Comparing Spatial Fields with SpatialVx: Spatial Forecast Verification in R

Eric Gilleland
National Center for Atmospheric Research

Abstract

SpatialVx is a software package written in R for performing spatial weather forecast verification primarily on gridded verification sets (i.e., a gridded observation field compared against a forecast field on the same grid). In a more general sense, sptial verification is all about comparing spatial fields and has a broader application area, such as comparing photographic images; though unique challenges in the realm of forecast verification exist. Background on spatial verification methods is given along with a tutorial for the key functions of the package. Beyond a description of the software, a look at some of the many challenges associated with spatial verification is explored and key areas for further statistical development are highlighted.

Keywords: **SpatialVx**, R, comparing spatial fields, spatial verification, image warping, cluster analysis, smoothing, binary image metrics, feature-based verification.

1. Introduction: Background on spatial forecast verification

Spatial forecast verification arose out of a need to provide consistent information about forecast performance wtih an expert's subjective opinion, as well as the need to provide meaningful diagnostic measures and plots. In particular, as the grid resolution for weather forecasts became increasingly fine, traditional grid-point-to-grid point verification techniques tended to favor the coarser scale models (cf. Mass, Ovens, Westrick, and Colle 2002; Baldwin and Kain 2006). One reason for this phenomenon is the double penalty incurred for a single timing or spatial alignment error. For example, the top left panel of Figure 1 shows an illustration of a forecast feature that could represent a phenomenon of interest, such as a storm cell, that perfectly matches the observed feature except that it is displaced spatially to where there is no overlap between the two. Traditional verification measures will penalize this otherwise perfect forecast once for all of the grid points over the observed storm cell (that do not overlap with the forecast) and again for all of the grid points over that for the forecast (that do not overlap with those in the observation).

Moreover, the top right panel of Figure 1, and the panel directly below it, will produce the same traditional verification measures as the panel in the top left. Indeed, the panel in the bottom of the figure may produce a better score than any of the other cases, with some of the traditional measures, simply because it is the only one where the forecast feature overlaps with the observed one. Subsequently, it was clear that there was a need for methods that could provide more diagnostic information about how a forecast does well and how it performs poorly. Such information must have the forecast user in mind as is illustrated in Figure 2.

