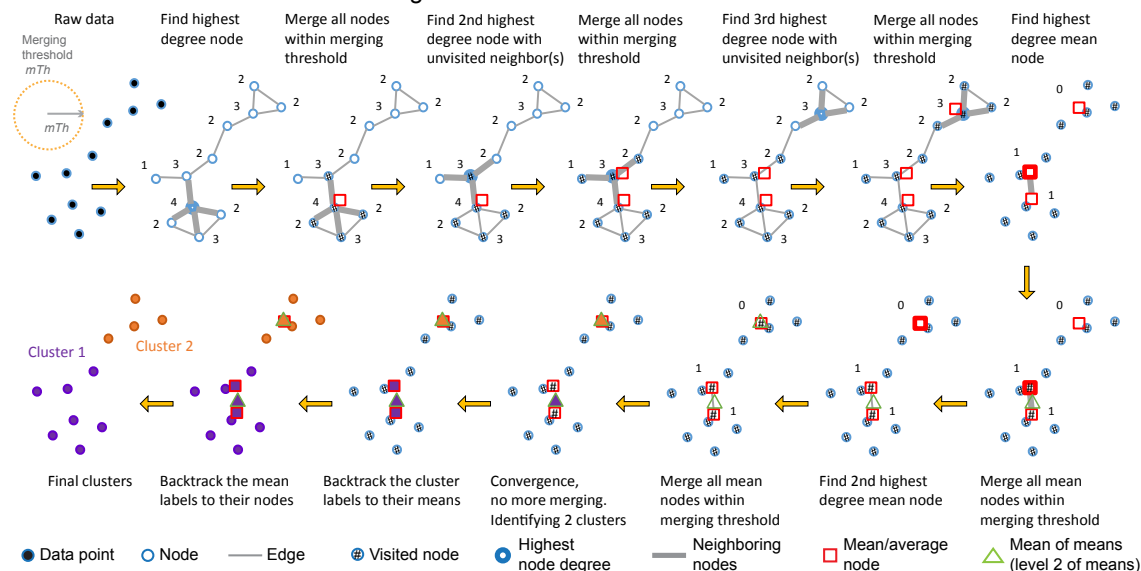
Supplementary Figure 2 Qualitative analysis comparison of the clustering methods based on the real SMLM dataset. Qualitative analysis showing the results when applying both HDCluster and DBSCAN methods to the Caveolin-1 data of more than 1.5 million localizations³³. The visualizations show the noisy localizations and the clusters. The arrows in the zoom-in regions show the quality of clustering for the overlapped clusters. The run time analysis for the HDCluster and DBSCAN algorithms on this data is shown in Figure 2E.

Supplementary Figure 3 Applying HDCluster to reconstruct the binding sites from dSTORM nanoruler and DNA-PAINT origami nanostructures datasets. (A) Gatta-PAINT 80 nm nanoruler imaged using ONI dSTORM Nanoimager. The HDCluster algorithm is used to denoise, cluster, and reconstruct the emitters of the imaged localizations of the nanorulers. The 1st–5th nearest neighbor (NN) distances of the reconstructed emitters are calculated. The

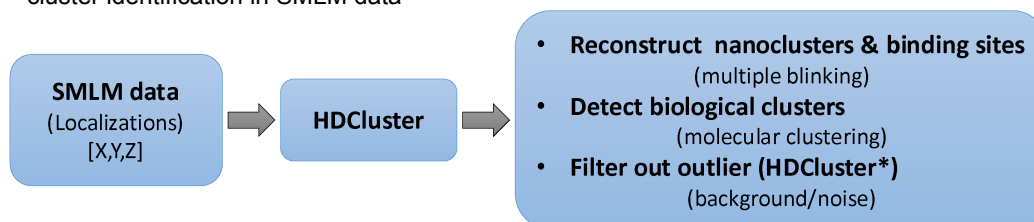
histogram of the 1st–NN distances shows that the majority of reconstructed emitters are distant ~80 nm from each other, and the second peak of the histogram is at the 2nd–NN of ~160 nm. (B) DNA-PAINT for two DNA origami nanostructures³⁴ is used to illustrate and validate HDCluster denoising, clustering, binding sites reconstruction, and quantification. For the linear nanostructures, HDCluster retrieves the binding sites, and the NN analysis shows the pattern of the binding sites' stoichiometry. Also, for the triangular nanostructures, HDCluster retrieves the binding sites, and the NN analysis shows the pattern of binding sites stoichiometry that corresponds to the designed origami nanostructures.

Supplementary Figure 4 (A) Generating Gaussian clusters with various standard deviations (σ) and the separation distance values. The cluster's density (i.e., number of localizations), the spread of localizations around the emitters, and the emitters' distances can be controlled to assess the performance of the emitters' reconstruction. σ and separation distance values can be used to set the merging threshold parameter of the HDCluster algorithm (Equation(2)). (B) 20 3D grids of Gaussian clusters with various σ and separation distance values are generated for each pair of σ and separation distance values for a specific cluster density. The counting (*error_C*) and localization errors (*error_L1* and *error_L2*) are calculated for every pair of σ and separation distance values to assess the quality of the reconstruction procedure for emitters/binding sites. Examples for low- and high-density clusters are shown.

