MINIAN · PIPELINE_NO_DECONV.IPYNB · PARAMETER TUNING

Running & tuning Minian, conceptually

How the deconvolution-free workshop pipeline actually works and how to tune every knob — preprocessing, the seed cascade, the spatial update (what `dl_wnd` really is), and why the temporal step still earns its place even though deconvolution is skipped.

The big picture: CNMF as an alternating fit

Minian models the movie as `Y ≈ A·C + b·f` — spatial footprints `A` (where cells are) times temporal traces `C` (when they fire), plus a background. Neither is known, so CNMF **alternates**: fix one, solve the other, repeat. Everything before that loop exists to give it a clean movie and a good starting guess.

PREP	MC	SEED	INIT	CNMF ×2	SAVE
Preprocess glow · denoise · background	**Motion** rigid alignment	**Seed cascade** over-complete → refine	**Initialize** grow A, project C	**spatial → temporal → merge** alternating refinement	**C / C_lp / A** → calab → CaMAP

What "deconvolution-free" changes

Standard Minian's temporal update solves a convex **AR model** that simultaneously denoises the trace `C` and produces deconvolved spikes `S` — the AR model *is* the deconvolution, so you can't keep `C` and turn spikes off. This workshop variant swaps that solve for a **low-pass filter** of the extracted fluorescence `YrA` (`lowpass_temporal`): it denoises enough for the loop to run, produces no `S`, and leaves deconvolution to the downstream tool (calab). The final saved `C` is the raw `YrA`; `C_lp` is the low-pass copy for display.

The one idea behind half the parameters: neuron size in pixels

Several knobs are really the *same physical scale* — your cell's size measured in pixels — wearing different hats. Calibrate them together: open the max-projection, measure a typical soma's diameter in pixels, and set these from it. Get this right and most of the pipeline falls into place.

RADIUS VS DIAMETER — MIND THE CONVENTION

Some of these are benchmarked against the cell **radius**, others against the **diameter**. They are not interchangeable.

PARAMETER	STAGE	MEANS	SET TO ≈	WORKSHOP
`denoise ksize`	preprocess	median kernel side (odd)	½ largest cell **diameter**	3
`tophat wnd`	preprocess	disk **radius**	largest cell **radius**	20
`max_wnd`	seeds_init	local-max disk **radius**	largest cell **radius**	15
`thres_dist`	seeds_merge	merge distance	~one cell **radius**	10
`wnd`	initialize	footprint correlation **radius**	largest cell **radius**	10
`dl_wnd`	spatial update	dilation disk **radius**	~one cell **radius**	10

All radii are in pixels. The workshop's larger values (15/10/10 vs Minian's 10/5/10 library defaults) suit somewhat larger / lower-resolution cells.

Preprocessing — clean the 1-photon movie

One-photon miniscopes excite a whole volume, so each frame carries heavy out-of-focus glow and background that 2-photon doesn't. If you skip this, CNMF fits the background instead of cells. Each step preserves the linear 0–255 scale so intensities stay interpretable.

`glow removal` `no params` — min-projection

Subtracts the **per-pixel minimum across all frames** — each pixel's floor over the recording is treated as static vignetting/glow and removed. Runs first; complements tophat (which removes the slowly-varying background that remains per frame).

`denoise · ksize` `7 → 3` `SIZE · PX` — median kernel, odd

Frame-by-frame **median filter** that kills salt-and-pepper sensor noise. Side length in pixels, must be odd. Guidance: ≈ half the largest cell *diameter*. The notebook sweeps `[1,3,5,7,9]` and picks **3** (this data is fairly clean / detailed).

↑ larger → blurs cells together, erases small/dim ones

↓ smaller → residual speckle survives

`remove_background · wnd` `15 → 20` **SIZE · PX RADIUS** tophat disk

Morphological **tophat**: a disk of *radius* `wnd` erodes then dilates each frame, removing any bright feature smaller than the disk — a size filter that strips slow background while keeping cell bodies. Set to ≈ the largest cell *radius*. Sweeps `[5,10,15,20]` → **20** (larger cells here).

↑ too large → background/halo not removed, FOV stays bright

↓ too small (< cell) → disk fits inside cells, hollows them out (donut artifacts)

HOW TO CHOOSE

Both use the interactive `visualize_preprocess(...)` sweep — it shows before/after frames plus contour plots. Pick `ksize` where speckle vanishes but cell edges stay sharp; `wnd` where background flattens but cells aren't hollowed.

