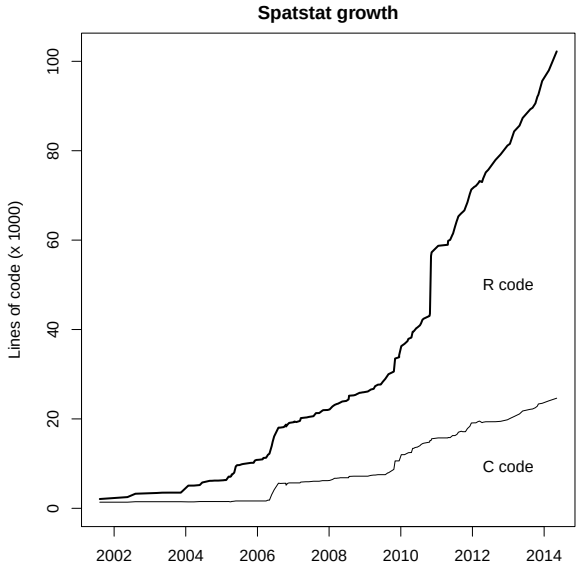
Summary of recent updates to `spatstat`

Adrian Baddeley, Rolf Turner and Ege Rubak

For `spatstat` version `1.37-0`

This is a summary of changes that have been made to the `spatstat` package since 2010, when the last set of workshop notes [1] was published. Those notes covered `spatstat` version `1.21-2`.

The current version of `spatstat` is `1.37-0`. `Spatstat` has almost tripled in size since 2010, and now contains over 100,000 lines of code.



Contents

1 Overview

Following is an overview of the most important changes.

- `spatstat` and all its dependencies are now Free Open Source.
- polygon clipping is now performed by the Free Open Source library `polyclip`. The library `gpclib` is no longer used.
- `spatstat` now 'Imports' (rather than 'Depends' on) the libraries `mgcv`, `deldir`, `abind`, `tensor`, `polyclip`. This means that these libraries are not accessible to the user unless the user explicitly loads them by typing `library(mgcv)` and so on.
- polygon calculations are now enabled, always.
- geometrical errors in polygon data are now repaired automatically.
- Improvements to Gibbs models (`ppm`):
  - now accepts syntax `ppm(X ~ Z)` similar to `lm` and `glm`, as well as old syntax `ppm(X, ~Z)`
  - Covariates no longer have to be passed in the argument `covariates` but can be objects in the R session;
  - new 'logistic likelihood' method;
  - variance-covariance matrix;
  - print method shows standard errors and CI for coefficients;
  - new interactions (`Triplets`, `Concom`);
  - hybrids of interactions;
  - self-starting feature of interactions;
  - better control over dummy points;
  - improved quadrature schemes (removing a source of bias);
  - spatial covariate functions can depend on additional parameters;
  - extract interaction from a fitted model;
  - check whether model is valid;
  - project to a valid model;
  - calculate *intensity* of fitted model;
  - calculate K-function and pair correlation function of fitted model;
  - confidence intervals using `confint`;
  - improvements/extensions to `rmh`;
  - exact simulation for many models;
  - predict conditional intensity for another point pattern
  - compute confidence intervals and prediction intervals for intensity, and for total number of points in a region.
  - covariates can be tessellations.
- Diagnostics for Gibbs models: Leverage, influence, partial residuals, score residuals, residual summary functions (`Kcom`, `Kres`, `Gcom`, `Gres`, `psst`, `psstA`, `psstG`, `compareFit`)

- Improvements to cluster/Cox models (`kppm`):
  - now accepts syntax `kppm(X ~ Z)` similar to `lm` and `glm`, as well as old syntax `kppm(X, ~Z)`
  - Covariates no longer have to be passed in the argument `covariates` but can be objects in the R session;
  - Guan's second order composite likelihood;
  - new cluster types `"Cauchy"`, `"VarGamma"`;
  - log-Gaussian Cox processes;
  - fitted intensity;
  - variance-covariance of trend parameters;
  - generic support e.g. `coef`, `model.matrix`;
  - conversion e.g. `as.fv`
  - extract K function (`Kmodel`, `pcfmodel`);
  - more control over trend (more arguments passed to `ppm`);
  - confidence intervals using `confint`
  - covariates can be tessellations.
- Support for point patterns on a linear network, including geometrically-corrected linear K function, pair correlation function, point process models, envelopes, tests.
- Support for replicated point patterns: An object of class `mppm` represents a point process model fitted to several point patterns. The patterns may be treated as independent replicate observations of the same model, or as the responses from different units in a designed experiment. Methods for this class include `plot`, `print`, `summary`, `fitted`, `predict`, `simulate`, ...
- Nonparametric/summary statistics e.g.
  - `rhohat` (nonparametric estimate of intensity as function of covariate)
  - `idw` (inverse distance weighted smoothing)
  - `convolve.im` (image convolution)
  - k-th nearest neighbours (`nndist`, `nncross`, `nnwhich`)
  - more methods for `intensity`.
- Statistical tools e.g.
  - several algorithms for bandwidth selection (`bw.*`)
  - bootstrap confidence intervals for summary functions (`lohboot`)
  - pooling operations (`pool.*`)
- Changes to `ppx` format: An object of class `ppx` may now include 'local' coordinates as well as 'spatial' and 'temporal' coordinates, and marks.
- Improvements to plots :
  - colour map objects
  - graphics symbol map objects
  - texture map objects

  - automatic legend in `plot.ppp`
  - collision avoidance in `plot.fv`
  - `plot.im(log=TRUE)`
  - `plot.listof(equal.ribbon=TRUE)`
  - improved labels in `plot.fv`
  - Switch all plots from colour to greyscale by setting `spatstat.options(monochrome=TRUE)`
- simulation:
  - `rmh` interface restructured
  - `rmh` can save transition history
  - visual debugger for `rmh`
  - exact simulation for many models
  - quasirandom point patterns and sequences
- envelopes: Extended functionality; Envelopes can be pooled and transformed.
- Many, many, many bug fixes.
- Substantially increased performance.
- data in `spatstat` are now lazy-loaded, so you don't have to type `data(amacrine)`, etc.
- user interrupt: Long calculations in `spatstat` now respond to the Interrupt/Stop signal.
- vignettes: New vignettes 'Getting Started with the Spatstat Package', 'Analysing Replicated Point Patterns in Spatstat' and 'Summary of Recent Changes to Spatstat'
- demonstrations: New demonstration `demo(sumfun)` shows the nonparametric summary functions in `spatstat`.

2 New classes

Important new classes of objects are listed here.

- `linnet`: An object of class '`linnet`' represents a linear network, i.e. a connected network of line segments, such as a road network. Methods for this class include plot, print, summary etc.
- `lpp`: An object of class '`lpp`' represents a point pattern on a linear network, such as a record of the locations of road accidents on a road network. Methods for this class include plot, print, summary etc.
- `lppm`: An object of class '`lppm`' represents a point process model fitted to a point pattern on a linear network. Methods for this class include plot, print, summary, fitted, predict, simulate, ...
- `linim`: Pixel image on a linear network.
- `linfun` Function defined on a linear network (e.g. a distance function)

- `mppm`: An object of class '`mppm`' represents a point process model fitted to several point pattern datasets. The patterns may be treated as independent replicate observations of the same model, or as the responses from different units in a designed experiment. Methods for this class include plot, print, summary, fitted, predict, simulate, ...
- `rhohat`: Nonparametric estimate of intensity as function of covariate (result of '`rhohat`'). Can be used as an R function. Methods include `predict`
- `layered` : A simple mechanism for controlling plots that consist of several successive layers of data.
- `timed`: A simple mechanism for recording the computation time taken.
- `rat`: 'Ratio objects' (suitable for pooling)
- `objsurf`: An object of class '`objsurf`' contains values of the likelihood or objective function in a neighbourhood of the maximum.
- `simplepanel`: An object of class '`simplepanel`' represents a spatial arrangement of buttons that respond to mouse clicks, supporting a simple, robust graphical interface.
- `symbolmap`: An object of class '`symbolmap`' represents a mapping from data values to graphical symbols. This makes it easy to ensure that the *same* graphics map is used in two different plots. The graphics map can also be plotted in its own right as a legend to the plots.
- `texturemap`: An object of class '`texturemap`' represents a mapping from data values to graphical textures.

3 New datasets

The following datasets have been added to the package.

