How R Helped Provide Tools for Spatial Data Analysis

Roger Bivand

29 February 2020, 10:25-11:10

Slides and script

The slides and script for parts of the code used are at:

https://github.com/rsbivand/celebRation20_files

Outline

- Twenty years ago, most labs and universities had little access to tools for spatial data analysis.
- The access they had was mostly closed source and costly.
- We do not know specific adoption rates of open source software for spatial data analysis, nor the shares of R packages.
- However, the proliferation of reverse depencences on core packages in the Spatial Task View and pagerank analyses do suggest that needs have been met.
- This gives strong incentives both to maintain backwards compatibility, and to adapt to emerging data sources and methods of analysis.

Spatial data

Spatial data typically combine position data in 2D (or 3D), attribute data and metadata related to the position data. Much spatial data could be called map data or GIS data. We collect and handle much more position data since global navigation satellite systems (GNSS) like GPS came on stream 20 years ago, earth observation satellites have been providing data for longer. Here we use **osmdata** (Padgham et al. 2017, 2020) , **mapview** (Appelhans et al. 2019) and **sf** (E. Pebesma 2018a; Pebesma 2020):

```
> suppressPackageStartupMessages(library(osmdata))
> library(sf)
## Linking to GEOS 3.8.0, GDAL 3.1.0dev-e52a02d452, PROJ 7.0.0
> bbox <- opq(bbox = 'bergen norway')
> byb0 <- osmdata_sf(add_osm_feature(bbox, key = 'railway',
+   value = 'light_rail'))$osm_lines
> tram <- osmdata_sf(add_osm_feature(bbox, key = 'railway',
+   value = 'tram'))$osm_lines
> byb1 <- tram[!is.na(tram$name),]
> o <- intersect(names(byb0), names(byb1))
> byb <- rbind(byb0[,o], byb1[,o])
> library(mapview)
> mapview(byb)
```

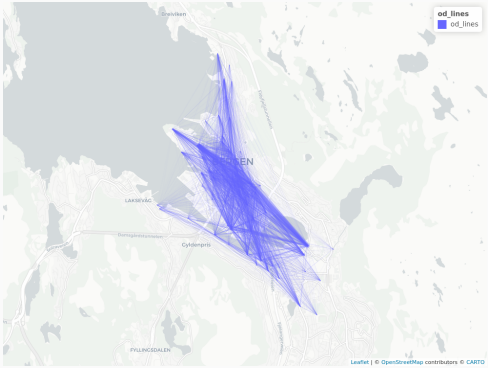
Data handling

We can download monthly CSV files of city bike use, and manipulate the input to let us use the **stplanr** package (Lovelace and Ellison 2018; Lovelace, Ellison, and Morgan 2020) to aggregate origin-destination data. One destination is in Oslo, some are round trips, but otherwise things are OK. We can use CycleStreets to route the volumes onto OSM cycle paths, via an API and API key. We'd still need to aggregate the bike traffic by cycle path segment for completeness.

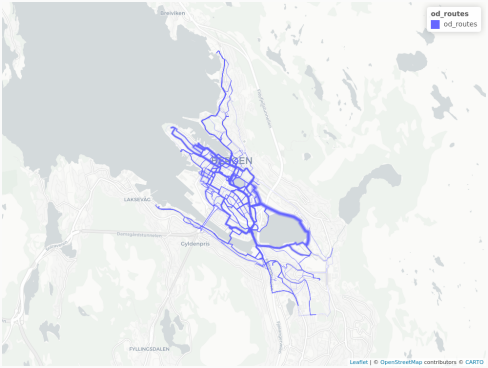
```
> bike_fls <- list.files("bbs")
> trips0 <- NULL
> for (fl in bike_fls) trips0 <- rbind(trips0,
+   read.csv(file.path("bbs", fl), header=TRUE))
> trips0 <- trips0[trips0[, 8] < 6 & trips0[, 13] < 6,]
> trips <- cbind(trips0[,c(1, 4, 2, 9)], data.frame(count=1))
> from <- unique(trips0[,c(4,5,7,8)])
> names(from) <- substring(names(from), 7)
> to <- unique(trips0[,c(9,10,12,13)])
> names(to) <- substring(names(to), 5)
> stations0 <- st_as_sf(merge(from, to, all=TRUE),
+   coords=c("station_longitude", "station_latitude"))
> stations <- aggregate(stations0, list(stations0$station_id),
+   head, n=1)
> suppressWarnings(stations <- st_cast(stations, "POINT"))
> st_crs(stations) <- 4326
> od <- aggregate(trips[,-(1:4)], list(trips$start_station_id,
+   trips$end_station_id), sum)
> od <- od[-(which(od[,1] == od[,2])),]
> library(stplanr)
> od_lines <- od2line(flow=od, zones=stations, zone_code="Group.1",
+   origin_code="Group.1", dest_code="Group.2")
> Sys.setenv(CYCLESTREET="XxXxXxXxXxXxXx")
> od_routes <- line2route(od_lines, "route_cyclestreet",
+   plan = "fastest")
```

Data handling

Origin-destination lines



Routed lines along cycle routes



Late 1990s spatial data analysis

Teaching and research 20 years ago

- In the early and mid 1990s, those of us who were teaching courses in spatial data analysis beyond the direct application of geographical information systems (GIS) found the paucity of software limiting.
- In institutions with funding for site licenses for GIS, it was possible to write or share scripts for Arc/Info (in AML), ArcView (in Avenue), or later in Visual Basic for ArcGIS.
- If site licenses and associated dongles used in the field were a problem (including students involved in fieldwork in research projects), there were few alternatives, but opportunities were discussed on mailing lists.

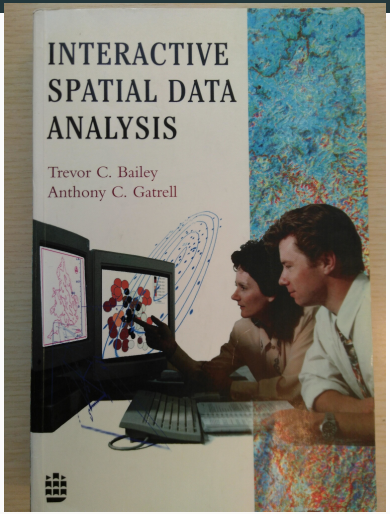
Software for teaching and research

- From late 1996, the R programmimg language and environment began to be seen as an alternative for teaching and research involving spatial analysis.
- R uses much of the syntax of S, then available commercially as S-Plus, but was and remains free to install, use and extend under the GNU General Public License (GPL).
- In addition, it could be installed portably across multiple operating systems, including Windows and Apple MACOS.
- At about the same time, the S-Plus SpatialStats module was published (Kaluzny et al. 1998), and a meeting occurred in Leicester to which many of those looking for solutions took part (Bivand 1998).
- Much of the porting of S code to R for spatial statistics was begun by Albrecht Gebhardt as soon as the R package mechanism matured. Since teachers moving courses from S to R needed access to the S libraries previously used, porting was a crucial step.

Packages

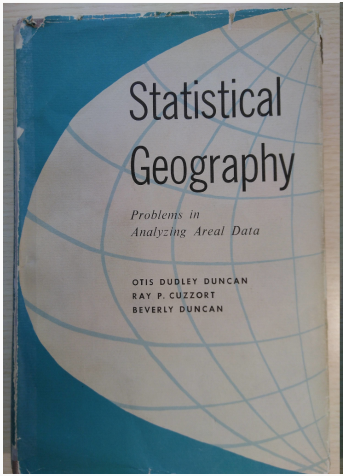
- CRAN listings show **tripack** (Renka and Gebhardt 2016) and **akima** (Akima and Gebhardt 2016) — both with non-open source licenses — available from August 1998 ported by Albrecht Gebhardt; **ash** and **sgeostat** (Majure and Gebhardt 2016) followed in April 1999.
- The **spatial** package was available as part of **MASS** (Venables and Ripley 2002), also ported in part by Albrecht Gebhardt.
- In the earliest period, CRAN administrators helped practically with porting and publication.
- Albrecht and I presented an overview of possibilities of usin R for research and teaching in spatial analysis and statistics in August 1998 (Bivand and Gebhardt 2000).

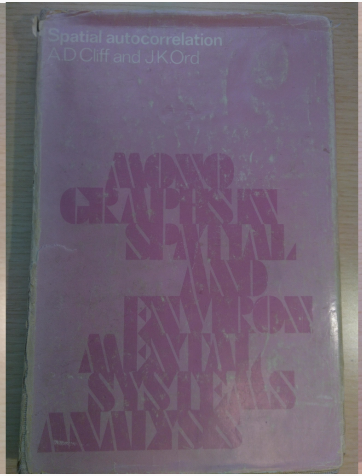
Using R in the computer lab

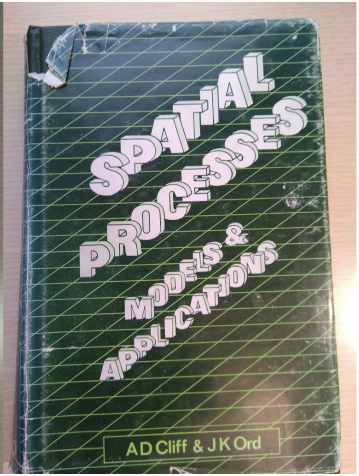


(Bailey and Gatrell 1995)

Books 1961-1981

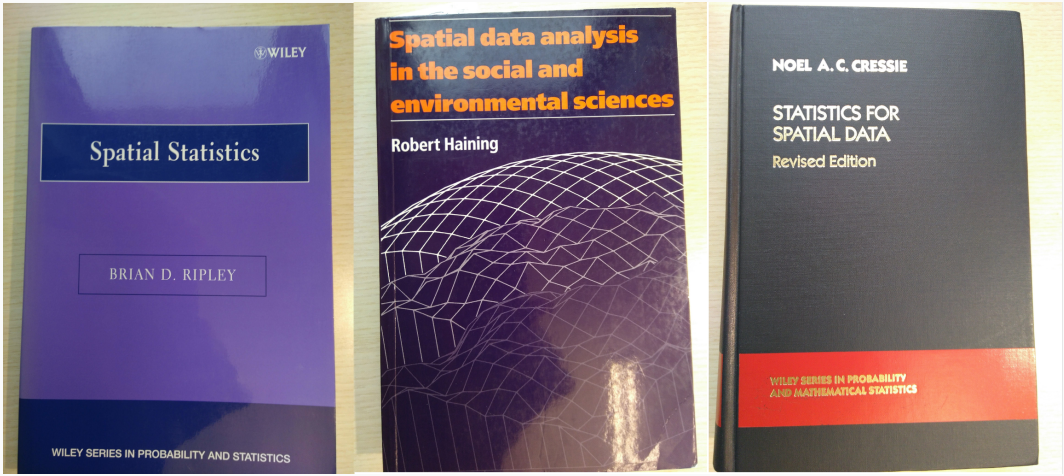






(Duncan, Cuzzort, and Duncan 1961; Cliff and Ord 1973, 1981)

