Session 4 — Super-Resolution Cluster Analysis

Using LLMs for cluster analysis of reconstructed SR images · keep this; it builds on Sessions 1–3

The one idea: SMLM data is a point cloud, not an image — the primary data is a coordinate table (x, y, photons, frame, uncertainty), not a pixel grid. The moment you prompt as though it is an image and threshold intensity, you are measuring the wrong thing. *Everything else in this session follows from this pivot.*

How this session works — describe your reconstructed SR image → build the cluster-analysis script with Claude → validate it against a randomization control. Same three-move arc as every session; only the modality changes.

The conceptual pivot: from pixels to point clouds

Conventional microscopy / earlier sessions	Localization microscopy (SMLM / STORM)
The data IS an image — a grid of pixels with intensity / bit depth	The data is NOT an image — a table: x, y, (z), photons, frame, uncertainty
Threshold intensity to find signal	No intensity to threshold — work with positions and counts
Resolution set by pixel size + diffraction limit	Resolution set by localization precision + sampling density
Ask: how bright is this pixel?	*Ask: where are the molecules, and how certain am I about each position?*
Prompt: describe channels, bit depth, threshold	Prompt: describe the reconstructed image — nm/pixel, cell boundary, cluster scale, channel(s)

How a localization is made — the pipeline; Claude analysis starts after step 04

#	Step	What happens	Output
01	**Blinking**	Sparse fluorophores switch ON per frame (photoswitching) — PSFs don't overlap	Raw movie of isolated blinks
02	**Fitting**	Each isolated PSF is fit (Gaussian / MLE) to sub-pixel precision — this is the super-resolution	x, y center + uncertainty per blink
03	**Localization**	Center + uncertainty + photon count + frame → one row in a table	One row per detected blink
04	**Table → image**	Thousands of frames → point cloud, then rendered into a reconstructed SR image	Localization table + SR image

Pipeline: *raw movie → ThunderSTORM / Picasso / commercial software → localization table + SR image.* ***Claude's cluster analysis starts at the reconstructed image (clusters already established) — not at the raw movie.***

Describe the reconstructed SR image in four dimensions

#	Dimension	What it covers	Thin → rich
01	**Sample**	Organism, cell type, target protein(s), treatment, timepoint	*"smooth muscle cell" → "adult mouse vascular SMC, Protein X, TAC 16 wk vs. sham"*
02	**The image**	Reconstructed SR image with clusters established — nm/pixel calibration, channels, noisy background outside cell?	*"fluorescent image" → "5 nm/pixel STORM reconstruction, single channel, background noisy outside cell"*
03	**Structure**	Expected cluster scale and organization — size, spacing, one protein or two interacting?	*"bright spots" → "discrete clusters ~60 nm diam., ~200 nm spacing, clusters already segmented"*
04	**Question**	Cluster size? Density? NND? Intermolecular proximity? Colocalization? — and compared against what	*"analyze clusters" → "cluster size + density + NND inside cell footprint; proximity ≤ 30 nm vs. CSR null"*

"The image" replaces "the raw data" from Sessions 1–3. *Always include: nm/pixel calibration, channel identity, whether the background is noisy. These three determine whether any measurement is trustworthy.*

Two modes of working

EXECUTION — you know what to measure
DESCRIBE → SPECIFY → VERIFY

Rich four-dimension description, then a numbered spec listing exactly what the script should do, in order — cell mask first, then clusters, then each metric. End with: *"test on one image first and show me the overlay."*

DISCOVERY — you want to know what's possible
SHOW → WONDER → PROBE

Rich description, don't direct the analysis — upload and ask: *"What spatial relationships in these clusters could I quantify that I might not have considered?"* Then chase what surprises you.

TODAY'S PORTFOLIO PIECE · Keep the four-dimension description and spec prompt you ran today. They are the literal input to your re-runnable protocol: image + spec + script + parameters. Portfolio accumulates across all four sessions — it doesn't reset.

WHAT THE NEXT STEPS ASSUME YOU HAVE · Describe in four dimensions · nm/pixel calibration · cell boundary · threshold as a decision · validate against a control · overlay is the interface · lossless files.

