T-LoCoH Data Classes

Andy Lyons,
UC Berkeley

January 2014

Introduction

This vignette describes the structure of data classes used in T-LoCoH. There are two main data classes in T-LoCoH, and most of the functions in the package operate on one of these objects. A LoCoH-xy object (class `locoh.lxy`) contains locations for one or more individuals; it will often store ancillary values associated with each location (e.g., the time stamp, temperature) and/or a lookup table of nearest neighors. A LoCoH-hullset object (class `locoh.lhs`) is a collection of hulls (local MCPs) for a set of points. A LoCoH-hullset object may store multiple sets of hulls for the same set of locations if different hull methods (e.g., k, a, or r method) and/or parameter values were used. LoCoH-hullset objects also store hull metrics, isopleths (aggregations of hulls), and hull scatterplots (saved scatterplot settings).

LoCoH-xy and LoCoH-hullset objects are both S3 data types (i.e., lists). The spatial objects themselves (i.e., points and polygons) are saved as data classes from the `sp` package (Pebesma and Bivand, 2005). You can use standard R methods for lists to extract information and/or manipulate LoCoH-xy and LoCoH-hullset objects, but before you do too much surgery with native R commands please check the existing functions as numerous utility functions are available for common manipulations (see appendix of the T-LoCoH tutorial). In addition, if the functionality you desire has broader applications and you notify the package author, we may be able to incorporate your function into the package. For example, one user wanted to know how many points were enclosed in each region of a saved hull scatter plot. Given knowledge of the data structure below, one could readily write an expression to return this, but it was equally easy to incorporate that functionality in the `summary` function.

LoCoH-xy objects

LoCoH-xy objects (often referred to in documentation by a *lxy* suffix) have class `locoh.lxy`. This data class stores 2-dimensional point locations and associated ancillary variables (e.g., time stamps, point id, animal id tag, etc.). These items are bundled together and 'ready to go' as inputs into T-LoCoH functions. The

benefits of bundling in a single object all of the variables related to a set of locations include:

1. Cleaning and error checking only has to be done once when the lxy object is initially created (e.g., with `xyt.lxy`).
2. Having all of the ancillary variables together simplifies the task of passing these values to other functions, as well as saving or retrieving your work to/from disk.
3. The nearest neighbor lookup table (which can take a long time to compute) can be reused.
4. Locations for multiple individuals can be saved in one object.

The list elements of a LoCoH-xy object are named, and it is suggested that you always refer to elements by name rather than order because the order of elements may change over time (i.e., use `bobcat$pts` instead of `bobcat[[1]]`). The named elements include:

- `pts` A SpatialPointsDataFrame of the locations. Columns in the associated data frame vary but typically include:
  - `ptid` An integer vector of unique id values for each point
  - `id` A factor or character vector containing the ids of the individuals (e.g., name of the animal, GPS device) associated with each point
  - `dt` A vector of date-stamps (class `POSIXct`) for each point
  - `col` A factor or character vector of color values (single value or one per location)
  - any other ancillary variables associated with each location (e.g., temperature, vegetation type)
- `anv` A two-column data frame containing meta-data for the ancillary variables associated with each point (or NULL if none). Columns are `anv` (the name of the column in the `data` slot of the `pts` SpatialPointsDataFrame) and `desc` (a short description of the variable). See `lxy.anv.add` and `lxy.gridanv.add`.
- `dt.int` A four-column data frame containing a frequency table of the time interval between points. Columns include `id`, `interval`, `count`, and `rtn` (where rtn is the round-to-nearest value (in seconds) that was used in binning the delta.t values).
- `rw.params` A four-column data frame containing the parameters used to compute the time scaled distance for any pair of points as a function of their separation in both space and time (see Eq.1 in Lyons, Turner and Getz, 2013). Columns in the data frame include: `id`, `time.step.median`, `d.bar` (median step-length), and `vmax` (maximum observed velocity).

