

# Supplementary Materials for ldmppr: Location Dependent Marked Point Processes in R

by Lane Drew and Andee Kaplan

## 1 Additional details on initialization defaults for `estimate_process_parameters()`

We initialize self-correcting model parameters using data-adaptive heuristics designed to place the optimizer in a stable region of the parameter space. The baseline log-intensity is anchored by the empirical event rate via  $\alpha_{\text{base}} = \log(n/\Delta t)$ . The self-correction strength  $\gamma_1$  is initialized on a conservative  $\log(n)/n$ -type scale to reduce unstable behavior early in optimization. To capture monotone temporal trends,  $\beta_1$  is initialized from a Poisson regression of binned event counts on time (with an offset for bin width), with a fallback that induces a modest multiplicative change across the observation window.

Spatial inhibition range  $\alpha_2$  is initialized using the median nearest-neighbor distance (or a conservative geometric fallback). Spatio-temporal interaction scales are bounded by the observation window: the spatial scale parameter is capped by the window diagonal, and the temporal interaction scale is capped by the observed time span (equal to 1 under the default time mapping). These choices provide stable starting values and natural finite bounds for optimization.

## 2 Stage-wise runtime benchmarks

To quantify fitting cost, we benchmarked runtime for `estimate_process_parameters()` and `train_mark_model()` on three contiguous datasets: (i) `medium_example_data`, (ii) a contiguous window with approximately 200 points, and (iii) a contiguous window with approximately 400 points. The larger datasets were selected as contiguous windows from the full domain (rather than random point subsamples) to preserve local dependence structure.

Stage-wise timing artifacts can be regenerated by running:

```
LDMPPR_STAGE_PROFILE=full
LDMPPR_STAGE_REPS=10
LDMPPR_STAGE_CORES=1,7
LDMPPR_STAGE_DATASETS=medium,win200,win400
Rscript "scripts/supplement_stage_timing_study.R".
```

If `LDMPPR_RASTER_DIR` is not set, required external rasters are downloaded automatically from ESS-DIVE into `data/ess_dive_rasters`.

**Table 1:** Stage-wise runtime summary (seconds): process estimation and mark-model training.

| Dataset | Points | Window |       | Runs | Est.  |      | Train |      | Total |      |
|---------|--------|--------|-------|------|-------|------|-------|------|-------|------|
|         |        | Side   | Cores |      | Mean  | SD   | Mean  | SD   | Mean  | SD   |
| medium  | 106    | 50     | 1     | 10   | 2.91  | 0.33 | 23.34 | 0.76 | 26.25 | 0.74 |
| medium  | 106    | 50     | 7     | 10   | 2.23  | 0.27 | 23.76 | 0.15 | 25.99 | 0.28 |
| win200  | 207    | 75     | 1     | 10   | 7.72  | 1.09 | 36.05 | 0.16 | 43.76 | 1.14 |
| win200  | 207    | 75     | 7     | 10   | 4.77  | 0.79 | 35.45 | 0.19 | 40.22 | 0.72 |
| win400  | 413    | 100    | 1     | 10   | 19.95 | 1.87 | 60.20 | 0.52 | 80.15 | 1.82 |
| win400  | 413    | 100    | 7     | 10   | 7.74  | 2.02 | 60.08 | 0.29 | 67.82 | 2.06 |

Using one core, process-estimation time increases from 2.91s ( $n=106$ ) to 19.95s ( $n=413$ ), while mark-model training increases from 23.34s to 60.20s over the same range.

### 3 Simulation-check runtime benchmarks

We benchmarked runtime for `check_model_fit()` using `n_sim` values of 200, 500, 1000, and 2500, with fixed seeds and matched settings otherwise. This isolates simulation-based goodness-of-fit cost and provides practical guidance for exploratory versus final checks.

Simulation-check timing artifacts can be regenerated by running:

```
LDMPPR_TIMING_MODE=full
LDMPPR_TIMING_N_SIM=200,500,1000,2500
LDMPPR_TIMING_REPS=10
LDMPPR_TIMING_CORES=1,2
Rscript "scripts/supplement_sim_timing_study.R".
```

**Table 2:** Runtime summary (seconds) for simulation-based model checking.

| Simulations | Cores | Runs | Elapsed Mean | Elapsed SD | Combined <i>p</i> -value Mean |
|-------------|-------|------|--------------|------------|-------------------------------|
| 200         | 1     | 10   | 4.33         | 0.10       | 0.133                         |
| 200         | 2     | 10   | 4.08         | 0.17       | 0.136                         |
| 500         | 1     | 10   | 10.63        | 0.13       | 0.136                         |
| 500         | 2     | 10   | 9.43         | 0.19       | 0.146                         |
| 1000        | 1     | 10   | 21.79        | 0.16       | 0.215                         |
| 1000        | 2     | 10   | 19.37        | 0.72       | 0.151                         |
| 2500        | 1     | 10   | 52.74        | 0.35       | 0.227                         |
| 2500        | 2     | 10   | 49.21        | 0.44       | 0.217                         |

At one core, mean check time grows from 4.33s at `n_sim=200` to 52.74s at `n_sim=2500`.

