Summary of recent updates to `spatstat`

Adrian Baddeley, Rolf Turner and Ege Rubak

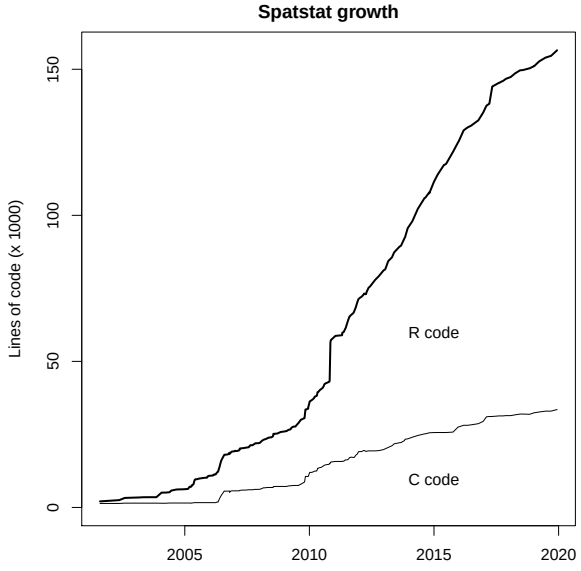
For `spatstat` version `1.62-2`

This is a summary of changes that have been made to the `spatstat` package since the publication of the accompanying book [2].

The book [2], published in December 2015, covers everything in `spatstat` up to version `1.42-0`, released in May 2015.

The `spatstat` package has grown by 33% since the book was published. This document summarises the most important changes.

The current version of `spatstat` is `1.62-2`. It contains 701 new functions and 3 new datasets introduced after May 2015.



Contents

1 spatstat is splitting into parts

spatstat is being split into sub-packages, to satisfy the requirements of CRAN. This should not affect the user: existing code will continue to work in the same way.

Typing `library(spatstat)` will load the familiar spatstat package which can be used as before.

1.1 The parts of spatstat

Currently there are three sub-packages, called `spatstat.utils`, `spatstat.data` and `spatstat`.

- The `spatstat` package contains the main code.
- The `spatstat.data` package now contains all the datasets for `spatstat`.
- The `spatstat.utils` package contains utility functions for `spatstat`.

Typing `library(spatstat)` will automatically load `spatstat.data` and silently "import" `spatstat.utils`.

To access the functions in `spatstat.utils` directly, you would need to type `library(spatstat.utils)`.

1.2 Extension packages

There are also extension packages which provide additional capabilities and must be loaded explicitly when you need them. Currently there are two extension packages:

- `spatstat.local` for local model-fitting,
- `spatstat.sphere` for analysing point patterns on a sphere.

2 Precis of all changes

Here is the text from the 'overview' sections of the News and Release Notes for each update.

- `spatstat` now Imports the package `spatstat.utils`.
- `spatstat` now requires the package `spatstat.data` which contains the datasets.
- `spatstat` now suggests the package `fftwtools`.
- Tessellations on a linear network can now have marks.
- More functions for manipulating tessellations on a linear network.
- New functions for simulating point processes on a linear network.
- Nearest Neighbour Index function can now return mark values.
- Index of repulsion strength for determinantal point process models.
- Nearest neighbours between two point patterns in any number of dimensions.
- More options for handling bad simulation outcomes in `envelope`.
- `mppm` accepts case weights.
- Bandwidth selectors warn about extreme values of bandwidth.
- Fast kernel estimation on a linear network using 2D kernels.
- Extension of Scott's rule for bandwidth selection.
- Cross-validated bandwidth selection on a linear network.
- Random thinning and random labelling of spatial patterns extended to different types of pattern.
- Confidence intervals for multitype K function.
- Envelopes for balanced two-stage test
- Extensions to adaptive intensity estimators
- 'Dartboard' tessellation using polar coordinates.
- Standard error calculation for inverse-distance weighting.
- Kernel estimate of intensity as a `function(x,y)`.
- Extract discrete and continuous components of a measure.
- Improvements and extensions to leverage and influence code.
- Plot a line segment pattern using line widths.
- Find connected components of each tile in a tessellation.
- Geometrical operations on `distfun` objects.
- Join vertices in a linear network.

- Distance map and contact distribution for rectangular structuring element.
- Lurking variable plot for models fitted to several point patterns.
- New dataset `cetaceans`.
- Gamma correction for colour maps and image plots.
- Class `units` has been renamed `unitname` to avoid package collision.
- More support for tessellations.
- Fixed longstanding bug in leverage and influence diagnostics.
- Improvements and bug fixes for leverage and influence diagnostics.
- Tighter bounding box for `psp`, `lpp`, `linnet` objects.
- Improved layout in `plot.solist`
- Tools to increase colour saturation.
- Connected components of a 3D point pattern.
- Accelerated computations on linear networks.
- Accelerated simulation of determinantal point processes.
- Improved printing of 3D point patterns.
- Minor corrections to handling of unitnames.
- Improvements to `ppm` and `update.ppm`.
- Correction to `lohboot`
- Numerous bug fixes for linear networks code.
- Now handles disconnected linear networks.
- Effect function is now available for all types of fitted model.
- Geometric-mean smoothing.
- A model can be fitted or re-fitted to a sub-region of data.
- New fast algorithm for kernel smoothing on a linear network.
- Leverage and influence diagnostics extended to Poisson/Gibbs models fitted by logistic composite likelihood.
- Two-stage Monte Carlo test.
- Dirichlet/Voronoi tessellation on a linear network.
- Thinning of point patterns on a linear network.
- More support for functions and tessellations on a linear network.

- Bandwidth selection for pair correlation function.
- Pooling operations improved.
- Operations on signed measures.
- Operations on lists of pixel images.
- Improved pixellation of point patterns.
- Stieltjes integral extended.
- Subset operators extended.
- Greatly accelerated `rmh` when using `nsave`
- Sufficient Dimension Reduction for point processes.
- Alternating Gibbs Sampler for point process simulation.
- New class of spatially sampled functions.
- ROC and AUC extended to other types of point patterns and models.
- More support for linear networks.
- More support for infinite straight lines.
- `spatstat` now depends on the packages `nlme` and `rpart`.
- Important bug fix in `linearK`, `linearpcf`
- Changed internal format of `linnet` and `lpp` objects.
- Faster computation in linear networks.
- Bias correction techniques.
- Bounding circle of a spatial object.
- Option to plot marked points as arrows.
- Kernel smoothing accelerated.
- Workaround for bug in some graphics drivers affecting image orientation.
- Non-Gaussian smoothing kernels.
- Improvements to inhomogeneous multitype K and L functions.
- Variance approximation for pair correlation function.
- Leverage and influence for multitype point process models.
- Functions for extracting components of vector-valued objects.
- Recursive-partition point process models.
- Minkowski sum, morphological dilation and erosion with any shape.

- Minkowski sum also applicable to point patterns and line segment patterns.
- Important bug fix in Smooth.ppp
- Important bug fix in spatial CDF tests.
- More bug fixes for replicated patterns.
- Simulate a model fitted to replicated point patterns.
- Inhomogeneous multitype F and G functions.
- Summary functions recognise `correction="all"`
- Leverage and influence code handles bigger datasets.
- More support for pixel images.
- Improved progress reports.
- New dataset `redwood3`
- Fixed namespace problems arising when spatstat is not loaded.
- Important bug fix in leverage/influence diagnostics for Gibbs models.
- Surgery with linear networks.
- Tessellations on a linear network.
- Laslett's Transform.
- Colour maps for point patterns with continuous marks are easier to define.
- Pair correlation function estimates can be pooled.
- Stipulate a particular version of a package.
- More support for replicated point patterns.
- More support for tessellations.
- More support for multidimensional point patterns and point processes.
- More options for one-sided envelopes.
- More support for model comparison.
- Convexifying operation.
- Subdivide a linear network.
- Penttinen process can be simulated (by Metropolis-Hastings or CFTP).
- Calculate the predicted variance of number of points.
- Accelerated algorithms for linear networks.
- Quadrat counting accelerated, in some cases.