A Illustration of the HDCluster algorithm



B General framework of HDCluster for cluster identification in SMLM data

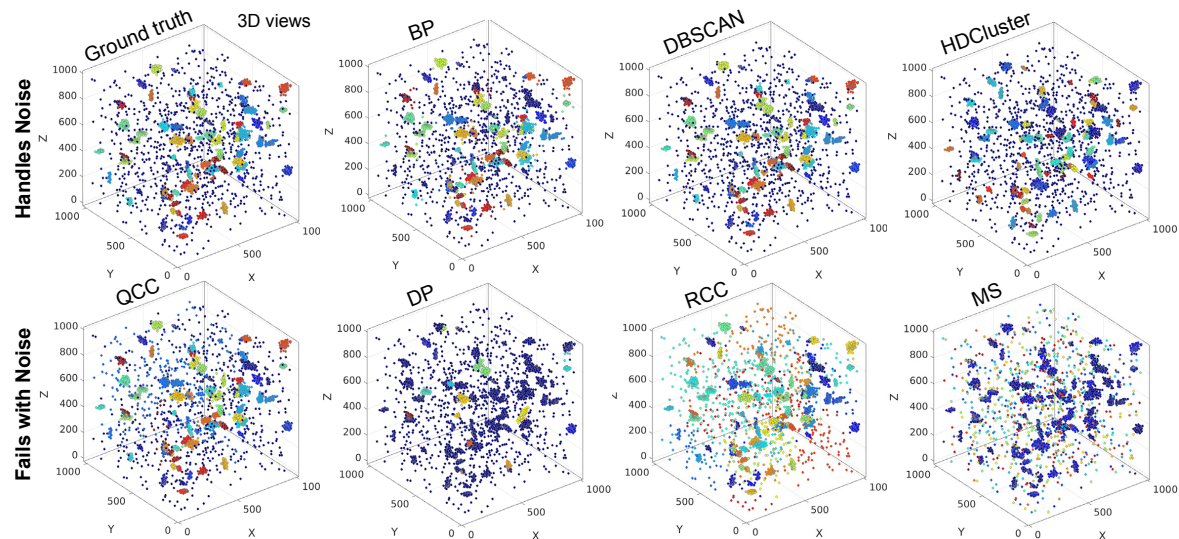


C Quantitative comparison of clustering methods on the benchmark shape dataset

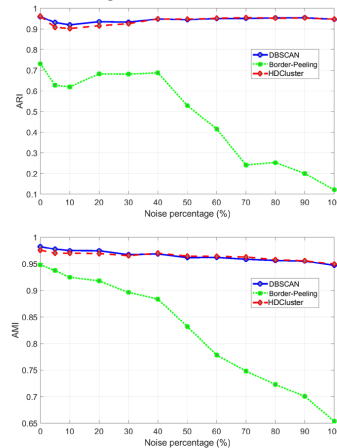
	Aggregation (x=7 clusters)			Flame (x=2 clusters)			R15 (x=15 clusters)			D31 (x=31 clusters)			Error = $\sum_{i=1}^4 x_i - \hat{x}_i $
	ARI	AMI	#clust ($\hat{x}$)	ARI	AMI	#clust ($\hat{x}$)	ARI	AMI	#clust ($\hat{x}$)	ARI	AMI	#clust ($\hat{x}$)	
BP	0.993	0.988	7	0.955	0.907	2	0.987	0.988	15	0.909	0.936	<u>32</u>	1
QCC	0.99	0.985	7	0.955	0.899	2	0.98	0.982	15	0.893	0.952	<u>29</u>	2
RCC	0.995	0.991	7	0.445	0.578	<u>5</u>	0.975	0.979	15	0.929	0.953	<u>33</u>	5
DP	0.998	0.996	7	0.327	0.411	2	0.993	0.994	15	0.935	0.955	31	0
MS	0.731	0.78	<u>6</u>	0.886	0.811	2	0.937	0.952	15	0.856	0.918	31	1
DBSCAN	0.99	0.984	7	0.922	0.861	2	0.933	0.941	15	0.845	0.911	<u>30</u>	1
HDCluster	0.99	0.986	7	0.922	0.861	2	0.982	0.984	15	0.929	0.949	31	0

Figure 1

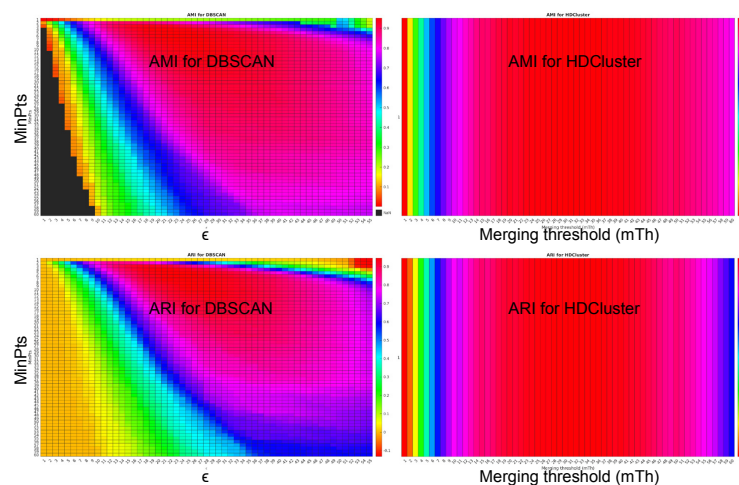
A Qualitative comparison of clustering methods on simulated datasets containing clusters with 20% noise



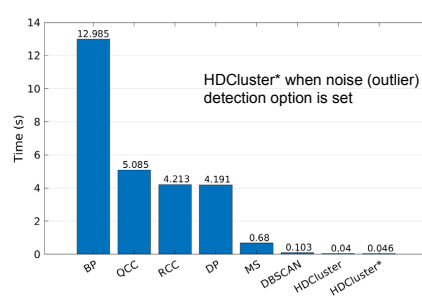
B Robustness of clustering methods across varying noise levels (percentage of total points)



