

# Computing LISA statistics with fastLISA

Lizhong Chen

July 1, 2026

## 1 Introduction

`fastLISA` computes Local Indicators of Spatial Association (LISA) using a plain-C backend with optional OpenMP multi-threading and a modern `xoshiro256++` random number generator for permutation inference. It accepts any `spdep::listw` spatial weights object — including custom and non-contiguity (e.g. distance-decay) weights — and returns compact statistic-specific matrices. Cluster codes follow `rgeoda` conventions, including an *Isolated* category for observations with no neighbours.

The package exposes seven functions:

- `local_moran()` — univariate local Moran's  $I$
- `local_moran_bv()` — bivariate local Moran's  $I$
- `local_moran_eb()` — Empirical-Bayes-rate local Moran's  $I$
- `local_geary()` — univariate local Geary's  $C$
- `local_multigeary()` — multivariate local Geary's  $C$
- `local_g()` — Getis-Ord local  $G$
- `local_gstar()` — Getis-Ord local  $G^*$

All take `nsim` permutations, an optional integer seed `iseed` for reproducibility, a significance cutoff `p.value`, and `n.cores` (default 1L; raise it to use multiple OpenMP threads).

## 2 A worked weights object

We use a small regular grid so the vignette runs quickly. In practice `listw` typically comes from `spdep::poly2nb()` on polygon data or `spdep::dnearneigh()` for distance-based neighbours.

```
> library(spdep)
> library(fastLISA)
> nb <- cell2nb(7, 7)           # 49 cells on a 7 x 7 grid
> lw <- nb2listw(nb, style = "W")
> x <- as.numeric(seq_len(49)) # a simple gradient
> y <- rev(x)
```

### 3 Local Moran's I

```
> res <- local_moran(x, lw, nsim = 199L, iseed = 1L, n.cores = 1L)
> head(res)
```

```
      Ii      Z.Ii Pr(folded) Sim
1:1 2.351020 2.089101      0.005
2:1 2.328163 2.606961      0.005
3:1 2.119184 2.402184      0.015
4:1 1.920000 2.393575      0.020
5:1 1.730612 2.050042      0.015
6:1 1.551020 2.338245      0.015
```

The result is a matrix with the observed statistic (*I<sub>i</sub>*), a permutation-based *z*-score (*Z.I<sub>i</sub>*), and the folded pseudo *p*-value (*Pr(folded)* *Sim*). Cluster classification and scatter-plot quadrants are attached as attributes:

```
> table(attr(res, "cluster"))
```

```
Not significant      High-High      Low-Low      Low-High      High-Low
           27              11              11              0              0
      Undefined      Isolated
           0              0
```

Setting `moments = TRUE` appends the permutation-distribution moments (*E.I<sub>i</sub>*, *Var.I<sub>i</sub>*, *Skew.I<sub>i</sub>*, *Kurt.I<sub>i</sub>*).

### 4 Bivariate and Empirical-Bayes Moran's I

```
> bv <- local_moran_bv(x, y, lw, nsim = 199L, iseed = 1L, n.cores = 1L)
> head(bv)
```

```
      Ibvi      Z.Ibvi Pr(folded) Sim
1:1 -2.351020 -2.089101      0.005
2:1 -2.328163 -2.606961      0.005
3:1 -2.119184 -2.402184      0.015
4:1 -1.920000 -2.393575      0.020
5:1 -1.730612 -2.050042      0.015
6:1 -1.551020 -2.338245      0.015
```

```
> event <- as.numeric(seq_len(49))
```

```
> base <- rep(100, 49)
```

```
> eb <- local_moran_eb(event, base, lw, nsim = 199L, iseed = 1L, n.cores = 1L)
```

```
> head(eb)
```

```
      Ii      Z.Ii Pr(folded) Sim
1:1 2.351020 2.089101      0.005
2:1 2.328163 2.606961      0.005
3:1 2.119184 2.402184      0.015
4:1 1.920000 2.393575      0.020
5:1 1.730612 2.050042      0.015
6:1 1.551020 2.338245      0.015
```

## 5 Local Geary's C (univariate and multivariate)

```
> c_uni <- local_geary(x, lw, nsim = 199L, iseed = 1L, n.cores = 1L)
> head(c_uni)
```

	Ci	Z.Ci	Pr	Sim
1:1	0.12244898	-1.537244	0.005	
2:1	0.08326531	-1.774296	0.005	
3:1	0.08326531	-1.777591	0.015	
4:1	0.08326531	-1.851907	0.025	
5:1	0.08326531	-1.631051	0.025	
6:1	0.08326531	-1.928682	0.005	

```
> c_multi <- local_multigeary(cbind(x, y), lw, nsim = 199L, iseed = 1L, n.cores = 1L)
> head(c_multi)
```

	Ci	Z.Ci	Pr	Sim
1:1	0.12244898	-1.537244	0.005	
2:1	0.08326531	-1.774296	0.005	
3:1	0.08326531	-1.777591	0.015	
4:1	0.08326531	-1.851907	0.025	
5:1	0.08326531	-1.631051	0.025	
6:1	0.08326531	-1.928682	0.005	

## 6 Getis-Ord G and G\*

```
> g <- local_g(x, lw, nsim = 199L, iseed = 1L, n.cores = 1L)
> gs <- local_gstar(x, lw, nsim = 199L, iseed = 1L, n.cores = 1L)
> head(g)
```

	Gi	Z.Gi	Pr(folded)	Sim
1:1	0.004084967	-2.089101		0.005
2:1	0.003543200	-2.606961		0.005
3:1	0.004364430	-2.402184		0.015
4:1	0.005187005	-2.393575		0.020
5:1	0.006010929	-2.050042		0.015
6:1	0.006836205	-2.338245		0.015

```
> head(gs)
```

	G*i	Z.G*i	Pr(folded)	Sim
1:1	0.002993197	-2.089101		0.005
2:1	0.003061224	-2.606961		0.005
3:1	0.003877551	-2.402184		0.015
4:1	0.004693878	-2.393575		0.020
5:1	0.005510204	-2.050042		0.015
6:1	0.006326531	-2.338245		0.015

## 7 Notes on conventions

- **Standardisation.** Variables are standardised with the *sample* standard deviation ( $n - 1$  denominator) before the statistic is computed where the function exposes a `scale` argument. Spatial weight values are row-standardised internally.
- **P-values.** Moran, G, and G\* statistics use the `rgeoda`-style folded form  $p = (\min(\#\{perm \geq obs\}, \#\{perm < obs\}) + 1) / (nsim + 1)$ . Geary statistics use the tail selected by the observed statistic relative to its permutation mean.
- **Isolates.** Observations with no neighbours are assigned the *Isolated* cluster code regardless of the significance cutoff.
- **Reproducibility.** Supplying `iseed` with `n.cores = 1L` makes permutation inference reproducible. Dynamic scheduling means multi-core permutation results may differ between runs.