# Watersheds: spatial watershed aggregation and drainage-network analysis

Jairo A. Torres-Matallana
*Climate Service Center Germany (GERICS), Helmholtz-Zentrum Hereon GmbH, Hamburg, Germany*
jairo.torres@hereon.de

November 4, 2026

## 1 Introduction

This vignette introduces `Watersheds`, an R package for following a drainage network upstream from a location of interest [2, 4]. Given a station, the package finds the hydrological unit containing it, identifies the local inlet and outlet nodes, and then traces its first- and second-order tributaries.

The public interface uses `sp` spatial classes [3]. Geometry predicates and measurements are performed internally with `sf`, so existing workflows based on `sp` can continue to use the package.

### 1.1 Workflow

The examples follow the same sequence used with a new hydrographic network:

1. choose a station and assemble the required basin, hydrological-unit, river and node layers in a `Watershed` object;
2. use `Watershed.Order` to find the focal unit, its connected river stretches, and its inlet and outlet nodes;
3. use `Watershed.Order2` when tributary catchments need to be expanded one level further; and
4. apply `Watershed.IOR1`–`Watershed.IOR4` according to the number of connected inlet/outlet river stretches.

## 2 The data: river Weser basin, Germany

The package includes an ECRINS v1.1 example for the river Weser basin, Germany. Its aggregation and drainage-network logic was developed specifically for the topology and attributes of the 2012 European Environment Agency (EEA) Catchments and Rivers Network System (ECRINS). ECRINS is based on previous work by the Joint Research Centre (JRC) Catchment Characterisation and Modelling (CCM) and the EEA's European Lakes, Dams and Reservoirs Database (Eldred2) and European Rivers and Catchments (ERICA) datasets [1]. Applying the package to another hydrographic network requires checking that its topology and required attributes are analogous.

### 2.1 Subsets

The dataset contains the following subsets:

- `basin`: an object `SpatialPolygonsDataFrame` as is defined in package `sp` that represents the river Weser basin. The `data` slot contains 6 variables as attributes of 1 observation.

- `ctry`: an object `SpatialPolygonsDataFrame` as is defined in package `sp` that represents the administrative boundary of Germany. The `data` slot contains 6 variables as attributes of 1 observation.

- `node:` an object `SpatialPointsDataFrame` as is defined in package `sp` that represents the nodes of the ECRINS river network of the river Weser basin. The `data` slot contains 13 variables as attributes of 3882 observations.

- `rAller` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the basin of the river Aller, a major tributary of the river Weser. The `data` slot contains 74 variables as attributes of 88 observations.

- `rDiemel` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the basin of the river Diemel, a major tributary of the river Weser. The `data` slot contains 74 variables as attributes of 39 observations.

- `rFulda` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the basin of the river Fulda, a major tributary of the river Weser. The `data` slot contains 74 variables as attributes of 82 observations.

- `rHunte` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the basin of the river Hunte, a major tributary of the river Weser. The `data` slot contains 74 variables as attributes of 34 observations.

- `river` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the ECRINS river network of the river Weser basin. The `data` slot contains 52 variables as attributes of 3874 observations.

- `rWerra` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the basin of the river Werra, a major tributary of the river Weser. The `data` slot contains 74 variables as attributes of 120 observations.

- `rWeser` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the basin of the river Weser. The `data` slot contains 74 variables as attributes of 104 observations.

- `rWiumme` an object `SpatialLinesDataFrame` as is defined in package `sp` that represents the basin of the river Wiumme, a major tributary of the river Weser. The `data` slot contains 74 variables as attributes of 18 observations.

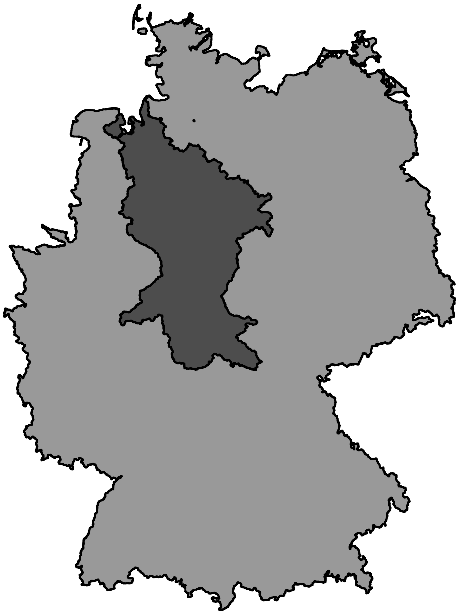
- `station` an object `SpatialPoints` as is defined in package `sp` that represents a point of interest for which the watershed will be aggregated an ordered. Could be a point with the coordinates of a measurement station.