C DBSCAN and HDCluster parameter tuning with 100% noise data



D Runtime comparison based on the simulated cluster dataset



E Runtime comparison on a super-resolution dataset of Caveolin-1 localizations in a HeLa cell

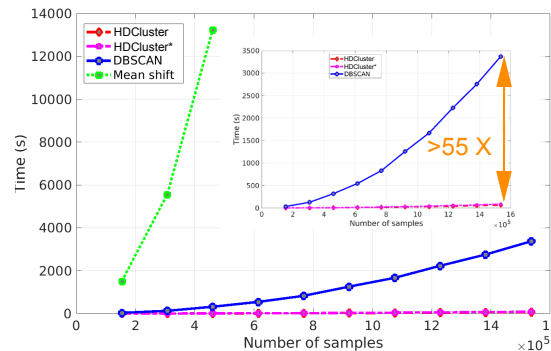
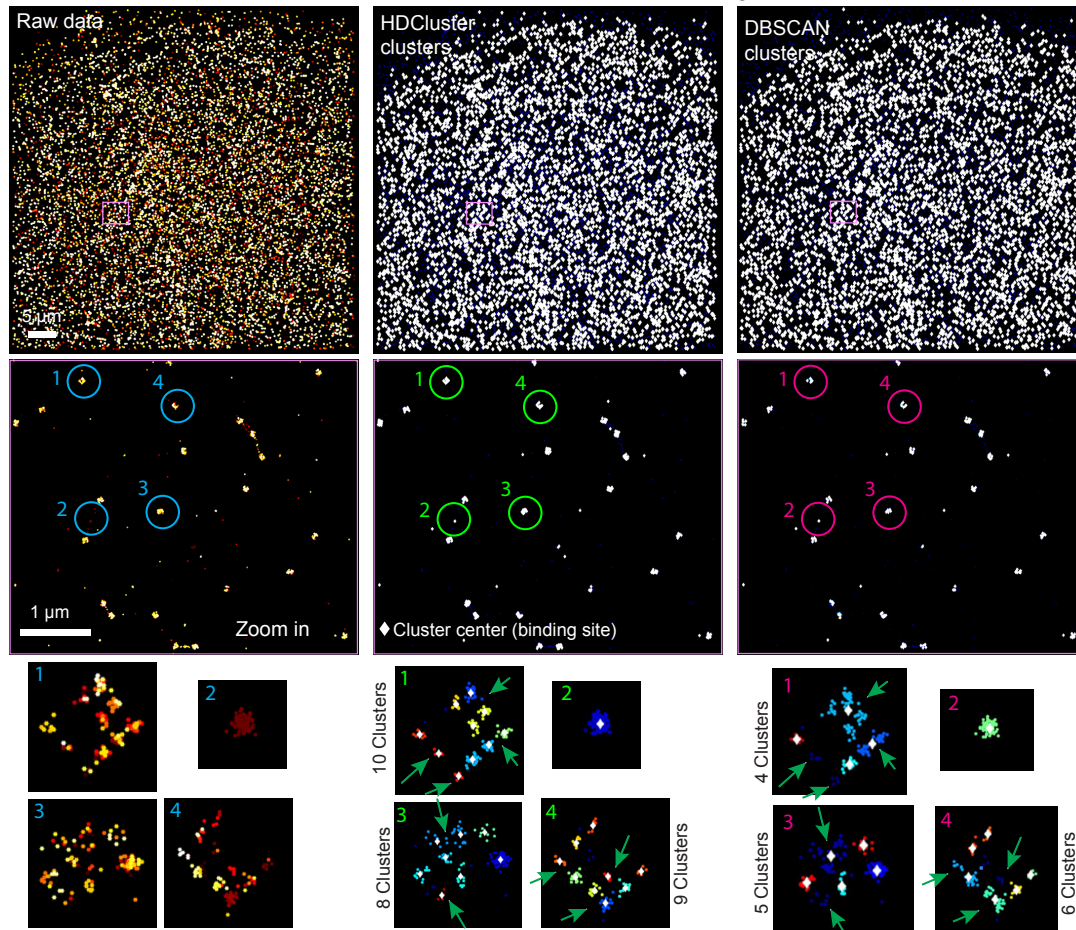


Figure 2

A Qualitative comparison of HDCluster and DBSCAN in binding sites reconstruction



B 1st–5th NN distances of binding sites reconstructed by HDCluster and DBSCAN on 3×4 DNA origami 20 nm grids