- `pyramidal`: Diggle-Lange-Benes data on pyramidal neurons in cingulate cortex. 31 point patterns divided into 3 groups.
- `waterstriders`: Nummelin-Penttinen waterstriders data. Three independent replicates of a point pattern formed by insects.
- `simba`: Simulated data
- `demohyper`: Simulated data example for `mppm`. Point patterns and pixel image covariates, in two groups with different regression coefficients.
- `clmfires`: Forest fires in Castilla-La Mancha
- `gordon`: People sitting on the grass in Gordon Square, London
- `hyytiala`: Mixed forest in Hyytiala, Finland (marked by species)
- `paracou`: Kimboto trees in Paracou, French Guiana (marked as adult/juvenile)
- `waka`: Trees in Waka national park (marked with diameters)
- `mucosa`: Cells in gastric mucosa

- `gorillas`: Gorilla nest sites in a National Park in Cameroon.
- `chicago`: Street crimes in the University district of Chicago. A point pattern on a linear network.
- `flu`: Spatial point patterns giving the locations of influenza virus proteins on cell membranes. Kindly released by Dr George Leser and Dr Robert Lamb.
- `simplenet`: Simple example of a linear network

4 New Functions

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

- `split.hyperframe`: method for `split` for hyperframes.
- `complementarycolour`: Compute the complementary colour value of a given colour value, or the complementary colour map of a given colour map.
- `dmixpois`, `pmixpois`, `qmixpois`, `rmixpois`: (log-)normal mixture of Poisson distributions.
- `gauss.hermite`: Gauss-Hermite quadrature approximation to the expectation of any function of a normally-distributed random variable.
- `minnndist`, `maxnndist`: faster ways to compute `min(nndist(X))` and `max(nndist(X))`
- `boundingbox`: Generic function, replaces `bounding.box`.
- `Smoothfun`: create a `function(x,y)` equivalent to the result of `Smooth.ppp`
- `vdCorput`, `Halton`, `Hammersley`, `rQuasi`: quasirandom point patterns and sequences.
- `ellipse`: create an elliptical window.
- `clickbox`: interactively specify a rectangle by point-and-click on a graphics device.
- `symbolmap`: Create a graphics symbol map (a mapping from data values to graphical symbols)
- `update.symbolmap`: Modify a graphics symbol map
- `invoke.symbolmap`: Apply graphics symbol map to data values, and plot the symbols
- `plot.symbolmap`: Plot the graphics symbol map in the style of a legend
- `add.texture`: Draw a simple texture inside a specified region.
- `textureplot`: Display a factor-valued pixel image using texture fill.
- `texturemap`: Create a graphics texture map (a mapping from data values to graphical textures)
- `plot.texturemap`: Plot the graphics texture map in the style of a legend
- `reload.or.compute`: Utility for use in R scripts. Either reload results from file, or compute them.
- `im.apply`: Apply a function to corresponding pixel values in several images.
- `hexgrid`, `hextess`: Create a hexagonal grid of points, or a tessellation of hexagonal tiles

- `shift.tess`, `rotate.tess`, `reflect.tess`, `scalardilate.tess`, `affine.tess`: Apply a geometrical transformation to a tessellation.
- `quantile.ewcdf`: Extract quantiles from a weighted cumulative distribution function.
- `scanLRTS`: Evaluate the spatially-varying test statistic for the scan test.
- `intensity.splitppp`: Estimate intensity in each component of a split point pattern.
- `intensity.quadratcount`: Use quadrat counts to estimate intensity in each quadrat.
- `as.owin.quadratcount`, `as.owin.quadrattest`: Extract the spatial window in which quadrat counts were performed.
- `mppm`: Fit a Gibbs model to several point patterns. The point pattern data may be organised as a designed experiment and the model may depend on covariates associated with the design.
- `anova.mppm`: Analysis of Deviance for models of class `mppm`
- `coef.mppm`: Extract fitted coefficients from a model of class `mppm`
- `fitted.mppm`: Fitted intensity or conditional intensity for a model of class `mppm`
- `kstest.mppm`: Kolmogorov-Smirnov test of goodness-of-fit for a model of class `mppm`
- `logLik.mppm`: log likelihood or log pseudolikelihood for a model of class `mppm`
- `plot.mppm`: Plot the fitted intensity or conditional intensity of a model of class `mppm`
- `predict.mppm`: Compute the fitted intensity or conditional intensity of a model of class `mppm`
- `quadrat.test.mppm`: Quadrat counting test of goodness-of-fit for a model of class `mppm`
- `residuals.mppm`: Point process residuals for a model of class `mppm`
- `subfits`: Extract point process models for each individual point pattern dataset, from a model of class `mppm`.
- `vcov.mppm`: Variance-covariance matrix for a model of class `mppm`
- `integral.msr`: Integral of a measure.
- `objsurf`: For a model fitted by optimising an objective function, this command computes the objective function in a neighbourhood of the optimal value.
- `contour.objsurf`, `image.objsurf`, `persp.objsurf`, `plot.objsurf`: Plot an 'objsurf' object.
- `fvnames`: Define groups of columns in a function value table, for use in `plot.fv`, etc
- `multiplicity`: New generic function for which `multiplicity.ppp` is a method.
- `unique.ppx`, `duplicated.ppx`, `multiplicity.ppx`: Methods for `unique`(), `duplicated`() and `multiplicity`() for 'ppx' objects. These also work for 'pp3' and `lpp` objects.
- `closepairs`, `crosspairs`, `closepaircounts`, `crosspaircounts`: Low-level functions for finding all close pairs of points

- `nnfun.ppp`, `distfun.ppp`: New argument `k` allows these functions to compute k-th nearest neighbours.
- `nndensity`: Estimate point process intensity using k-th nearest neighbour distances
- `simplepanel`, `run.simplepanel`: Support for a simple point-and-click interface for general use.
- `as.lpp`: Convert data to a point pattern on a linear network.
- `distfun.lpp`: Distance function for point pattern on a linear network.
- `eval.linim`: Evaluate expression involving pixel images on a linear network.
- `linearKcross`, `linearKdot`, `linearKcross.inhom`, `linearKdot.inhom`: Multitype K functions for point patterns on a linear network
- `linearmarkconnect`, `linearmarkequal`: Mark connection function and mark equality function for multitype point patterns on a linear network
- `linearpcfcross`, `linearpcfdot`, `linearpcfcross.inhom`, `linearpcfdot.inhom`: Multitype pair correlation functions for point patterns on a linear network
- `linfun`: New class of functions defined on a linear network
- `nndist.lpp`, `nnwhich.lpp`, `nncross.lpp`: Methods for `nndist`, `nnwhich`, `nncross` for point patterns on a linear network
- `nnfun.lpp`: Method for `nnfun` for point patterns on a linear network
- `vcov.lppm`: Variance-covariance matrix for parameter estimates of a fitted point process model on a linear network.
- `bilinearform` Computes a bilinear form
- `tilenames`, `tilenames<-`: Extract or change the names of tiles in a tessellation.
- `bw.ppl`: Likelihood cross-validation technique for bandwidth selection in kernel smoothing.
- `is.lppm`, `is.kppm`, `is.slrm`: Tests whether an object is of class 'lppm', 'kppm' or 'slrm'
- `Smooth`: New generic function for spatial smoothing.
- `Smooth.ppp`, `Smooth.fv`, `Smooth.msr`: Methods for `Smooth` (identical to `smooth.ppp`, `smooth.fv`, `smooth.msr` respectively)
- `fitted.kppm`: Method for `fitted` for cluster/Cox models
- `nncross.pp3`: Method for `nncross` for point patterns in 3D
- `nnmark`: Mark of nearest neighbour - can be used for interpolation
- `dclf.progress`, `mad.progress`: Progress plots (envelope representations) for the DCLF and MAD tests.
- `deriv.fv`: Numerical differentiation for `fv` objects.
- `interp.colourmap`: Smooth interpolation of colour map objects - makes it easy to build colour maps with gradual changes in colour

- `tweak.colourmap` Change individual colour values in a colour map object
- `beachcolourmap`: Colour scheme appropriate for 'altitudes' (signed numerical values)
- `as.fv`: Convert various kinds of data to an 'fv' object
- `quadscheme.logi`: Generates quadrature schemes for the logistic method of ppm.
- `beginner` Introduction for beginners.
- `nnmap`: Given a point pattern, finds the k-th nearest point in the pattern from each pixel in a raster.
- `coef.fii`, `coef.summary.fii`: Extract the interaction coefficients of a fitted interpoint interaction
- `edges2vees`: Low-level function for finding triples in a graph.
- `affine.lpp`, `shift.lpp`, `rotate.lpp`, `rescale.lpp`, `scalardilate.lpp`: Geometrical transformations for point patterns on a linear network
- `affine.linnet`, `shift.linnet`, `rotate.linnet`, `rescale.linnet`, `scalardilate.linnet`: Geometrical transformations for linear networks
- `[.linnet` Subset operator for linear networks
- `timed`: Records the computation time taken
- `connected.ppp`: Find clumps in a point pattern.
- `kstest.lpp`, `kstest.lppm`: The spatial Kolmogorov-Smirnov test can now be applied to point patterns on a linear network (class `lpp`) and point processes on a linear network (class `lppm`).
- `bermantest.lpp`, `bermantest.lppm`: Berman's Z1 and Z2 tests can now be applied to point patterns on a linear network (class `lpp`) and point processes on a linear network (class `lppm`).
- `rhohat.lpp`, `rhohat.lppm`: Nonparametric estimation of the dependence of a point pattern on a spatial covariate: `rhohat` now applies to objects of class `lpp` and `lppm`.
- `intensity.lpp`: Empirical intensity of a point pattern on a linear network.
- `as.function.rhohat`: Converts a `rhohat` object to a function, with extrapolation beyond the endpoints.
- `[.layered`: Subset operator for layered objects.
- `shift`, `rotate`, `affine`, `rescale`, `reflect`, `flipxy`, `scalardilate`: These geometrical transformations now work for `layered` objects.
- `iplot.layered`: Interactive plotting for `layered` objects.
- `as.owin.layered`: Method for as.owin for layered objects.
- `[.owin`: Subset operator for windows, equivalent to `intersect.owin`.
- `rcellnumber`: Generates random integers for the Baddeley-Silverman counterexample.