For the largest run (`n_sim=2500`), moving from one core to two cores reduces mean wall-clock time by 6.7%.

For iterative model development, moderate simulation counts can reduce turnaround time. For final reporting, larger simulation counts provide more stable Monte Carlo *p*-value estimates.

For a direct, reproducible benchmark of the improved paper workflow (estimation + mark-model training + model check, matching the script in the main paper), run:

```
LDMPPR_BENCH_REPS=10
Rscript "scripts/benchmark_improved_workflow_timing.R".
```

### 4 Description and interpretation of the spatstat comparison framework

The comparison script `scripts/spatstat_comp.R` implements a two-stage baseline intended to approximate the improved manuscript workflow as closely as possible while utilizing the `spatstat` modeling framework:

1. A spatial point process is fit with `spatstat::ppm`, using raster-derived covariates and a Strauss interaction term for regularity/inhibition.
2. A separate XGBoost mark model is fit from location/raster features with optional competition-index features. To align with the improved paper workflow, tuning uses MAE with 10-fold cross-validation and a tuning grid size of 150.
3. Simulated spatial patterns from the fitted `ppm` model are marked using the fitted XGBoost model then evaluated with the same LGFJEV and combined GET-rank envelope machinery used in the package-facing checks.

For the direct comparison below we use `n_sim = 2500` for both methods, matching the improved-paper model-check setting. For the `ldmppr` workflow, we match the paper's

specifications exactly: estimation and model-check steps run without parallelization, while mark-model training uses `parallel = TRUE`. Comparison artifacts can be regenerated by running:

```
LDMPPR_COMP_N_SIM=2500
LDMPPR_COMP_PARALLEL=true
Rscript "scripts/ldmppr_comp_timing.R" and
LDMPPR_SPATSTAT_N_SIM=2500
LDMPPR_SPATSTAT_CV_FOLDS=10
LDMPPR_SPATSTAT_TUNING_GRID=150
LDMPPR_SPATSTAT_FG_CORRECTION=km
LDMPPR_SPATSTAT_PARALLEL=true
LDMPPR_SPATSTAT_NUM_CORES=7
Rscript "scripts/spatstat_comp.R".
```

If `LDMPPR_RASTER_DIR` is not set, required external rasters are downloaded automatically from ESS-DIVE into `data/ess_dive_rasters`.

**Table 3:** Side-by-side timing and combined  $p$ -value comparison (ldmppr vs spatstat two-stage) for  $n = 2500$  simulations in seconds.

| Method             | Process fit | Mark fit | Check fit | Total (s) | Combined $p$ |
|--------------------|-------------|----------|-----------|-----------|--------------|
| ldmppr             | 7.49        | 136.54   | 53.84     | 197.87    | 0.2819       |
| spatstat two-stage | 0.02        | 128.59   | 443.81    | 443.81    | 0.0016       |