- `comment` A named list of comments that describe the locations. There is one element per id, and the element names are the id values. The default comment for each id is a constructed string consisting of the id, number of points, and start/end times. May also be NULL.
- `nn` Nearest neighbor lookup table (or NULL if no neighbors have been identified with `lxy.nn.add`). Note:
  - A different set of nearest neighbors will exist for each value of s, because s determines how point-to-point distance is computed.
  - Once nearest neighbors have been identified, they can be used to construct hulls for any nearest neighbor selection method (e.g., k, r, or a method).
  - For a set of nearest neighbors, the maximum possible value for a, k, and r are noted and saved as list elements. If you want to construct hulls for larger values of a, k, or r, you must first identify additional nearest neighbors using `lxy.nn.add`.

  `nn` is a list of lists. The name of each list element is a pipe-delimited character string that uniquely identifies the parameters for that set of neighbors (e.g., *ag206|vmax|s0|n17206|kmax10|rmax4.1|amax17.3*). Each element of this list is another list with the following elements:
  - `id` The id (name) of the individual.
  - `s` The value of s used in the time-scaled-distance metric.
  - `kmax` The maximum value of k that this set of nearest neighbors can be used to create hulls with the k-method.
  - `rmax` The maximum value of r that this set of nearest neighbors can be used to create hulls with the r-method.
  - `amax` The maximum value of a that this set of nearest neighbors can be used to create hulls with the a-method.
  - `ptid` A vector of ptid values (from `pts@data` that have nearest neighbors identified. This will normally be NULL (all points have neighbors identified) unless the `ptid` parameter was passed when `lxy.nn.add` was called.
  - `auto.a.df` A data frame of the auto-a values that have been calculated. See `lxy.amin.add`. Columns include:
    - `meth` The method used to identify a (`enc` = enclosed).
    - `ptp` Proportion of total points.
    - `nnn` Number of nearest neighbors.
    - `tct` Temporal continuity threshhold.
    - `aVal` The value of a that will result in *ptp* proportion of points being 1 of *nnn* nearest neighbors in at least one hull.

- `time.taken` The time taken for the script to identify the nearest neighbors (in seconds).
- `nn.df` A data frame with the following five columns:
  - `pp.idx` index of the parent point (from the `pts` SpatialPoints-DataFrame).
  - `nn.rank` Nearest neighbor rank (integer), starting from 0 for the parent point. Points with no identified neighbors will be have only the parent point itself as the 0th neighbor.
  - `nn.idx` Index of the nearest neighbor point (from `pts`). For the 0th neighbor this will be the same as `pp.idx`.
  - `tsd` Time-scaled-distance (see Eq.1 in Lyons, Turner and Getz, 2013).
  - `tsd.cumsum` Cumulative tsd.

- `ptsh` List of the proportion of time-selected hulls (i.e., the proportion of hulls constructed from sequential locations) for different values of s. This table is normally computed from a random sample of hulls with k=10 (see `lxy.ptsh.add`), and is used as a guide to select a value for s that balances space and time. `ptsh` is a named list with one element for each id in the LoCoH-xy object. Each list element is another list that represents the results of one computation of the ptsh table (which will differ because they are based on a random selection of parent points). List elements include:
  - `id` The id (name of the individual).
  - `samp.idx` A vector of the indices of the locations (from pts) that were used to compute the ptsh table.
  - `n` The number of the orginal locations sampled.
  - `k` The number of nearest neighbors found when computing the ptsh table.
  - `target.ptsh` The target ptsh. `lxy.ptsh.add` interatively tries to find a value of s such that these proportions of hulls (from the sampled parent points) are time-selected).
  - `target.s` The values of s that resulted in the targetted ptsh (within a margin specified by the ptsh.buf parameter in lxy.ptsh.add).
  - `s.ptsh` A two-column matrix containing 1) the found values of s and 2) the actual proportion of time-select hulls (from the sample). This matrix will include the targetted ptsh as well as others that were computed in an effort to find the targeted ptsh.
  - `time.taken` An object of class `difftime` that stores the amount of time it took for the script to compute the ptsh table.