- `is.lpp`: Tests whether an object is a point pattern on a linear network.
- `is.stationary.lppm`, `is.poisson.lppm`: New methods for `is.stationary` and `is.poisson` for class `lppm`
- `sessionLibs`: Print library names and version numbers (for use in `Sweave` scripts)
- `round.ppp`: Round the spatial coordinates of a point pattern to a specified number of decimal places.
- `rounding`: Determine whether a dataset has been rounded.
- `Hybrid`: The hybrid of several point process interactions [Joint research with Jorge Mateu and Andrew Bevan]
- `is.hybrid`: Recognise a hybrid interaction or hybrid point process model.
- `Finhom`, `Ginhom`, `Jinhom` Inhomogeneous versions of the F, G and J functions
- `delaunay.distance`: Graph distance in the Delaunay triangulation.
- `distcdf`: Cumulative distribution function of the distance between two independent random points in a given window.
- `bw.frac`: Bandwidth selection based on window geometry
- `shortside.owin`, `sidelengths.owin`: Side lengths of (enclosing rectangle of) a window
- `clusterset`: Allard-Fraley estimator of high-density features in a point pattern
- `pool.quadrattest`: Pool several quadrat tests
- `nnfun`: Nearest-neighbour map of a point pattern or a line segment pattern
- `as.ppm`: Converts various kinds of objects to `ppm`.
- `crossdist.lpp`: Shortest-path distances between pairs of points in a linear network
- `nobs.lppm`: Method for `nobs` for `lppm` objects.
- `as.linim`: Converts various kinds of objects to `linim`.
- `model.images.slrm`: Method for `model.images` for `slrm` objects
- `rotate.im`: Rotate a pixel image
- `parres`: Partial residual plots for spatial point process models. A diagnostic for the form of a covariate effect.
- `addvar`: Added variable plots for spatial point process models. A diagnostic for the existence of a covariate effect.
- `intensity`, `intensity.ppp`, `intensity.ppm` Calculate the intensity of a dataset or fitted model. Includes new approximation to the intensity of a fitted Gibbs model
- `LambertW`: Lambert's W-function

- `exactMPLEstrauss`: Fits the stationary Strauss point process model using an exact maximum pseudolikelihood technique. This is mainly intended for technical investigation of algorithms.
- `split.ppx`: Method for `split` for multidimensional point patterns (class `ppx`). This also works for point patterns on a linear network (class `lpp`).
- `model.images`: This function is now generic, with methods for classes `ppm`, `kppm`, `lppm`
- `model.frame`, `model.matrix`: These generic functions now have methods for classes `kppm`, `lppm`
- `as.owin.kppm`, `as.owin.lppm`: New methods for 'as.owin' for objects of class `kppm`, `lppm`
- `as.linnet.lppm`: Extracts the linear network in which a point process model was fitted.
- `dclf.test`: Perform the Diggle (1986)/ Cressie (1991)/ Loosmore and Ford (2006) test of CSR (or another model)
- `mad.test`: Perform the Maximum Absolute Deviation test of CSR (or another model).
- `convolve.im`: Compute convolution of pixel images.
- `Kmulti.inhom`: Counterpart of `Kmulti` for spatially-varying intensity.
- `rmhexpand`: Specify a simulation window, or a rule for expanding the simulation window, in Metropolis-Hastings simulation (`rmh`)
- `transect.im`: Extract pixel values along a line transect.
- `affine.im`: Apply an affine transformation to a pixel image.
- `scalardilate`: Perform scalar dilation of a geometrical object relative to a specified origin.
- `reflect`: Reflect a geometrical object through the origin.
- `[.lpp`, `[.ppx`: Subset operators for the classes `lpp` (point pattern on linear network) and `ppx` (multidimensional space-time point pattern).
- `is.rectangle`, `is.polygonal`, `is.mask`: Determine whether a window w is a rectangle, a domain with polygonal boundaries, or a binary pixel mask.
- `has.offset`: Determines whether a fitted model object (of any kind) has an offset.
- `lohboot`: Computes bootstrap confidence bands for pair correlation function and K function using Loh's (2008) mark bootstrap.
- `rho2hat`: Bivariate extension of `rhohat` for estimating spatial residual risk, or intensity as a function of two covariates.
- `rStraussHard`: Perfect simulation for Strauss-hardcore process (with $\gamma \leq 1$)
- `simulate.slrm`: Method for `simulate` for spatial logistic regression models.
- `labels.ppm`, `labels.kppm`, `labels.slrm`: Methods for `labels` for fitted point process models.
- `commonGrid`: Determine a common spatial domain and pixel resolution for several pixel images and/or binary masks

- `as.matrix.owin`: Converts a window to a logical matrix.
- `rDiggleGratton`, `rDGS`, `rHardcore`: Perfect simulation for the Diggle-Gratton process, Diggle-Gates-Stibbard process, and Hardcore process.
- `bw.scott`: Scott's rule of thumb for bandwidth selection in multidimensional smoothing
- `valid.ppm`: Checks whether a fitted point process model is a valid point process
- `project.ppm`: Forces a fitted point process model to be a valid point process
- `leverage.ppm`, `influence.ppm`, `dfbetas.ppm`: Leverage and influence for point process models
- `ippm`: Experimental extension to 'ppm' which fits irregular parameters in trend by Fisher scoring algorithm.
- `Tstat`: Third order summary statistic for point patterns based on counting triangles.
- `rCauchy`, `rVarGamma`: simulation of a Neyman-Scott process with Cauchy clusters or Variance Gamma (Bessel) clusters. Contributed by Abdollah Jalilian.
- `rPoissonCluster`: simulation of a general Poisson cluster process
- `model.covariates`: Identify the covariates involved in a model (`lm`, `glm`, `ppm` etc)
- `as.im.distfun`: Converts a 'distfun' to a pixel image.
- `cauchy.estK`, `cauchy.estpcf`, `vargamma.estK`, `vargamma.estpcf`: Low-level model-fitting functions for the Neyman-Scott process with Cauchy or Variance-Gamma cluster kernel. Contributed by Abdollah Jalilian.
- `Triplets`: Geyer's triplet interaction, for point process models
- `coef.summary.ppm`: You can now type `coef(summary(fit))` to extract a table of the fitted coefficients of the point process model `fit`
- `scan.test`: Spatial scan test of clustering
- `rat`: New class of 'ratio objects'
- `pool.rat`: New method for `pool`. Combines K function estimates for replicated point patterns (etc) by computing ratio-of-sums
- `unnormdensity`: Weighted kernel density with weights that do not sum to 1 and may be negative.
- `compatible`: New generic function with methods for `fv`, `im`, `fasp` and `units`.
- `imcov`: Spatial covariance function of pixel image or spatial cross-covariance function of two pixel images
- `harmonise.im`: Make several pixel images compatible by converting them to the same pixel grid
- `contour.listof`, `image.listof`: Methods for `contour`() and `image`() for lists of objects
- `dummify`: Convert data to numeric values by constructing dummy variables.
- `pool`: Pool data from several objects of the same class

- `pool.envelope` Pool simulated data from several envelope objects and create a new envelope
- `pool.fasp`: Pool simulated data from several function arrays and create a new array
- `envelope.envelope`: Recalculate an envelope from simulated data using different parameters
- `bw.diggle`: Bandwidth selection for density.ppp by mean square error cross-validation.
- `bw.smoothppp`: Bandwidth selection for smooth.ppp by least-squares cross-validation.
- `layered`, `plot.layered`: A simple mechanism for controlling plots that consist of several successive layers of data.
- `model.depends`: Given a fitted model (of any kind), identify which of the covariates is involved in each term of the model.
- `model.is.additive`: Determine whether a fitted model (of any kind) is additive, in the sense that each term in the model involves at most one covariate.
- `sumouter`, `quadform`: Evaluate certain quadratic forms.
- `flipxy`: Exchange x and y coordinates.
- `idw`: Inverse-distance weighted smoothing.
- `localKinhom`, `localLinhom`, `localpcfinhom`: Inhomogeneous versions of `localK`, `localL`, `localpcf`:
- `envelope.lpp`: Simulation envelopes for point patterns on a linear network
- `lineardisc`: Compute the 'disc' of radius r in a linear network
- `linearpcf`: Pair correlation for point pattern on a linear network
- `linearKinhom`, `linearpcfinhom`: Inhomogeneous versions of the K function and pair correlation function for point patterns on a linear network
- `lppm`: Fit point process models on a linear network.
- `anova.lppm`: Analysis of deviance for point process models on a linear network.
- `predict.lppm`: Prediction for point process models on a linear network.
- `envelope.lppm`: Simulation envelopes for point process models on a linear network.
- `linim`: Pixel image on a linear network
- `plot.linim`: Plot a pixel image on a linear network
- `Gcom`, `Gres`, `Kcom`, `Kres`: New diagnostics for fitted Gibbs or Poisson point process models based on score residuals. `Gcom` is the compensator of the G function; `Gres` is the residual of the G function; `Kcom` is the compensator of the K function; `Kres` is the residual of the K function.
- `psst`, `psstA`, `psstG`: New diagnostics for fitted Gibbs or Poisson point process models based on pseudoscore residuals. `psst` is the pseudoscore diagnostic for a general alternative; `psstA` is the pseudoscore diagnostic for an Area-interaction alternative; `psstG` is the pseudoscore diagnostic for a Geyer saturation alternative.