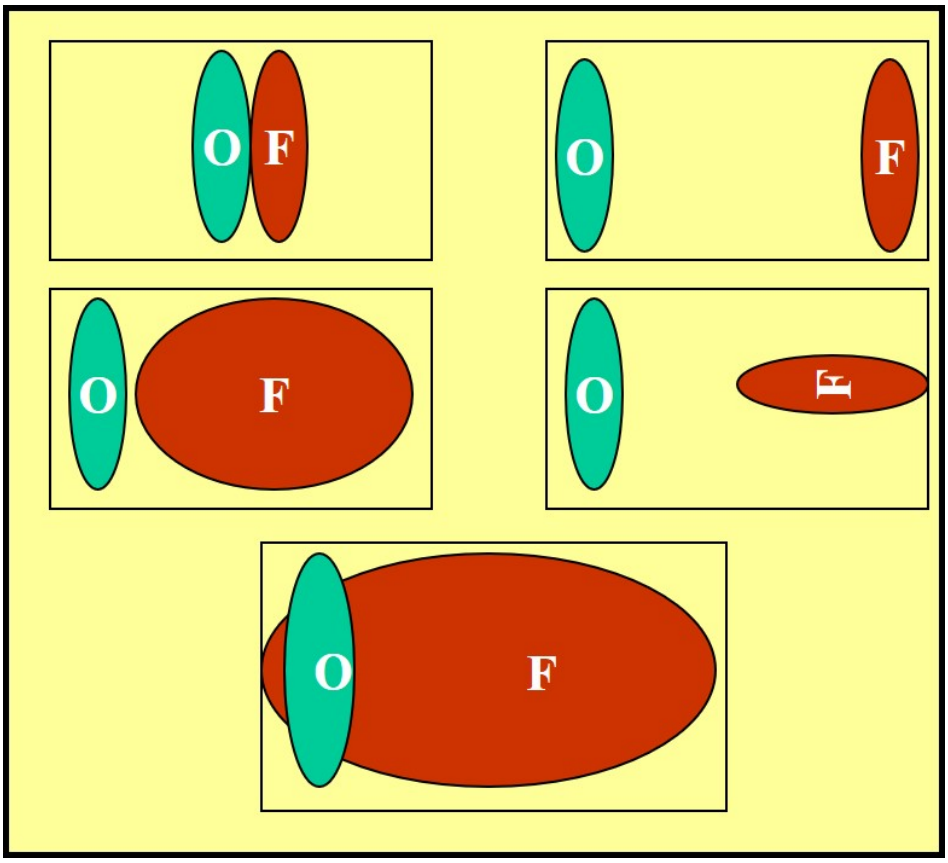

Figure 1: Illustration of some of the issues associated with traditional forecast verification applied to high-resolution verification sets. Figure used by permission from Barbara G. Brown.

As high-resolution forecasts were developed and these issues of verification identified, a multitude of methods were proposed. The rapid pace of development was spurred on in part because of existing methods that had previously been developed in other application areas, such as image analysis, computer vision and spatial statistics (see, e.g., Wikle, Zammit-Mangion, and Cressie 2019). To make sense of the dizzying cornucopia of these new approaches, an inter-comparison project, dubbed the ICP, was established (Gilleland, Ahijevych, Brown, and Ebert 2010a). A major contribution of this project was the creation of a series of test cases that could be used by method developers in order to test their methods against the wide array of other approaches without the daunting task of having to duplicate these myriad, often highly complex, methods.

The cases included contrived geometric shapes that were inspired by Figure 1 and are described in Ahijevych, Gilleland, Brown, and Ebert (2009), as well as real and purturbed real cases. A set of nine single-time-point snapshots of 24-h accumulated precipitation (the real cases) from the 2005 Spring Experiment sponsored by the Storm Prediction Center (SPC) and the National Severe Storms Laboratory (NSSL) included three competing forecast models and one analysis field for each of the nine cases serving as the "observations" (see Kain, Weiss, Bright, Baldwin, Levit, Carbin, Schwartz, Weisman, Droegemeier, Weber, and Thomas 2008; Ahijevych *et al.* 2009, for more information about these cases). All of them are available

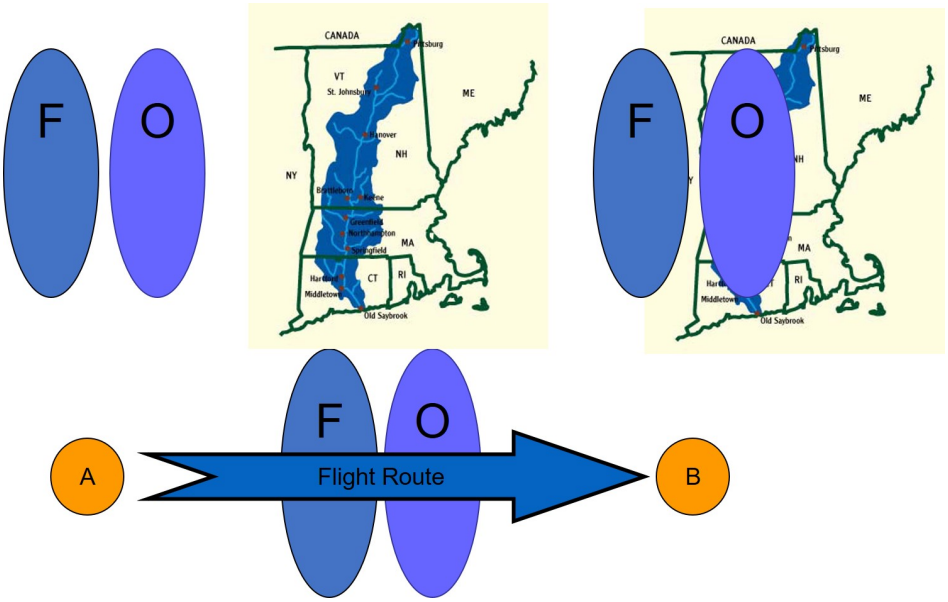

Figure 2: Illustration of the need for user-specific diagnostic information beyond what a traditional forecast verification measure can provide. The top left panel shows a forecast and an observation feature. The two are similar in size and shape but do not overlap. Is this forecast good? The answer may depend on the user. If the user is concerned about the watershed shown in the top middle, then it is a very bad forecast as depicted on the top right. On the other hand, a flight route planner might need to divert the route by the same amount regardless of the spatial displacement error (bottom panel), making it a good forecast for this purpose. Figure used by permission from Barbara G. Brown.