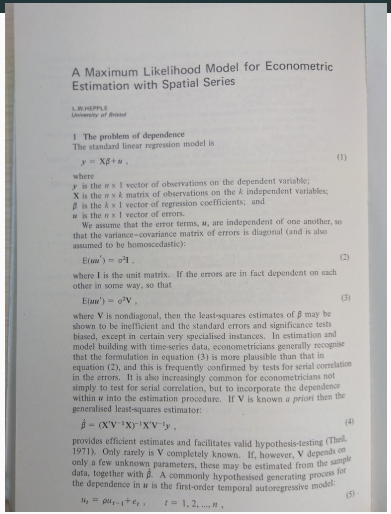
Books 1981-1991



(Ripley 1981; Haining 1990; Cressie 1993)

Intended directions

The need was to cover the sections on spatial autocorrelation and spatial regression in Bailey and Gatrell (1995) for areal data. As with time series, spatial “series” cannot just assume that proximate observations are independent of each other. I’d written Fortan code, and code to create an open source SYSTAT module for testing for spatial autocorrelation and fitting spatial regression models, and some AWK code for for testing for spatial autocorrelation (Bivand 1996, 1997).



A Maximum Likelihood Model for Econometric Estimation with Spatial Series

L.W.HEPPLE
University of Bristol

1 The problem of dependence

The standard linear regression model is

$$y = X\beta + u \,, \tag{1}$$

where
y is the $n \times 1$ vector of observations on the dependent variable;
$\mathbf{X}$ is the $n \times k$ matrix of observations on the k independent variables;
β is the $k \times 1$ vector of regression coefficients; and
u is the $n \times 1$ vector of errors.

We assume that the error terms, u_i, are independent of one another, so that the variance–covariance matrix of errors is diagonal (and is also assumed to be homoscedastic):

$$\mathrm{E}(uu') = \sigma^2\mathbf{I} \,, \tag{2}$$

where $\mathbf{I}$ is the unit matrix. If the errors are in fact dependent on each other in some way, so that

$$\mathrm{E}(uu') = \sigma^2\mathbf{V} \,, \tag{3}$$

where $\mathbf{V}$ is nondiagonal, then the least-squares estimates of β may be shown to be inefficient and the standard errors and significance tests biased, except in certain very specialised instances. In estimation and model building with time-series data, econometricians generally recognise that the formulation in equation (3) is more plausible than that in equation (2), and this is frequently confirmed by tests for serial correlation in the errors. It is also increasingly common for econometricians not simply to test for serial correlation, but to incorporate the dependence within u into the estimation procedure. If $\mathbf{V}$ is known *a priori* then the generalised least-squares estimator:

$$\hat{\beta} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{-1}y \,, \tag{4}$$

provides efficient estimates and facilitates valid hypothesis-testing (Theil, 1971). Only rarely is $\mathbf{V}$ completely known. If, however, $\mathbf{V}$ depends on only a few unknown parameters, these may be estimated from the sample data, together with $\hat{\beta}$. A commonly hypothesised generating process for the dependence in u is the first-order temporal autoregressive model:

$$u_t = \rho u_{t-1} + \epsilon_t \,, \quad t = 1, 2, \ldots, n \,, \tag{5}$$

(Hepple 1976)

Contemporary replication

Using **sf** (Pebesma 2020) to read the data, and **spdep** (Bivand 2019) and **spatialreg** (Bivand and Piras 2019) for analysis, we can replicate some of the classes of twenty years ago. Then, data was input without polygon boundaries; these were shared by Michael Tiefelsdorf.

```
> library(sf)
> new_eire <- st_read(system.file("shapes/eire.shp", package="spData")[1]
## Reading layer `eire' from data source `/home/rsb/lib/r_libs/spData/shapes
## Simple feature collection with 26 features and 10 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: -4.12 ymin: 5768 xmax: 300.82 ymax: 6119.25
## CRS:            NA
```

Finding spatial neighbours

In early work, the neighbours used to determine the spatial weights were found manually and stored in sparse formats. When **spdep** began, automating these steps became important once we could read the geometries associated with spatial objects:

```
> library(spdep)
## Loading required package: sp
## Loading required package: spData
> eire.nb <- poly2nb(new_eire)
> eireW <- nb2listw(eire.nb, style="W")
> eireB <- nb2listw(eire.nb, style="B")
> plot(st_geometry(new_eire))
> plot(eire.nb, st_coordinates(st_centroid(new_eire)), add=TRUE)
```

The standard representation of the measures is:

$$I = \frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(x_i - \bar{x})(x_j - \bar{x})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

for Moran's I, and for Geary's C:

$$C = \frac{(n-1)}{2\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(x_i - x_j)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

where $x_i, i = 1, \dots, n$ are n observations on the numeric variable of interest, and w_{ij} are the spatial weights (Bivand and Wong 2018).

Standard tests

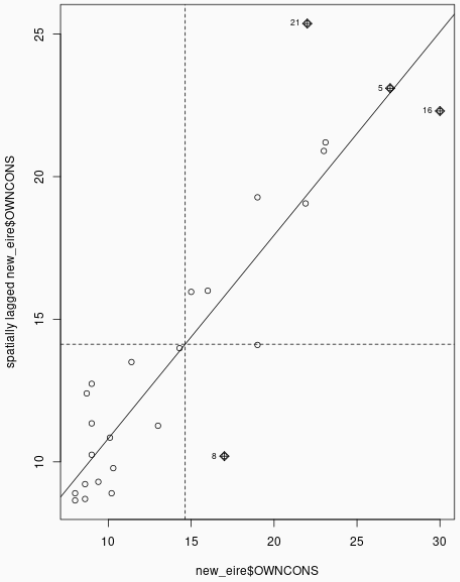
```
> moran.test(new_eire$OWNCONS, eireW)
##
##  Moran I test under randomisation
##
## data:  new_eire$OWNCONS
## weights: eireW
##
## Moran I statistic standard deviate = 5.8637, p-value
## = 2.263e-09
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##        0.71281837       -0.04000000        0.01648309
> geary.test(new_eire$OWNCONS, eireW)
##
##  Geary C test under randomisation
##
## data:  new_eire$OWNCONS
## weights: eireW
##
## Geary C statistic standard deviate = 5.6363, p-value
## = 8.685e-09
## alternative hypothesis: Expectation greater than statistic
## sample estimates:
## Geary C statistic       Expectation          Variance
##        0.24078478        1.00000000        0.01814409
```

```
> moran.test(new_eire$OWNCONS, eireB)
##
##  Moran I test under randomisation
##
## data:  new_eire$OWNCONS
## weights: eireB
##
## Moran I statistic standard deviate = 5.7187, p-value
## = 5.366e-09
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##        0.63262789       -0.04000000        0.01383411
> geary.test(new_eire$OWNCONS, eireB)
##
##  Geary C test under randomisation
##
## data:  new_eire$OWNCONS
## weights: eireB
##
## Geary C statistic standard deviate = 5.1184, p-value
## = 1.541e-07
## alternative hypothesis: Expectation greater than statistic
## sample estimates:
## Geary C statistic       Expectation          Variance
##        0.24485527        1.00000000        0.02176675
```

Moran plots

Luc Anselin continues to be very supportive, earlier in comparing **spdep** output with successive versions of his implementations (SpaceStat, GeoDa), and in contributing code to **spdep** for importing and exporting spatial weights objects. One of his innovations is the Moran plot, setting values of a variable on the x-axis, and the averages of neighbours (the spatially lagged values) on the y-axis, and representing Moran's I as a linear relationship (Anselin 1996):

```
> moran.plot(new_eire$OWNCONS, eireW)
```



Testing regression residuals

When spatial autocorrelation is present, the regression coefficients and their standard errors may be biassed. A set of Lagrange Multiplier tests can be used to check for alternatives, such as omitted spatial dependence in the error term, in the lagged response, in robust versions of the tests and in a portmanteau test (Anselin et al. 1996):

```
> f <- OWNCONS ~ ROADACC
> res0 <- lm(f, data=new_eire)
> summary(res0)
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -8.44782    3.19292  -2.646   0.0142
## ROADACC      0.00527    0.00071   7.423 1.16e-07
##
## Residual standard error: 3.685 on 24 degrees of freedom
## Multiple R-squared:  0.6966, Adjusted R-squared:  0.684
## F-statistic:  55.1 on 1 and 24 DF,  p-value: 1.156e-07
> resLM <- lm.LMtests(res0, listw=eireW, test="all")
> t(sapply(resLM, function(x) unlist(x[1:3])))
##        statistic.LMerr parameter.df      p.value
## LMerr         5.792543            1 0.0160942957
## LMlag        14.425831            1 0.0001457888
## RLMerr        1.044632            1 0.3067466400
## RLMlag        9.677920            1 0.0018649564
## SARMA        15.470463            2 0.0004371513
```

The standard linear regression model is:

$$\mathbf{y} = \mathbf{X}\beta + \varepsilon, \qquad \varepsilon \sim N(0, \sigma^2)$$

where $\mathbf{y}$ is an $(N \times 1)$ vector of observations on a dependent variable taken at each of N locations, $\mathbf{X}$ is an $(N \times k)$ matrix of exogenous variables, β is an $(k \times 1)$ vector of parameters, and ε is an $(N \times 1)$ vector of disturbances. The spatially lagged exogenous variables (the "Durbin" term) are added to the model (spatially lagged X, SLX) (Halleck Vega and Elhorst 2015):

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{W}\mathbf{X}\gamma + \varepsilon,$$

where γ is an $((k-1) \times 1)$ vector of parameters where $\mathbf{W}$ is row-standardised, and a $(k \times 1)$ vector otherwise. It is clear that these two models are estimated in the same way.