Motion correction — essentially parameter-free

Rigid translation by FFT phase correlation, estimated recursively over frame chunks. The workshop runs it with defaults; the notebook itself calls this step "usually parameter-free."

`estimate_motion · dim` `"frame"`

Axis to estimate motion along — always `"frame"`. Internals you rarely touch: `npart=3` (chunk recursion), `alt_error=5` px (rejects implausible shifts when activity is sparse), `mesh_size=None` (set it to enable experimental *non-rigid* correction).

`subset_mc (ROI)` `None` optional box

Restrict motion estimation to a drawn sub-region. Useful when a bright artifact, vessel, or drifting non-neural structure would dominate the phase correlation — pick a stable, cell-rich patch.

HOW TO TELL IT WORKED

Compare the max-projection **before vs after**: good correction makes cell borders crisp; bad correction leaves cells smeared across pixels. Big black borders after `apply_transform` = large drift.

Seeding — cast a wide net, then refine

CNMF can't find cells directly; it needs a starting guess. Minian scatters a deliberately **over-complete** set of candidate points, then filters them by temporal signal quality. The bias is intentional: a spurious seed is cheap to drop later, but a *missed* cell can never be recovered downstream — so over-seed and under-merge when unsure. (This is where most of your tuning attention pays off.)

seeds_init — over-complete generation

`max_wnd` `15` SIZE · PX RADIUS

Radius of the disk used to find local maxima — effectively the **radius of the largest cell you want to detect**.

↑ too large → two nearby cells collapse to one peak (missed cells), slower

↓ too small → one big cell yields several seeds (cleaned up at merge)

`diff_thres` `3 → 8` raw intensity

A local max is kept only if it's brighter than its neighbors by at least this much — a prominence filter against flat/noise peaks. In raw intensity units (data-dependent), so re-check it if you change preprocessing. Workshop raises it to **8** (stricter, fewer junk seeds).

↑ higher → dim cells never seed (missed cells)

↓ lower → background fluctuations pass (spurious seeds)

`wnd_size / stp_size` `1000 / 500` frames

Chunk length and step for the rolling max-projection that seeds come from. Overlap (step < window) gives sparsely-active cells more chances to be caught. Mostly a robustness/speed knob, not precision.

pnr_refine — peak-to-noise ratio

`noise_freq` `0.06` fraction of frame rate

High-pass cutoff that defines a seed's "noise" band. PNR = signal swing ÷ noise swing; real cells have big slow transients and little high-frequency noise → high PNR. See the noise_freq section.

thres `1` ratio · or "auto"

Keep seeds with PNR above this. `"auto"` fits a 2-component Gaussian mixture to the PNR distribution and keeps the higher-mean group. `1` is a permissive floor (signal swing = noise swing).

↑ higher → drops dim-but-real cells

↓ lower → noise seeds survive

ks_refine — normality test

sig `0.05` p-value

Kolmogorov–Smirnov test: a real cell's trace is bimodal (baseline + active) → strongly non-normal. A seed is kept when `p < sig` (normality rejected). Complements PNR by catching noise regardless of amplitude.

↑ higher (0.05) → more seeds pass (permissive)

↓ lower (0.01) → stricter, may drop quiet cells

seeds_merge & initialize — de-duplicate, then grow footprints

thres_dist `10` SIZE · PX

Two seeds merge if within this many pixels *and* temporally correlated. Merging is **transitive**, so keep it small in dense tissue to avoid chaining distinct cells into one.

thres_corr `0.8` correlation

Min trace correlation to merge (seeds_merge) and the footprint-boundary cutoff (initialize). Higher = more conservative: keeps units separate, tighter footprints.

↑ higher → separate units, tight footprints (may fragment)

↓ lower → over-merge / bloated footprints with cross-talk

initialize · wnd `10` SIZE · PX RADIUS

Radius within which each seed's footprint can grow (pixels correlated to the seed's trace above `thres_corr` become the footprint). An upper bound on footprint extent — set to ~cell radius.

Spatial update — where `dl_wnd` lives

Given the current traces `C`, re-solve each cell's footprint `A`: for every candidate pixel, an L1-penalized non-negative regression decides which cells own it and how strongly. Three knobs.

YOUR QUESTION: WHAT IS DL_WND, EXACTLY?

It's the dilation window — a disk *radius* in pixels, not a threshold. Minian takes each unit's current footprint, `cv2.dilate`s it with a disk of radius `dl_wnd`, binarizes that, and uses it as a mask: only pixels inside the dilated mask may be assigned to that unit this round. So `dl_wnd` is literally *how far a footprint is allowed to grow outward, in pixels* — set it around one cell radius. It's Minian's analogue of CaImAn's expected-neuron-size `gSig`.