- `subbasin` an object `SpatialPolygonsDataFrame` as is defined in package `sp` that represents the subbasins of the tributaries of the river Weser. The `data` slot contains 4 variables as attributes of 4 observations.

- `zhyd` an object `SpatialPolygonsDataFrame` as is defined in package `sp` that contains the primary hydrological units of the river Weser basin accordingly with ECRINS. The `data` slot contains 50 variables as attributes and 915 observations.

The maps below move from the whole Weser basin to its tributary basins, elementary hydrological units and river network. They establish the spatial context for the analysis that follows.

**River Weser basin, Germany**



# 3 The `Watersheds` objects

A `Watershed` object brings together the station with the layers needed by the algorithms. The station must use the same projected coordinate reference system as the spatial layers. The short setup below creates an object for the bundled example; the remaining maps use the same pattern with different station locations.

```
>         station1 = WatershedsData["station"][[1]]
>         subbasin1 = WatershedsData["subbasin"][[1]]
>         zhyd1 = WatershedsData["zhyd"][[1]]
>         river1 = WatershedsData["river"][[1]]
>         node1 = WatershedsData["node"][[1]]
>         station1 = SpatialPoints(station1,
+                 proj4string=slot(subbasin1,"proj4string"))
>         watershed = new("Watershed",station=station1,subbasin=subbasin1,
```

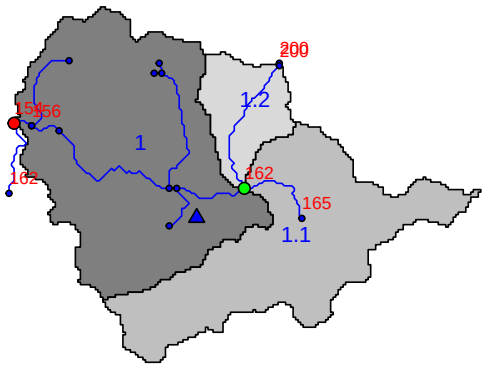
```
+                 zhyd=zhyd1,river=river1,c1=subbasin1,node=node1)
>         class(watershed)
```

```
[1] "Watershed"
attr(,"package")
[1] "Watersheds"
```

# 4 The `Watersheds.Order` method

`Watershed.Order` is the main entry point. It finds the subbasin containing the station, selects the focal `zhyd` unit, identifies its river stretches and candidate inlet/outlet nodes, and returns the first-order tributary units.

In the map, the dark polygon is the focal unit, lighter polygons are first-order tributaries, blue lines are river stretches, the blue triangle is the station, and green and red points mark the inlet and outlet. This is the key diagnostic plot: it lets the analyst check that the topology selected by the method agrees with the intended drainage direction.

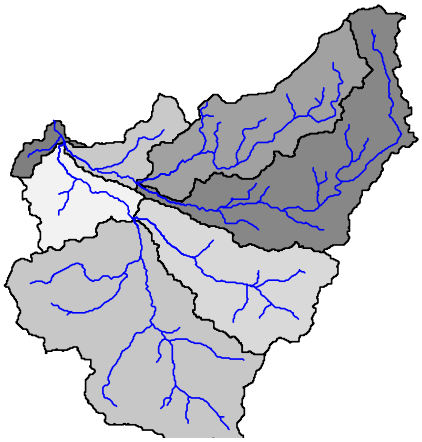



First order tributary watersheds (1.1, 1.2)

# 5 The `Watersheds.Order2` method

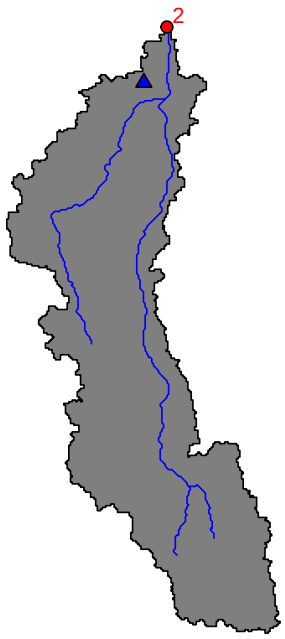
`Watershed.Order2` expands the analysis to the tributaries of a focal unit. When `Watershed.Order` returns two tributary nodes, this method applies the ordering procedure at each node and returns the two results. The resulting map shows the focal unit together with its first- and second-order tributaries, making the upstream structure easier to inspect.