- `compareFit`: Computes and compares several point process models fitted to the same dataset, using a chosen diagnostic.
- `as.interact`: Extracts the interpoint interaction structure (without parameters) from a fitted point process model or similar object.
- `is.stationary`, `is.poisson`: New generic functions for testing whether a point process model is stationary and/or Poisson. Methods for `ppm`, `kppm`, `slrm` etc.
- `raster.xy`: raster coordinates of a pixel mask
- `zapsmall.im`: 'zapsmall' for pixel images
- `linearK`: Computes the Okabe-Yamada network K-function for a point pattern on a linear network.
- `pairdist.lpp`: Shortest-path distances between each pair of points on a linear network.
- `vcov.kppm`: Asymptotic variance-covariance matrix for regression parameters in kppm object. [Contributed by Abdollah Jalilian and Rasmus Waagepetersen]
- `rLGCP`: Simulation of log-Gaussian Cox processes [Contributed by Abdollah Jalilian and Rasmus Waagepetersen]
- `predict.rhohat`: Method for `predict` for objects of class `rhohat`. Computes a pixel image of the predicted intensity.
- `Kmodel`, `pcfmodel`: Generic functions that compute the K-function or pair correlation function of a point process *model*. There are methods for `kppm` and `ppm`.
- `as.function.fv` Converts a function value table (class `fv`) to a function in R
- `coef.kppm`: Method for `coef` for objects of class `kppm`
- `unitname`, `unitname<-`: These generic functions now have methods for fitted model objects (classes `ppm`, `slrm`, `kppm`, `minconfit`) and quadrature schemes (`quad`).
- `nobs.ppm`: Method for `nobs` for class `ppm`. Returns the number of points in the original data.
- `rgbim`, `hsvim`: Specify three colour channels. These functions convert three pixel images with numeric values into a single image whose pixel values are strings representing colours.
- `scaletointerval`: Generic utility function to rescale data (including spatial data) to a specified interval
- `runiflpp`: Uniformly distributed random points on a linear network
- `rpoislpp`: Poisson point process on a linear network
- `clickjoin`: Interactive graphics to create a linear network
- `superimpose`: The function 'superimpose' is now generic, with methods for ppp, psp and a default method.
- `as.ppp.psp`: New method for `as.ppp` extracts the endpoints and marks from a line segment pattern

- `runiflpp`: Uniformly distributed random points on a linear network
- `rpoislpp`: Poisson point process on a linear network
- `clickjoin`: Interactive graphics to create a linear network
- `superimpose`: The function `superimpose` is now generic, with methods for `ppp`, `psp` and a default method.
- `nsegments`: Number of segments in a line segment pattern
- `clarkevans.test` Classical Clark-Evans test of randomness
- `msr` New class `msr` of signed measures and vector-valued measures supporting residual analysis.
- `quadrat.test.quadratcount`: Method for `quadrat.test` for objects of class `quadratcount` (allows a χ^2 test to be performed on quadrat counts rather than recomputing from the original data)
- `tile.areas`: Computes areas of tiles in a tessellation (efficiently)

5 Precis of all changes

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

- Likelihood cross-validation for smoothing bandwidth.
- Faster computations for pairwise interaction models.
- More flexible models of intensity in cluster/Cox processes.
- New generic function for smoothing.
- Substantial acceleration of many functions
- New 'logistic likelihood' method for fitting Gibbs models.
- Nearest neighbours for point patterns in 3D
- Nearest-neighbour interpolation in 2D
- New 'progress plots'
- Hard core thresholds can be estimated automatically.
- More support for colour maps
- More support for 'fv' objects
- More support for pooling of envelopes.
- More functionality for nearest neighbours
- More support for 'layered' objects.
- Find clumps in a point pattern.

- Connected component interaction model.
- Improvements to interactive plots.
- Visual debugger for Metropolis-Hastings algorithm.
- Rounding of spatial coordinates
- The spatstat manual now exceeds 1000 pages.
- Hybrids of Gibbs point process models.
- Second order composite likelihood method for kppm.
- Inhomogeneous F, G and J functions.
- Delaunay graph distance
- Pixel images can now be displayed with a logarithmic colour map.
- random sequential packing
- Allard-Fraley estimator
- method for pooling several quadrat tests
- better control over dummy points in ppm
- nearest neighbour map
- changes to subsetting of images
- New code for Partial Residual Plots and Added Variable Plots.
- maximum profile pseudolikelihood computations vastly accelerated.
- now possible to capture every k-th state of Metropolis-Hastings algorithm.
- scope of 'intensity.ppm' extended.
- quadrat.test can now perform Monte Carlo tests and one/two-sided tests
- improvements to 'plot.fv'
- improvement to 'rescale'
- some datasets reorganised/corrected.
- New approximation to the intensity of a fitted Gibbs model.
- Improvements to 3D summary functions.
- A multidimensional point pattern (ppx) can now have 'local' coordinates as well as spatial and temporal coordinates and marks.
- More support for fitted cluster models (kppm).
- split method for multidimensional point patterns (ppx) and point patterns on a linear network (lpp).

- Variance estimates are now available for all Gibbs point process models.
- Cressie-Loosmore-Ford test implemented
- plot.fv now avoids collisions between the legend and the graphics.
- Extension to predict.ppm
- Improvements to envelopes and multitype summary functions.
- Line transects of a pixel image.
- Changes to defaults in Metropolis-Hastings simulations.
- More geometrical operations
- Variance-covariance matrix for Gibbs point process models.
- Bootstrap confidence bands for pair correlation function and K function.
- Nearest neighbour distance computations accelerated.
- Improved support for fitted point process models.
- Improved mechanism for handling 'invalid' point processes
- New functions for perfect simulation
- Code for ensuring a fitted point process model is a valid point process
- Leverage and influence for point process models
- New cluster models (support for model-fitting and simulation).
- Fit irregular parameters in trend of point process model
- Third order summary statistic.
- Geyer's triplet interaction
- more functionality for replicated point patterns
- changed default for simulation window in point process simulation
- changed default for edge correction in Kcom, Gcom
- data in spatstat is now lazy-loaded
- Spatial Scan Test
- Functionality for replicated point patterns
- Spatial covariance functions of windows and pixel images.
- Area-interaction models can now be fitted in non-rectangular windows
- New vignette on 'Getting Started with Spatstat'
- Functions to pool data from several objects of the same class.

- Bandwidth selection for density.ppp and smooth.ppp
- Layered plots.
- Model-handling facilities.
- Inverse-distance weighted smoothing.
- Inhomogeneous versions of neighbourhood density functions.
- changes to renormalisation of estimates in Kinhom and pcfinhom
- new diagnostics based on score residuals
- Fitting and simulation of log-Gaussian Cox processes with any covariance function
- More support for 'kppm' and 'rhohat' objects
- Metropolis-Hastings algorithm now saves its transition history
- Easier control of dummy points in ppm
- Convert an 'fv' object to an R function
- marked line segment patterns can now be plotted
- multitype point process models are now 'self-starting'
- new functions to manipulate colour images
- `superimpose` is now generic
- improved mathematical labels when plotting functions
- A line segment pattern can now have a data frame of marks.
- Increased numerical stability.
- New 'self-starting' feature of interpoint interactions.
- Point process model covariates may now depend on additional parameters.
- New class of signed measures, for residual analysis.
- Spatstat now has version nicknames.
- Improved quadrature schemes in `ppm`
- More methods for `intensity`
- Counterpart of `apply` for lists of images.
- Hexagonal grids and tessellations.
- Extensions to scan test and Allard-Fraley cluster set estimator
- Extensions and improvements to plotting functions.
- Improvements to labelling of `fv` objects.

- Geometrical errors in polygons are now repaired automatically.
- Switch all plots from colour to greyscale using a single command.
- Change parameters of fitted model before simulating it.
- Improvements to layout of printed output.
- Plots using texture fill.
- Changed default behaviour of perfect simulation algorithms.
- New syntax for point process models (`ppm`, `kppm`, `lppm`) equivalent to syntax of `lm`, `glm`, ...
- Covariates in `ppm`, `kppm` can be tessellations.