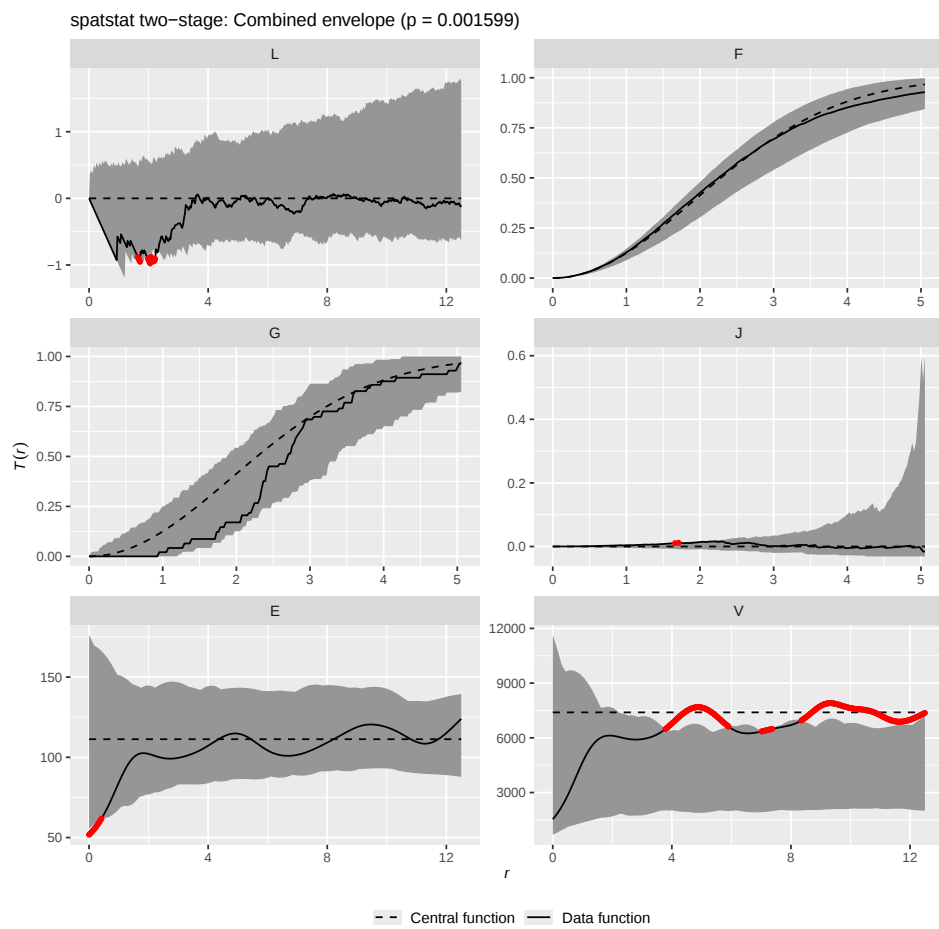
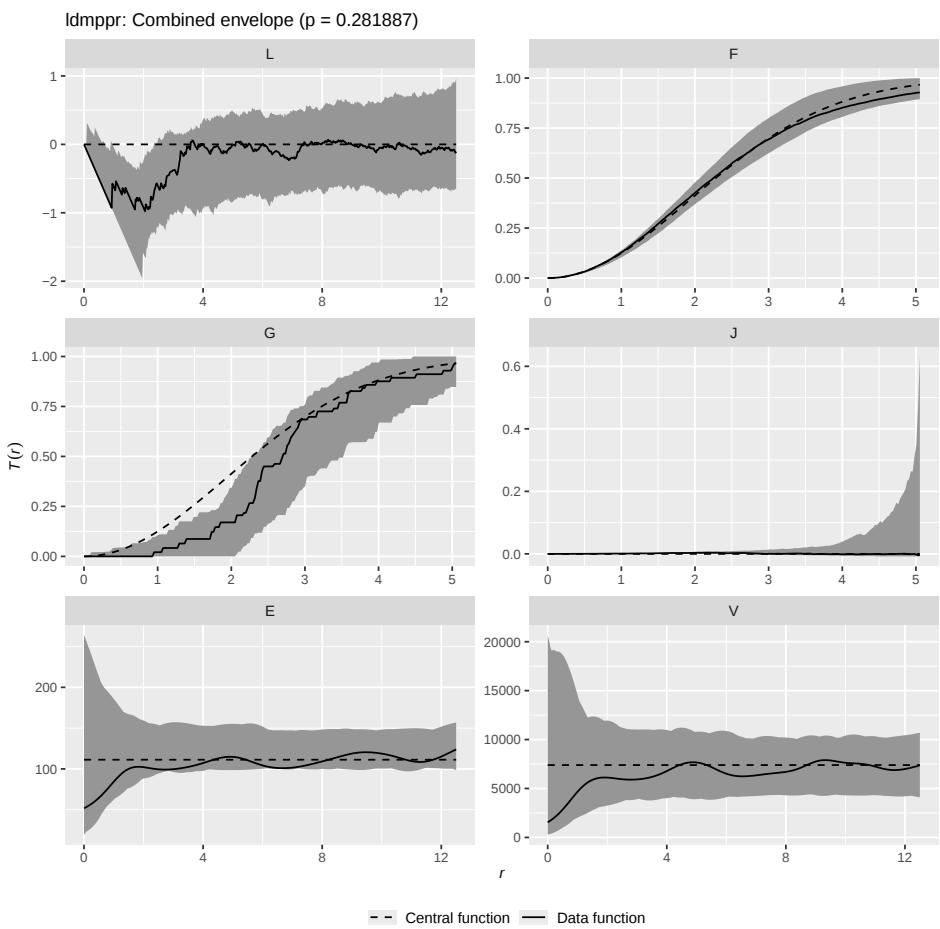
Under matched comparison settings ( $n_{\text{sim}}=2500$ ), the combined GET  $p$ -value is 0.2819 for ldmppr and 0.0016 for the two-stage spatstat baseline.