- Simulation algorithms have been accelerated; simulation outcomes are *not* identical to those obtained from previous versions of `spatstat`.
- Determinantal point process models.
- Random-effects and mixed-effects models for replicated patterns.
- Dao-Genton test, and corresponding simulation envelopes.
- Simulated annealing and simulated tempering.
- spatstat colour tools now handle transparent colours.
- Improvements to `[` and `subset` methods
- Extensions to kernel smoothing on a linear network.
- Support for one-dimensional smoothing kernels.
- Mark correlation function may include weights.
- Cross-correlation version of the mark correlation function.
- Penttinen pairwise interaction model.
- Improvements to simulation of Neyman-Scott processes.
- Improvements to fitting of Neyman-Scott models.
- Extended functionality for pixel images.
- Fitted intensity on linear network
- Triangulation of windows.
- Corrected an edge correction.

3 New datasets

The following datasets have been added to the package.

- `austates`: The states and large mainland territories of Australia represented as polygonal regions forming a tessellation.
- `redwood3`: a more accurate version of the `redwood` data.
- `cetaceans`: point patterns of whale and dolphin sightings.

4 New classes

- `ssf`: Class of spatially sampled functions.

5 New Functions

Following is a list of all the functions that have been added.

- `rcelllpp`: Simulate the cell point process on a linear network.
- `rSwitzerlpp`: Simulate the Switzer-type point process on a linear network.
- `intersect.lintess`: Form the intersection of two tessellations on a linear network.
- `chop.linnet`: Divide a linear network into tiles using infinite lines.
- `repairNetwork`: Detect and repair inconsistencies in internal data in a `linnet` or `lpp` object.
- `marks<-.lintess`, `unmark.lintess`: Assign marks to the tiles of a tessellation on a linear network.
- `marks.lintess`: Extract the marks of the tiles of a tessellation on a linear network.
- `tilenames.lintess`: Extract the names of the tiles in a tessellation on a linear network
- `tilenames<-.lintess`: Change the names of the tiles in a tessellation on a linear network
- `nobjects.lintess`: Count the number of tiles in a tessellation on a linear network
- `as.data.frame.lintess`: Convert a tessellation on a linear network into a data frame.
- `repul`: Repulsiveness index for a determinantal point process model.
- `reach.kppm`: Reach (interaction distance) for a Cox or cluster point process model.
- `summary.dppm`, `print.summary.dppm`: Summary method for determinantal point process models.
- `nncross.ppx`: Nearest neighbours between two point patterns in any number of dimensions.
- `rthinclumps`: Divide a spatial region into clumps and randomly delete some of them.
- `densityQuick.lpp`: Fast kernel estimator of point process intensity on a network using 2D smoothing kernel.
- `data.lppm`: Extract the original point pattern dataset (on a linear network) to which the model was fitted.
- `bw.scott.iso`: Isotropic version of Scott's rule (for point patterns in any dimension).
- `bits.envelope`: Global simulation envelope corresponding to `bits.test`, the balanced independent two-stage Monte Carlo test.
- `extrapolate.psp`: Extrapolate line segments to obtain infinite lines.
- `uniquemap`: Map duplicate points to unique representatives. Generic with methods for `ppp`, `lpp`, `ppx`
- `uniquemap.data.frame`, `uniquemap.matrix`: Map duplicate rows to unique representatives
- `localKcross`, `localLcross`, `localKdot`, `localLdot`, `localKcross.inhom`, `localLcross.inhom`: Multitype local K functions.

- `polartess`: tessellation using polar coordinates.
- `densityVoronoi`: adaptive estimate of point process intensity using tessellation methods.
- `densityAdaptiveKernel`: adaptive estimate of point process intensity using variable kernel methods.
- `bw.abram`: compute adaptive smoothing bandwidths using Abramson's rule.
- `coords.quad`: method for `coords`, to extract the coordinates of the points in a quadrature scheme.
- `lineartileindex`: low-level function to classify points on a linear network according to which tile of a tessellation they fall inside.
- `markmarkscatter`: Mark–mark scatterplot.
- `bw.CvL`: Cronie-van Lieshout bandwidth selection for density estimation.
- `subset.psp`: subset method for line segment patterns.
- `densityfun`, `densityfun.ppp`: Compute a kernel estimate of intensity of a point pattern and return it as a function of spatial location.
- `as.im.densityfun`: Convert `function(x,y)` to a pixel image.
- `measureDiscrete`, `measureContinuous`: Extract the discrete and continuous components of a measure.
- `connected.tess`: Find connected components of each tile in a tessellation and make a new tessellation composed of these pieces.
- `dffit.ppm`: Effect change diagnostic `DFFIT` for spatial point process models.
- `shift.distfun`, `rotate.distfun`, `reflect.distfun`, `flipxy.distfun`, `affine.distfun`, `scalardilate.distfun`: Methods for geometrical operations on `distfun` objects.
- `rescale.distfun`: Change the unit of length in a `distfun` object.
- `plot.indicfun`: Plot method for indicator functions created by `as.function.owin`.
- `Smooth.leverage.ppm`, `Smooth.influence.ppm`: Smooth a leverage function or an influence measure.
- `integral.leverage.ppm`, `integral.influence.ppm`: Compute the integral of a leverage function or an influence measure.
- `mean.leverage.ppm`: Compute the mean value of a leverage function.
- `rectdistmap`: Distance map using rectangular metric.
- `rectcontact`: Contact distribution function using rectangular structuring element.
- `joinVertices`: Join specified vertices in a linear network.
- `summary.ssf`: Summary method for a spatially sampled function (class `ssf`).

- `unstack.tess`: Given a tessellation with multiple columns of marks, take the columns one at a time, and return a list of tessellations, each carrying only one of the original columns of marks.
- `contour.leverage.ppm`: Method for `contour` for leverage functions of class `leverage.ppm`
- `lurking`: New generic function for lurking variable plots.
- `lurking.ppp`, `lurking.ppm`: These are equivalent to the original function `lurking`. They are now methods for the new generic `lurking`.
- `lurking.mppm`: New method for class `mppm`. Lurking variable plot for models fitted to several point patterns.
- `print.lurk`: Prints information about the object returned by the function `lurking` representing a lurking variable plot.
- `model.matrix.mppm`: Method for `model.matrix` for models of class `mppm`.
- `test.crossing.psp`, `test.selfcrossing.psp`: Previously undocumented functions for testing whether segments cross.
- `to.saturated`: Convert a colour value to the corresponding fully-saturated colour.
- `intensity.psp`: Compute the average total length of segments per unit area.
- `boundingbox.psp`: Bounding box for line segment patterns. This produces a tighter bounding box than the previous default behaviour.
- `boundingbox.lpp`: Bounding box for point patterns on a linear network. This produces a tighter bounding box than the previous default behaviour.
- `boundingbox.linnet`: Bounding box for a linear network. This produces a tighter bounding box than the previous default behaviour.
- `"Frame<-.default"`: New default method for assigning bounding frame to a spatial object.
- `connected.pp3`: Connected components of a 3D point pattern.
- `colouroutputs`, `"colouroutputs<-"`: Extract or assign colour values in a colour map. (Documented a previously-existing function)
- `fitin.profilepl`: Extract the fitted interaction from a model fitted by profile likelihood.
- `[<-.linim`: Subset assignment method for pixel images on a linear network.
- `nnfromvertex`: Given a point pattern on a linear network, find the nearest data point from each vertex of the network.
- `tile.lengths`: Calculate the length of each tile in a tessellation on a network.
- `text.ppp`, `text.lpp`, `text.psp`: Methods for `text` for spatial patterns.
- `as.data.frame.envelope`: Extract function data from an envelope object, including the functions for the simulated data ('simfuns') if they were saved.
- `is.connected`, `is.connected.default`, `is.connected.linnet`: Determines whether a spatial object consists of one topologically connected piece, or several pieces.