Enter Durbin (WX)

The fit of the linear model seems improved, and much of the spatial autocorrelation seems to have been captured by the inclusion of the spatially lagged covariates:

```
> library(spatialreg)
> res0D <- lmSLX(f, data=new_eire, listw=eireW)
> summary(res0D)
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.299e+01  4.711e+00  -4.881 6.27e-05
## ROADACC      2.832e-03  8.753e-04   3.235  0.00366
## lag.ROADACC  5.938e-03  1.608e-03   3.693  0.00120
##
## Residual standard error: 2.983 on 23 degrees of freedom
## Multiple R-squared:  0.8095, Adjusted R-squared:  0.793
## F-statistic: 48.88 on 2 and 23 DF,  p-value: 5.223e-09
```

```
> resLMa <- lm.LMtests(res0D, listw=eireW, test="all")
> t(sapply(resLMa, function(x) unlist(x[1:3])))
##        statistic.LMerr parameter.df    p.value
## LMerr         2.539740            1 0.11101306
## LMlag         4.632957            1 0.03136346
## RLMerr        1.325690            1 0.24957342
## RLMlag        3.418907            1 0.06445368
## SARMA         5.958647            2 0.05082721
```

Enter impacts too

For exogenous variable r, $\partial y_i / \partial x_{ir} = \beta_r$, and $\partial y_i / \partial x_{jr} = 0$ for $i \neq j$, are the direct impacts of changes in x_{ir} on y_i; γ_r are the indirect impacts of the neighbours of i, and $\beta_r + \gamma_r$ the total impacts, here for the single covariate ROADACC. A unit increase in ROADACC (reducing accessibility) will affect OWNCONS at i directly through β_r and indirectly through γ_r. For many models, we can evaluate the variance of the impacts by linear combination:

```
> summary(impacts(res0D))
## Impact measures (SLX, estimable, n-k):
##               Direct    Indirect       Total
## ROADACC 0.002831854 0.005937642 0.008769497
## ========================================================
## Standard errors:
##               Direct    Indirect       Total
## ROADACC 0.000875304 0.001607855 0.001108193
## ========================================================
## Z-values:
##           Direct Indirect    Total
## ROADACC 3.235281 3.692897 7.913331
##
## p-values:
##         Direct    Indirect   Total
## ROADACC 0.0012152 0.00022171 2.4425e-15
```

There are a number of alternative forms of spatial regression models (LeSage and Pace 2009). Here we will start with the simultaneous autoregressive form (also known as the spatial error model — SEM), which may be written as (Ord 1975):

$$\mathbf{y} = \mathbf{X}\beta + \mathbf{u}, \qquad \mathbf{u} = \rho_{\text{Err}}\mathbf{W}\mathbf{u} + \varepsilon,$$

where $\mathbf{y}$ is an $(N \times 1)$ vector of observations on a dependent variable taken at each of N locations, $\mathbf{X}$ is an $(N \times k)$ matrix of exogenous variables, β is an $(k \times 1)$ vector of parameters, ε is an $(N \times 1)$ vector of disturbances ($\varepsilon \sim N(0, \sigma^2)$). $\mathbf{W}$ is a fixed $(N \times N)$ spatial weights matrix, ρ_{Err} is a scalar spatial error parameter, and $\mathbf{u}$ is a spatially autocorrelated disturbance vector. The data generation process of the spatial error model is:
$(\mathbf{I} - \rho_{\text{Err}}\mathbf{W})\mathbf{y} = (\mathbf{I} - \rho_{\text{Err}}\mathbf{W})\mathbf{X}\beta + \varepsilon$, and rewriting: $\mathbf{y} = \mathbf{X}\beta + (\mathbf{I} - \rho_{\text{Err}}\mathbf{W})^{-1}\varepsilon$.

Fitting spatial regressions with maximum likelihood

The log-likelihood function for the spatial error model is:

$$\ell(\beta, \rho_{\text{Err}}, \sigma^2) = -\frac{N}{2}\ln 2\pi - \frac{N}{2}\ln\sigma^2$$
$$+\ln|\mathbf{I} - \rho_{\text{Err}}\mathbf{W}| - \frac{1}{2\sigma^2}\left[(\mathbf{y} - \mathbf{X}\beta)^\top(\mathbf{I} - \rho_{\text{Err}}\mathbf{W})^\top\right.$$
$$\left.(\mathbf{I} - \rho_{\text{Err}}\mathbf{W})(\mathbf{y} - \mathbf{X}\beta)\right]$$

The eigenvalue method for finding the Jacobian is:

$$\ln|\mathbf{I} - \rho_{\text{Err}}\mathbf{W}| = \sum_{i=1}^{N}\ln(1 - \rho_{\text{Err}}\zeta_i)$$

where ζ_i are the eigenvalues of $\mathbf{W}$. A Hausman test checks for further misspecification.

(Ord 1975; Pace and LeSage 2008)

```
> res1 <- errorsarlm(f, data=new_eire, listw=eireW)
> summary(res1, Hausman=TRUE)
## Type: error
## Coefficients: (asymptotic standard errors)
##               Estimate Std. Error z value  Pr(>|z|)
## (Intercept) 2.8927196  4.0228319  0.7191    0.4721
## ROADACC     0.0028009  0.0006746  4.1519 3.297e-05
##
## Lambda: 0.78397, LR test value: 11.282, p-value: 0.00078267
## Asymptotic standard error: 0.11337
##     z-value: 6.915, p-value: 4.6791e-12
## Wald statistic: 47.817, p-value: 4.679e-12
##
## Log likelihood: -64.12465 for error model
## ML residual variance (sigma squared): 6.6005, (sigma: 2.5691)
## Number of observations: 26
## Number of parameters estimated: 4
## AIC: 136.25, (AIC for lm: 145.53)
## Hausman test: -20.108, df: 2, p-value: 4.3011e-05
```

and with Durbin

```
> res1D <- errorsarlm(f, data=new_eire, listw=eireW, Durbin=TRUE)
> summary(res1D, Hausman=TRUE)
## Type: error
## Coefficients: (asymptotic standard errors)
##                 Estimate  Std. Error z value  Pr(>|z|)
## (Intercept) -17.5971453   6.3108327 -2.7884  0.005297
## ROADACC       0.0030347   0.0006593  4.6029 4.166e-06
## lag.ROADACC   0.0044932   0.0016052  2.7990  0.005125
##
## Lambda: 0.50117, LR test value: 3.2261, p-value: 0.072475
## Asymptotic standard error: 0.19858
##     z-value: 2.5238, p-value: 0.011611
## Wald statistic: 6.3693, p-value: 0.011611
##
## Log likelihood: -62.10006 for error model
## ML residual variance (sigma squared): 6.4945, (sigma: 2.5484)
## Number of observations: 26
## Number of parameters estimated: 5
## AIC: 134.2, (AIC for lm: 135.43)
## Hausman test: 13.435, df: 3, p-value: 0.0037837
```

```
> summary(impacts(res1D))
## Impact measures (SDEM, estimable, n):
##              Direct    Indirect       Total
## ROADACC 0.003034738 0.004493158 0.007527896
## ========================================================
## Standard errors:
##               Direct    Indirect       Total
## ROADACC 0.0006593039 0.001605248 0.001468245
## ========================================================
## Z-values:
##           Direct Indirect    Total
## ROADACC 4.602942 2.799043 5.127139
##
## p-values:
##         Direct     Indirect  Total
## ROADACC 4.1656e-06 0.0051254 2.9418e-07
```

The spatial lag model is the most frequently encountered specification in spatial econometrics (termed SAR in some contexts):

$$\mathbf{y} = \rho_{\text{Lag}}\mathbf{W}\mathbf{y} + \mathbf{X}\beta + \varepsilon,$$

In ths model, $\partial y_i/\partial x_{jr} = ((\mathbf{I} - \rho_{\text{Lag}}\mathbf{W})^{-1}\mathbf{I}\beta_r)_{ij}$, where $\mathbf{I}$ is the $N \times N$ identity matrix, and $(\mathbf{I} - \rho_{\text{Lag}}\mathbf{W})^{-1}$ is known to be dense. The awkward matrix term needed to calculate impact measures $S_r(\mathbf{W}) = ((\mathbf{I} - \rho_{\text{Lag}}\mathbf{W})^{-1}\mathbf{I}\beta_r)$ for the lag model (LeSage and Pace 2009), and $S_r(\mathbf{W}) = ((\mathbf{I} - \rho_{\text{Lag}}\mathbf{W})^{-1}(\mathbf{I}\beta_r - \mathbf{W}\gamma_r))$ for the spatial Durbin model, may be approximated using traces of powers of the spatial weights matrix as well as analytically. The average direct impacts are represented by the sum of the diagonal elements of the matrix divided by N for each exogenous variable; the average total impacts are the sum of all matrix elements divided by N for each exogenous variable. Inference on impacts is carried out by sampling from the fitted model.

Why the interaction between β and ρ_{Lag}?