`dl_wnd` `10` SIZE · PX RADIUS

Dilation radius defining each unit's candidate-pixel neighborhood.

↑ too large → footprints bleed into neighbors, slower compute

↓ too small → footprint can't reach the cell's real edge (truncated, may be culled)

`sparse_penal` `0.01 → 0.005`

L1 weight

L1 sparsity penalty on the footprint (scaled per-pixel by noise `sn`). Controls footprint size/density. The notebook sweeps `[0, 0.002, 0.005, 0.01, 0.05]`; workshop picks **0.005** (less sparse, fewer cells dropped).

↑ higher → sparser/smaller footprints, units can be dropped

↓ lower → denser/larger footprints, overlap, captured noise

`size_thres` `(25,None) → (16,None)`

(min, max) pixels

Footprint **pixel-area** filter: drop units smaller than min (noise fragments) or larger than max (merged blobs). Workshop loosens the floor to **16** px to keep small/dim cells — pairs with the lower `sparse_penal`.

HOW TO PICK SPARSE_PENAL

Run the `visualize_spatial_update` exploration cell (it renders footprints at each candidate penalty). Choose the value where footprints are compact around their centroids — not bleeding/overlapping (penalty too low) and not dropping real cells (too high). Judge by *your* plot, not a fixed number.

Temporal update — do we still need it without deconvolution?

Your core question. Standard Minian's temporal step jointly denoises `C` and deconvolves spikes `S` via an AR model; here it's replaced by a low-pass filter producing only a denoised `C`. So is the step still doing anything useful?

VERDICT — YES, KEEP IT (BUT REFRAME WHY)

Its essential job without deconvolution is to hand a **denoised regressor `C` to the next spatial update** — `update_spatial` regresses each pixel onto `C`, so a cleaner `C` gives a cleaner footprint demixing. And **merging never depended on deconvolution anyway**: `unit_merge` low-pass-smooths traces internally before correlating, so a low-pass `C` is exactly what it wants. The low-pass substitute is therefore legitimate. There's precedent: CaImAn's `p=0` mode and OASIS-as-a-post-step both treat deconvolution as a separable final stage.

`noise_freq` `0.06` — fraction of frame rate

The only live temporal knob in this variant — and now your master denoising dial. The low-pass cutoff applied to `YrA`. Because it shapes the trace the downstream deconvolver will consume, set it slightly conservative (preserve transient shape). See below.

`p, sparse_penal, add_lag, jac_thres` `1, 1, 20, 0.2/0.4` INERT HERE

The AR-deconvolution parameters: AR order `p` (1 = pure exponential decay, 2 = adds rise time), spike L1 penalty, autocovariance lags, and the Jaccard overlap threshold for grouping cells in the joint solve. **All inert in the low-pass variant** — kept in the dict only for parity with stock Minian. They'd matter if you restored the AR solve.

Why run spatial + temporal twice — and the grow → merge → tighten idea

CNMF is non-convex, solved by **alternating block-coordinate descent**: each factor is only as good as the other it was conditioned on. So Minian sweeps the cycle twice, with a merge in between:

1	2	3	4
spatial #1	**temporal #1 + merge**	**spatial #2**	**temporal #2 + merge**
A given initial C	cleaner C, fewer dup units	A given the improved C	final C

KEY MECHANISM — THE MERGE IS GATED ON SHARED PIXELS

This is what makes the next idea work. `unit_merge` only merges cells **that overlap in space**: it first computes the shared-pixel count for every footprint pair (`A_inter = (A>0)·(A>0)ᵀ`), and *only among overlapping pairs* does it then check temporal correlation against `thres_corr` . Two units with no common pixels are never merged, however correlated their traces. (Verified in Minian's `cnmf.py` : docstring "merges all cells that have *common pixels* based on correlation of their temporal components.")

A LEGITIMATE STRATEGY: GROW → MERGE → TIGHTEN

Because the merge fires *only on overlapping footprints*, deliberately letting cells **grow in pass 1** (lower `sparse_penal` , larger `dl_wnd` , low `size_thres` floor) makes more genuine duplicates *share pixels* and become merge candidates — so the in-between merge can actually collapse them. Then **tighten pass 2** (raise `sparse_penal` / `size_thres`) to shrink the merged footprints back to compact, accurate extents. This is a sound, mechanism-aware schedule — implement it by editing `param_first_spatial` and `param_second_spatial` independently.