- `is.connected.ppp`: Determines whether a point pattern is connected after all pairs of points closer than distance R are joined.
- `hist.funxy`: Histogram of values of a spatial function.
- `model.matrix.ippm`: Method for `model.matrix` which allows computation of regular and irregular score components.
- `harmonise.msr`: Convert several measures (objects of class `msr`) to a common quadrature scheme.
- `bits.test`: Balanced Independent Two-Stage Monte Carlo test, an improvement on the Dao-Genton test.
- `lineardirichlet`: Computes the Dirichlet-Voronoi tessellation associated with a point pattern on a linear network.
- `domain.lintess`, `domain.linfun`: Extract the linear network from a `lintess` or `linfun` object.
- `summary.lintess`: Summary of a tessellation on a linear network.
- `clicklpp`: Interactively add points on a linear network.
- `envelopeArray`: Generate an array of envelopes using a function that returns `fasp` objects.
- `bw.pcf`: Bandwidth selection for pair correlation function.
- `grow.box3`: Expand a three-dimensional box.
- `hexagon`, `regularpolygon`: Create regular polygons.
- `Ops.msr`: Arithmetic operations for measures.
- `Math.imlist`, `Ops.imlist`, `Summary.imlist`, `Complex.imlist`: Arithmetic operations for lists of pixel images.
- `measurePositive`, `measureNegative`, `measureVariation`, `totalVariation`: Positive and negative parts of a measure, and variation of a measure.
- `as.function.owin`: Convert a spatial window to a `function(x,y)`, the indicator function.
- `as.function.ssf`: Convert an object of class `ssf` to a `function(x,y)`
- `as.function.leverage.ppm` Convert an object of class `leverage.ppm` to a `function(x,y)`
- `sdr`, `dimhat`: Sufficient Dimension Reduction for point processes.
- `simulate.rhohat`: Simulate a Poisson point process with the intensity estimated by `rhohat`.
- `rlpp`: Random points on a linear network with a specified probability density.
- `cut.lpp`: Method for `cut` for point patterns on a linear network.
- `has.close`: Faster way to check whether a point has a close neighbour.
- `psib`: Sibling probability (index of clustering strength in a cluster process).
- `rags`, `ragsAreaInter`, `ragsMultiHard`: Alternating Gibbs Sampler for point processes.

- `bugfixes`: List all bug fixes in recent versions of a package.
- `ssf`: Create a spatially sampled function
- `print.ssf`, `plot.ssf`, `contour.ssf`, `image.ssf`: Display a spatially sampled function
- `as.im.ssf`, `as.ppp.ssf`, `marks.ssf`, `marks<-.ssf`, `unmark.ssf`, `[.ssf`, `with.ssf`: Manipulate data in a spatially sampled function
- `Smooth.ssf`: Smooth a spatially sampled function
- `integral.ssf`: Approximate integral of spatially sampled function
- `roc.kppm`, `roc.lppm`, `roc.lpp`: Methods for `roc` for fitted models of class `"kppm"` and `"lppm"` and point patterns of class `"lpp"`
- `auc.kppm`, `auc.lppm`, `auc.lpp`: Methods for `auc` for fitted models of class `"kppm"` and `"lppm"` and point patterns of class `"lpp"`
- `timeTaken`: Extract the timing data from a `"timed"` object or objects.
- `rotate.infline`, `shift.infline`, `reflect.infline`, `flipxy.infline`: Geometrical transformations for infinite straight lines.
- `whichhalfplane`: Determine which side of an infinite line a point lies on.
- `matrixpower`, `matrixsqrt`, `matrixinvsqrt`: Raise a matrix to any power.
- `points.lpp`: Method for `points` for point patterns on a linear network.
- `pairs.linim`: Pairs plot for images on a linear network.
- `closetriples`: Find close triples of points.
- `anyNA.im`: Method for `anyNA` for pixel images.
- `bc`: Bias correction (Newton-Raphson) for fitted model parameters.
- `rex`: Richardson extrapolation for numerical integrals and statistical model parameter estimates.
- `boundingcircle`, `boundingcentre`: Find the smallest circle enclosing a window or point pattern.
- `[.linim` : Subset operator for pixel images on a linear network.
- `mean.linim`, `median.linim`, `quantile.linim`: The mean, median, or quantiles of pixel values in a pixel image on a linear network.
- `weighted.median`, `weighted.quantile`: Median or quantile of numerical data with associated weights.
- `"[.linim"`: Subset operator for pixel images on a linear network.
- `mean.linim`, `median.linim`, `quantile.linim`: The mean, median, or quantiles of pixel values in a pixel image on a linear network.
- `boundingcircle`, `boundingcentre`: Smallest circle enclosing a spatial object.
- `split.msr`: Decompose a measure into parts.

- `unstack.msr`: Decompose a vector-valued measure into its component measures.
- `unstack.ppp`, `unstack.psp`, `unstack.lpp`: Given a spatial pattern with several columns of marks, separate the columns and return a list of spatial patterns, each having only one column of marks.
- `kernel.squint`: Integral of squared kernel, for the kernels used in density estimation.
- `as.im.data.frame`: Build a pixel image from a data frame of coordinates and pixel values.
- `covering`: Cover a window using discs of a given radius.
- `dilationAny`, `erosionAny`, `%(-)%` : Morphological dilation and erosion by any shape.
- `FmultiInhom`, `GmultiInhom` Inhomogeneous multitype/marked versions of the summary functions `Fest`, `Gest`.
- `kernel.moment` Moment or incomplete moment of smoothing kernel.
- `MinkowskiSum`, `%(+)%`: Minkowski sum of two windows: `A %(+)% B`, or `MinkowskiSum(A,B)`
- `nobjects`: New generic function for counting the number of 'things' in a dataset. There are methods for `ppp`, `ppx`, `psp`, `tess`.
- `parameters.interact`, `parameters.fii`: Extract parameters from interpoint interactions. (These existing functions are now documented.)
- `ppmInfluence`: Calculate `leverage.ppm`, `influence.ppm` and `dfbetas.ppm` efficiently.
- `rppm`, `plot.rppm`, `predict.rppm`, `prune.rppm`: Recursive-partition point process models.
- `simulate.mppm` Simulate a point process model fitted to replicated point patterns.
- `update.interact`: Update the parameters of an interpoint interaction. [This existing function is now documented.]
- `where.max`, `where.min` Find the spatial location(s) where a pixel image achieves its maximum or minimum value.
- `compileK`, `compilepcf`: make a K function or pair correlation function given the pairwise distances and their weights. [These existing internal functions are now documented.]
- `laslett`: Laslett's Transform.
- `lintess`: Tessellation on a linear network.
- `divide.linnet`: Divide a linear network into pieces demarcated by a point pattern.
- `insertVertices`: Insert new vertices in a linear network.
- `thinNetwork`: Remove vertices and/or segments from a linear network etc.
- `connected.linnet`: Find connected components of a linear network.
- `nvertices`, `nvertices.linnet`, `nvertices.owin`: Count the number of vertices in a linear network or vertices of the boundary of a window.

- `as.data.frame.linim`, `as.data.frame.linfun`: Extract a data frame of spatial locations and function values from an object of class `linim` or `linfun`.
- `as.linfun`, `as.linfun.linim`, `as.linfun.lintess`: Convert other kinds of data to a `linfun` object.
- `requireversion`: Require a particular version of a package (for use in stand-alone R scripts).
- `as.function.tess`: Convert a tessellation to a `function(x,y)`. The function value indicates which tile of the tessellation contains the point (x, y).
- `tileindex`: Determine which tile of a tessellation contains a given point (x, y).
- `persp.leverage.ppm`: Method for persp plots for objects of class `leverage.ppm`
- `AIC.mppm`, `extractAIC.mppm`: AIC for point process models fitted to replicated point patterns.
- `nobs.mppm`, `terms.mppm`, `getCall.mppm`: Methods for point process models fitted to replicated point patterns.
- `rPenttinen`: Simulate the Penttinen process using perfect simulation.
- `varcount`: Given a point process model, compute the predicted variance of the number of points falling in a window.
- `inside.boxx`: Test whether multidimensional points lie inside a specified multidimensional box.
- `lixellate`: Divide each segment of a linear network into smaller segments.
- `nsegments.linnet`, `nsegments.lpp`: Count the number of line segments in a linear network.
- `grow.boxx`: Expand a multidimensional box.
- `deviance.ppm`, `deviance.lppm`: Deviance for a fitted point process model.
- `pseudoR2`: Pseudo-R-squared for a fitted point process model.
- `tiles.empty` Checks whether each tile of a tessellation is empty or nonempty.
- `summary.linim`: Summary for a pixel image on a linear network.
- Determinantal Point Process models:
  - `dppm`: Fit a determinantal point process model.
  - `fitted.dppm`, `predict.dppm`, `intensity.dppm`: prediction for a fitted determinantal point process model.
  - `Kmodel.dppm`, `pcfmodel.dppm`: Second moments of a determinantal point process model.
  - `rdpp`, `simulate.dppm`: Simulation of a determinantal point process model.
  - `logLik.dppm`, `AIC.dppm`, `extractAIC.dppm`, `nobs.dppm`: Likelihood and AIC for a fitted determinantal point process model.
  - `print.dppm`, `reach.dppm`, `valid.dppm`: Basic information about a `dpp` model.
  - `coef.dppm`, `formula.dppm`, `print.dppm`, `terms.dppm`, `labels.dppm`, `model.frame.dppm`, `model.matrix.dppm`, `model.images.dppm`, `is.stationary.dppm`, `reach.dppm`, `unitname.dppm`, `unitname<-.dppm`, `Window.dppm`: Various methods for `dppm` objects.