**Current zhyd watershed and**
**1st and 2nd order tributary watersheds**



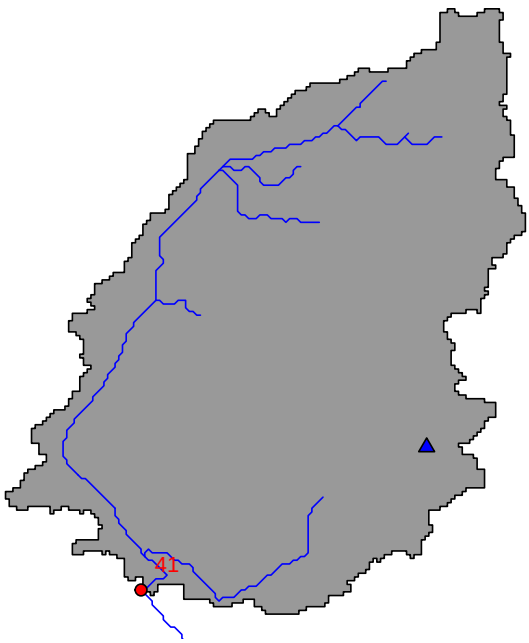
## 6 The `Watersheds.IOR1` function

The IOR functions resolve the inlet and outlet nodes after `Watershed.Order` has identified the connected river stretches. Case I applies when there is a single connected stretch. There is only an outlet in this situation, shown in red in the map.



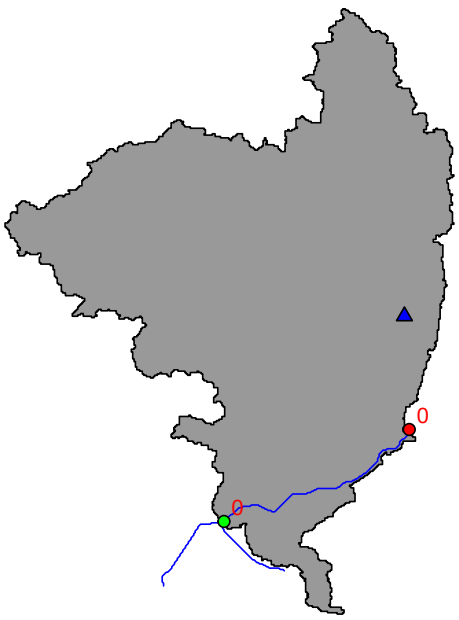

## 7 The `Watersheds.IOR2` function

Case II applies when two connected river stretches are present. The function uses the candidate nodes and their distances to the watershed boundary to determine the outlet shown in red.



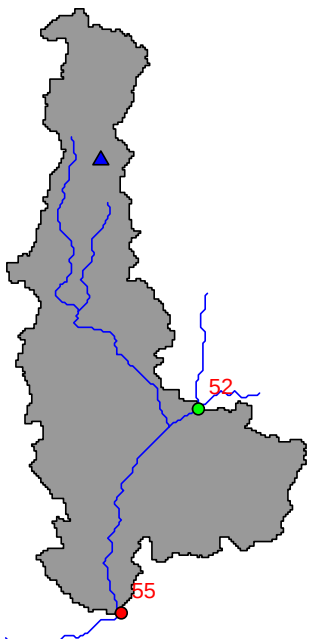

## 8 The `Watersheds.IOR3` function

Case III applies when three connected river stretches are present. The method distinguishes one outlet (red) from the inlet (green) using the river topology and the node distances to the watershed boundary.




## 9 The `Watersheds.IOR4` function

Case IV is the most connected example in the vignette: four river stretches meet the focal watershed. `Watersheds.IOR4` combines their topology and boundary distances to identify the inlet (green) and outlet (red). This final plot is Figure 9 in the vignette.




# References

[1] EAA. EEA catchments and rivers network system, ECRINS v1.1. rationales, building and improving for widening uses to Water Accounts and WISE applications. EEA Technical report 7/2012, European Environment Agency - EAA, Copenhagen, 2012. Luxembourg: Publications Office of the European Union.

[2] Ross Ihaka and Robert Gentleman. R: a languague for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5:299–314, 1996.

[3] Edzer Pebesma and Roger Bivand. *Package "sp": classes and methods for spatial data.* The Comprehensive R Archive Network, CRAN, 0.9-99 edition, May 2005-2012.

[4] R Development Core Team. R: A Language and Environment for Statistical Computing, 2013.