6 Alphabetical list of changes

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

- `adaptive.density`: Now has argument 'verbose'
- `affine`: Argument `vec` can be in various formats.
- `as.function.fv`: Now allows multiple columns to be interpolated.
- `alltypes` Now works for lpp objects
- `ants`: The function `ants.extra$plot()` has been renamed plotit() for conformity with other datasets. The auxiliary data `ants.extra` now includes a function called `side` determining whether a given location is in the scrub or field region. Can be used as a covariate in `ppm`, `kppm`, `slrm`.
- `applynbd` The arguments `N`, `R` and `criterion` may now be specified together.
- `areaGain`, `areaLoss` These now handle arbitrary windows `W`. They are now more accurate when `r` is very small.
- `AreaInter` Point process models with the `AreaInter` interaction can now be fitted to point pattern data in any window.
- `as.box3`: Now accepts objects of class `ppx` or `boxx`.
- `as.fv`: Now handles objects of class `kppm` or `minconfit`.
- `as.mask`: Argument `eps` can now be a 2-vector, specifying x and y resolutions.
- `as.owin.ppm`, `as.owin.kppm`: New argument `from` allows the user to extract the spatial window of the point data (`from="points"`) or the covariate images (`from="covariates"`)
- `betacells`: This dataset has been restructured. The vector of cell profile areas, formerly given by `betacells.extra$area`, has now been included as a column of marks in the point pattern `betacells`.
- `bw.relrisk`: Computation in large datasets accelerated. New arguments `hmin`, `hmax` control the range of trial values of bandwidth.

- `clusterset`: The argument `result` has been renamed `what`. It is now possible to give multiple values to `what` so that both types of result can be computed together.
- `colourmap`: The argument `col` can now be any kind of colour data
- `connected`: `connected` is now generic, with methods for `im`, `owin` and `ppp`.
- `crossdist.ppp`, `crossdist.pp3`, `crossdist.default`: New argument `squared` allows the squared distances to be computed (saving computation time in some applications)
- `crossing.psp`: New argument `fatal` allows the user to handle empty intersections
- `dclf.test`, `mad.test`: The argument X can now be an object produced by a previous call to `dclf.test` or `mad.test`.
- `default.dummy`: Can now generate dummy points as a quasirandom sequence.
- `default.rmhcontrol`, `default.expand`: These functions now work with models of class `rmhmodel` as well as `ppm`.
- `density.ppp`:
  - The argument `weights` can now be a matrix or an `expression`.
  - Numerical underflow no longer occurs when `sigma` is very small and `at="points"`. A warning is no longer issued.
  - New argument `diggle` allows choice of edge correction.
  - New argument `adjust` makes it easy to adjust the smoothing bandwidth.
- `disc`: Argument `centre` can be in various formats.
- `distcdf`: Arguments W and V can now be point patterns.
- `distfun`: The internal format of objects of class `distfun` has been changed.
- `duplicated.ppp`, `unique.ppp` New argument `rule` allows behaviour to be consistent with package `deldir`.
- `effectfun`: Now has argument `se.fit` allowing calculation of standard errors and confidence intervals.
- `envelope`: All methods for `envelope` now handle `fun=NULL`. New argument `envir.simul` determines the environment in which to evaluate the expression `simulate`.
- `envelope.lpp`: Now handles multitype point patterns.
- `envelope.envelope`: New argument `transform` allows the user to apply a transformation to previously-computed summary functions.
- `envelope.envelope`: In `envelope(E, fun=NULL)` if `E` does not contain simulated summary functions, but does contain simulated point patterns, then `fun` now defaults to `Kest`, instead of flagging an error.
- `envelope`: Can now calculate an estimate of the true significance level of the "wrong" test (which declares the observed summary function to be significant if it lies outside the pointwise critical boundary anywhere). Controlled by new argument `do.pwrong`.

- `envelope`: Improved plot labels for envelopes that were generated using the `transform` argument.
- `erosion`, `dilation`, `opening`, `closing`: If the original set is a polygon, the result is a polygon.
- `eval.fv`, `eval.fasp`: Improved behaviour when plotted. Evaluation is now applied only to columns that contain values of the function itself (rather than values of the derivative, hazard rate, etc). This is controlled by the new argument `dotonly`.
- `eval.im`: Images with incompatible dimensions are now resampled to make them compatible (if `harmonize=TRUE`). When the argument `envir` is used, `eval.im` now recognises functions as well as variables in `envir`
- `exactMPLEstrauss`: New argument `project` determines whether the parameter gamma is constrained to lie in $[0, 1]$.
- `expand.owin`: Functionality extended to handle all types of expansion rule.
- `F3est`: Calculation of theoretical Poisson curve (`theo`) has changed, and is now controlled by the argument `sphere`.
- `fitted.ppm`, `predict.ppm`: Argument `new.coef` specifies a vector of parameter values to replace the fitted coefficients of the model.
- `fryplot`: Now has arguments `to` and `from`, allowing selection of a subset of points.
- `fryplot`, `frypoints`: These functions now handle marked point patterns properly.
- `Hardcore`, `StraussHard`: The hard core distance `hc` can now be omitted; it will be estimated from data.
- `identify.ppp`: now handles multivariate marks.
- `im`, `as.im`: The pixel coordinates in an image object are now generated more accurately. This avoids a numerical error in `plot.im`.
- `image.listof`: New argument `equal.ribbon` allows several images to be plotted with the same colour map. If `equal.ribbon=TRUE`, the colour ribbon will no longer be displayed repeatedly for each panel, but will now be plotted only once, at the right hand side of the plot array.
- `inside.owin`: Now accepts the form `list(x,y)` for the first argument.
- `intensity.ppp`: Now has argument `weights`.
- `intensity.ppm`: Now works for stationary point process models with the interactions `Diggle-Gratton`, `DiggleGatesStibbard`, `Fiksel`, `PairPiece` and `Softcore`.
- `intersect.owin`, `union.owin`, `setminus.owin`: If A and B are polygons, the result is a polygon.
- `intersect.tess`, `dirichlet`: The tiles of the resulting tessellation are polygons if the input was polygonal.
- `iplot`: `iplot` is now generic, with methods for `ppp`, `layered` and `default`. `iplot` methods now support zoom and pan navigation.
- `Kest`:

  - Now includes fast algorithm for `correction="none"` which will handle patterns containing millions of points.
  - New option: `correction = "good"` selects the best edge correction that can be computed in reasonable time.
  - Options `correction='border'` and `correction='none'` now run about 4 times faster, thanks to Julian Gilbey.
  - New argument `ratio` determines whether the numerator and denominator of the estimate of the K-function will be stored. This enables analysis of replicated point patterns, using `pool.rat` to pool the K function estimates.

- `Kcross`, `Kdot`, `Kmulti`: New argument `ratio` determines whether the numerator and denominator of the estimate of the multitype K-function will be stored. This enables analysis of replicated point patterns, using `pool.rat` to pool the K function estimates.

- `Kinhom`
  - New argument `normpower` allows different types of renormalisation.
  - Now accepts `correction="good"`.
  - New argument `update`. If `lambda` is a fitted model (class `"ppm"` or `"kppm"`) and `update=TRUE`, the model is re-fitted to the data pattern, before the intensities are computed.

- `Kmulti`, `Gmulti`, `Jmulti`: The arguments `I`, `J` can now be any kind of subset index or can be functions that yield a subset index.

- `Kcom`, `Gcom`: New argument `conditional` gives more explicit control over choice of edge correction in compensator. Simplified defaults for edge correction.

- `Kscaled`, `Lscaled`: Argument `lambda` can now be a fitted model object (class `"ppm"`).

- `kppm`
  - now accepts syntax `kppm(X ~ Z)` similar to `lm` and `glm`, as well as old syntax `kppm(X, ~Z)`
  - Covariates no longer have to be passed in the argument `covariates` but can be objects in the R session;
  - Now has the option of fitting models using Guan's (2006) second order composite likelihood.
  - More flexible models of the intensity, and greater control over the intensity fitting procedure, are now possible using the arguments `covfunargs`, `use.gam`, `nd`, `eps` passed to `ppm`.
  - Also the argument `X` may now be a quadrature scheme.
  - Error messages from the optimising function `optim` are now trapped and handled.
  - Confidence intervals for the fitted trend parameters can now be obtained using `confint`
  - Now accepts `clusters="Cauchy"` or `clusters="VarGamma"` for the Neyman-Scott process with Cauchy or Variance-Gamma cluster kernel.
  - Can now fit a log-Gaussian Cox process with any covariance function implemented by the `RandomFields` package.
  - Covariates can now be tessellations.

- `latest.news`: The text is now displayed one page at a time. Can now be executed by typing the name of the function without parentheses.