- It has emerged over time that unlike the spatial error model, the spatial dependence in the parameter ρ_{Lag} feeds back
- These feedbacks are discussed as emanating effects (Kelejian, Tavlas, and Hondroyiannis 2006; Kelejian and Piras 2017), also known as impacts (LeSage and Fischer 2008; LeSage and Pace 2009), simultaneous spatial reaction function/reduced form (Anselin and Lozano-Gracia 2008) and equilibrium effects (Ward and Gleditsch 2008)
- This feedback comes from the fact that the elements of the Hessian matrix for the maximum likelihood spatial error model linking ρ_{Err} and β are zero, $\partial^2\ell/(\partial\beta\partial\rho_{\text{Err}}) = \mathbf{0}$
- In the spatial lag model (and by extension in the spatial Durbin model) (Ord 1975; LeSage and Pace 2009): $\partial^2\ell/(\partial\beta\partial\rho_{\text{Lag}}) = \sigma^2\mathbf{X}^\top(\mathbf{I} - \rho_{\text{Lag}}\mathbf{W})^{-1}\mathbf{W}\mathbf{X}\beta$

ML models with spatially lagged response

```
> res2 <- lagsarlm(f, data=new_eire, listw=eireW)
> summary(res2)
## Coefficients: (asymptotic standard errors)
##                Estimate  Std. Error z value Pr(>|z|)
## (Intercept) -6.59344143  2.14828712 -3.0692 0.002147
## ROADACC      0.00270528  0.00058678  4.6104 4.02e-06
##
## Rho: 0.66416, LR test value: 15.793, p-value: 7.0669e-05
## Asymptotic standard error: 0.13015
##     z-value: 5.1032, p-value: 3.3403e-07
## Wald statistic: 26.042, p-value: 3.3403e-07
##
## Log likelihood: -61.86917 for lag model
## ML residual variance (sigma squared): 5.979, (sigma: 2.4452)
## Number of observations: 26
## Number of parameters estimated: 4
## AIC: 131.74, (AIC for lm: 145.53)
## LM test for residual autocorrelation
## test value: 0.46276, p-value: 0.49634
```

```
> tr <- trW(as(eireW, "CsparseMatrix"))
> set.seed(1)
> summary(impacts(res2, tr=tr, R=2000), short=TRUE, zstats=TRUE)
## Impact measures (lag, trace):
##               Direct    Indirect       Total
## ROADACC 0.003159734 0.004895523 0.008055257
## ========================================================
## Simulation results (asymptotic variance matrix):
## ========================================================
## Simulated standard errors
##                Direct    Indirect       Total
## ROADACC 0.0006193466 0.004692186 0.004956921
##
## Simulated z-values:
##           Direct Indirect    Total
## ROADACC 5.225919 1.293278 1.877165
##
## Simulated p-values:
##         Direct     Indirect Total
## ROADACC 1.7329e-07 0.19591  0.060496
```

and with Durbin

```
> res2D <- lagsarlm(f, data=new_eire, listw=eireW, Durbin=TRUE)
> summary(res2D)
## Type: mixed
## Coefficients: (asymptotic standard errors)
##                Estimate  Std. Error z value Pr(>|z|)
## (Intercept) -1.2232e+01  4.7850e+00 -2.5563 0.010580
## ROADACC      2.4107e-03  7.4056e-04  3.2552 0.001133
## lag.ROADACC  2.1302e-03  1.7416e-03  1.2231 0.221283
##
## Rho: 0.51393, LR test value: 4.8985, p-value: 0.02688
## Asymptotic standard error: 0.18732
##     z-value: 2.7436, p-value: 0.0060771
## Wald statistic: 7.5273, p-value: 0.0060771
##
## Log likelihood: -61.26384 for mixed model
## ML residual variance (sigma squared): 6.0655, (sigma: 2.4628)
## Number of observations: 26
## Number of parameters estimated: 5
## AIC: 132.53, (AIC for lm: 135.43)
## LM test for residual autocorrelation
## test value: 0.057165, p-value: 0.81103
```

```
> summary(impacts(res2D, tr=tr, R=2000), short=TRUE, zstats=TRUE)
## Impact measures (mixed, trace):
##              Direct   Indirect       Total
## ROADACC 0.002950977 0.00639107 0.009342047
## ========================================================
## Simulation results (asymptotic variance matrix):
## ========================================================
## Simulated standard errors
##               Direct    Indirect       Total
## ROADACC 0.0006521133 0.003997453 0.004025574
##
## Simulated z-values:
##           Direct Indirect    Total
## ROADACC 4.586809  1.75151 2.482304
##
## Simulated p-values:
##         Direct     Indirect Total
## ROADACC 4.5007e-06 0.079858 0.013054
```

What followed for this subarea

- From my perspective, learning S3 classes for objects needed for testing for spatial autocorrelation and for fitted regression models led to the question of how to write `predict()` methods (Bivand 2002; Goulard, Laurent, and Thomas-Agnan 2017); this led to understanding impacts better.
- **sphet** provided implementations of GMM estimators for cross-sectional models (Piras 2010, 2018), and **splm** for spatial panel models (Millo and Piras 2012, 2018)
- I worked with colleagues comparing measures of spatial autocorrelation (Bivand and Wong 2018), and model fitting approaches (Bivand, Hauke, and Kossowski 2013; Bivand and Piras 2015; Bivand et al. 2017)
- The neighbour objects and measures of spatial autocorrelation have been used across a wide range of packages, in ecology, environmental data analysis, epidemiology and phytogenetics among others

Packages for spatial data analysis

Learning from S-PLUS

- The S-PLUS version of **splancs** provided point pattern analysis (Rowlingson and Diggle 1993, 2017).
- I had contacted Barry Rowlingson in 1997 but only moved forward with porting as R's ability to load shared objects advanced.
- In September 1998, I wrote to him: "It wasn't at all difficult to get things running, which I think is a result of your coding, thank you!"
- However, I added this speculation: "An issue I have thought about a little is whether at some stage Albrecht and I wouldn't integrate or harmonize the points and pairs objects in **splancs**, **spatial** and **sgeostat** — they aren't the same, but for users maybe they ought to appear to be so".
- This concern with class representations for geographical data turned out to be fruitful.

- A further step was to link GRASS and R (Bivand 2000), and followed up at several meetings and working closely with Markus Neteler.
- A consequence of this work was that the CRAN team suggested that I attend a meeting in Vienna in early 2001 to talk about the GRASS GIS interface.
- The meeting gave unique insights into the dynamics of R development, and very valuable contacts.
- Later the same year Luc Anselin and Serge Rey asked me to take part in a workshop in Santa Barbara, which again led to many fruitful new contacts (Bivand 2006).
- Further progress was made in spatial econometrics (Bivand 2002).

2003 Vienna workshop

- During the second half of 2002, it seemed relevant to propose a spatial statistics paper session at the next Vienna meeting to be held in March 2003, together with a workshop to discuss classes for spatial data.
- I had reached out to Edzer Pebesma as an author of the stand-alone open source program `gstat` (Pebesma and Wesseling 1998); it turned out that he had just been approached to wrap the program for S-Plus.
- He saw the potential of the workshop immediately, and in November 2002 wrote in an email: "I wonder whether I should start writing S classes. I'm afraid I should."
- Virgilio Gómez-Rubio had been developing two spatial packages, **RArcInfo** (Gómez-Rubio and López-Quílez 2005; Gómez-Rubio 2011) and **DCluster** (Gómez-Rubio, Ferrándiz-Ferragud, and Lopez-Quílez 2005; Gómez-Rubio, Ferrándiz-Ferragud, and López-Quílez 2015), and was committed to participating.

2003 Vienna workshop

- Although he could not get to the workshop, Nicholas Lewin-Koh wrote in March 2003 that: “I was looking over all the DSC material, especially the spatial stuff. I did notice, after looking through peoples’ packages that there is a lot of duplication of effort. My suggestion is that we set up a repository for spatial packages similar to the Bioconductor mode, where we have a base spatial package that has S-4 based methods and classes that are efficient and general.”
- Straight after the workshop, a collaborative repository for the development of software using SourceForge was established, and the R-sig-geo mailing list (still with over 3,600 subscribers) was created to facilitate interaction.

Beginnings of sp

- So the mandate for the development of the **sp** package emerged in discussions between interested contributors before, during, and especially following the 2003 Vienna workshop.
- Coding meetings were organized by Barry Rowlingson in Lancaster in November 2004 (and kindly hosted by Peter Diggle) and by Virgilio Gómez-Rubio in Valencia in May 2005, at both of which the class definitions and implementations were stress-tested and often changed radically; the package was first published on CRAN in April 2005.
- The underlying model adopted was for S4 (new-style) classes to be used, for `"Spatial"` objects, whether raster or vector, to behave like `"data.frame"` objects, and for visualization methods to make it easy to show the objects.

Relationships with other packages

- From an early point in time, object conversion (known as coercion in S and R) to and from **sp** classes and classes in for example the **spatstat** package (Baddeley and Turner 2005; Baddeley, Rubak, and Turner 2015; Baddeley, Turner, and Rubak 2019).
- Packages could choose whether they would use **sp** classes and methods directly, or rather use those classes for functionality that they did not provide themselves through coercion.
- Reading and writing ESRI Shapefiles had been possible using the **maptools** package (Bivand and Lewin-Koh 2019) available from CRAN since August 2003, but **rgdal** (Bivand, Keitt, and Rowlingson 2019), on CRAN from November 2003, initially only supported raster data read and written using the external GDAL library (Warmerdam 2008).
- Further code contributions by Barry Rowlingson for handling projections using the external PROJ.4 library and the vector drivers in the then OGR part of GDAL were folded into **rgdal**, permitting reading into **sp**-objects and writing from **sp**-objects of vector and raster data.

Completing the sp-verse

- For vector data it became possible to project coordinates, and in addition to transform them where datum specifications were available.
- Until recently, the interfaces to external libraries GDAL and PROJ have been relatively stable, and upstream changes have not led to breaking changes for users of packages using **sp** classes or **rgdal** functionalities, although they have involved significant maintenance effort.
- The final part of the framework for spatial vector data handling was the addition of the **rgeos** package interfacing the external GEOS library in 2011, thanks to Colin Rundell's 2010 Google Summer of Coding project (Bivand and Rundel 2019).
- The **rgeos** package provided vector topological predicates and operations typically found in GIS such as intersection; note that by this time, both GDAL and GEOS used the Simple Features vector representation internally.

By the publication of ASDAR (Bivand, Pebesma, and Gomez-Rubio 2008), a few packages not written or maintained by the authors and their nearest collaborators had begun to use **sp** classes. By the publication of the second edition (Bivand, Pebesma, and Gomez-Rubio 2013), we had seen that the number of packages depending on **sp** had grown strongly. In late 2014, a clear spatial cluster was found from a page rank graph of CRAN packages (de Vries 2014). This cluster is from mid-February 2020:

The raster package

- The **raster** package (Hijmans 2020) was written to provide wrappers for **sp** "`SpatialGrid`" objects, for file access through **rgdal** and other packages, and **rgeos** predicates and operations
- Among other benefits, large rasters could be handled by reading and writing chunks using facilities in **rgdal** and GDAL
- In ASDAR 2, we wrote "We have not attempted to cover the **raster** package in the detail that it deserves, hoping that a **raster** book will appear before long"
- Perhaps **raster** has continued to evolve so fast that writing a book has not been seen as possible; Google Scholar notes over 3,000 citations of the package
- The **terra** package to replace **raster** has just been submitted to CRAN ...