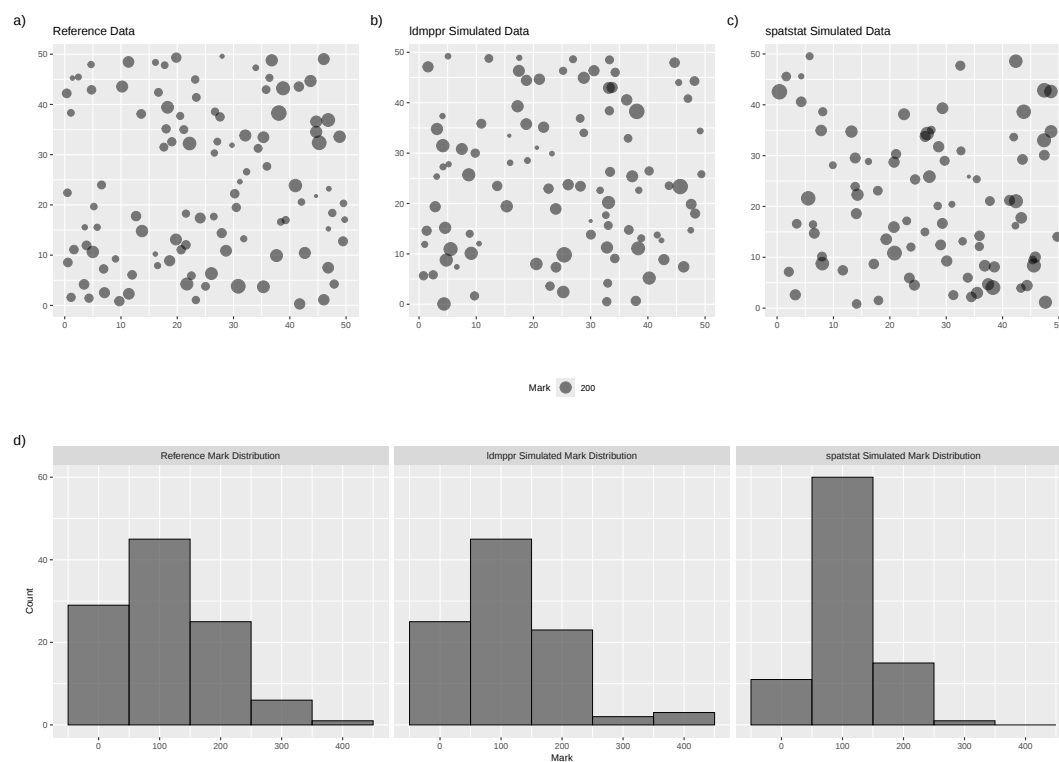
Total runtime is 197.87s for ldmppr versus 443.81s for the two-stage spatstat baseline; in the table above, the spatstat check time reports simulation plus envelope-check computation, while the ldmppr check time reports the corresponding integrated model-check step.

**Table 4:** Side-by-side  $p$ -values by summary function, including the combined test  $p$ -value.

| Statistic | ldmppr $p$ -value | spatstat two-stage $p$ -value |
|-----------|-------------------|-------------------------------|
| L         | 0.4650            | 0.0144                        |
| F         | 0.0728            | 0.5614                        |
| G         | 0.7137            | 0.1236                        |
| J         | 0.1443            | 0.0324                        |
| E         | 0.0688            | 0.0140                        |
| V         | 0.9100            | 0.0024                        |
| Combined  | 0.2819            | 0.0016                        |

At the 0.05 level, ldmppr shows significant departures for: none. The two-stage spatstat baseline shows significant departures for: L, J, E, V.





This baseline provides useful context but is not a fully comparable benchmark for the mechanistic ldmppr workflow. In particular, temporal/self-correcting dynamics are not estimated in a single unified model, and mark generation is modularized as a second stage. We therefore interpret these results as contextual sensitivity evidence rather than a definitive head-to-head comparison.