- `layered`, `layerplotargs`, `plot.layered`: The plotting argument can now be a list of length 1, which will be replicated to the correct length.
- `layerplotargs<-`: Now handles any spatial object, converting it to a `layered` object.
- `Lest`: Now handles theoretical variance estimates (using delta method) if `var.approx=TRUE`
- `LennardJones` Increased numerical stability. New (optional) scaling argument `sigma0`. Inter-point distances are automatically rescaled using 'self-starting' feature.
- `lgcp.estK`, `Kmodel`: Computation can be greatly accelerated by setting `spatstat.options(fastK.lgcp=TRUE)`.
- `logLik.ppm`: New argument `new.coef` allows the user to evaluate the likelihood for a different value of the parameter.
- `lohboot`: When the result is plotted, the confidence limits are now shaded. New argument `global` allows global (simultaneous) confidence bands instead of pointwise confidence intervals.
- `lppm`:
  - The model-fitting function `lppm` now accepts a syntax similar to `lm` or `glm`, for example `lppm(X ~ Z)`, but still accepts the older syntax `lppm(X, ~Z)`. To support both kinds of syntax, the function `lppm` is now generic, with methods for the classes `formula` and `lppp`. The `formula` method handles a syntax like `lppm(X ~ Z)` while the `lpp` method handles the old syntax `lppm(X, ~Z)`.
  - Covariates no longer have to be passed in the argument `covariates`: variable names appearing in the model formula can simply be the names of objects in the R session.
  - New arguments `eps` and `nd` control the quadrature scheme.
  - Stepwise model selection using `step` now works for `lppm` objects.
- `markstat`: The arguments `N` and `R` may now be specified together.
- `marks.ppp`, `marks<-.ppp`: New argument `drop` determines whether a data frame of marks with only one column will be converted to a vector.
- `model.depends`: Now also recognises 'offset' terms.
- `MultiStrauss`, `MultiHard`, `MultiStraussHard`:
  - The argument `types` can now be omitted or NULL; it will be inferred from the point pattern data.
  - The ordering of the arguments has changed. (The old syntax will still work.)
- `nbfires`: For conformity with other datasets, there is now an object `nbfires.extra`.
- `nnclean`: Extra arguments are now passed to hist.default. The result now has attributes which give the fitted parameter values, information about the fitting procedure, and the histogram bar heights.
- `nncross`: This function is now generic, with methods for `ppp` and `default`. `nncross.ppp` can now find the k-th nearest neighbours, for any k. New arguments added to control the return value and the sorting of data.

- `nndist`, `nnwhich`: New argument `by` makes it possible to find nearest neighbours belonging to specified subsets in a point pattern, for example, the nearest neighbour of each type in a multitype point pattern.
- `nncorr`, `nnmean`, `nnvario` These functions now handle data frames of marks.
- pairwise interaction Gibbs models: Many calculations for these models have been accelerated.
- `owin`: Polygon data are no longer subjected to strict checks on their geometrical validity (self-intersections, overlaps, etc). Instead, geometrical inconsistencies in polygon data are automatically repaired.
- `pcf.ppp`: New argument `domain` allows the computation to be restricted to a subset of the window.
- `pcf.ppp`, `pcfinhom`: New argument `divisor` enables better performance of the estimator of pair correlation function for distances close to zero.
- `pcfcross, pcfdot`: Algorithms have been reimplemented using a single-pass kernel smoother and now run much faster. Bandwidth selection rule improved.
- `pcfinhom`: Now performs renormalisation of estimate. Default behaviour changed - estimates are now renormalised by default.
- `persp.im`: The `colmap` argument can now be a `colourmap` object
- pixel images and grids: The default size of a pixel grid, given by `spatstat.options("npixel")`, has been changed from 100 to 128. A power of 2 gives faster and more accurate results in many cases.
- `plot.fasp`, `plot.fv`, `plot.im`, `plot.ppp`, `plot.owin`, `plot.im`: Can be switched from colour to greyscale by setting `spatstat.options(monochrome=TRUE)`.
- `plot.colourmap`: Now passes arguments to `axis` to control the plot. Appearance of plot improved.
- `plot.envelope`: If the upper envelope is NA but the lower envelope is finite, the upper limit is now treated as +Infinity
- `plot.hyperframe`: The argument `e` now has a different format. Instead of `plot(h, plot(XYZ))` one must now type `plot(h, quote(plot(XYZ)))` This is necessary in order to avoid problems with 'S4 method dispatch'
- `plot.fasp`: New argument `samey` controls whether all panels have the same y limits. Changed default value of `samex`.
- `plot.fii`: Increased resolution of the plot obtained from `plot(fitin(ppm(...)))`
- `plot.fv`:
  - Improved collision-avoidance algorithm (for avoiding overlaps between curves and legend).
  - Labelling of plots has been improved in some cases.
  - New argument `log` allows plotting on logarithmic axes.

  - The formula can now include the symbols `.x` and `.y` as abbreviation for the function argument and the recommended function value, respectively.
  - New argument `add`
  - Argument `legendmath` now defaults to TRUE.
  - New argument `legendargs` gives more control over appearance of legend.
  - Increased default spacing between lines in legend.
  - Improved handling of argument `shade`.

- `plot.fv`, `plot.envelope`: New argument `limitsonly` allows calculation of a common x,y scale for several plots.

- `plot.im`:
  - Now handles many arguments recognised by plot.default such as `cex.main`. Also handles argument `box`.
  - New argument `ribargs` contains parameters controlling the ribbon plot only.
  - Now returns the colour map used.
  - Argument `col` can now be a function.
  - New argument `log` allows colour map to be equally spaced on a log scale.
  - The colour ribbon can now be placed left, right, top or bottom using new argument `ribside`.
  - Can now plot images whose pixel values are strings representing colours. New argument `valuesAreColours`
  - New arguments `add`, `do.plot`, `clipwin`.
  - More arguments recognised by `ribargs`

- `plot.kstest`: Can now perform P-P and Q-Q plots as well.

- `plot.layered`: New arguments `add`, `main`. Better argument handling.

- `plot.listof`, `plot.splitppp`, `contour.listof`, `image.listof`:
  - Default behaviour has changed: panels are now plotted on different scales.
  - When `equal.scales=TRUE` the panels are plotted on exactly equal scales and are exactly aligned (under certain conditions).
  - The arguments `panel.begin` and `panel.end` can now be objects such as windows.
  - New arguments `hsep, vsep` allow increased spacing between plots without adding white space around the outer margin.
  - Argument `mar.panel` may have length 1, 2 or 4.

- `plot.lpp`: New arguments. Automatically plots a legend.

- `plot.owin`:
  - Polygons with holes can now be plotted with filled colours on any device.
  - The window can now be filled with one of 8 different textures.

- `plot.ppp`:

  - Automatically plots a legend
  - Return value is now a graphics map object
  - Now behaves differently if there are multiple columns of marks. Each column of marks is plotted, in a series of separate plots arranged side-by-side.
  - An 'empty' plot can now be generated by setting `type="n"`.
  - New arguments `show.window`, `show.all`, `do.plot`, `clipwin`

- `plot.psp`: Now handles marked line segment patterns and plots the marks as colours.

- `plot.scan.test`, `as.im.scan.test`: These functions can now give the optimal value of circle radius `r`.

- `plot.tess`: For a tessellation represented by a pixel image, `plot.tess` no longer treats the pixel labels as palette colours.

- `plot.ppp`, `plot.owin`, `plot.im`, `plot.psp`, `plot.tess`, `plot.layered`: New universal argument `show.all` determines what happens when a plot is added to an existing plot. If `show.all = TRUE` then everything is plotted, including the main title and colour ribbon.

- `plot.quadratcount`, `plot.quadrattest`: These functions have now been documented.

- `print.fv`: improved layout.

- `pool.envelope`: New arguments `savefuns` and `savepatterns`. Envelopes generated with `VARIANCE=TRUE` can now be pooled. The plot settings of the input data are now respected.

- `ppm`:
  - The model-fitting function `ppm` now accepts a syntax similar to `lm` or `glm`, for example `ppm(X ~ Z)`, but still accepts the older syntax `ppm(X, ~Z)`. To support both kinds of syntax, the function `ppm` is now generic, with methods for the classes `formula`, `ppp` and `quad`. The `formula` method handles a syntax like `ppm(X ~ Z)` while the `ppp` method handles the old syntax `ppm(X, ~Z)`.
  - Covariates no longer have to be passed in the argument `covariates`: variable names appearing in the model formula can simply be the names of objects in the R session.
  - Can now fit models with 'hybrid' interactions.
  - New option: `method = "logi"` for the 'logistic likelihood' method.
  - New argument `nd` controls the number of dummy points.
  - New argument `eps` determines the spacing between dummy points. (also works for related functions `quadscheme`, `default.dummy`, ...)
  - The default quadrature scheme for a point pattern has been improved (in the case of a non-rectangular window) to remove a possible source of bias.
  - Confidence intervals for the fitted trend parameters can now be obtained using `confint` .
  - New argument `project` determines whether the fitted model is forced to be a valid point process.
  - Interaction objects may now be 'self-starting' i.e. initial parameter estimates can be computed from the point pattern dataset.

  - The spatial trend can now depend on additional parameters. This is done by allowing spatial covariate functions to have additional parameters: `function(x, y, ...)` where `...` is controlled by the new argument `covfunargs` to `ppm`.
  - The print and summary methods for `ppm` objects now show standard errors for parameter estimates if the model is Poisson.
  - Covariates can now be tessellations.

-

- `ppp`: New argument `drop` determines whether a data frame of marks with only one column will be converted to a vector.