Ways forward

The sf package

- The **sf** package provides the input/output and geometry manipulation functionalities found in **rgdal** and **rgeos**, and an alternative class representation for vector data based on the Simple Features standard and the **units** package (Pebesma, Mailund, and Hiebert 2016; E. Pebesma 2018a; Ucar, Pebesma, and Azcorra 2018)
- The **stars** package (Pebesma 2019) adds some facilities for handling spatio-temporal raster and vector data, building in part on work with the **spacetime** package (E. Pebesma 2018b).
- The **stars** package uses **sf** to interface GDAL for reading and writing rasters, and can handle data by proxy, with the work potentially executed on a back-end, or read from a back-end on demand and in a resolution suited to a chosen output device
- By extension **gdalcubes** builds on the **stars** proxy approach and extends itfor building earth observation workflows (Appel and Pebesma 2019; Appel 2020)

Other packages; reproducible research

- Newer visualization packages, such as **tmap** (Tennekes 2018, 2020), **mapview** and **cartography** (Giraud and Lambert 2016, 2019), give broader scope for data exploration and communication.
- An overview of modelling and analysis packages shows the considerable range of approaches now available in contributed packages and other R code present in supplementary material to published papers (Pebesma, Bivand, and Ribeiro 2015).
- This provides a helpful mechanism supporting reproducible research and hands-on reviewing in which readers can read the code and scripts used in calculating the results presented in published work.

Package dependencies

- Because contributed packages form an ecosystem, some packages are used by others in turn in dependency trees.
- Class representations of data are central, with the data frame conceptualisation shaping much of the whole R ecosystem.
- For the modelling infrastructure to perform correctly, the relationships between objects containing data and formula interfaces constructing model responses and matrices are crucial.
- Because both **sp** and **sf** provide similar interfaces, transition from **sp** to **sf** representations is convenient by design.

Reverse dependencies of the sp and sf packages

- Forward or upstream dependencies are typically on R itself, a small number of packages whose functionalities are used in the package in question (by loading and attaching the package (dependencies), or just loading the namespace of the package (imports)), and possibly external software libraries.
- Reverse or downstream dependencies are packages that themselves use the package in question by loading and attaching it, only loading its namespace, or using it on demand (suggests).
- **sp** and **sf** were written carefully to minimise forward dependencies, with **sp** only depending on and importing packages included in every R distribution by default.
- **sf** adds CRAN contributed packages **Rcpp** and **units** required to build the package, and **classInt** for class intervals, **DBI** for interfacing spatial databases, and **magrittr** for piped operations, where none of these extra forward dependencies draws in many other packages.

Reverse dependencies of the sp and sf packages

The table shows the structure of reverse dependency counts for **sp** and **sf** in mid-Febuary 2020. Recursive dependencies traverse through the whole CRAN dependency tree; the first column of the table shows counts of “depends” and “imports” dependencies counted across the whole tree. These split into 1167 only involving **sp**, 211 involving both packages, and 50 only involving **sf**. If we additionally include “suggests” dependencies, both packages may be used at least indirectly by all CRAN packages.

```
##         Recursive Recursive w/Suggests Not recursive Not recursive w/Suggests
## Sum sp       1378                15698           504                       609
## Sum sf        261                15698           148                       222
## Only sp      1167                    0           445                       502
## Only sf        50                    0            89                       115
## Both          211                15698            59                       107
```

Reverse dependencies of the sp and sf packages

- The two right columns show the same counts but only for packages first-order dependencies on **sp**, **sf** or both.
- The number of packages only using **sf** is encouraging, given that it first entered CRAN in October 2016; it is also encouraging that a fair number use both packages, showing existing packages preserving legacy workflows but also opening up for more modern object representations.
- It takes time and effort to communicate the desirability of migrating from **sp** representations to **sf** and probably **stars**.

Upstream software dependencies of the R-spatial ecosystem

Upstream dependencies of sp workflows

In **sp**, the compiled code (written in C) is self-contained and is made available to other packages, chiefly **rgdal** and **rgeos** to link to their compiled code.

rgdal links to **sp**, and to the external libraries PROJ and GDAL. GDAL itself links to PROJ, and can link to GEOS and many other libraries needed for specific drivers.

The versions vary between platforms and by the installation method used; for **rgdal**, the versions of GDAL, PROJ and **sp** are reported, together with a test showing whether GDAL was built linking to GEOS, something that affects the behaviour of some drivers, while the report for **rgeos** is simpler, only listing the versions of GEOS itself and **sp**.

```
> rgdal::rgdal_extSoftVersion()

##                  GDAL        GDAL_with_GEOS                  PROJ                    sp
## "3.1.0dev-e52a02d452"                "TRUE"              "7.0.0"              "1.4-0"

> rgeos::rgeos_extSoftVersion()

##    GEOS      sp
## "3.8.0" "1.4-0"
```

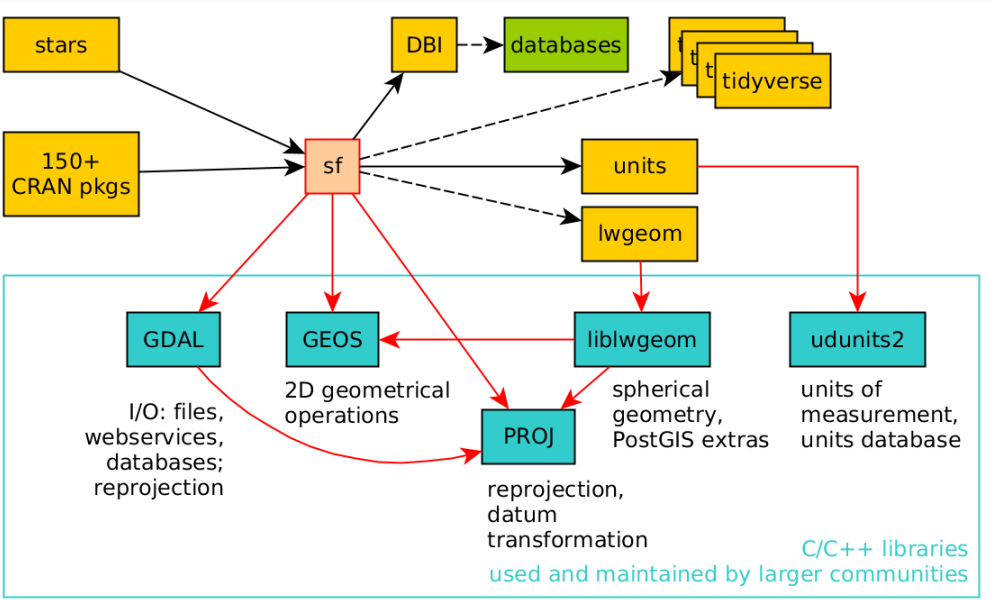
In the case of **sf**, and because it brings together access to the GDAL and GEOS external libraries through Simple Features representation for vector objects, the external software versions supported are the union of those seen above, omitting linkage to a separate package defining classes for objects.

In addition, it reports which API is used for PROJ, either `proj.h` or not (the earlier `proj_api.h`).

```
> sf_extSoftVersion()
```

```
##                  GEOS                  GDAL                proj.4        GDAL_with_GEOS            USE_PROJ_H
##               "3.8.0" "3.1.0dev-e52a02d452"               "7.0.0"                "true"                "true"
```

Upstream dependencies of sf workflows



Upstream software dependencies of the R-spatial ecosystem

- When changes occur in upstream external software, R packages using these libraries often need to adapt, but package maintainers try very hard to shield users from any consequences, so that legacy workflows continue to provide the same or at least similar results from the same data.
- ASDAR code (Bivand, Pebesma, and Gomez-Rubio 2008, 2013) is almost all run nightly on a platform with updated R packages and external software; obviously, this does not test **sf** workflows.
- This does not necessarily trap all differences (figures are not compared), but is helpful in detecting impacts of changes in packages or external software.
- It is also very helpful that CRAN servers using the released and development versions of R, and with different levels of external software also run nightly checks, often on updated versions of external software.

- Again, sometimes changes are first noticed by users, but quite often checks run by maintainers and by CRAN alert us to impending challenges.
- Tracking the development mailing lists of the external software communities, all open source, can also show how thinking is evolving, although sometimes code tidying in external software can have unexpected consequences, breaking not **sf** or **sp** with **rgdal** or **rgeos**, but a package further downstream.
- I discuss open source geospatial software stacks more generally elsewhere (Bivand 2014), but here we will consider ongoing changes in PROJ.

Approaching and implemented changes in PROJ

- PROJ developers not only point out how the world has changed since a World Geodetic System of 1984 (WGS84) was adopted as a hub for coordinate transformation in PROJ, but also introduced transformation pipelines (Knudsen and Evers 2017; Evers and Knudsen 2017).
- In using a transformation hub, PROJ had worked adequately when the errors introduced by transforming first to WGS84 and then from WGS84 to the target coordinate reference system, but with years passing from 1984, the world has undergone sufficient tectonic shifts for errors to increase.
- In addition, the need for accuracy has risen in agriculture and engineering.
- So PROJ, as it was, risked ceasing to be fit for purpose as a fundamental component of the geospatial open source software stack.