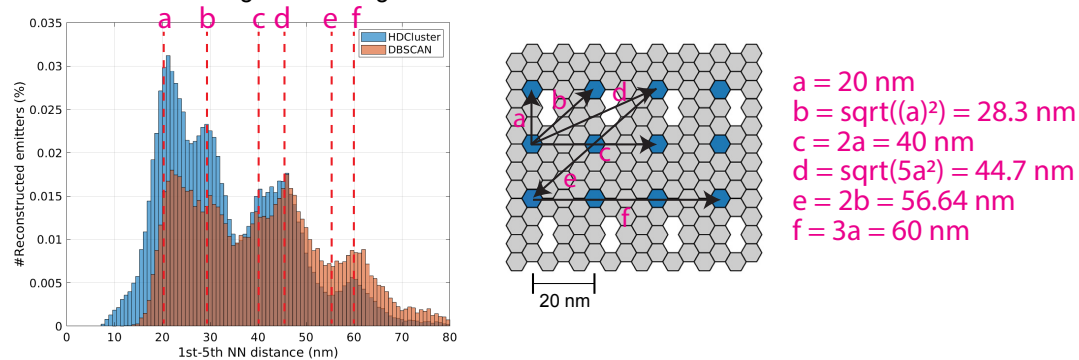
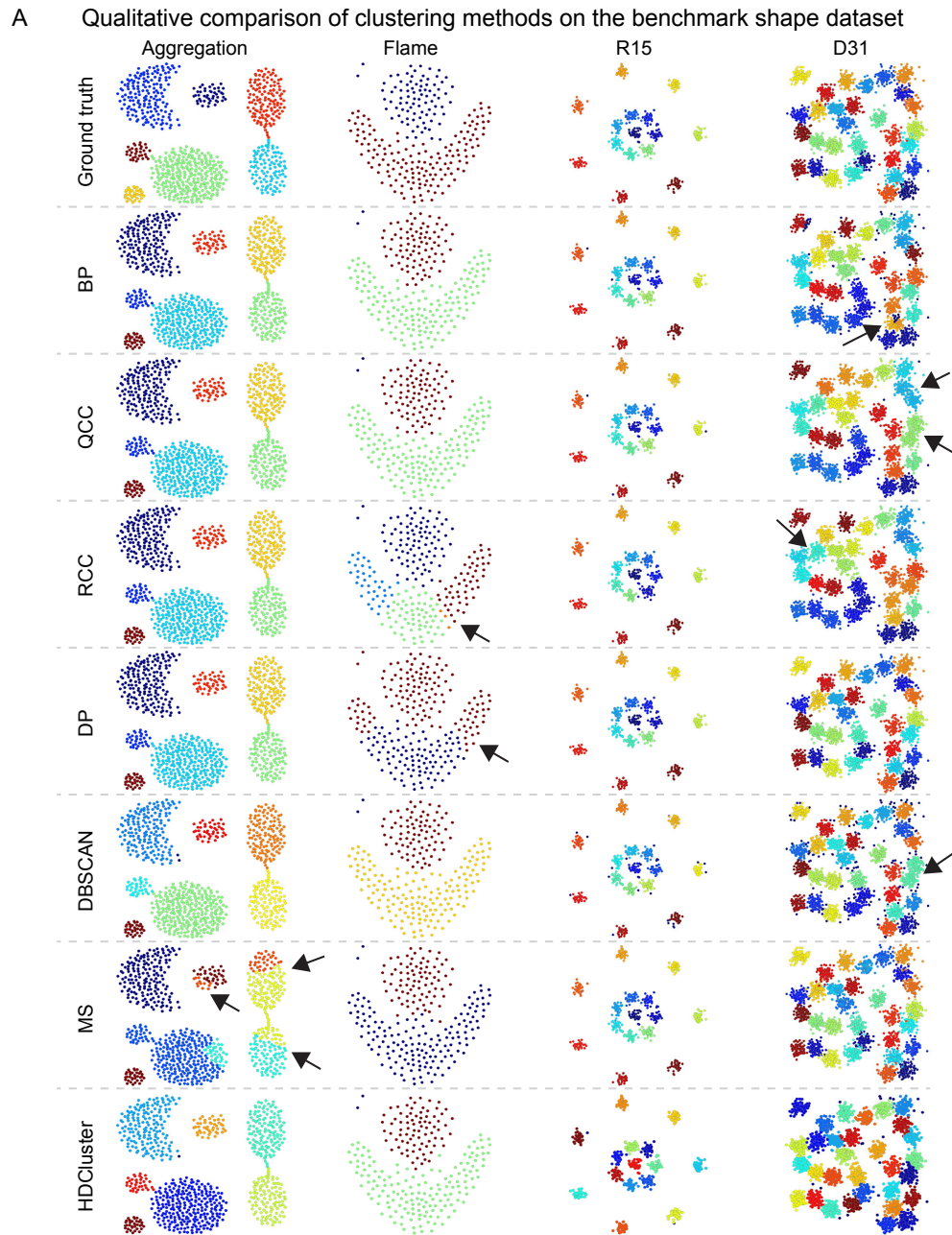
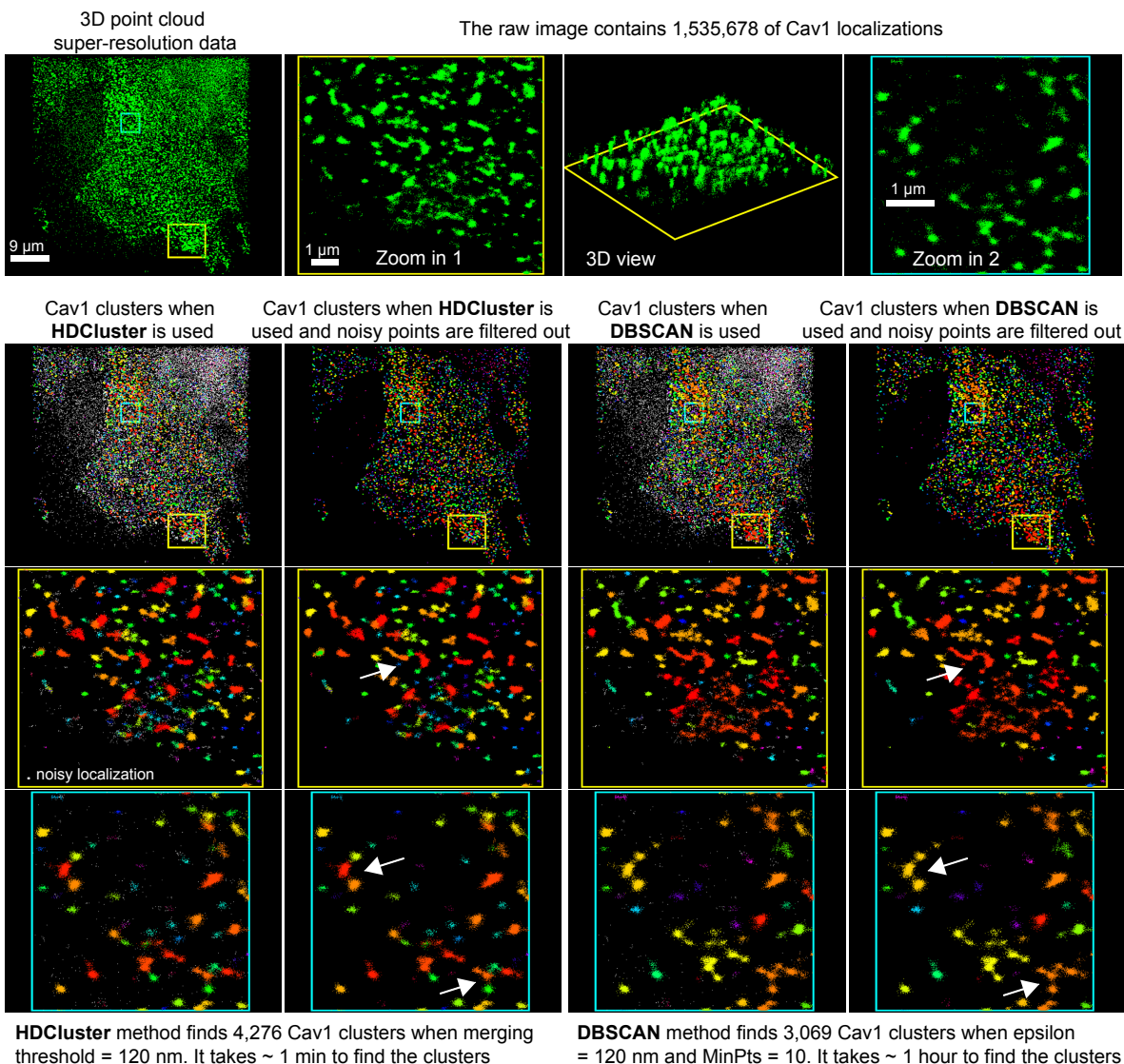