from the ICP website (`https://projects.ral.ucar.edu/icp/`) in ASCII format, including a larger set of 32 real cases as analyzed in Davis, Brown, Bullock, and Halley Gotway (2009); Gilleland, Lindström, and Lindgren (2010c); Gilleland (2013a, 2017, 2021). There is also an R (R Core Team 2020) package called **ICPtestcases** available at this same website.[1]

A further product of the ICP was a special collection of papers in the *Weather and Forecasting* journal (Ahijevych *et al.* 2009; Brill and Mesinger 2009; Casati 2010; Davis *et al.* 2009; Ebert 2009; Ebert and Gallus 2009; Gilleland, Ahijevych, Brown, Casati, and Ebert 2009; Gilleland *et al.* 2010c; Keil and Craig 2009; Lack, Limpert, and Fox 2010; Lakshmanan and Kain 2010; Marzban, Sandgathe, Lyons, and Lederer 2009; Mittermaier and Roberts 2010; Nachamkin 2009; Wernli, Hofmann, and Zimmer 2009). A major result of the project included a preliminary classification of the methods into two categories, which were further divided into two sub-categories each (Gilleland *et al.* 2009), namely: filtering, which can be subdivided into smoothing (dubbed neighborhood methods) and spectral (dubbed scale separation), and displacement, which can be subdivided into feature-based and field deformation methods. Independently, Wealands, Grayson, and Walker (2005) proposed essentially the same methods for spatial verification in the hydrology literature.

The ICP was focused on the central United States, which is an area known for severe storm ac-

[1]These cases were originally included in **SpatialVx** but needed to be removed because of changes in CRAN policies on memory allocations.

tivity and is characterized by a large swath of relatively flat land. A second inter-comparison project was subsequently introduced that aimed at testing the methods over central Europe, which is characterized by complex terrain, in order to further test the methods; namely, the Mesoscale Verification Intercomparison in Complex Terrain (MesoVICT Dorninger, Mittermaier, Gilleland, Ebert, Brown, and Wilson 2013) project. The cases for MesoVICT were established over a tiered system going from the same types of simple snapshots as the ICP, called the core case, to much more elaborate and realistic verification sets that included multiple variables beyond precipitation and weather systems that cover multiple days. These cases are also available at the ICP website in ASCII format, with instructions for uploading them into R. For more about MesoVICT and results from the project, see Dorninger *et al.* (2013); Dorninger, Gilleland, Casati, Mittermaier, Ebert, Brown, and Wilson (2018); Gilleland (2017, 2021); Mariani and Casaioli (2018); Radanovics, Vidal, and Sauquet (2018); Skok and Hladnik (2017).

Spatial verification methods have also started to receive some attention in the climate literature as they present new, more physically meaningful ways to compare future climates with past ones (e.g., Jun, Knutti, and Nychka 2008; Livina, Edwards, Goswami, and Lenton 2008; Moise and Delage 2011; AghaKouchak and Mehran 2013; Levy, Ingram, Jenkinson, Huntingford, Hugo Lambert, and Allen 2013; Hobæk Haff, Frigessi, and Maraun 2015; Konca-Kedzierska 2015). Other climate application areas include downscaling (e.g., De Haan, Kanamitsu, De Sales, and Sun 2015; De Sales and Xue 2010), identifying believable scales for climate projections (e.g., Kwiotkowski, Halloran, Mumby, and Stephenson 2014), and identifying model bias (Siongco, Hohenegger, and Stevens 2014; Yorgun and Rood 2014). In the context of extreme values, the propinquity model introduced in Gilleland, Brown, and Ammann (2013)[2] lends itself particularly well to spatial verification comparisons (e.g., Gilleland, Bukovsky, Williams, McGinnis, Ammann, Brown, and Mearns 2016).