LoCoH-hullset objects

A LoCoH-hullset is an object of class `locoh.lhs` is a S3 object (named list) containing hulls and their associated objects. A hullset collection may contain hullsets for multiple individuals (ids), nearest neighbor selection methods, and/or parameter values. The name of each element of a hullset collection takes the following form:

individual name + num points + method + s value + kmin value
e.g. *ag214.pts10702.a3700.s0.01.kmin0*

In code examples and documentation, LoCoH-hullset objects are typically noted by a *lhs* suffix, and functions that are designed to work on LoCoH-hullset objects typically start with *lhs*. For example, a single hullset can be extracted from a hullset collection with the `lhs.select` function, and two or more hullsets can be joined using the `lhs.merge` function. Each element of a LoCoH-hullset object is a hullset that contains:

- `id` The name of the individual. Hulls must always be for the same individual, to create hulls from the locations of multiple devices combined merge the locations into a single LoCoH-xy object (`lxy.merge`) and then assign them a new id (`lxy.id.new`).
- `pts` A SpatialPointsDataFrame of the locations. This is copied from the LoCoH-xy object (although duplicate locations will be randomly offset by a small amount if the `offset.dups` parameter was > 0 when the hullset was created, see `lxy.lhs`). Columns in the associated data frame typically include:
  - `ptid`. An integer vector of unique id values for each point
  - `id`. A factor or character vector containing the ids of the individuals (e.g., name of the animal, GPS device) associated with each point
  - `dt`. A vector of date-stamps (class `POSIXct`) for each point
  - `col`. A factor or character vector of color values (single value or one color value per location)
  - any other ancillary variables associated with each location (e.g., temperature, vegetation type)
- `anv`. A two-column data frame containing meta-data for the ancillary variables associated with each point (or NULL if none). Columns are `anv` (the name of the column in the `data` slot of the `pts` SpatialPointsDataFrame) and `desc` (a short description of the variable).
- `rw.params`. A four-column data frame containing the parameters used to compute the maximum distance for any pair of points as a function of the difference in time (see Eq.1 in Lyons, Turner and Getz, 2013). Columns in the data frame include: `id`, `time.step.median`, `d.bar` (median step-length), and `vmax` (maximum observed velocity).

- `mode` A character for the method used to make hulls: `k`, `a`, or `r`.
- `k` The value of k (NULL if k-method not used).
- `r` The value of r (NULL if r-method not used).
- `a` The value of a (NULL if a-method not used).
- `auto.a` A data frame with the auto.a parameters: `meth`, `ptp`, `nnn`, and `tct`. `auto.a` is an alternative way for the user to specify the a parameter value for the adaptive method (which is not intuitive to select because it represents a cummulative distance). The parameters tell the script to find the smallest value of a that will result in ptp proportion of total points having nnn nearest neighbors, excluding those points whose temporal distance from the closest point in time is more than tct times the median time interval for the entire series (see help page for `lxy.amin.add`).
- `s` The value of the space-time scaling parameter s.
- `kmin` The value of $kmin$.
- `dups` A list with two elements that records the random offset (if any) applied duplicate locations (which create problems when making hulls). The list elements include:
  - `dup.idx` Indices of duplicate points (NULL if no duplicates).
  - `offset` Displacement distance (same for all points). NULL if no duplicates, and 0 if no offset was requested. The displacement angle is not saved.
- `desc` A short description of the hulls (automatically generated by `lxy.lhs`).
- `hulls` A SpatialPolygonsDataFrame containing the hulls and hull metrics. Columns in the associated data frame include `pts.idx` (the index of the parent point taken from `pts` SpatialPointsDataFrame). Hull metrics will vary but several are automatically created (see separate vignette on hull metrics).
- `hm` A named list containing meta data for the hull metrics (hull metric values themselves are saved as columns in associated data frame of `hulls`). The name of each list element is the name of a hull metric (i.e., matches a column `hulls@data`). Each list element of the list is a list with two items:
  - `type` The type of hull metric (e.g., `area`, `mnlv`, `nsv`). In many cases, this will simply be the name of the hull metric. But other hull metrics take one or more auxillary parameters (e.g., the number of separate visits or `nsv` metric requires an additional parameter for the inter-visit gap). In these cases, the `type` will simply be the family name of the metric (`nsv`) while the hull metric name in the SpatialPolygons-DataFrame is a concatenation of the type and the auxiliary parameter

value(s) (e.g., *nsv.86400* where *86400* is the ivg value in seconds). See also vignette on hull metrics.

  - `aux` A named list of additional parameter values that were used in computing the hull metric. The name of each element is the name of the parameter, and the list elements are the parameter values for which metrics have been computed. For example, for the revisitation metrics (*nsv* and *mnlv*), `aux` will contain a list with one element named *ivg* which contains a vector of all the ivg values that have been used (each one corresponding to a different column in the `hulls` SpatialPolygonsDataFrame. For metrics that don't take auxillary parameters (e.g., geometric properties of a polygon), `aux` will be NULL.