Figure 3



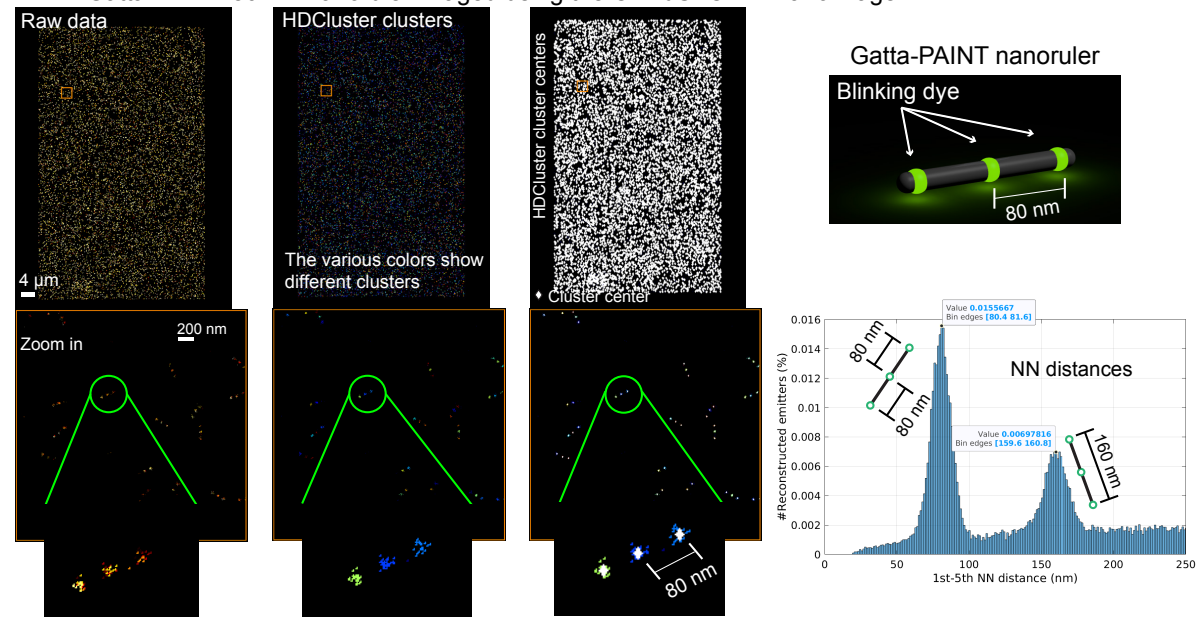
Supp. Figure 1

Clustering analysis of Caveolin-1 protein localizations (HeLa cell, super-resolution data)