  - `parameters.dppm`: Extract meaningful list of model parameters.
  - `objsurf.dppm`: Objective function surface of a `dppm` object.
  - `residuals.dppm`: Residual measure for a `dppm` object.

- Determinantal Point Process model families:
  - `dppBessel`, `dppCauchy`, `dppGauss`, `dppMatern`, `dppPowerExp`: Determinantal Point Process family functions.
  - `detpointprocfamilyfun`: Create a family function.
  - `update.detpointprocfamily`: Set parameter values in a determinantal point process model family.
  - `simulate.dppm`: Simulation.
  - `is.stationary.detpointprocfamily`, `intensity.detpointprocfamily`, `Kmodel.detpointprocfamil` `pcfmodel.detpointprocfamily`: Moments.
  - `dim.detpointprocfamily`, `dppapproxkernel`, `dppapproxpcf`, `dppeigen`, `dppkernel`, `dpp-parbounds`, `dppspecdenrange`, `dppspecden`: Helper functions.

- `dg.envelope`: Simulation envelopes corresponding to Dao-Genton test.
- `dg.progress`: Progress plot (envelope representation) for the Dao-Genton test.
- `dg.sigtrace`: significance trace for the Dao-Genton test.
- `markcrosscorr`: Mark cross-correlation function for point patterns with several columns of marks.
- `rtemper`: Simulated annealing or simulated tempering.
- `rgb2hsva`: Convert RGB to HSV data, like `rgb2hsv`, but preserving transparency.
- `superimpose.ppplist`, `superimpose.splitppp`: New methods for 'superimpose' for lists of point patterns.
- `dkernel`, `pkernel`, `qkernel`, `rkernel`: Probability density, cumulative probability, quantiles and random generation from distributions used in basic one-dimensional kernel smoothing.
- `kernel.factor`: Auxiliary calculations for one-dimensional kernel smoothing.
- `spatdim`: Spatial dimension of any object in the `spatstat` package.
- `as.boxx`: Convert data to a multi-dimensional box.
- `intensity.ppx`: Method for `intensity` for multi-dimensional space-time point patterns.
- `fourierbasis`: Evaluate Fourier basis functions in any number of dimensions.
- `valid`: New generic function, with methods `valid.ppm`, `valid.lppm`, `valid.dppm`.
- `emend`, `emend.ppm`, `emend.lppm`: New generic function with methods for `ppm` and `lppm`. `emend.ppm` is equivalent to `project.ppm`.
- `Penttinen`: New pairwise interaction model.
- `quantile.density`: Calculates quantiles from kernel density estimates.

- `CDF.density`: Calculates cumulative distribution function from kernel density estimates.
- `triangulate.owin`: decompose a spatial window into triangles.
- `fitted.lppm`: fitted intensity values for a point process on a linear network.
- `parameters`: Extract all parameters from a fitted model.

6 Alphabetical list of changes

Here is a list of all changes made to existing functions, listed alphabetically.

- `adaptive.density`: This function can now perform adaptive estimation by two methods: either tessellation-based methods or variable-bandwidth kernel estimation. The calculations are performed by either `densityVoronoi` or `densityAdaptiveKernel`.
- `affine.owin`: Allows transformation matrix to be singular, if the window is polygonal.
- `alltypes`: If `envelope=TRUE` and the envelope computation reaches the maximum permitted number of errors (`maxnerr`) in evaluating the summary function for the simulated patterns, then instead of triggering a fatal error, the envelope limits will be set to `NA`.
- `anova.mppm`: Now handles Gibbs models, and performs the adjusted composite likelihood ratio test. New argument `fine`.
- `anyDuplicated.ppp`: Accelerated.
- `append.psp`: arguments may be `NULL`.
- `as.function.tess`: New argument `values` specifies the function values.
- `as.im.distfun`: New argument `approx` specifies the choice of algorithm.
- `as.im.function`:
  - New argument `strict`.
  - New argument `stringsAsFactors`.
- `as.im.leverage.ppm`: New argument `what`.
- `as.im.nnfun`: New argument `approx` chooses between a fast, approximate algorithm and a slow, exact algorithm.
- `as.im.smoothfun`: New argument `approx` chooses between a fast, approximate algorithm and a slow, exact algorithm.
- `as.layered`: Default method now handles a (vanilla) list of spatial objects.
- `as.linfun.lintess`:
  - New argument `values` specifies the function value for each tile.
  - The default `values` are the marks, if present.
  - New argument `navalue`.
  - Computation accelerated.

- as.linim.default: New argument delta controls spacing of sample points in internal data.
- as.linnet.psp:
  - If the line segment pattern has marks, then the resulting linear network also carries these marks in the $lines component.
  - Computation accelerated.
- as.owin.default:
  - Now refuses to convert a box3 to a two-dimensional window.
  - Now accepts a structure with entries named xmin,xmax, ymin, ymax in any order. This handles objects of class bbox in the sf package.
  - Now detects objects of class SpatialPolygons and issues a more helpful error message.
- as.owin.data.frame: New argument step
- as.polygonal:
  - Can now repair errors in polygon data, if repair=TRUE.
  - Accelerated when w is a pixel mask.
- as.psp: now permits a data frame of marks to have only one column, instead of coercing it to a vector.
- as.solist: The argument x can now be a spatial object; as.solist(cells) is the same as solist(cells).
- bdist.pixels: Accelerated for polygonal windows. New argument method.
- bdist.points: Accelerated for polygonal windows.
- beachcolours, beachcolourmap: Improved positioning of the yellow colour band.
- bind.fv: New argument clip.
- blur: New argument kernel.
- bw.abram:
  - New argument smoother determines how the pilot estimate is computed.
  - Formal arguments rearranged.
- bw.diggle, bw.ppl, bw.relrisk, bw.smoothppp:
  - These functions now extract and store the name of the unit of length from the point pattern dataset. When the bandwidth selection criterion is plotted, the name of the unit of length is shown on the x-axis.
  - A warning is issued if the optimal value of the cross-validation criterion occurs at an end-point of the search interval. New argument warn.
- bw.ppl:
  - New arguments weights and sigma.

  - New argument `shortcut` allows faster computation.
  - Additional arguments `...` are now passed to `density.ppp`.

- `bw.scott`:
  - the two bandwidth values in the result now have names `sigma.x` and `sigma.y`.
  - Now handles point patterns of any dimension.
  - New arguments `isotropic` and `d`.

- `bw.stoyan`: The rule has been modified so that, if the pattern is empty, it is now treated as if it contained 1 point, so that a finite bandwidth value is returned.

- `cbind.hyperframe`: The result now retains the `row.names` of the original arguments.

- `cdf.test`:
  - Calculations are more robust against numerical rounding effects.
  - The methods for classes `ppp`, `ppm`, `lpp`, `lppm`, `slrm` have a new argument `interpolate`.
  - Monte Carlo test runs much faster.
  - More jittering is applied when `jitter=TRUE`. Warnings about tied values should not occur any more.

- `cdf.test.mppm`:
  - Now handles Gibbs models.
  - Now recognises `covariate="x"` or `"y"`.

- `clarkevans`: The argument `correction="all"` is now recognised: it selects all the available options. [This is also the default.]

- `clickpoly`: The polygon is now drawn progressively as the user clicks new vertices.

- `closepairs.ppp`: New argument `periodic`.

- `closepairs.ppp`, `closepairs.pp3`:
  - New arguments `distinct` and `neat` allow more options.
  - Argument `ordered` has been replaced by `twice` (but `ordered` is still accepted, with a warning).
  - Performance improved (computation time and memory requirements reduced.) This should improve the performance of many functions in `spatstat`.

- `closepairs.pp3`: Argument `what` can take the value `"ijd"`

- `clusterset`: Improved behaviour.

- `clusterfit`:
  - New argument `algorithm` specifies the choice of optimisation algorithm.
  - Changed precedence rule for handling the algorithm parameters in the minimum contrast algorithm. Individually-named arguments `q,p,rmax,rmin` now take precedence over entries with the same names in the list `ctrl`.

  - New argument `verbose`.