1.1. Other spatial analysis packages in R

Spatial analysis is a fairly broad topic that includes many types of applications. There are, subsequently, many other spatial analysis packages available for R. Various methods and classes for analyzing spatial data are provided by **sp** (Pebesma and Bivand 2005; Bivand, Pebesma, and Gomez-Rubio 2013), while **sf** (Pebesma 2018) is a modern implementation and standardization for some of **sp**, particularly for accessing vector data, that is supported by the R Consortium (`https://www.r-consortium.org`). The package **spacetime** (Pebesma 2012; Bivand *et al.* 2013) further extends the methods and classes of **sp** to space-time data analysis. The **raster** (Hijmans 2021) pacakge contains visualization and some image analysis methodology.

Several geostatistical spatial analysis packages that are available in R contain a wide array of spatial methods, such as **spatial** (available with base R; Venables and Ripley 2002),**gstat** (Gräler, Pebesma, and Heuvelink 2016), **fields** (Nychka, Furrer, Paige, and Sain 2017), **geoR** (Diggle and Ribeiro Jr. 2006), **spBayes** (Finley, Banerjee, and Carlin 2007; Finley, Banerjee, and E.Gelfand 2015), **ramps** (Smith, Yan, and Cowles 2008), **FRK** (specifically for large data sets; Sainsbury-Dale, Zammit-Mangion, and Cressie 2021; Zammit-Mangion and Cressie 2021), **INLA** (`https://www.r-inla.org/`; Rue, Martino, and Chopin 2009; Lindgren, Rue,

[2]The term propinquity model was introduced later. See `http://n2t.net/ark:/85065/d7tm7f52` last downloaded on 1 March 2021.

and Lindström 2011; Lindgren and Rue 2015), and **spatstat** (Baddeley, Rubak, and Turner 2015).

Other geostatistical spatial packages in R include: **RandomFields** (for simulating spatial fields; Schlather, Malinowski, Menck, Oesting, and Strokorb 2015; Schlather, Malinowski, Oesting, Boecker, Strokorb, Engelke, Martini, Ballani, Moreva, Auel, Menck, Gross, Ober, Ribeiro, Ripley, Singleton, Pfaff, and R Core Team 2022), **SpatialExtremes** (for analyzing extreme values spatially; Ribatet 2020), **CompRandFld** (a set of procedures for using likelihood and non-standard likelihood methods; Padoan and Bevilacqua 2015), **geospt** (analysis and design of optimal sampling networks; Melo, Santacruz, and Melo 2012), **rtop** (interpolation for data with irregular spatial support; Skoien, Bloschl, Laaha, Pebesma, Parajka, and Viglione 2014).

SpatialVx is more specifically aimed at making direct comparisons between two spatial fields than any of these other packages. An exception is **spatstat**, and many of these functions are directly imported to **SpatialVx**, but the specific purpose of verifying weather and climate forecasts is unique to **SpatialVx** and therefore is much more specialized than any other package to this author's knowledge for this purpose.

Operational forecast verification requires considerable heavy lifting and an ability to work with atmospheric data sets, such as GRIB files,[3] as well as an ability to digest extremely large files over many instances. Therefore, a software package meant for operational use that included some of the new spatial methods was introduced, called the Model Evaluation Tools (Brown, Jensen, Halley Gotway, Bullock, Gilleland, Fowler, Newman, Adriaansen, Blank, Burek, Harrold, Hertneky, Kalb, Kucera, Nance, and Wolff 2021) and based in C++, was developed. **SpatialVx** was originally developed for the personal use of the author in order to be able to test the different approaches more easily. However, it became clear that there was a demand for more research-oriented software, so **SpatialVx** was submitted to CRAN for general use.

1.2. Package dependencies

SpatialVx depends on several packages and imports from several more, most notably, **spatstat** (Baddeley and Turner 2005; Baddeley *et al.* 2015) and **smoothie** (Gilleland 2013b). The former includes considerable functionality for analyzing spatial fields and streamlined the development of **SpatialVx**. The latter was originally a collection of functions included in **SpatialVx** but was ultimately made into its own package because these functions are useful more generally than in comparing spatial fields; it contains functions for executing spatial convolution filters using the fast Fourier transform, including a collection of kernel choices. The package **fields** (Nychka *et al.* 2017) is also used extensively because it provides complementary spatial analysis routines to those in **spatstat** along with other useful tools.