Approaching and implemented changes in PROJ

PROJ will also become more tightly linked to authorities responsible for the specification components. While the original well-known text (WKT1) descriptions also contained authorities, WKT2 (2019) is substantially more stringent. PROJ continues to use the European Petroleum Survey Group (EPSG) database, the local copy PROJ uses is now an SQLite database, with a large number of tables:

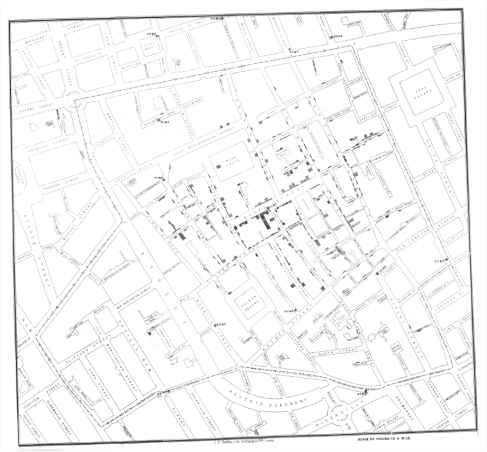
```
> library(RSQLite)
> db <- dbConnect(SQLite(), dbname="/usr/local/share/proj/proj.db")
> cat(strwrap(paste(dbListTables(db), collapse=", ")), sep="\n")
## alias_name, area, authority_list,
## authority_to_authority_preference, axis,
## celestial_body, compound_crs, concatenated_operation,
## concatenated_operation_step, conversion,
## conversion_method, conversion_param,
## conversion_table, coordinate_operation_method,
## coordinate_operation_view,
## coordinate_operation_with_conversion_view,
## coordinate_system, crs_view, deprecation, ellipsoid,
## geodetic_crs, geodetic_datum, geoid_model,
## grid_alternatives, grid_packages,
## grid_transformation, helmert_transformation,
## helmert_transformation_table, metadata, object_view,
## other_transformation, prime_meridian, projected_crs,
## supersession, unit_of_measure, vertical_crs,
## vertical_datum
> dbDisconnect(db)
```

- The initial use of coordinate reference systems for objects defined in **sp** was based on the PROJ string representation, which built on a simplified key=value form.
- Keys began with plus (`+`), and the value format depended on the key.
- If essential keys were missing, some might be added by default from a file that has now been eliminated as misleading; if `+ellps=` was missing and not added internally from other keys, `+ellps=WGS84` would be added silently to refer to the World Geodetic System 1984 ellipsoid definition.

- Accurate coordinate transformation has always been needed for the integration of data from different sources, but has become much more pressing as web mapping has become available in R, through the **leaflet** package (Cheng, Karambelkar, and Xie 2019), on which **mapview** and the `"view"` mode of **tmap**.
- As web mapping provides zooming and panning, possible infelicities that were too small to detect as mismatches in transformation jump into prominence.
- The web mapping workflow transforms input objects to EPSG:4326 (geographical CRS WGS 84, World area of relevance, WGS84 datum) as expected by **leaflet**, then on to EPSG:3857 (WGS 84 / Pseudo-Mercator) for display on web map backgrounds (this is carried out internally in **leaflet**).
- Objects shown in **mapview** and **tmap** are now coerced to **sf** classes, then `st_transform()` transforms to EPSG:4326 if necessary

Broad Street Cholera Data

John Snow did not use maps to *find* the Broad Street pump, the polluted water source behind the 1854 cholera epidemic in Soho in central London, because he associated cholera with water contaminated with sewage, based on earlier experience (Brody et al. 2000). The basic data to be used here were made available by Jim Detwiler, who had collated them for David O'Sullivan for use on the cover of O'Sullivan and Unwin (2003), based on earlier work by Waldo Tobler and others.



Where is the Broad Street pump (sf)?

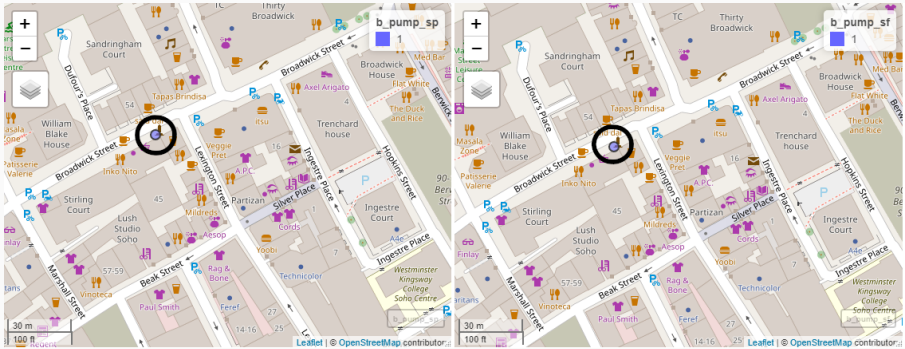
We'll use the example of the location of the Broad Street pump in Soho, London, to be distributed with **sf** (now from a branch of the development version that is being used here); the object has planar coordinates in the OSGB British National Grid projected CRS with the OSGB datum:

```
> library(sf)
> bp_file <- "b_pump.gpkg"
> b_pump_sf <- st_read(bp_file)
## Reading layer `b_pump' from data source `/home/rsb/presentations/celebRat
## Simple feature collection with 1 feature and 1 field
## geometry type:  POINT
## dimension:      XY
## bbox:        xmin: 529393.5 ymin: 181020.6 xmax: 529393.5 ymax: 181020.6
## projected CRS:  Transverse_Mercator
```

Where is the Broad Street pump (sp)?

sp workflows also need to be prepared for changes in CRS representations (note the warning now given when `+datum=` disappears from the rendering of the input coordinate reference system to PROJ string format with GDAL 3):

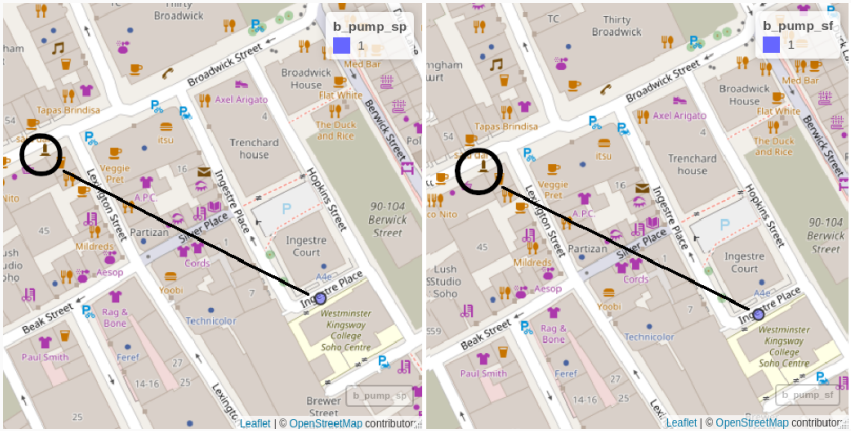
```
> b_pump_sp <- rgdal::readOGR(bp_file)
## Warning in OGRSpatialRef(dsn, layer, morphFromESRI =
## morphFromESRI, dumpSRS = dumpSRS, : Discarded datum
## OSGB_1936 in CRS definition: +proj=tmerc +lat_0=49 +lon_0=-2
## +k=0.999601 +x_0=400000 +y_0=-100000 +ellps=airy +units=m
## +no_defs
## OGR data source with driver: GPKG
## Source: "/home/rsb/presentations/celebRation20/b_pump.gpkg", layer: "b_p
## with 1 features
## It has 1 fields
## Integer64 fields read as strings:  cat
```



mapview displays of the Broad Street pump (black circle) and plotted point location (filled blue symbol), using PROJ 4.9.3 and GDAL 2.2.3 to read the file and transform to EPSG:4326: left panel: `"Spatial"` object using only PROJ string CRS; right panel: `"sf"` object using only PROJ string CRS.

PROJ 6 and GDAL 3

- Once PROJ and GDAL began to progress, particularly to PROJ 6 and GDAL 3, business-as-usual led to regressions, because the objects read from file had lost the deprecated `+datum=` key from their PROJ strings
- The consequences for released **sp** (1.3-2 and 1.4-1), and for **sf** up to and including 0.8-1 are dramatic, with the pump being placed in Ingestre Place, not Broad Street (about 125m in straight-line distance)
- **sf** (development branch `SetFromUserInput`) corrects the regression using WKT2 (2019) strings in place of PROJ strings when PROJ 6 and GDAL 3 are used



mapview displays using PROJ 7.0.0 and GDAL 3.1.0; shifts shown by black lines: left panel: `"Spatial"` object (**sp** 1.3-2 released November 2019); right panel: `"sf"` object (**sf** 0.8-1 released January 2020)

Changes in sf

Released versions of **sf** define the contents of the `"crs"` object, the coordinate reference system of an `"sf"` or `"sfc"` object as an S3 object with `epsg` and `proj4string` components (the `proj4string` compenent is vulnerable to regression). The object can be retreived or assigned using `st_crs()`:

```
> st_crs(b_pump_sf)

## List of 2
##  $ epsg       : int NA
##  $ proj4string: chr "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.999601 +x_0=400000 +y_0=-100000 +ell"| __truncated__
##  - attr(*, "class")= chr "crs"
```

Changes in sf

The development version of **sf** (branch `SetFromUserInput`) modifies the contents of the `"crs"` object. If set, the `input` component is what was passed by the user or instantiating function, such as `st_read()`, while the `wkt` component contains the WKT2 (2019) representation in multi-line format exported from the object read into GDAL:

```
## List of 2
##  $ input: chr "Transverse_Mercator"
##  $ wkt  : chr "PROJCRS[\"Transverse_Mercator\",\n    BASEGEOGCRS[\"GCS_OSGB 1936\",\n   "| __truncated__
##  - attr(*, "class")= chr "crs"
```

The representation of CRS in **sp** and packages using **sp** classes has been unchanged for over 15 years. Up to **sp** 1.3-2, the PROJ string representation was used, and worked without problems before PROJ 6 and GDAL 3. Here the file was read with PROJ 6 and GDAL 3, so has lost the `+datum=` key, with a warning issued as we saw above:

```
> str(slot(b_pump_sp, "proj4string"))

## Formal class 'CRS' [package "sp"] with 1 slot
##   ..@ projargs: chr "+proj=tmerc +lat_0=49 +lon_0=-2 +k=0.999601 +x_0=400000 +y_0=-100000 +ellps=airy +units=m +no_defs"
```

Changes in sp and rgdal

From PROJ 6 and GDAL 3, we noticed that important key-value pairs were being dropped from PROJ strings when reading files. `sp` uses S4 classes, so serialized objects (for example stored in RDS files) break if the S4 definition of `"CRS"` objects is changed. We have to keep PROJ strings in the legacy S4 object, but add a `comment()` to the S4 object containing a WKT2 (2019) string:

```
> str(comment(slot(b_pump_sp, "proj4string")))
```

```
##  chr "PROJCRS[\"Transverse_Mercator\",\n    BASEGEOGCRS[\"GCS_OSGB 1936\",\n        DATUM[\"OSGB 1936\",\n            "| __truncated__
```

Progress so far, PROJ 6+ and GDAL 3

mapview display of three objects: left panel: `"Spatial"` object without WKT2 falling back to the PROJ string without the `+datum=` key; centre panel: `"Spatial"` object with WKT2 correctly coerced to `"sf"` in **sf** (`SetFromUserInput` branch); right panel: `"sf"` object with WKT2.