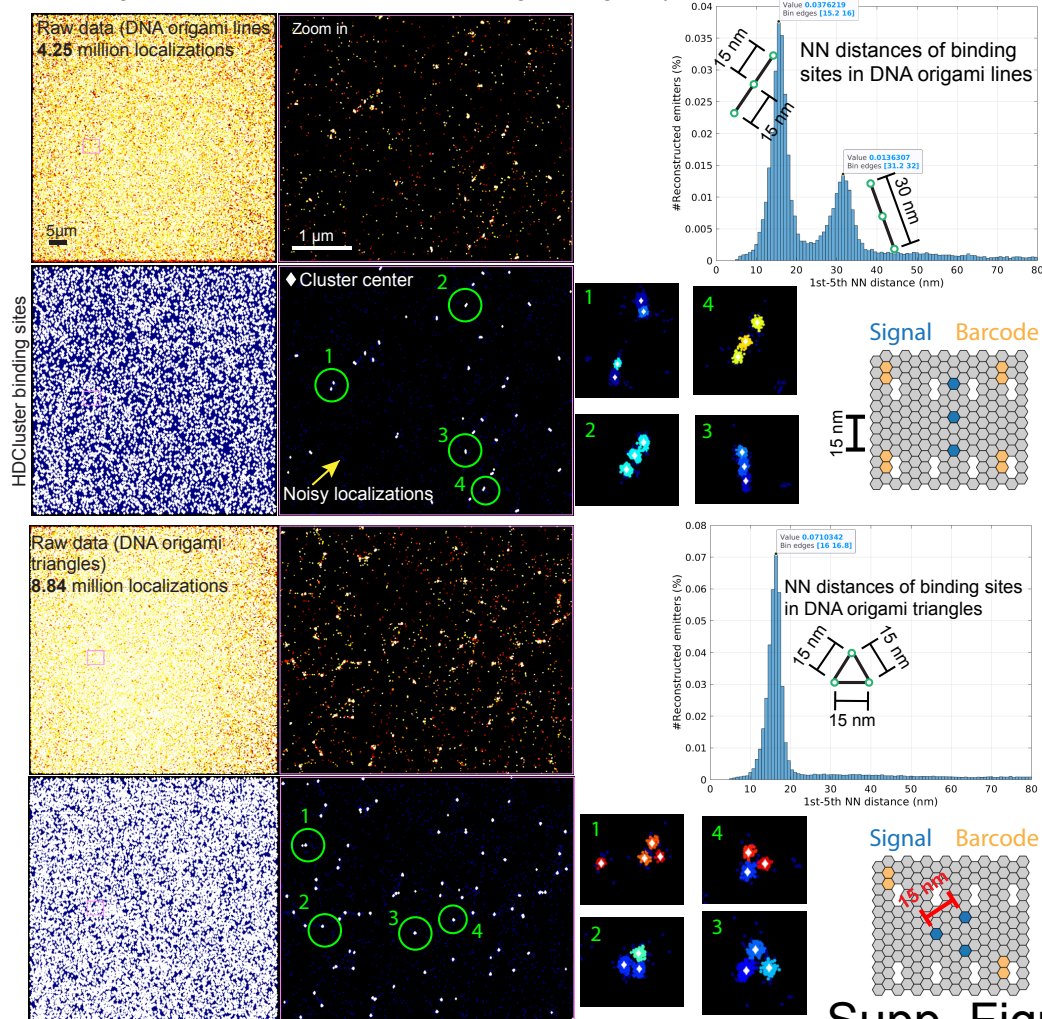


Supp. Figure 2

A Gatta-PAINT 80 nm nanoruler imaged using the ONI dSTORM Nanoimager

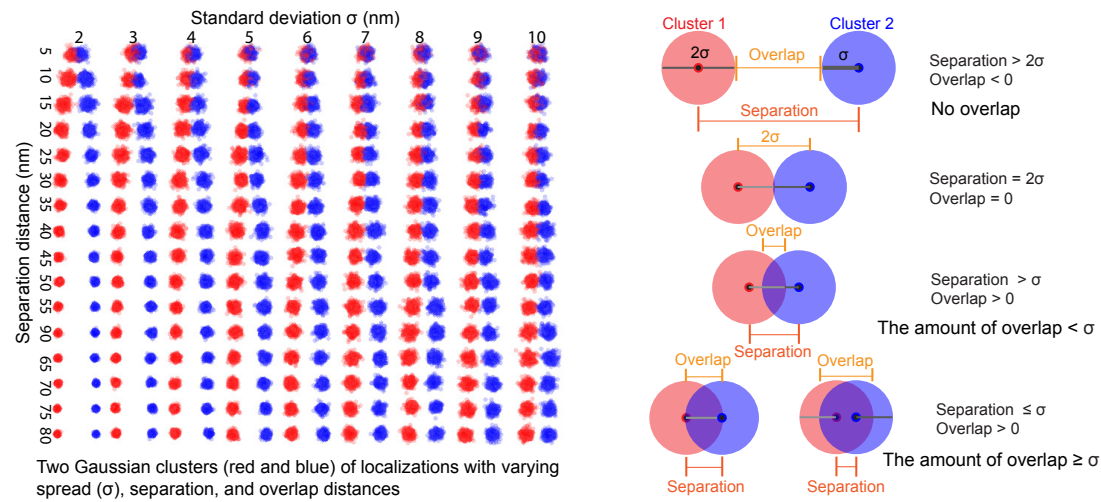


B DNA origami nanostructures of different designs imaged by DNA-PAINT

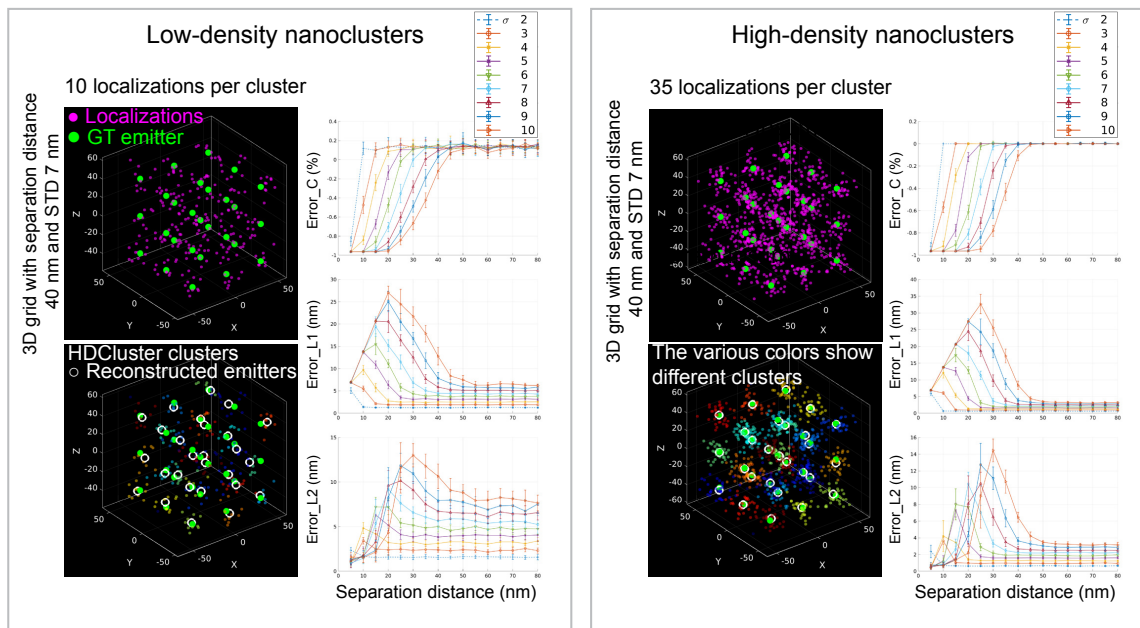


Supp. Figure 3

A Gaussian standard deviation and separation distance between neighboring emitters in the grids



B Reconstruction errors in simulated data grids



Supp. Figure 4

Supp. Table 1: Comparison of state-of-the-art spatial clustering methods^{20–25} based on their key characteristics and performance criteria. The table lists state-of-the-art clustering methods with their characteristics, with a particular focus on their applicability to SMLM data. Based on our study, we enumerate each method's robustness to noise, ability to explicitly label noisy localizations or points, scalability to large datasets, and effectiveness when dealing with a high number of clusters. We also assess the number of required parameters, the ease of parameter tuning, the ability to detect clusters of varying shapes and densities, and the computational efficiency when applied to large datasets containing hundreds of thousands of points.

Method †	Implicit Denoising Support §	Robust to noise §	Scalability for big data ‡	Ability to Handle Many Clusters	Parametric	Easiness of parameter selection ^	Arbitrary shape clusters	Various density clusters	Speed (when using 100's of thousands of samples)
BP	Supported (-1 label is given to noisy points)	Robust to a certain noise level (border points)	Not scalable (due to memory constraints)	Weak	Non-parametric	NA (non-parametric)	Limited	Limited	NA (due to lack of scalability)
QCC	Moderate support	Not robust	Not scalable (due to memory constraints)	Weak	2 parameters (k, α)	Moderate (the algorithm is not very sensitive to the parameters)	Limited	Good	NA (due to lack of scalability)
RCC	Not supported	Not robust	Not scalable (due to memory constraints)	Weak	Non-parametric (requires k for graph construction)	Easy (set k to construct the graph via $mkNN$ algorithm)	Limited	Good	NA (due to lack of scalability)