- `colourmap`: argument `col` have have length 1, representing a trivial colour map in which all data values are mapped to the same colour.
- `collapse.fv`: This is now treated as a method for the `nlme` generic `collapse`. Its syntax has been adjusted slightly.
- `connected.im`: Now handles a logical-valued image properly. Arguments `...` now determine pixel resolution.
- `connected.owin`: Arguments `...` now determine pixel resolution.
- `contour.im`: New argument `col` specifies the colour of the contour lines. If `col` is a colour map, then the contours are drawn in different colours.
- `convolve.im`: the name of the unit of length is preserved.
- `crossing.psp`: New argument `details` gives more information about the intersections between the segments.
- `crosspairs.pp3`: Argument `what` can take the value `"ijd"`
- `cut.ppp`: Argument `z` can be `"x"` or `"y"` indicating one of the spatial coordinates.
- `dclf.test, mad.test, dclf.progress, mad.progress, dclf.sigtrace, mad.sigtrace, dg.progress, dg.sigtrace`:
  - New argument `clamp` determines the test statistic for one-sided tests.
  - New argument `rmin` determines the left endpoint of the test interval.
  - New argument `leaveout` specifies how to calculate discrepancy between observed and simulated function values.
  - New argument `scale` allows summary function values to be rescaled before the comparison is performed.
  - New argument `interpolate` supports interpolation of p-value.
  - New argument `interpolate` supports interpolation of critical value of test.
  - Function values which are infinite, `NaN` or `NA` are now ignored in the calculation (with a warning) instead of causing an error. Warning messages are more detailed.
- `default.rmhcontrol, default.rmhexpand`: New argument `w`.
- `density.lpp`:
  - New fast algorithm (up to 1000 times faster) for the default case where `kernel="gaussian"` and `continuous=TRUE`. Generously contributed by Greg McSwiggan.
  - Fast algorithm has been further accelerated.
  - New argument `kernel` specifies the smoothing kernel. Any of the standard one-dimensional smoothing kernels can be used.
  - Now supports both the 'equal-split continuous' and 'equal-split discontinuous' smoothers. New argument `continuous` determines the choice of smoother.
  - New arguments `weights` and `old`.

  - New argument `distance` offers a choice of different kernel methods.

- `density.ppp`:
  - A non-Gaussian kernel can now be specified using the argument `kernel`.
  - Argument `weights` can now be a pixel image.
  - Infinite bandwidth `sigma=Inf` is supported.
  - Accelerated by about 30% when `at="pixels"`.
  - Accelerated by about 15% in the case where `at="points"` and `kernel="gaussian"`.
  - Accelerated in the cases where weights are given or `diggle=TRUE`.
  - New argument `verbose`.

- `density.psp`:
  - New argument `method`.
  - Accelerated by 1 to 2 orders of magnitude.

- `dfbetas.ppm`:
  - For Gibbs models, memory usage has been dramatically reduced, so the code can handle larger datasets and finer quadrature schemes.
  - Increased the default resolution of the pixel images. Spatial resolution can now be controlled by the arguments `dimyx`, `eps`.

- `diagnose.ppm`:
  - Infinite values of `rbord` are now ignored and treated as zero. This ensures that `diagnose.ppm` has a sensible default when the fitted model has infinite reach.
  - Accelerated, when `type="inverse"`, for models without a hard core.

- `diagnose.ppm, plot.diagppm`:
  - New arguments `col.neg, col.smooth` control the colour maps.
  - Accelerated, when `type="inverse"`, for models without a hard core.

- `dilation.ppp`: Improved geometrical accuracy. Now accepts arguments to control resolution of polygonal approximation.

- `discs`:
  - Now accepts a single numeric value for `radii`.
  - New argument `npoly`.
  - Accelerated in some cases.

- `distfun`: When the user calls a distance function that was created by `distfun`, the user may now give a `ppp` or `lpp` object for the argument `x`, instead of giving two coordinate vectors `x` and `y`.

- `dppm`: Changed precedence rule for handling the algorithm parameters in the minimum contrast algorithm. Individually-named arguments `q,p,rmax,rmin` now take precedence over entries with the same names in the list `ctrl`.

- `duplicated.ppp`: accelerated.
- `edge.Trans`: New argument `gW` for efficiency.
- `effectfun`:
  - Now works for `ppm`, `kppm`, `lppm`, `dppm`, `rppm` and `profilepl` objects.
  - New argument `nvalues`.
- `envelope`:
  - New argument `clamp` gives greater control over one-sided envelopes.
  - New argument `funargs`
  - New argument `scale` allows global envelopes to have width proportional to a specified function of r, rather than constant width.
  - New argument `funYargs` contains arguments to the summary function when applied to the data pattern only.
  - The argument `simulate` can now be a function (such as `rlabel`). The function will be applied repeatedly to the original data pattern.
  - `rejectNA` and `silent`.
- `envelope.lpp`, `envelope.lppm`:
  - New arguments `fix.n` and `fix.marks` allow envelopes to be computed using simulations conditional on the observed number of points.
  - New arguments `maxnerr`, `rejectNA` and `silent`.
- `eval.im`: New argument `warn`.
- `eval.linim`: New argument `warn`.
- `ewcdf`:
  - Argument `weights` can now be `NULL`.
  - New arguments `normalise` and `adjust`.
  - Computation accelerated.
  - The result does not inherit class `"ecdf"` if `normalise=FALSE`.
- `Fest`: Additional checks for errors in input data.
- `Finhom`:
  - A warning is issued if bias is likely to occur because of undersmoothing.
  - New arguments `warn.bias` and `savelambda`.
- `fitted.lppm`: New argument `leaveoneout` allows leave-one-out computation of fitted value.
- `fitted.ppm`:
  - New option, `type="link"`.
  - New argument `ignore.hardcore`.

- `funxy`:
  - When the user calls a function that was created by `funxy`, the user may now give a `ppp` or `lpp` object for the argument `x`, instead of giving two coordinate vectors `x` and `y`.
  - Functions of class `"funxy"` can now be applied to quadrature schemes.
- `Geyer`: The saturation parameter `sat` can now be less than 1.
- `Ginhom`:
  - A warning is issued if bias is likely to occur because of undersmoothing.
  - New arguments `warn.bias` and `savelambda`.
- `grow.rectangle`: New argument `fraction`.
- `Hest`:
  - Argument `X` can now be a pixel image with logical values.
  - New argument `W`. [Based on code by Kassel Hingee.]
  - Additional checks for errors in input data.
- `hist.im`: New argument `xname`.
- `identify.psp`: Improved placement of labels. Arguments can be passed to `text.default` to control the plotting of labels.
- `idw`: Standard errors can now be calculated by setting `se=TRUE`.
- `imcov`: the name of the unit of length is preserved.
- `im.apply`:
  - Computation accelerated
  - New argument `fun.handles.na`
  - New argument `check`
- `influence.ppm`: For Gibbs models, memory usage has been dramatically reduced, so the code can handle larger datasets and finer quadrature schemes.
- `integral.linfun`:
  - New argument `delta` controls step length of approximation to integral.
  - Argument `domain` can be a tessellation.
- `integral.linim`: Argument `domain` can be a tessellation.
- `integral.ssf`: Argument `domain` can be a tessellation.
- `intensity.ppm`: Intensity approximation is now implemented for area-interaction model, and Geyer saturation model.
- `interp.im`: New argument `bilinear`.
- `ippm`:

  - Accelerated.
  - The internal format of the result has been extended slightly.
  - Improved defaults for numerical algorithm parameters.

- `Jinhom`:
  - A warning is issued if bias is likely to occur because of undersmoothing.
  - New arguments `warn.bias` and `savelambda`.

- `Kcross.inhom`, `Kdot.inhom`, `Kmulti.inhom`:
  - These functions now allow intensity values to be given by a fitted point process model.
  - New arguments `update`, `leaveoneout`, `lambdaX`.
  - Leave-one-out calculation is now implemented when `lambbdaX` is a fitted model of class `"dppm"`.

- `Kest`
  - Accelerated computation (for translation and rigid corrections) when window is an irregular shape.
  - Calculation of isotropic edge correction for polygonal windows has changed slightly. Results are believed to be more accurate. Computation has been accelerated by about 20 percent in typical cases.

- `Kest.fft`: Now has `...` arguments allowing control of spatial resolution.

- `Kinhom`:
  - New argument `ratio`.
  - Stops gracefully if `lambda` contains any zero values.
  - Leave-one-out calculation is implemented when `lambda` is a fitted model of class `"dppm"`.