Thin vs. rich — spell out the spec before any code

THIN — "Analyze this image."	RICH — a numbered spec that names every decision
Arbitrary threshold, no cell mask, no calibration, no defined metric. Whatever the model guesses — and you can't reproduce it. A thin prompt doesn't fail visibly: it returns a confident, arbitrary result that *looks* fine.	1. Detect the cell boundary — analyze only inside it (background is noisy) 2. Detect clusters by intensity threshold — they are already established in the image 3. Report cluster size, density (clusters per µm²), and coverage fraction inside the cell 4. Intramolecular NND — nearest-neighbor distance between clusters, one channel 5. **Two channels:** intermolecular distance, proximity ≤ 30 nm, colocalization / overlap 6. Randomize cluster positions inside same cell boundary (CSR null model); rerun same metrics 7. Export per-cluster measurements to CSV *Each line is a decision you can see, change, and report — and a number you can defend.*

The decisions you own — set deliberately, report them, sweep the sensitive ones

Decision	What to set and why it matters
Cell mask	Which region counts — critical when background is non-zero. Detect the cell edge first and confirm the overlay before measuring anything inside it.
Cluster threshold	How the intensity cut defines each cluster's extent. Set from background statistics — mean + k×SD (k ≈ 3) — never auto-selected per image. Same principle as every earlier session: a threshold is a decision, not a default.
Proximity threshold	What counts as "close" across two channels — e.g. 30 nm. This number drives your biological claim. Report it explicitly; sweep at least two values to show the result is not threshold-sensitive.
nm/pixel calibration	Sets every distance you report. Read from the file metadata — don't retype. A wrong calibration scales all distances proportionally and corrupts every downstream metric silently.
Cell-edge dilation	How far the boundary extends beyond outermost localizations. Too tight → clips peripheral clusters. Too loose → includes background. Always show the overlay and confirm before measuring.

The metrics menu — what commercial software gives you vs. what Claude scripts add

Commercial software already gives you	What you extract with a Claude script from the same reconstructed image
• Cluster size (area / diameter) • Cluster density (clusters per µm²) • Cluster count	• **Intramolecular NND** — nearest-neighbor distance between clusters, one channel; mean, median, full distribution • **Intermolecular distance** — NND between channels in two-color images • **Proximity fraction** — proportion of clusters within a defined distance (e.g. ≤ 30 nm) of a partner channel • **Colocalization / overlap** — spatial coincidence of two-channel clusters • **Cluster morphology** — circularity, eccentricity, solidity per cluster • **Spatial statistics** — Ripley's K/L-function and pair-correlation g(r) to test whether clustering is non-random at each spatial scale

Validation — is it real? The randomization control (S · R · C)

Step	Action
S — Shuffle	Randomize cluster positions inside the same cell boundary (CSR null model). Cluster sizes and appearances are preserved — only positions are shuffled.
R — Recompute	Run the exact same proximity / colocalization / NND script on the shuffled data — identical parameters and pipeline.
C — Compare	Observed ≫ randomized → the pattern is real. Observed ≈ randomized → consistent with chance. Use Ripley's K/L-function and pair-correlation g(r): if the original lies above the 95% Monte Carlo envelope and shuffled data tracks within it, the result stands.

The overlay is the interface — the map is the proof

You don't need to read the code. You read the map. If the overlay marks clusters where a biologist would mark them, the analysis is right — however elegant the code looks. If the mask cuts off the cell edge, or the NND colors land on background, it's wrong — however clean the script seems. **Code literacy is optional. Biological judgment is not.**

Before any result goes in a figure

- ☐ Described the image in four dimensions before writing any code
- ☐ Cell boundary detected and overlay confirmed as biologically sensible before measuring inside it
- ☐ Cluster threshold set deliberately from background statistics — not auto-selected
- ☐ nm/pixel calibration read from file metadata — not retyped
- ☐ Tested on one image first; overlay of detected clusters matches manual expectation
- ☐ 5–10 clusters cross-checked by hand or against an established tool

- ☐ Randomization control run — observed metric compared to shuffled null (Ripley's K/L, g(r), or equivalent)
- ☐ Proximity threshold reported; result swept across ≥ 2 threshold values
- ☐ Batch parameters fixed — no per-image auto-adjustment; prompt + model version archived with raw data; known limitations written down

TODAY'S PORTFOLIO PIECE · Keep the four-dimension description, spec prompt, and script from today. Image + spec + parameterized script + randomization result = a fully re-runnable protocol. That is this session's anchor principle: **Reproducibility.** The portfolio accumulates across all four sessions.

WHAT THIS SESSION ASSUMED YOU HAD · Describe in four dimensions · read calibration & channels · treat a threshold as a decision · validate against known ground truth · the overlay is the interface · lossless files, read metadata, never JPEG · batch runs use fixed parameters.