…BUT IT'S NOT THE DEFAULT, AND DON'T OVER-GROW

Stock and workshop Minian use *identical* params in both passes by default (verified: both end at `sparse_penal=0.005, size_thres=(16,None)`), so out of the box the second pass benefits from refined *inputs*, not changed params — the grow→tighten schedule is an opt-in refinement, not built in. The risk: grow pass 1 too far and footprints bleed into real **neighbors**, creating spurious overlaps → **over-merging distinct cells** (worse because merging is transitive). `thres_corr` guards this (overlapping-but-uncorrelated cells won't merge), but co-active neighbors can still fuse — so grow, but not so low that footprints swallow neighbors.

FOR A TRACES-ONLY DELIVERABLE, THE 2ND CYCLE IS MARGINAL

Most footprint/trace cleanup happens in cycle 1. When you only want denoised fluorescence (deconvolution deferred), the second cycle gives diminishing returns — keep it by default, but dropping to one cycle (+ a final `compute_trace`) is defensible if runtime is tight. Don't skip the temporal step *between* two spatial updates, though — pass 2 needs a denoised `C`.

noise_freq — the frequency that recurs everywhere

It appears in `pnr_refine`, `seeds_merge`, `initialize`, and the temporal low-pass. Worth understanding once.

It's a fraction of your frame rate

`noise_freq` is a **normalized cutoff = fraction of the sampling (frame-rate) frequency**, not Hz. At `0.06` with a ~20 Hz miniscope that's roughly a ~1.2 Hz cutoff — comfortably above real calcium-transient bandwidth but below sensor noise. Calcium signal is slow and low-frequency; noise is broadband and high-frequency, so the cutoff separates them.

Tie it to your indicator + frame rate: slow indicators (GCaMP6s) → lower cutoff; fast ones (GCaMP6f/8f) or higher frame rates → it can go higher. After temporal downsampling, it's a fraction of the *downsampled* rate.

↑ too high → noise leaks in: fuzzy traces, cells wrongly rejected by PNR

↓ too low → over-smoothed: transients blunted/merged, real cells lost

Tune it with the interactive frequency-exploration cell in the seed-refinement section: pick the cutoff that flattens the noise while preserving transient *shape*. Since deconvolution is deferred here, lean slightly conservative (preserve transients) — this filter shapes the trace your downstream deconvolver sees.

Tuning cheat-sheet

SYMPTOM	LIKELY CAUSE	MOVE
Many junk / noise units	seeds too permissive	↑ `diff_thres`, ↑ PNR `thres` (or `"auto"`), ↓ `ks sig`
Real / dim cells missing	seeds too strict or scale wrong	↓ `diff_thres`, ↓ PNR `thres`, check `noise_freq`, ↓ `max_wnd` if cells crowd
One cell → several units	over-segmentation	↑ `max_wnd`; ↑ `thres_dist` (cautiously); ↓ merge `thres_corr`
Two cells → one unit	over-merging	↓ `thres_dist`; ↑ merge `thres_corr`
Footprints truncated / culled	spatial too tight	↑ `dl_wnd`; ↓ `sparse_penal`; ↓ `size_thres` floor
Footprints bloated / cross-talk	spatial too loose	↓ `dl_wnd`; ↑ `sparse_penal`
Cells hollowed (donut artifacts)	tophat `wnd` < cell	↑ `wnd` toward cell radius
Traces over-smoothed / noisy	`noise_freq` too low / too high	tune to preserve transient shape
Cells smeared after MC	bad motion estimate	restrict to a stable ROI (`subset_mc`)

THE FAST WAY TO RUN IT THE FIRST TIME

① Measure a soma's radius in pixels → set `tophat wnd`, `max_wnd`, init `wnd`, `dl_wnd` from it. ② Use the three interactive sweeps (`denoise` / `background`, `noise_freq`, `sparse_penal`) and pick by eye. ③ Over-seed and under-merge — you can cull later, but missed cells are gone for good. ④ Watch the dask dashboard and set `MINIAN_MEM_LIMIT` for your RAM (6GB on a 32GB machine).

Grounded in the workshop's `tutorials/minian/pipeline_no_deconv.ipynb` and verified against the installed Minian 2.0.2 source + Dong et al. 2022 (eLife 70661), Pnevmatikakis 2016, and CaImAn (Giovannucci 2019). Values shown as `default → workshop` where the notebook overrides them. Companion to `WORKSHOP_GUIDE.html`, `EZTRACK_GUIDE.html`, and `PLACECELL_GUIDE.html`.