- `hm.params` A list of the hull metric auxiliary parameters that have been used in the computation of hull metrics. The name of each list element is the name of the parameter, while the values are the values that have been analyzed. TO BE PHASED OUT, USE HM ELEMENT ABOVE INSTEAD.

- `enc.pts` A two-element list that record the points enclosed by each hull and the points used to construct each hull:

  - `idx` A list of the indices (from the `pts` SpatialPointsDataFrame) of the points enclosed by each hull. There will be one list element for each hull, in the same order as the `hulls` SpatialPolygonsDataFrame.

  - `nn` A list of Boolean (TRUE/FALSE) vectors that indicate whether each enclosed point was also a nearest neighbor (used in the construction of the hull as opposed to merely being enclosed by the hull). For example, if there are 22 points enclosed by hull 4, there will be 22 numeric values in `enc.pts$idx[[4]]` (indices of the points enclosed by the hull), and 22 Boolean values in `enc.pts$nn[[4]]` that indicate which of the 22 points in `enc.pts$idx[[4]]` was also a nearest neighbor. When s=0 (time ignored in point-to-point distance calculations), all enclosed points will also be nearest neighbors.

- `ellipses` A six-column data frame with parameters for the bounding ellipse for each hull (see `lhs.ellipses.add`). NULL if ellipses have not been computed or were not saved. Columns include:

  - `pts.idx` Index of the hull parent point. Depending on the hull parameters, not every point may have a hull and therefore an ellipse.

  - `cx` The x coordinate of the ellipse center.

  - `cy` The y coordinate of the ellipse center.

  - `a` Length of the semi-major axis.

  - `b` Length of the semi-minor axis.

  - `alpha` Angle of rotation from the x-axis in radians.

- `hsp` Saved scatterplots of hull metrics, or more precisely a list of the parameters used to create scatterplots of hull metrics. Saved hull scatterplots can be used for plot symbolizations and/or subsetting, and can also include saved regions (manually defined areas of the scatterplot space). Each list element is a S3 object of class `locoh.hsp`. List element names take the form *xaxis.yaxis.hmapvars.reg-n.i*, where *xaxis* and *yaxis* are the names of hull metrics plotted on the x and y axes respectively, *hmapvars* is a concatention of hull metric auxillary parameters (if any), *reg-n* is the number of manually defined regions, and i is an incremental counter that starts at 1. Hull scatterplot objects are returned by `lhs.plot.scatter` and added to a LoCoH-hullset object with `lhs.hsp.add`. List elements include:

  - `hs.name` The name of the set of hulls the scatterplot parameters were first generated from.
  - `x.axis` The hull metric for the x-axis.
  - `y.axis` The hull metric for the y-axis.
  - `limx` A vector of the lower and upper limits for the x-axis.
  - `limy` A vector of the lower and upper limits for the y-axis.
  - `trans.x` The name of a function used to transform the x-axis values.
  - `trans.y` The name of a function used to transform the y-axis values.
  - `jiggle.x` The range of a random offset for x-axis values (for better visualization of point density when x a categorical).
  - `hmap` A named list of auxiliary parameters that were used in computing the hull metrics. The name of each list element is the name of the parameter.
  - `title` A plot title.
  - `ufat` Whether to substitute user-friendly axis titles (pre-defined for each hull metric), T/F.
  - `bg` Background color.
  - `cex` The plot symbol expansion factor.
  - `col` A vector of color values or 'spiral'.
  - `hue.offset` The hue offset for a spiral color scheme (see `lhs.plot.scatter`).
  - `sat.base` The saturation base for a spiral color scheme (see `lhs.plot.scatter`).
  - `val.base` The value base for a spiral color scheme (see `lhs.plot.scatter`).
  - `center.method` How to center the spiral (see `lhs.plot.scatter`).
  - `regions` A list of manually defined regions in scatterplot space. Each list element is a list with three elements:
    * `poly.pts` Data frame of xy coordinates of the nodes that define the region