DP	Supported (via <i>IsHalo</i> option, 0 label is given to noisy points)	Not robust	Not scalable (due to memory constraints)	Weak	2 parameters can be set via decision graph (ρ, δ)	Moderate (manual/visual selection of the parameters. Not applicable when #clusters is very large)	Moderate (Requires changing the kernel)	Good	NA (due to lack of scalability)
DBSCAN ‡‡	Strong support (-1 label is given to noisy points)	Robust to high level of noises	Scalable	Strong	2 parameters (ϵ - radius, minPts)	Hard	Very good	Limited	Slow (tens of minutes)
MS	Not supported	Not robust	Moderate scalability (limited by memory constraints)	Moderate	1 parameter (Kernel bandwidth)	Easy (bandwidth is set based on clusters scale)	Limited	Good	Very slow (many hours)
HDCluster	Strong support (via <i>IsNoise</i> option, 0 label is given	Robust to high level of noises (Designed	Superior scalability	Strong	1 parameter (mTh); β is an optional parameter for filtering	Very easy (merging threshold is set based on physical	Limited	Very good	Very fast (seconds)

	to noisy points)	for SMLM-like noise)				scale of the clusters)			
--	------------------	----------------------	--	--	--	------------------------	--	--	--

† None of the methods in this table require prior knowledge of the number of clusters.

‡ By *big data* we refer to algorithms that are scalable and capable of handling hundreds of thousands to millions of points.

‡‡ For DBSCAN, results are reported based on MATLAB 2024b implementation.

^ Parameter selection refers to 2D or 3D spatial data. For non-parametric methods, parameter selection is implicitly determined by the algorithm based on the input data.

\$ This indicates whether the algorithm accounts for labeling noisy points. However, this does not necessarily mean the method is robust to noise or outlier detection.

§ This criterion was evaluated using generated datasets containing both clustered and noisy points (see Figure 2).