Progress so far, PROJ 6+ and GDAL 3

- **sp** 1.4-1 is being released fixing problems in a nested condition found by CRAN team in 1.4-0; the 1.4 series have the code to use WKT2 (2019) strings; this is not used until the >= 1.5-1 **rgdal** series is released (available from R-Forge); this will read and write files, and project and transform objects using WKT2 in comments to `"CRS"` objects
- **sf** (development branch `SetFromUserInput`) uses WKT2 (2019) strings when reading and writing files, and projecting and transforming objects; **sp** `"CRS"` objects will be coerced correctly in both directions using WKT2 (2019)
- then we just have to inform package maintainers and script authors that from **sp** 1.4, **rgdal** 1.5 and **sf** when the branch is merged into master and released that they need to review their code when PROJ is >=6 and GDAL >= 3; we've tried hard, lots of github issues filed; then we need to upgrade CRAN binary packages to build using PROJ 6+ and GDAL 3 ...

Conclusions

Conclusions i

- Progress with the **sf** and **stars** packages, including regular spatio-temporal data structures, continues; proxy access to data via APIs or cloud repositories and raster and vector tiles are of growing importance
- Work on **sf** and its use in **stars** is linked to using **Rcpp** to interface external C and C++ libraries; should this be extended to other spatial analysis packages (**gdalcubes**, **terra**)?
- Changes in handling coordinate reference systems through PROJ are continuing, and will impact most work; web mapping depends on knowing the correct CRS of the data
- The legacy shapefile format for vector data should be discontinued as soon as possible, with binary SQLite-based GeoPackage (GPKG) a more robust choice than text-based GeoJSON or GML for non-database storage

Conclusions ii

- Visualization does not need to be web mapping; the **tmap** and **cartography** packages provide modern thematic mapping functionality; **ggplot2** also supports `"sf"` objects through `geom_sf()` and `geom_stars()`
- An important reason for increasing attention on prediction is that it is fundamental for machine learning approaches, in which prediction for validation and test data sets drives model specification choice
- Using spatial data requires care in splitting training data from other data; see **mlr3** (Schratz et al. 2019; Lang et al. 2020), (https://mlr3spatiotempcv.mlr-org.com/), and **blockCV** (Valavi et al. 2019, 2020).

Aftermatter

R's sessionInfo() i

```
> sessionInfo()

## R version 3.6.2 (2019-12-12)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Fedora 31 (Workstation Edition)
##
## Matrix products: default
## BLAS:   /home/rsb/topics/R/R362-share/lib64/R/lib/libRblas.so
## LAPACK: /home/rsb/topics/R/R362-share/lib64/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_GB.UTF-8       LC_NUMERIC=C               LC_TIME=en_GB.UTF-8        LC_COLLATE=en_GB.UTF-8
##  [5] LC_MONETARY=en_GB.UTF-8    LC_MESSAGES=en_GB.UTF-8    LC_PAPER=en_GB.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C             LC_MEASUREMENT=en_GB.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] wordcloud_2.6      RColorBrewer_1.1-2 RSQLite_2.2.0      spatialreg_1.1-5   Matrix_1.2-18      spdep_1.1-3
##  [7] spData_0.3.3       sp_1.4-0           mapview_2.7.0      sf_0.8-2           osmdata_0.1.3      extrafont_0.17
##
## loaded via a namespace (and not attached):
```

R's sessionInfo() ii

```
##  [1] nlme_3.1-144        satellite_1.0.2     lubridate_1.7.4     bit64_0.9-7         webshot_0.5.2       gmodels_2.18.1
##  [7] httr_1.4.1          tools_3.6.2         backports_1.1.5     utf8_1.1.4          rgdal_1.5-5         R6_2.4.1
## [13] KernSmooth_2.23-16 rgeos_0.5-2         DBI_1.1.0           colorspace_1.4-1    raster_3.0-12       tidyselect_1.0.0
## [19] leaflet_2.0.3      bit_1.1-15.2        curl_4.3            compiler_3.6.2      extrafontdb_1.0     cli_2.0.1
## [25] rvest_0.3.5        expm_0.999-4        xml2_1.2.2          scales_1.1.0        classInt_0.4-2      stringr_1.4.0
## [31] digest_0.6.23      rmarkdown_2.1       base64enc_0.1-3     pkgconfig_2.0.3     htmltools_0.4.0     fastmap_1.0.1
## [37] highr_0.8          htmlwidgets_1.5.1   rlang_0.4.4         shiny_1.4.0         generics_0.0.2      jsonlite_1.6.1
## [43] crosstalk_1.0.0    gtools_3.8.1        dplyr_0.8.4         magrittr_1.5        Rcpp_1.0.3          munsell_0.5.0
## [49] fansi_0.4.1        lifecycle_0.1.0     stringi_1.4.5       yaml_2.2.1          MASS_7.3-51.5       blob_1.2.1
## [55] grid_3.6.2         gdata_2.18.0        promises_1.1.0      crayon_1.3.4        deldir_0.1-25       lattice_0.20-38
## [61] splines_3.6.2      knitr_1.28          pillar_1.4.3        boot_1.3-24         codetools_0.2-16    stats4_3.6.2
## [67] LearnBayes_2.15.1  glue_1.3.1          evaluate_0.14       png_0.1-7           vctrs_0.2.2         httpuv_1.5.2
## [73] Rttf2pt1_1.3.8     purrr_0.3.3         tidyr_1.0.2         assertthat_0.2.1    xfun_0.12           mime_0.9
## [79] xtable_1.8-4       broom_0.5.4         e1071_1.7-3         coda_0.19-3         later_1.0.0         class_7.3-15
## [85] viridisLite_0.3.0  tibble_2.1.3        memoise_1.1.0       units_0.6-5         spDataLarge_0.3.1
```

Akima, H., and A. Gebhardt. 2016. *akima: Interpolation of irregularly and regularly spaced data*. https://CRAN.R-project.org/package=akima.

Anselin, L. 1996. The Moran scatterplot as an ESDA tool to assess local instability in spatial association. In *Spatial analytical perspectives on GIS*, eds. M. M. Fischer, H. J. Scholten, and D. Unwin, 111–125. London: Taylor & Francis.

Anselin, L., A. K. Bera, R. Florax, and M. J. Yoon. 1996. Simple diagnostic tests for spatial dependence. *Regional Science and Urban Economics* 26:77–104.

Anselin, L., and N. Lozano-Gracia. 2008. Errors in variables and spatial effects in hedonic house price models of ambient air quality. *Empirical Economics* 34:5–34.

Appel, M. 2020. *Gdalcubes: Earth observation data cubes from satellite image collections*. https://CRAN.R-project.org/package=gdalcubes.

Appel, M., and E. Pebesma. 2019. On-demand processing of data cubes from satellite image collections with the gdalcubes library. *Data* 4 (3). https://www.mdpi.com/2306-5729/4/3/92.

Appelhans, T., F. Detsch, C. Reudenbach, and S. Woellauer. 2019. *mapview: Interactive viewing of spatial data in R*. https://CRAN.R-project.org/package=mapview.

Baddeley, A., E. Rubak, and R. Turner. 2015. *Spatial point patterns: Methodology and applications with R*. London: Chapman; Hall/CRC Press. http://www.crcpress.com/Spatial-Point-Patterns-Methodology-and-Applications-with-R/Baddeley-Rubak-Turner/9781482210200/.

Baddeley, A., and R. Turner. 2005. spatstat: An R package for analyzing spatial point patterns. *Journal of Statistical Software* 12 (6):1–42. http://www.jstatsoft.org/v12/i06/.

Baddeley, A., R. Turner, and E. Rubak. 2019. *Spatstat: Spatial point pattern analysis, model-fitting, simulation, tests*. https://CRAN.R-project.org/package=spatstat.

Bailey, T. C., and A. C. Gatrell. 1995. *Interactive spatial data analysis*. Harlow: Longman.

Bivand, R. 1996. Scripting and toolbox approaches to spatial analysis in a GIS context. In *Spatial analytical perspectives on GIS*, eds. M. M. Fischer, H. J. Scholten, and D. Unwin, 39–52. London: Taylor & Francis.

———. 1997. Scripting and tool integration in spatial analysis: Prototyping local indicators and distance statistics. In *Innovations in GIS 4*, ed. Z. Kemp, 127–138. London: Taylor & Francis.

Bivand, R. 1998. Software and software design issues in the exploration of local dependence. *The Statistician* 47:499–508.

———. 2000. Using the R statistical data analysis language on GRASS 5.0 GIS database files. *Computers & Geosciences* 26 (9):1043–1052.

———. 2002. Spatial econometrics functions in R: Classes and methods. *Journal of Geographical Systems* 4:405–421.

———. 2006. Implementing spatial data analysis software tools in R. *Geographical Analysis* 38:23–40.

———. 2014. Geocomputation and open source software: Components and software stacks. In *Geocomputation*, eds. R. J. Abrahart and L. M. See, 329–355. Boca Raton: CRC Press http://hdl.handle.net/11250/163358.

———. 2019. *spdep: Spatial dependence: Weighting schemes, statistics*. https://github.com/r-spatial/spdep/.

Bivand, R., and A. Gebhardt. 2000. Implementing functions for spatial statistical analysis using the R language. *Journal of Geographical Systems* 2 (3):307–317. https://doi.org/10.1007/PL00011460.

Bivand, R., J. Hauke, and T. Kossowski. 2013. Computing the Jacobian in Gaussian spatial autoregressive models: An illustrated comparison of available methods. *Geographical Analysis* 45 (2):150–179. https://doi.org/10.1111/gean.12008.

Bivand, R., T. Keitt, and B. Rowlingson. 2019. *rgdal: Bindings for the 'geospatial' data abstraction library*. https://CRAN.R-project.org/package=rgdal.

Bivand, R., and N. Lewin-Koh. 2019. *Maptools: Tools for handling spatial objects*. https://CRAN.R-project.org/package=maptools.

Bivand, R., E. Pebesma, and V. Gomez-Rubio. 2008. *Applied spatial data analysis with R*. Springer, NY. https://asdar-book.org/.

———. 2013. *Applied spatial data analysis with R, second edition*. Springer, NY. https://asdar-book.org/.