    * `col` Color to use when plotting the region
    * `label` Label for the region

- `isos` A list of isopleths. NULL if isopleths have not been created. The name of each list element takes the form *iso.sort-method.hmap-vals.iso-method.num-hulls.num-isopleth-levels* where *sort-method* is the name of the hull metric used to sort hulls, *hmap-vals* is a concatenated string of any hull metric auxillary parameters used, *iso-method* is point quantiles or hull metric values, *num-hulls* is the total number of hulls unioned, and *num-isopleth-levels* is the number of isopleths created. List elements include:
  - `desc` A short description of the isopleth (which may be used as a caption for a plot).
  - `ufipt` A user-friendly title for isopleth plots.
  - `sort.metric` The name of the hull metric that was used to sort hulls prior to unioning.
  - `iso.method` What the isopleth levels represent: `pt.quantiles` (quantiles of enclosed points) or `hm.vals` (hull metric values).
  - `hmap` A named list of hull metric auxiliary parameters that were used in computing the sort metric. The name of each list element is the name of the parameter. NULL if the sort metric didn't require additional parameters.
  - `polys` A SpatialPolygonsDataFrame containing the isopleths. Columns in the data frame include:
    * `iso.level` The isopleth level (usually representing a desired proportion of total points enclosed).
    * `area` The isopleth area (in map units squared)
    * `edge.len` The total length of the isopleth perimeter (including holes)
    * `nep` Actual number of enclosed points.
    * `ptp` Actual proportion of total points enclosed.
    * `hm.val` The value of the sort metric in the last hull added to the isopleth.
    * `num.hulls` The number of hulls unioned to produce the isopleth.
  - `subset.metric` Name of a second hull metric used to subset the hulls that went into the isopleth construction (i.e., stratified isopleths). NULL if not used.
  - `subset.vals` Numeric vector containing the lower and upper limits of `subset.metric` for each strata of hulls, where the lower bound is closed (>=) and the upper bound is open (<). NULL if not used.
  - `subset.vals.fr` The full range (all strata) of `subset.metric` values, used for setting the color ramp when plotting stratified isopleths.

  - `rast` A raster version of the isopleths (object of class `raster`). See `lhs.iso.rast`.

- `dr` List of directional routes (temporally contiguous hull parent points selected by an elongation hull metric above a certain threshold), created by `lhs.dr.add`. The name of each list element takes the form: *dr.elongtation-metric.threshhold.smoothing-factor* (e.g., *dr.par.q0.9.sm1*). List elements include:

  - `metric` The name hull metric that proxies directional movement (e.g, *par* (perimeter:area ratio) or *ecc* (eccentricity of the bounding ellipsoid)).
  - `thresh.val` The threshhold value above which a hull must fall to be considered part of a directional route.
  - `thresh.type` The threshhold type: `q` for quantile or `v` for hull metric value.
  - `smooth` Smoothing factor (see `lhs.dr.add`).
  - `lines` An unnamed list of indices (of `pts`) that comprise the directional route segments.

- `hin` A list of hull intersections (with the hulls from another set of hulls in the same LoCoH-hullset object). Intesecting hulls are used when computing hull metrics that use the spatial overlap of two individuals as the basis for metrics of association. For documentation purposes, the current set of hulls are called the reference hulls, and the second set of hulls are called the comparison hulls. The name of each list element in `hin` identifies the comparison set of hulls and is constructed in the form *id2Val.sVal.karVal* (e.g., *cilla.s0.1.k10*). Each element is a list that contains:

  - `id` The id of the comparison hulls.
  - `s` The s value of the comparison hulls.
  - `mode` The mode of the comparison hulls.
  - `k` The k value of the comparison hulls.
  - `a` The a value of the comparison hulls.
  - `r` The r value of the comparison hulls.
  - `hidx` A list of length equal to the number of reference hulls containing the indices of the intersecting hulls in the comparison hulls.

- `hto` A list of hull temporal overlap info with a comparison set of hulls (within the same LoCoH-hullset and presumably for a different individual). The name of each element is the name of the comparison set of hulls. List elements include:

– `id`, `s`, `mode`, `k`, `a`, and `r` Parameters that identify the comparison set of hulls.

– `maxdt` The maximum temporal distance (in seconds) of the parent points for two hulls to be considered temporally overlapping.

– `to.lst` A list of the indices of the temporally overlapping hulls.