- `ppx`: Arguments have changed. An object of class `ppx` may now include 'local' coordinates as well as 'spatial' and 'temporal' coordinates, and marks.

- `predict.ppm`:

  - Can now calculate the conditional intensity of a model relative to any point pattern `X` (not just the original data pattern).
  - New argument `correction` allows choice of edge correction when calculating the conditional intensity.
  - New argument `interval` allows confidence intervals or prediction intervals to be calculated.
  - New argument `total` allows for prediction of the total number of points in a specified region.

- `predict.slrm`: New argument `window`

- `print.ppp`, `print.summary.ppp`, `print.owin`, `print.summary.owin`, `print.im`, `print.summary.im`, `print.fv`, `print.msr`: These functions now avoid over-running the text margin (i.e. they respect `options('width')` where possible).

- `print.ppp`, `summary.ppp`: If the point pattern x was generated by Metropolis-Hastings simulation using `rmh`, then `print(x)` and `summary(x)` show information about the simulation parameters.

- `print.ppm`: Standard errors for the parameter estimates, and confidence intervals for the parameters, can now be printed for all Gibbs models (but are printed only for Poisson models by default). Now has argument `what` to allow only selected information to be printed.

- `print.summary.ppm`: Output gives a more precise description of the fitting method.

- `profilepl`: Accelerated (typically by a factor of 5). Does not generate warnings when some of the candidate models have zero likelihood - for example when fitting model with a hard core. Can now maximise over trend parameters as well as interaction parameters

- `progressreport`: Improved output (also affects many functions which print progress reports)

- `project.ppm`: Improved algorithm. Now handles terms in the trend formula as well as the interaction. The projected point process is now obtained by re-fitting the model, and is guaranteed to be the maximum pseudolikelihood fit.

- `psp`: A line segment pattern (object of class 'psp') can now have a data frame of marks.

- `psst`: Argument `funcorrection` changed to `funargs` allowing greater flexibility.
- `quad.ppm`: Now accepts `kppm` objects. New argument `clip`.
- `quadscheme`: Can now generate dummy points from a quasirandom sequence.
- `quadrat.test.splitppp`: The result is now a single object of class `quadrattest`
- `quadrat.test`: Can now perform Monte Carlo test as well (for use in small samples where the χ^2 approximation is inaccurate) New argument `alternative` allows choice of alternative hypothesis and returns one-sided or two-sided p-values as appropriate. The print method for `quadrattest` objects now displays information about the quadrats.
- `raster.x`, `raster.y`, `raster.xy`: These functions have a new argument `drop`
- `rDiggleGratton`, `rDGS`, `rHardcore`, `rStrauss`, `rStraussHard`: By default the point pattern is now generated on a larger window, and trimmed to the original window. New argument `expand=TRUE`.
- `redwoodfull` (Full redwood data): The function `redwoodfull.extra$plot()` has been renamed `plotit` for conformity with other datasets, and improved.
- `relrisk`: New argument `casecontrol` determines whether a bivariate point pattern should be treated as case-control data. New argument `case` allows the user to specify which mark value corresponds to the cases in a case-control dataset.
- `rescale`:
  - If scale argument `s` is missing, then the data are rescaled to native units. For example if the current unit is 0.1 metres, coordinates will be re-expressed in metres.
  - New argument `unitname` makes it possible to change the name of the unit of length.
- `residuals.ppm`: New arguments `new.coef` and `quad` for advanced use (make it possible to compute residuals from a modified version of the fitted model.) The value returned by `residuals.ppm` is now an object of class `msr`. It can be plotted directly.
- `rhohat`: Now has arguments `eps`, `dimyx` to control pixel resolution. If `covariate = "x"` or `"y"`, the resulting object has the same `unitname` as the original point pattern data. Estimation can now be performed using local likelihood fitting with the `locfit` package, or using kernel smoothing.
- `rlabel`: Now works for `lpp`, `pp3`, `ppx` objects.
- `rlinegrid`, `rpoisline`: These functions now handle binary mask windows.
- `rmh`: If `track=TRUE`, the history of transitions of the Metropolis-Hastings algorithm is saved and returned.
- `rmh`, `rmhstart`: The initial state ('`start`') can now be missing or null.
- `rmh`, `rmhcontrol`: It is now possible to save every k-th iteration of the Metropolis-Hastings algorithm. The arguments `nsave` and `nburn` may be given to `rmh` or to `rmhcontrol`. They specify that the point pattern will be saved every `nsave` iterations, after an initial burn-in of `nburn` iterations.

- `rmh.default`: New argument `snoop` allows the user to activate a visual debugger for the Metropolis-Hastings algorithm.
- `rmh.default`: `track` is no longer a formal argument of `rmh.default`; it is now a parameter of `rmhcontrol`. However there is no change in usage: the argument `track` can still be given to `rmh.default`.
- `rmh.ppm`, `simulate.ppm`, `default.expand`: For point process models which have a trend depending only on x and y, the simulation window is now taken to be the same as the original window containing the data (by default). That is, 'expansion' does not take place, by default. (In previous versions of spatstat the simulation window was larger than the original data window.)
- `rmh.ppm`, `simulate.ppm`: The argument sequence for these functions has changed. New argument `expand` allows more explicit control over simulation domain.
- `rmh.ppm`, `simulate.ppm`, `rmhmodel.ppm`: New argument `new.coef` allows the user to change the parameters of a fitted model before it is simulated.
- `rmhcontrol`: The default value of the parameters `periodic` and `expand` has changed.
- `rmhcontrol`: The parameter `expand` can now be in any format acceptable to `rmhexpand`.
- `rmh.ppm`, `rmh.default`, `simulate.ppm`: Any `rmhcontrol` parameter can now be given directly as an argument to `rmh.ppm`, `rmh.default` or `simulate.ppm`.
- `rNeymanScott`: Argument `rcluster` may now take a different format.
- `rotate`: The user can specify the centre of rotation.
- `rotate.owin`, `affine.owin`: These functions now handle binary pixel masks. New argument `rescue` determines whether rectangles will be preserved.
- `rSSI`: Performs 'Random Sequential Packing' if `n=Inf`.
- `runifpointOnLines`, `rpoisppOnLines`, `runiflpp`, `rpoislpp`: Can now generate multitype point patterns.
- `rVarGamma`, `kppm`, `vargamma.estK`, `vargamma.estpcf`: New argument `nu.pcf` provides an alternative way to specify the kernel shape in the VarGamma model, instead of the existing argument `nu.ker`. Function calls that use the ambiguous argument name `nu` will no longer be accepted.
- second order summary functions (`Kest`, `Lest`, `Kinhom`, `pcf.ppp`, `Kdot`, `Kcross`, `Ldot` etc etc) These functions now accept `correction="translation"` as an alternative to `correction = "translate"`, for consistency.
- `scan.test`: Now handles multiple values of circle radius 'r'.
- `setcov`: Can now compute the 'cross-covariance' between two regions
- `shift`: Argument `vec` can be in various formats.
- `simulate.ppm`: New argument `singlerun` determines whether the simulated patterns are generated using independent runs of the Metropolis-Hastings algorithm or are obtained by performing one long run of the algorithm and saving every k-th iteration.
- `simulate.ppm`, `simulate.kppm`: The total computation time is also returned.

- `simulate.kppm`: Now catches errors (such as 'insufficient memory').
- `slrm`: Confidence intervals for the fitted trend parameters can now be obtained using `confint`
- `smooth.fv`: Further options added.
- `smooth.ppp`: Accelerated, when there are several columns of marks. Bandwidth `sigma` is now selected by least-squares cross-validation.
- `smooth.ppp`: Now handles bandwidths equal to zero (by invoking `nnmark`)
- `smooth.ppp`, `smooth.fv`, `smooth.msr`: These functions are now 'Deprecated' in favour of the methods `Smooth.ppp`, `Smooth.fv`, `Smooth.msr` respectively.
- `Smooth.ppp`: `sigma` is now a formal argument of `Smooth.ppp`
- `Softcore`: Improved numerical stability. New argument `sigma0` for manual control over rescaling.
- `spatstat.options`:
  - New option `nvoxel`.
  - New option `print.ppm.SE` controls whether standard errors and confidence intervals are printed for all Gibbs models, for Poisson models only, or are never printed.
  - New option `project.fast` allows a faster shortcut for `project.ppm`.
  - New options `rmh.p`, `rmh.q`, `rmh.nrep` determine the default values of the parameters `p`, `q` and `nrep` of the Metropolis-Hastings algorithm. See `rmhcontrol`.
  - New parameter `n.bandwidth`
  - New parameter `monochrome` switches all plots from colour to greyscale.
- `square` Now accepts vectors of length 2.
- `str` This generic function (for inspecting the internal structure of an object) now produces sensible output for objects of class `hyperframe`, `ppx`, `lpp`.
- `stratrand`, `overlap.owin`, `update.slrm`, `edge.Trans`, `edge.Ripley`: These already-existing functions are now documented.
- `summary.ppm`: Now reports whether the spatial coordinates have been rounded. Improved behaviour when the model covariates are a data frame.
- `superimposePSP`: This function is now deprecated in favour of `superimpose`
- `superimpose`: Now handles data frames of marks.
- `union.owin`: It is now guaranteed that if `A` is a subset of `B`, then `union.owin(A,B)=B`. Now handles a single argument: `union.owin(A)` returns `A`.
- `varblock`: Ugly legends have been repaired.
- `vcov.ppm`: This function now handles all Gibbs point process models. New argument `matrix.action` controls what happens when the matrix is ill-conditioned.
- `vcov.slrm`: Can now calculate correlation matrix or Fisher information matrix as well as variance-covariance matrix.