- `kppm`:
  - Fitting a model with `clusters="LGCP"` no longer requires the package `RandomFields` to be loaded explicitly.
  - New argument `algorithm` specifies the choice of optimisation algorithm.
  - Left hand side of formula can now involve entries in the list `data`.
  - refuses to fit a log-Gaussian Cox model with anisotropic covariance.
  - A warning about infinite values of the summary function no longer occurs when the default settings are used. Also affects `mincontrast`, `cauchy.estpcf`, `lgcp.estpcf`, `matclust.estpcf`, `thomas.estpcf`, `vargamma.estpcf`.
  - Changed precedence rule for handling the algorithm parameters in the minimum contrast algorithm. Individually-named arguments `q,p,rmax,rmin` now take precedence over entries with the same names in the list `ctrl`.
  - Improved printed output.

- `latest.news`: Now prints news documentation for the current major version, by default. New argument `major`.

- `Lcross.inhom`, `Ldot.inhom`: These functions now allow intensity values to be given by a fitted point process model. New arguments `update`, `leaveoneout`, `lambdaX`.
- `lengths.psp`: New argument `squared`.
- `Lest`, `Linhom`, `Ldot`, `Lcross`, `Ldot.inhom`, `Lcross.inhom`: These summary functions now have explicit argument `"correction"`.
- `leverage.ppm`:
  - For Gibbs models, memory usage has been dramatically reduced, so the code can handle larger datasets and finer quadrature schemes.
  - Increased the default resolution of the pixel images. Spatial resolution can now be controlled by the arguments `dimyx`, `eps`.
- `leverage.ppm`, `influence.ppm`, `dfbetas.ppm`:
  - These methods now work for models that were fitted by logistic composite likelihood (`method='logi'`).
  - Computation has been vastly accelerated for models with Geyer interaction fitted using isotropic or translation edge corrections.
  - Faster computation in many cases.
  - Virtually all models and edge corrections are now supported, using a "brute force" algorithm. This can be slow in some cases.
- `linearK`, `linearpcf` and relatives:
  - substantially accelerated.
  - ratio calculations are now supported.
  - new argument `ratio`.
- `linearKinhom`: new argument `normpower`.
- `linearKinhom`, `linearpcfinhom`:
  - Changed behaviour when `lambda` is a fitted model.
  - New arguments `update` and `leaveoneout`.
- `linearpcf`: new argument `normpower`.
- `linim`:
  - The image `Z` is now automatically restricted to the network.
  - New argument `restrict`.
- `linnet`:
  - The internal format of a `linnet` (linear network) object has been changed. Existing datasets of class `linnet` are still supported. However, computation will be faster if they are converted to the new format. To convert a linnet object `L` to the new format, use `L <- as.linnet(L)`.

  - If the argument `edges` is given, then this argument now determines the ordering of the sequence of line segments. For example, the `i`-th row of `edges` specifies the `i`-th line segment in `as.psp(L)`.
  - New argument `warn`.
  - When argument `edges` is specified, the code now checks whether any edges are duplicated.

- `lintess`:
  - Argument `df` can be missing or `NULL`, resulting in a tesellation with only one tile.
  - Tessellations can now have marks. New argument `marks`.

- `localpcfinhom`:
  - New arguments `update` and `leaveoneout`.

- `logLik.ppm`:
  - New argument `absolute`.
  - The warning about pseudolikelihood ('log likelihood not available') is given only once, and is not repeated in subsequent calls, within a spatstat session.

- `logLik.mppm`: new argument `warn`.

- `lohboot`:
  - Algorithm has been corrected and extended thanks to Christophe Biscio and Rasmus Waagepetersen.
  - New arguments `block`, `basicboot`, `Vcorrection`.
  - Accelerated when the window is a rectangle.
  - Now works for multitype K functions `Kcross`, `Kdot`, `Lcross`, `Ldot`, `Kcross.inhom`, `Lcross.inhom`
  - Confidence bands for `Lest`, `Linhom`, `Lcross`, `Ldot`, `Lcross.inhom` are now computed differently. First a confidence band is computed for the corresponding K function `Kest`, `Kinhom`, `Kcross`, `Kdot`, `Kcross.inhom` respectively. Then this is transformed to a confidence band for the L function by applying the square root transformation.

- `lpp`:
  - The internal format of an `lpp` object has been changed. Existing datasets of class `lpp` are still supported. However, computation will be faster if they are converted to the new format. To convert an `lpp` object `X` to the new format, use `X <- as.lpp(X)`.
  - `X` can be missing or `NULL`, resulting in an empty point pattern.

- `lpp`, `as.lpp`: These functions now handle the case where coordinates `seg` and `tp` are given but `x` and `y` are missing.

- `lppm`:
  - New argument `random` controls placement of dummy points.
  - Computation accelerated.

- `lurking.ppm`: accelerated.

- `lut`: argument `outputs` may have length 1, representing a lookup table in which all data values are mapped to the same output value.
- `markcorr`: New argument `weights` allows computation of the weighted version of the mark correlation function.
- `mppm`:
  - Now handles models with a random effect component. (This is covered in [2, Chap. 16].)
  - New argument `random` is a formula specifying the random effect. (This is covered in [2, Chap. 16].)
  - Performs more checks for consistency of the input data.
  - New arguments `gcontrol` and `reltol.pql` control the fitting algorithm.
  - New argument `weights` specifies case weights for each row of data.
- `msr`: Infinite and `NA` values are now detected (if `check=TRUE`) and are reset to zero, with a warning.
- `nbfires`:
  - the unit of length for the coordinates is now specified in this dataset.
  - This dataset now includes information about the different land and sea borders of New Brunswick.
- `nncorr,nnmean,nnvario`: New argument `na.action`.
- `nndist.lpp, nnwhich.lpp, nncross.lpp, distfun.lpp`: New argument `k` allows computation of k-th nearest point. Computation accelerated.

  `nnfun.lpp`:
  - New argument `k`.
  - New argument `value` specifies whether to return the index of the nearest neighbour or the mark value of the nearest neighbour.

  `nnfun.ppp`:
  - New argument `value` specifies whether to return the index of the nearest neighbour or the mark value of the nearest neighbour.

  `nnfun.psp`:
  - New argument `value` specifies whether to return the index of the nearest neighbour or the mark value of the nearest neighbour.
- `padimage`: New argument `W` allows an image to be padded out to fill any window.
- `pairorient`: Default edge corrections now include `"bord.modif"`.
- `parres`: the argument `covariate` is allowed to be missing if the model only depends on one covariate.
- `pcf.ppp`:

- New argument `close` for advanced use.
- New argument `ratio` allows several estimates of pcf to be pooled.
- Now calculates an analytic approximation to the variance of the estimate of the pair correlation function (when `var.approx=TRUE`).
- Now returns the smoothing bandwidth used, as an attribute of the result.
- New argument `close` for advanced use.
- Now accepts `correction="none"`.

- `pcfinhom`:
  - New argument `close` for advanced use.
  - Default behaviour is changed when `lambda` is a fitted model. The default is now to re-fit the model to the data before computing pcf. New arguments `update` and `leaveoneout` control this.
  - New argument `close` for advanced use.
  - Now handles `correction="good"`
  - Leave-one-out calculation is implemented when `lambda` is a fitted model of class `"dppm"`.

- `persp.funxy`: Improved z-axis label.

- `pixellate.ppp`:
  - If the pattern is empty, the result is an integer-valued image (by default) for consistency with the results for non-empty patterns.
  - Accelerated in the case where weights are given.
  - New arguments `fractional` and `preserve` for more accurate discretisation.
  - New argument `savemap`.

- `plot.anylist`:
  - If a list entry `x[[i]]` belongs to class `"anylist"`, it will be expanded so that each entry `x[[i]][[j]]` will be plotted as a separate panel.
  - New arguments `panel.begin.args`, `panel.end.args`
  - Result is now an (invisible) list containing the result from executing the plot of each panel.

- `plot.colourmap`:
  - Now handles a colour map for a zero-length interval [a,a]
  - New argument `increasing` specifies whether the colours are displayed in order left-to-right/bottom-to-top.
  - Changed default behaviour for discrete colour maps when `vertical=FALSE`.

- `plot.im`:
  - Now handles complex-valued images.
  - New argument `workaround` to avoid a bug in some MacOS device drivers that causes the image to be displayed in the wrong spatial orientation.