Supp. Table 2: Summary of clustering evaluation metrics across various scenarios and methods. Results are presented as the mean $\pm$ standard deviation, calculated from 50 independently generated files for each scenario in SMLM benchmarking dataset²⁶. DBSCAN adopts the parameters as reported in Nieves et al.²⁶ that were tuned based on ARI. Look at Supp. Table 3 for the parameters used to generate this table. Where, SEM is the standard error of the mean (SEM = std / sqrt(N)), where N=9 scenarios.

	Method	Ground Truth (GT) (No Added Blinking)		Multiple Blinking (MB)	
		ARI	AMI	ARI	AMI
Scenario 2	HDCluster	0.86 $\pm$ 0.02	0.85 $\pm$ 0.018	0.74 $\pm$ 0.059	0.82 $\pm$ 0.03
	DBSCAN	0.85 $\pm$ 0.023	0.84 $\pm$ 0.02	0.16 $\pm$ 0.069	0.36 $\pm$ 0.081
Scenario 3	HDCluster	0.90 $\pm$ 0.017	0.83 $\pm$ 0.025	0.72 $\pm$ 0.057	0.71 $\pm$ 0.045
	DBSCAN	0.84 $\pm$ 0.017	0.74 $\pm$ 0.022	0.69 $\pm$ 0.05	0.69 $\pm$ 0.042
Scenario 4	HDCluster	0.68 $\pm$ 0.064	0.62 $\pm$ 0.055	0.34 $\pm$ 0.099	0.41 $\pm$ 0.079
	DBSCAN	0.67 $\pm$ 0.057	0.60 $\pm$ 0.057	0.34 $\pm$ 0.103	0.41 $\pm$ 0.084
Scenario 5	HDCluster	0.55 $\pm$ 0.027	0.72 $\pm$ 0.009	0.38 $\pm$ 0.036	0.65 $\pm$ 0.012
	DBSCAN	0.56 $\pm$ 0.02	0.71 $\pm$ 0.011	0.42 $\pm$ 0.034	0.64 $\pm$ 0.018
Scenario 6	HDCluster	0.70 $\pm$ 0.02	0.74 $\pm$ 0.012	0.63 $\pm$ 0.036	0.70 $\pm$ 0.018
	DBSCAN	0.73 $\pm$ 0.017	0.75 $\pm$ 0.016	0.65 $\pm$ 0.027	0.71 $\pm$ 0.018
Scenario 7	HDCluster	0.55 $\pm$ 0.041	0.61 $\pm$ 0.027	0.43 $\pm$ 0.074	0.64 $\pm$ 0.032
	DBSCAN	0.53 $\pm$ 0.043	0.59 $\pm$ 0.036	0.45 $\pm$ 0.062	0.64 $\pm$ 0.032
Scenario 8	HDCluster	0.82 $\pm$ 0.034	0.82 $\pm$ 0.03	0.64 $\pm$ 0.067	0.76 $\pm$ 0.044
	DBSCAN	0.80 $\pm$ 0.037	0.81 $\pm$ 0.031	0.64 $\pm$ 0.061	0.74 $\pm$ 0.062
Scenario 9	HDCluster	0.57 $\pm$ 0.026	0.65 $\pm$ 0.014	0.54 $\pm$ 0.037	0.63 $\pm$ 0.02
	DBSCAN	0.58 $\pm$ 0.021	0.63 $\pm$ 0.02	0.55 $\pm$ 0.038	0.61 $\pm$ 0.028
Scenario 10	HDCluster	0.85 $\pm$ 0.026	0.84 $\pm$ 0.025	0.71 $\pm$ 0.065	0.81 $\pm$ 0.034
	DBSCAN	0.84 $\pm$ 0.028	0.84 $\pm$ 0.024	0.72 $\pm$ 0.066	0.82 $\pm$ 0.034

Metric	HDCluster (Mean $\pm$ SEM)	DBSCAN (Mean $\pm$ SEM)	Absolute diff. (HDCluster – DBSCAN)	Relative Improvement (%) $= \frac{\text{HDCluster Mean} - \text{DBSCAN Mean}}{\text{DBSCAN Mean}} \times 100$
GT ARI	0.72 $\pm$ 0.05	0.71 $\pm$ 0.04	0.01	1.41 %
GT AMI	0.74 $\pm$ 0.03	0.72 $\pm$ 0.03	0.02	2.78 %
MB ARI	0.57 $\pm$ 0.05	0.51 $\pm$ 0.06	0.06	11.76 %
MB AMI	0.68 $\pm$ 0.04	0.62 $\pm$ 0.05	0.06	9.68 %

Supp. Table 3: Methods' Parameters. DBSCAN parameters adopted from Nieves et al.²⁶. The HDCluster parameters are tuned based on the AMI measure.

	Ground Truth				Multiple Blinking			
	HDCluster		DBSCAN		HDCluster		DBSCAN	
	mTh (nm)	β	ϵ (nm)	MinPts	mTh (nm)	β	ϵ (nm)	MinPts
Scenario 2	59	0.8	45	7	67	0.9	35	50
Scenario 3	34	2.1	40	10	48	1.7	45	50
Scenario 4	59	1.9	50	4	59	1.6	60	25
Scenario 5	50	0.8	35	8	53	0.8	35	34
Scenario 6	58	0.6	30	6	63	0.8	40	42
Scenario 7	74	0.7	65	6	80	0.8	65	32
Scenario 8	62	0.6	50	4	70	0.9	50	27
Scenario 9	77	0.6	45	12	110	0.7	45	50
Scenario 10	59	0.9	45	7	68	1	50	33