- `with.fv`: Argument `drop` replaced by new argument `fun` (with different interpretation). New argument `enclos` allows evaluation in other environments.
- `[.fv` Now handles the argument 'drop'. Improved behaviour when plotted.
- `[.im` and `[<-.im` New argument 'j' allows any type of matrix indexing to be used.
- `[.im` Default behaviour changed in the case of a rectangular subset. New argument 'rescue' can be set to TRUE to reinstate previous behaviour.
- `[.lpp` Subset index may now be a window (class 'owin')
- `[.msr` Now handles character-valued indices.
- `[.psp` Now handles binary mask windows.

7 Serious Bugs Fixed

Hundreds of bugs have been detected and fixed in `spatstat` since the version covered by the workshop notes [1]. Bugs that may have affected the user are listed in the package `NEWS` file. To read all these bug reports, type

```
> news(grepl("^BUG", Category), package="spatstat")
```

which currently produces a list of 304 bugs, of which 245 were detected after publication of the workshop notes.

Following is a list of the **most serious bugs** only, in order of potential impact.

- `nncross.ppp`:

  Results were completely incorrect if $k > 1$.
  (Bug introduced in `spatstat 1.31-2`, april 2013; fixed in `spatstat 1.35-0`, december 2013)
- `nncross.pp3`: Results were completely incorrect in some cases.
  (Bug introduced in `spatstat 1.32-0`, august 2013; fixed in `spatstat 1.34-0`, october 2013)
- `rmh`:
  - Simulation was completely incorrect in the case of a multitype point process with an interaction that does not depend on the marks, such as `ppm(betacells, ~marks, Strauss(60))` due to a coding error in the `C` interface.
    (Bug introduced in `spatstat 1.22-3`, march 2010; fixed in `spatstat 1.22-3`, june 2011)
  - Simulation of the Area-Interaction model was completely incorrect.
    (Bug introduced in `spatstat 1.23-6`, october 2011; fixed in `spatstat 1.31-0`, january 2013)
  - Simulation of the Geyer saturation process was completely incorrect.
    (Bug introduced in `spatstat 1.31-0`, january 2013; fixed in `spatstat 1.31-1`, march 2013)
  - Simulation of the Strauss-Hard Core process was partially incorrect, giving point patterns with a slightly lower intensity.
    (Bug introduced in `spatstat 1.31-0`, january 2013; fixed in `spatstat 1.37-0`, may 2014)
  - Simulation of the Geyer saturation process was incorrectly initialised, so that the results of a short run (i.e. small value of `nrep`) were incorrect, while long runs were correct.
    (Bug introduced in `spatstat 1.17-0`, october 2009; fixed in `spatstat 1.31-1`, march 2013)

- `nncross`, `distfun`, `AreaInter`: Results of `nncross` were possibly incorrect when `X` and `Y` did not have the same window. This bug affected values of `distfun` and may also have affected ppm objects with interaction `AreaInter`.
  (Bug introduced in `spatstat 1.9-4`, june 2006; fixed in `spatstat 1.25-2`, january 2012)

- `envelope.ppm`: If the model was an inhomogeneous Poisson process, the resulting envelope object was incorrect (the simulations were correct, but the envelopes were calculated assuming the model was CSR).
  (Bug introduced in `spatstat 1.23-5`, september 2011; fixed in `spatstat 1.23-6`, october 2011)

- `leverage.ppm`, `influence.ppm`, `dfbetas.ppm`: Results were incorrect for non-Poisson processes.
  (Bug introduced in `spatstat 1.25-0`, december 2011; fixed in `spatstat 1.34-0`, october 2013)

- `rVarGamma`: Simulations were incorrect; they were generated using the wrong value of the parameter `nu.ker`.
  (Bug introduced in `spatstat 1.25-0`, december 2011; fixed in `spatstat 1.35-0`, december 2013)

- `rCauchy`: Simulations were incorrect; they were generated using the wrong value of the parameter `omega`.
  (Bug introduced in `spatstat 1.25-0`, december 2011; fixed in `spatstat 1.25-2`, january 2012)

- `kppm`, `matclust.estpcf`, `pcfmodel`: The pair correlation function of the Mátern Cluster Process was evaluated incorrectly at distances close to 0. This could have affected the fitted parameters in `matclust.estpcf()` or `kppm(clusters="MatClust")`.
  (Bug introduced in `spatstat 1.20-2`, august 2010; fixed in `spatstat 1.33-0`, september 2013)

- `ppm`: Results were incorrect for the Geyer saturation model with a non-integer value of the saturation parameter `sat`.
  (Bug introduced in `spatstat 1.20-0`, july 2010; fixed in `spatstat 1.31-2`, april 2013)

- `lppm`: For multitype patterns, the fitted model was completely incorrect due to an error in constructing the quadrature scheme.
  (Bug introduced in `spatstat 1.23-0`, july 2011; fixed in `spatstat 1.30-0`, december 2012)

- `Geyer`: For point process models with the `Geyer` interaction, `vcov.ppm` and `suffstat` sometimes gave incorrect answers.
  (Bug introduced in `spatstat 1.27-0`, may 2012; fixed in `spatstat 1.30-0`, december 2012)

- `vcov.ppm`, `suffstat`: These functions sometimes gave incorrect values for marked point process models.
  (Bug introduced in `spatstat 1.27-0`, may 2012; fixed in `spatstat 1.29-0`, october 2012)

- `linearK`, `linearKinhom`: If any data points were located exactly at a vertex of the linear network, the weights for Ang's correction were incorrect, due to numerical error. This sometimes produced infinite or NA values of the linear K function.
  (Bug introduced in `spatstat 1.23-0`, july 2011; fixed in `spatstat 1.27-0`, may 2012)

- `Kinhom`: the results were not renormalised (even if `renormalise=TRUE`) in some cases.
  (Bug introduced in `spatstat 1.21-0`, december 2010; fixed in `spatstat 1.37-0`, may 2014)

- `predict.ppm` Calculation of the conditional intensity omitted the edge correction if `correction='translate'` or `correction='periodic'`.
  (Bug introduced in `spatstat 1.17-0`, october 2009; fixed in `spatstat 1.31-3`, may 2013)

- `scan.test` Results were sometimes incorrect due to numerical instability (a 'Gibbs phenomenon'). (Bug introduced in `spatstat 1.24-1`, october 2011; fixed in `spatstat 1.26-1`, april 2012)
- `selfcrossing.psp`: y coordinate values were incorrect. (Bug introduced in `spatstat 1.23-2`, august 2011; fixed in `spatstat 1.25-3`, february 2012)
- `predict.slrm`: Results of `predict(object, newdata)` were incorrect if the spatial domain of `newdata` was larger than the original domain. (Bug introduced in `spatstat 1.21-0`, november 2010; fixed in `spatstat 1.25-3`, february 2012)
- `Lest`: The variance approximations (Lotwick-Silverman and Ripley) obtained with `var.approx=TRUE` were incorrect for `Lest` (although they were correct for `Kest`) due to a coding error. (Bug introduced in `spatstat 1.24-1`, october 2011; fixed in `spatstat 1.24-2`, november 2011)
- `bw.diggle`: Bandwidth was too large by a factor of 2. (Bug introduced in `spatstat 1.23-4`, september 2011; fixed in `spatstat 1.23-5`, september 2011)
- pair correlation functions (`pcf.ppp`, `pcfdot`, `pcfcross` etc:) The result had a negative bias at the maximum r value, because contributions to the pcf estimate from interpoint distances greater than `max(r)` were mistakenly omitted. (Bugs fixed in `spatstat 1.35-0`, december 2013)
- `Kest`, `Lest`: Gave incorrect values in very large datasets, due to numerical overflow. 'Very large' typically means about 1 million points in a random pattern, or 100,000 points in a tightly clustered pattern. [Overflow cannot occur unless there are at least 46,341 points.]
- `bw.relrisk`: Implementation of `method="weightedleastsquares"` was incorrect and was equivalent to `method="leastsquares"`. (Bug introduced in `spatstat 1.21-0`, november 2010; fixed in `spatstat 1.23-4`, september 2011)
- `bdist.tiles` Values were incorrect in some cases due to numerical error. (Bug fixed in `spatstat 1.29-0`, october 2012)
- polygon geometry The point-in-polygon test gave the wrong answer in some boundary cases. (Bug fixed in `spatstat 1.23-2`, august 2011)

References

[1] A.˜Baddeley. Analysing spatial point patterns in R. Technical report, CSIRO, 2010. Version 4. Available at `www.csiro.au/resources/pf16h.html`.