  - The number of tick marks in the colour ribbon can now be controlled using the argument `nint` in `ribargs`.
  - Improved behaviour when all pixel values are `NA`.
  - Improved handling of tickmarks on colour ribbon.
  - Improved behaviour when the image values are almost constant.
  - New argument `riblab`.
  - Axes are prevented from extending outside the image rectangle.
  - New argument `zap`.
  - Some warnings are suppressed when `do.plot=FALSE`.

- `plot.imlist`: Result is now an (invisible) list containing the result from executing the plot of each panel.

- `plot.influence.ppm`: New argument `multiplot`.

- `plot.kppm`:
  - New arguments `pause` and `xname`.
  - The argument `what="all"` is now recognised: it selects all the available options. [This is also the default.]

- `plot.leverage.ppm`:
  - New arguments `multiplot` and `what`.
  - A contour line showing the average value of leverage is now drawn on the colour ribbon, as well as on the main image. New argument `args.contour`.

- `plot.linfun`:
  - Now passes arguments to the function being plotted.
  - A scale bar is now plotted when `style="width"`.
  - New argument `legend`.
  - The return value has a different format.

- `plot.linim`:
  - The return value has a different format.
  - A scale bar is now plotted when `style="width"`.
  - When `style="width"`, negative values are plotted in red (by default). New argument `negative.args` controls this.
  - New argument `zlim` specifies the range of values to be mapped.
  - New explicit argument `box` determines whether to plot a bounding box; default is `FALSE` in all cases.

- `plot.lintess`:
  - Improved plot method, with more options.
  - Modified to display the marks attached to the tiles.

  - Options: `style=c("colour", "width", "image")`.

- `plot.lpp`:
  - New argument `show.network`.
  - For a point pattern with continuous marks ("real numbers") the colour arguments `cols`, `fg`, `bg` can now be vectors of colour values, and will be used to determine the default colour map for the marks.
- `plot.mppm`: New argument `se`.
- `plot.msr`:
  - Now handles multitype measures.
  - New argument `multiplot`.
  - New argument `massthresh`.
  - New arguments `equal.markscale` and `equal.ribbon`.
- `plot.onearrow`: Graphical parameters, specified when the object was created, are now taken as the defaults for graphical parameters to the plot.
- `plot.psp`:
  - Segments can be plotted with widths proportional to their mark values.
  - New argument `style`.
  - New argument `col` gives control over the colour map representing the values of marks attached to the segments.
- `plot.pp3`: New arguments `box.front`, `box.back` control plotting of the box.
- `plot.ppp`:
  - The default colour for the points is now a transparent grey, if this is supported by the plot device.
  - For a point pattern with continuous marks ("real numbers") the colour arguments `cols`, `fg`, `bg` can now be vectors of colour values, and will be used to determine the default colour map for the marks.
  - Now recognises graphics parameters for text, such as `family` and `srt`
  - When `clipwin` is given, any parts of the boundary of the window of `x` that lie inside `clipwin` will also be plotted.
  - Improved placement of symbol map legend when argument `symap` is given.
- `plot.tess`:
  - This plot method can now fill each tile with a different colour.
  - New arguments `do.col`, `values`, `col` and `ribargs`. Old argument `col` has been renamed `border` for consistency.
  - Now generates a separate plot panel for each column of marks, if `do.col=TRUE`.
  - New argument `multiplot`.

- `plot.profilepl` ,`plot.quadratcount`, `plot.quadrattest`, `plot.tess`: Now recognise graphics parameters for text, such as `family` and `srt`
- `plot.solist`:
  - New arguments `panel.begin.args`, `panel.end.args`
  - Result is now an (invisible) list containing the result from executing the plot of each panel.
- `plot.studpermutest`: This existing function now has a help file.
- `plot.symbolmap`: New argument `nsymbols` controls the number of symbols plotted.
- `ponderosa`: In this installed dataset, the function `ponderosa.extra$plotit` has changed slightly (to accommodate the dependence on the package `spatstat.utils`).
- `polynom`: This function now has a help file.
- `pool.fv`:
  - The default plot of the pooled function no longer includes the variance curves.
  - New arguments `relabel` and `variance`.
- `pool.rat`: New arguments `weights`, `relabel` and `variance`.
- `ppm`:
  - Argument `interaction` can now be a function that makes an interaction, such as `Poisson`, `Hardcore`, `MultiHard`.
  - Argument `subset` can now be a window (class `"owin"`) specifying the sub-region of data to which the model should be fitted.
- `ppm.ppp, ppm.quad`:
  - New argument `emend`, equivalent to `project`.
  - New arguments `subset` and `clipwin`.
- `ppmInfluence`: The result now belongs to class `ppmInfluence`, for which there are methods for `leverage`, `influence`, `dfbetas` which extract the desired component.
- `ppp`:
  - New argument `checkdup`.
  - If the coordinate vectors `x` and `y` contain `NA`, `NaN` or infinite values, these points are deleted with a warning, instead of causing a fatal error.
- `pp3`: New argument `marks`.
- `predict.kppm, residuals.kppm` Now issues a warning when the calculation ignores the cluster/Cox component and treats the model as if it were Poisson. (This currently happens in predict.kppm when se=TRUE or interval != "none", and in residuals.kppm when type != "raw").
- `predict.lppm`: Argument `locations` can now be an `lpp` object.
- `predict.mppm`: The argument `type="all"` is now recognised: it selects all the available options. [This is also the default.]

- `predict.ppm`:
  - Now recognises the arguments `dimyx` and `eps` for specifying the resolution of the grid of prediction points.
  - New argument `ignore.hardcore`.
  - Accelerated for models fitted with `method="VBlogi"`
- `predict.rhohat`: New argument `what` determines which value should be calculated: the function estimate, the upper/lower confidence limits, or the standard error.
- `print.linim`: More information is printed.
- `print.lintess`: Output includes information about marks.
- `print.quad`: More information is printed.
- `print.rmhmodel`: More information is printed.
- `progressreport`
  - Behaviour improved.
  - New arguments `state`, `tick`, `showtime`.
  - New option: `style="tk"`
- `pseudoR2.ppm`, `pseudoR2.lppm`:
  - The null model now includes any offset terms, by default.
  - New argument `keepoffset`.
- `quadratcount.ppp`: Computation accelerated in some cases.
- `quadrat.test.ppm`: Computation accelerated in some cases.
- `quantess`:
  - The covariate `Z` can now be `"rad"` or `"ang"` representing polar coordinates.
  - New argument `origin` specifies the origin of polar coordinates.
  - New argument `eps` controls the accuracy of the calculation.
- `quantile.ewcdf`: The function is now normalised to the range `[0,1]` before the quantiles are computed. This can be suppressed by setting `normalise=FALSE`.
- `qqplot.ppm` Argument `expr` can now be a list of point patterns, or an envelope object containing a list of point patterns.
- `rbind.hyperframe`: The result now retains the `row.names` of the original arguments.
- `rcellnumber`: New argument `mu`.
- `rebound.owin`: Now preserves unitnames of the objects.
- `rescale.owin`, `rescale.ppp`, `rescale.psp`: The geometrical type of the window is now preserved in all cases. (Previously if the window was polygonal but was equivalent to a rectangle, the rescaled window was a rectangle.)

- rgbim, hsvim: New argument A controls the alpha (transparency) channel.
- rgb2hex, col2hex, paletteindex, is.colour, samecolour, complementarycolour, is.grey, to.grey These colour tools now handle transparent colours.
- rgb2hex: New argument maxColorValue
- relrisk.ppp:
  - If se=TRUE and at="pixels", the result belongs to class solist.
  - The arguments adjust, edge, diggle are now explicit formal arguments.

  rhohat:
  - Nonparametric maximum likelihood estimation is now supported, assuming the intensity is a monotone function of the covariate.
  - New options smoother="increasing" and smoother="decreasing".
  - New argument subset allows computation for a subset of the data.
  - New argument positiveCI specifies whether confidence limits should always be positive.

  rhohat.lpp: New argument random controls placement of dummy points.
- rlabel:
  - New arguments nsim and drop.
  - X can now be a point pattern of any type (ppp, lpp, pp3, ppx) or a line segment pattern (psp).
- rLGCP:
  - Accelerated.
  - This function no longer requires the package RandomFields to be loaded explicitly.
- rMaternI, rMaternII: These functions can now generate random patterns in three dimensions and higher dimensions, when the argument win is of class box3 or boxx.
- rmh: Accelerated, in the case where multiple patterns are saved using nsave.
- rmh.ppm, rmhmodel.ppm, simulate.ppm: A model fitted using the Penttinen interaction can now be simulated.
- rmh.default, rmhmodel.default:
  - These functions now recognise cif='penttinen' for the Penttinen interaction.
  - New arguments nsim, saveinfo.
- rmhcontrol:
  - New parameter pstage determines when to generate random proposal points.
  - The parameter nsave can now be a vector of integers.
- rose.default New argument weights.