Bivand, R., and G. Piras. 2015. Comparing implementations of estimation methods for spatial econometrics. *Journal of Statistical Software* 63 (1):1–36.

———. 2019. *spatialreg: Spatial regression analysis*. https://CRAN.R-project.org/package=spatialreg.

Bivand, R., and C. Rundel. 2019. *rgeos: Interface to geometry engine - open source ('geos')*. https://CRAN.R-project.org/package=rgeos.

Bivand, R., Z. Sha, L. Osland, and I. S. Thorsen. 2017. A comparison of estimation methods for multilevel models of spatially structured data. *Spatial Statistics*. https://doi.org/10.1016/j.spasta.2017.01.002.

Bivand, R., and D. W. S. Wong. 2018. Comparing implementations of global and local indicators of spatial association. *TEST* 27 (3):716–748. https://doi.org/10.1007/s11749-018-0599-x.

Brody, H., M. R. Rip, P. Vinten-Johansen, N. Paneth, and S. Rachman. 2000. Map-making and myth-making in Broad Street: The London cholera epidemic, 1854. *Lancet* 356:64–68.

Cheng, J., B. Karambelkar, and Y. Xie. 2019. *Leaflet: Create interactive web maps with the javascript 'leaflet' library*. https://CRAN.R-project.org/package=leaflet.

Cliff, A. D., and J. K. Ord. 1973. *Spatial autocorrelation*. London: Pion.

———. 1981. *Spatial processes*. London: Pion.

Cressie, N. A. C. 1993. *Statistics for spatial data*. New York:Wiley.

de Vries, A. 2014. Finding clusters of CRAN packages using igraph. https://blog.revolutionanalytics.com/2014/12/finding-clusters-of-cran-packages-using-igraph.html.

Duncan, O. D., R. P. Cuzzort, and B. Duncan. 1961. *Statistical geography: Problems in analyzing areal data*. Glencoe, IL: Free Press.

Evers, K., and T. Knudsen. 2017. *Transformation pipelines for proj.4*. https://www.fig.net/resources/proceedings/fig/_proceedings/fig2017/papers/iss6b/ISS6B/_evers/_knudsen/_9156.pdf.

Giraud, T., and N. Lambert. 2016. cartography: Create and integrate maps in your R workflow. *Journal of Open Source Software* 1 (4). https://doi.org/10.21105/joss.00054.

———. 2019. *cartography: Thematic cartography*. https://CRAN.R-project.org/package=cartography.

Goulard, M., T. Laurent, and C. Thomas-Agnan. 2017. About predictions in spatial autoregressive models: Optimal and almost optimal strategies. *Spatial Economic Analysis* 12 (2-3):304–325. https://doi.org/10.1080/17421772.2017.1300679.

Gómez-Rubio, V. 2011. *RArcInfo: Functions to import data from arc/info v7.x binary coverages*. https://CRAN.R-project.org/package=RArcInfo.

Gómez-Rubio, V., J. Ferrándiz-Ferragud, and A. Lopez-Quílez. 2005. Detecting clusters of disease with R. *Journal of Geographical Systems* 7 (2):189–206.

Gómez-Rubio, V., J. Ferrándiz-Ferragud, and A. López-Quílez. 2015. *DCluster: Functions for the detection of spatial clusters of diseases*. https://CRAN.R-project.org/package=DCluster.

Gómez-Rubio, V., and A. López-Quílez. 2005. RArcInfo: Using gis data with r. *Computers & Geosciences* 31 (8):1000–1006.

Haining, R. P. 1990. *Spatial data analysis in the social and environmental sciences*. Cambridge: Cambridge University Press.

Halleck Vega, S., and J. P. Elhorst. 2015. The SLX model. *Journal of Regional Science* 55 (3):339–363. https://doi.org/10.1111/jors.12188.

Hepple, L. W. 1976. A maximum likelihood model for econometric estimation with spatial series. In *Theory and practice in regional science*, London papers in regional science., ed. I. Masser, 90–104. London: Pion.

Hijmans, R. J. 2020. *Raster: Geographic data analysis and modeling*. https://CRAN.R-project.org/package=raster.

Kaluzny, S., S. Vega, T. Cardoso, and A. Shelly. 1998. *S+SpatialStats*. New York, NY: Springer.

Kelejian, H. H., and G. Piras. 2017. *Spatial econometrics*. London: Academic Press.

Kelejian, H. H., G. S. Tavlas, and G. Hondroyiannis. 2006. A spatial modelling approach to contagion among emerging economies. *Open Economies Review* 17 (4-5):423–441.

Knudsen, T., and K. Evers. 2017. *Transformation pipelines for proj.4*. https://meetingorganizer.copernicus.org/EGU2017/EGU2017-8050.pdf.

Lang, M., B. Bischl, J. Richter, P. Schratz, and M. Binder. 2020. *Mlr3: Machine learning in r - next generation*. https://CRAN.R-project.org/package=mlr3.

LeSage, J., and M. Fischer. 2008. Spatial growth regression: Model specification, estimation and interpretation. *Spatial Economic Analysis* 3:275–304.

LeSage, J. P., and K. R. Pace. 2009. *Introduction to spatial econometrics*. Boca Raton, FL: CRC Press.

Lovelace, R., and R. Ellison. 2018. stplanr: A Package for Transport Planning. *The R Journal* 10 (2):7–23. https://doi.org/10.32614/RJ-2018-053.

Lovelace, R., R. Ellison, and M. Morgan. 2020. *Stplanr: Sustainable transport planning*. https://CRAN.R-project.org/package=stplanr.

Majure, J. J., and A. Gebhardt. 2016. *sgeostat: An object-oriented framework for geostatistical modeling in S+*. https://CRAN.R-project.org/package=sgeostat.

Millo, G., and G. Piras. 2012. Splm: Spatial panel data models in r. *Journal of Statistical Software, Articles* 47 (1):1–38. https://www.jstatsoft.org/v047/i01.

———. 2018. *Splm: Econometric models for spatial panel data*. https://CRAN.R-project.org/package=splm.

Ord, J. K. 1975. Estimation Methods for Models of Spatial Interaction. *Journal of the American Statistical Association* 70 (349):120–126.

O'Sullivan, D., and D. J. Unwin. 2003. *Geographical information analysis*. Hoboken, NJ: Wiley.

Pace, R. K., and J. P. LeSage. 2008. A spatial Hausman test. *Economics Letters* 101 (3):282–284.

Padgham, M., B. Rudis, R. Lovelace, and M. Salmon. 2017. Osmdata. *The Journal of Open Source Software* 2 (14). https://doi.org/10.21105/joss.00305.

———. 2020. *Osmdata: Import 'openstreetmap' data as simple features or spatial objects*. https://CRAN.R-project.org/package=osmdata.

Pebesma, E. 2018a. Simple Features for R: Standardized Support for Spatial Vector Data. *The R Journal* 10 (1):439–446. https://journal.r-project.org/archive/2018/RJ-2018-009/index.html.

———. 2018b. *spacetime: Classes and methods for spatio-temporal data*. https://CRAN.R-project.org/package=spacetime.

———. 2019. *stars: Spatiotemporal arrays, raster and vector data cubes*. https://CRAN.R-project.org/package=stars.

———. 2020. *sf: Simple features for r*. https://CRAN.R-project.org/package=sf.

Pebesma, E., R. Bivand, and P. Ribeiro. 2015. Software for spatial statistics. *Journal of Statistical Software, Articles* 63 (1):1–8. https://www.jstatsoft.org/v063/i01.

Pebesma, E. J., and C. G. Wesseling. 1998. Gstat, a program for geostatistical modelling, prediction and simulation. *Computers and Geosciences* 24:17–31.

Pebesma, E., T. Mailund, and J. Hiebert. 2016. Measurement units in R. *The R Journal* 8 (2):486–494.

Piras, G. 2010. Sphet: Spatial models with heteroskedastic innovations in r. *Journal of Statistical Software, Articles* 35 (1):1–21. https://www.jstatsoft.org/v035/i01.

———. 2018. *Sphet: Estimation of spatial autoregressive models with and without heteroscedasticity*. https://CRAN.R-project.org/package=sphet.

Renka, R. J., and A. Gebhardt. 2016. *tripack: Triangulation of irregularly spaced data*. https://CRAN.R-project.org/package=tripack.

Ripley, B. D. 1981. *Spatial statistics*. New York: Wiley.

Rowlingson, B., and P. Diggle. 2017. *Splancs: Spatial and space-time point pattern analysis*. https://CRAN.R-project.org/package=splancs.

Rowlingson, B., and P. J. Diggle. 1993. Splancs: Spatial point pattern analysis code in S-Plus. *Computers and Geosciences* 19:627–655.

Schratz, P., J. Muenchow, E. Iturritxa, J. Richter, and A. Brenning. 2019. Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. *Ecological Modelling* 406:109–120.

Tennekes, M. 2018. tmap: Thematic maps in R. *Journal of Statistical Software* 84 (6):1–39. https://www.jstatsoft.org/v084/i06.

———. 2020. *tmap: Thematic maps*. https://CRAN.R-project.org/package=tmap.

Ucar, I., E. Pebesma, and A. Azcorra. 2018. Measurement Errors in R. *The R Journal* 10 (2):549–557. https://journal.r-project.org/archive/2018/RJ-2018-075/index.html.

Valavi, R., J. Elith, J. Lahoz-Monfort, and G. Guillera-Arroita. 2020. *BlockCV: Spatial and environmental blocking for k-fold cross-validation*. https://CRAN.R-project.org/package=blockCV.

Valavi, R., J. Elith, J. J. Lahoz-Monfort, and G. Guillera-Arroita. 2019. BlockCV: An r package for generating spatially or environmentally separated folds for k-fold cross-validation of species distribution models. *Methods in Ecology and Evolution* 10 (2):225–232.

Venables, W. N., and B. D. Ripley. 2002. *Modern applied statistics with S* Fourth. New York: Springer. http://www.stats.ox.ac.uk/pub/MASS4.

Ward, M. D., and K. S. Gleditsch. 2008. *Spatial regression models*. Thousand Oaks, CA: Sage.

Warmerdam, F. 2008. The Geospatial Data Abstraction Library. In *Open source approaches in spatial data handling*, eds. G. B. Hall and M. Leahy, 87–104. Berlin: Springer-Verlag.