- `rose` New arguments `start` and `clockwise` specify the convention for measuring and plotting angles.
- `rotmean`:
  - New argument `padzero`.
  - Default behaviour has changed.
  - Improved algorithm stability.
- `rpoispp`: Accelerated, when `lambda` is a pixel image.
- `rpoisppx`: New argument `drop`.
- `rpoisline`: Also returns information about the original infinite random lines.
- `rStrauss, rHardcore, rStraussHard, rDiggleGratton, rDGS, rPenttinen:` New argument `drop`.
- `rtemper:` new argument `track`.
- `rthin`
  - Accelerated, when `P` is a single number.
  - `X` can now be a point pattern of any type (`ppp`, `lpp`, `pp3`, `ppx`) or a line segment pattern (`psp`).
- `rThomas, rMatClust, rCauchy, rVarGamma`:
  - When the model is approximately Poisson, it is simulated using rpoispp. This avoids computations which would require huge amounts of memory. New argument `poisthresh` controls this behaviour.
  - New argument `saveparents`.
- `runifpointOnLines`, `rpoisppOnLines`: New argument `drop`.
- `runifpointx`: New argument `drop`.
- `selfcut.psp`:
  - Computation accelerated.
  - The result now has an attribute `"camefrom"` indicating the provenance of each segment in the result.
- `setcov`: the name of the unit of length is preserved.
- `shapley`: In this installed dataset, the function `shapley.extra$plotit` has changed slightly (to accommodate the dependence on the package `spatstat.utils`).
- `shift.im`, `shift.owin`, `shift.ppp`, `shift.psp`: More options for the argument `origin`.
- Simulation: Several basic simulation algorithms have been accelerated. Consequently, simulation outcomes are not identical to those obtained with previous versions of `spatstat`, even when the same random seed is used. To ensure compatibility with previous versions of spatstat, revert to the slower code by setting `spatstat.options(fastthin=FALSE, fastpois=FALSE)`.

- `simulate.kppm`:
  - Accelerated for LGCP models.
  - Additional arguments `...` are now passed to the function that performs the simulation.
- `simulate.ppm`: New argument `w` controls the window of the simulated patterns. New argument `verbose`.
- `Smooth.ppp`:
  - A non-Gaussian kernel can now be specified using the argument `kernel`.
  - Argument `weights` can now be a pixel image.
  - Infinite bandwidth `sigma=Inf` is supported.
  - Accelerated by about 30% in the case where `at="pixels"`.
  - Accelerated by about 15% in the case where `at="points"` and `kernel="gaussian"`.
  - Now exits gracefully if any mark values are `NA`, `NaN` or `Inf`.
  - New argument `geometric` supports geometric-mean smoothing.
  - The arguments `adjust`, `edge`, `diggle` and `kernel` are now explicit formal arguments.
- `solist`: New argument `.NameBase`
- `spatialcdf`:
  - Computation accelerated.
  - The result does not inherit class `"ecdf"` if `normalise=FALSE`.
- `spatstat.options` New options `fastthin` and `fastpois` enable fast simulation algorithms. Set these options to `FALSE` to reproduce results obtained with previous versions of `spatstat`.
- `split.ppp`, `split.ppx`: The splitting variable `f` can now be a logical vector.
- `square`: Handles a common error in the format of the arguments.
- `step`: now works for models of class `"mppm"`.
- `stieltjes`: Argument `M` can be a stepfun object (such as an empirical CDF).
- `subset.ppp`, `subset.lpp`, `subset.pp3`, `subset.ppx`: The argument `subset` can now be any argument acceptable to the `"["` method.
- summary functions The argument `correction="all"` is now recognised: it selects all the available options.

  This applies to `Fest`, `F3est`, `Gest`, `Gcross`, `Gdot`, `Gmulti`, `G3est`, `Gfox`, `Gcom`, `Gres`, `Hest`, `Jest`, `Jmulti`, `Jcross`, `Jdot`, `Jfox`, `Kest`, `Kinhom`, `Kmulti`, `Kcross`, `Kdot`, `Kcom`, `Kres`, `Kmulti.inhom`, `Kcross.inhom`, `Kdot.inhom`, `Kscaled`, `Ksector`, `Kmark`, `K3est`, `Lscaled`, `markcorr`, `markcrosscorr`, `nnorient`, `pairorient`, `pcfinhom`, `pcfcross.inhom`, `pcfcross`, `pcf`, `Tstat`.
- `Summary.linim` family supporting `range`, `max`, `min` etc: Recognises the argument `finite` so that `range(x, finite=TRUE)` works for a linim object `x`.

- summary.distfun, summary.funxy:
  - More information is printed.
  - Pixel resolution can now be controlled.
- summary.kppm: prints more information about algorithm convergence.
- summary.lintess: prints information about marks.
- summary.ppm: New argument fine selects the algorithm for variance estimation.
- summary.owin, summary.im: The fraction of frame area that is occupied by the window/image is now reported.
- sumouter: New argument y allows computation of asymmetric outer products.
- symbolmap:
  - Now accepts a vector of colour values for the arguments col, cols, fg, bg if the argument range is given.
  - New option: shape="arrows".
- tess: Argument window is ignored when xgrid, ygrid are given.
- texturemap: Argument textures can be missing or NULL.
- textureplot: Argument x can now be something acceptable to as.im.
- tilenames, tilenames<-: These functions are now generic, with methods for tess and lintess.
- to.grey New argument transparent.
- union.owin: Improved behaviour when there are more than 2 windows.
- unstack.lintess: now handles marks.
- update: now works for models of class "mppm".
- update.kppm:
  - New argument evaluate.
  - Now handles additional arguments in any order, with or without names.
  - Changed arguments.
  - Improved behaviour.
- update.ppm: For the case update(model, X) where X is a point pattern, if the window of X is different from the original window, then the model is re-fitted from scratch (i.e. use.internal=FALSE).
- valid.ppm This is now a method for the generic function valid.
- vcov.mppm:
  - Now handles models with Gibbs interactions.
  - New argument nacoef.action specifies what to do if some of the fitted coefficients are NA, NaN or Inf.

- `vcov.ppm`:
  - Performance slightly improved, for Gibbs models.
  - New argument `nacoef.action` specifies what to do if some of the fitted model coefficients are `NA`, `NaN` or infinite.
- `[<-.im`
  - Accepts an array for `value`.
  - The subset index `i` can now be a linear network. Then the result of `x[i, drop=FALSE]` is a pixel image of class `linim`.
  - New argument `drop` controls behaviour when indices are missing as in `x[] <- value`
- `[.layered`:
  - Subset index `i` can now be an `owin` object.
  - Additional arguments `...` are now passed to other methods.
- `[.leverage.ppm`: New argument `update`.
- `[.linnet`:
  - New argument `snip` determines what to do with segments of the network that cross the boundary of the window. Default behaviour has changed.
  - More robust against artefacts when the subset index is a pixel mask.
- `[.linim`: More robust against artefacts.
- `[.lpp`: New argument `snip` determines what to do with segments of the network that cross the boundary of the window. Default behaviour has changed.
- `[.ppx`: The subset index `i` may now be a spatial domain of class `boxx` or `box3`.
- `[.ppp` New argument `clip` determines whether the window is clipped.
- `[.ppp` The previously-unused argument `drop` now determines whether to remove unused levels of a factor.
- `[.pp3`, `[.lpp`, `[.ppx`, `subset.ppp`, `subset.pp3`, `subset.lpp`, `subset.ppx`: These methods now have an argument `drop` which determines whether to remove unused levels of a factor.
- `[.psp`:
  - accelerated.
  - New argument `fragments` specifies whether to keep fragments of line segments that are cut by the new window, or only to retain segments that lie entirely inside the window.
- `[.solist`: Subset index `i` can now be an `owin` object.

References

[1] A. Baddeley. Analysing spatial point patterns in R. Technical report, CSIRO, 2010. Version 4. URL `https://research.csiro.au/software/r-workshop-notes/`

[2] A. Baddeley, E. Rubak, and R. Turner. *Spatial Point Patterns: Methodology and Applications with R.* Chapman & Hall/CRC Press, 2015.