The other dependencies are **smatr** (Warton, Duursma, Falster, and Taskinen 2012) for analyzing some feature-based properties, and **turboEM** (Bobb and Varadhan 2020) for improving the computational efficiency with the Gaussian Mixture Model method's estimation (Lakshmanan and Kain 2010). The imported packages are **distillery** (Gilleland 2020a,b, primarily for some useful method functions), **maps** (Becker, Wilks, Brownrigg, Minka, and Deckmyn 2018, for the ability to add maps to graphics), **boot** (Canty and Ripley 2020; Davison and Hinkley 1997), **CircStats** (Lund and Agostinelli 2018), **fastcluster** (Müllner 2013) and **waves-**

[3]See https://www.cpc.ncep.noaa.gov/products/wesley/reading_grib.html (last downloaded on 4 March 2021) for more information on GRIB files.

lim (Whitcher 2020).

1.3. Scope of this tutorial

There is a wealth of functionality in **SpatialVx** for instigating use of several of the proposed methods. For this treatment, however, only a relatively small subset of the more mature functions is addressed, as well as some lesser developed functions that help to shed light on specific challenges, as well as circumstances to consider when verifying high-resolution verification sets. For a complete summary of the functions included in the package, see `https://projects.ral.ucar.edu/icp/SpatialVx/` (last downloaded on 4 March 2021) under the "Contents" tab.

Conspicuously absent from this document are the filter-based methods. The function `hoods2d` provides many of the "neighborhood" sub-category of these methods (see Ebert 2008, for a thorough review of these methods). The help file (use `?hoods2d`) should prove adequate for learning how to apply this function, particularly after reading Sections 2 and 4 here.

Wavelet methods that were proposed early on (Briggs and Levine 1997; Casati, Ross, and Stephenson 2004; Casati 2010) are available through the functions `waverify2d` and `waveIS`, which make heavy use of **waveslim** (Whitcher 2020).[4] Weniger, Kapp, and Friederichs (2016) provide a thorough review of these and other wavelet methods, as well as introducing a new method (see also Buschow, Pidstrigach, and Friederichs 2019; Buschow and Friederichs 2021; Brune, Buschow, and Friederichs 2021). **SpatialVx** does not currently contain any newer wavelet methods that have been proposed.

The presentation, here, follows a path beginning with field deformation, specifically image warping, in Section 3. These capabilities are limited in **SpatialVx** but are nevertheless useful for understanding some of the issues invovled with all methods. Section 4 continues with the displacement-based topic to the feature-based setting. The first feature-based approach follows logically from the discussion of image warping as it can be seen as a crude version of the methodology, although it is considerably more flexible and can provide a wealth of other information. The next logical step goes from the rather complicated displacement-based approaches to the remarkably simple spatial alignment, or spatial dissimilarity, measures in Section 5. This topic is followed by the closely related field qualities in Section 6, which also present simple summaries but only over individual fields, rather than comparing one field to another directly; variograms are also discussed in this section. The tutorial then considers some more advanced topics, such as the spatial prediction comparison test (Section 7), ideas for analyzing performance using the bearing along with other previously discussed measures in conjunction with circle histograms as well as an approach to point-to-grid verification (Section 8).

1.4. Overview of SpatialVx

As detailed in Section 2 below, the main class in **SpatialVx** is a list object containing the verification set dubbed `"SpatialVx"`. The original intention was to have every function work on this class object in order to simplify code, and much of the functionality in the package works this way. Some newer functions, and some less used functions, do not always use this class of function, and some allow for both this class of object and using pairs of matrices

[4] Again, the help files for these functions should suffice for learning how to apply these functions.

separately. For the most part, however, the main methods of **SpatialVx** work on `"SpatialVx"` class objects. Therefore, it is a good idea to first learn how to set up this type of object (see Section 2).

This document follows a path of logic for verifying weather forecasts that begins with the "obvious" choice and why this choice may not be ideally suited to the task, and leads through different alternatives of varying complexity. While there are main functions for each broad category of spatial verification, the main functions for this author's recommended methods include:

1. Spatial dissimilarity measures: `locmeasures2d` calculates several of the spatial dissimilarity measures from Baddeley (1992a,b), as well as `Gbeta` and `TheBigG` for the new dissimilarity measures proposed by Gilleland (2021); the former operates on `"SpatialVx"` objects while the latter are newer and do not yet operate on these objects, as they will at some point be folded into `loceasures2d`.

2. Feature-based methods: `FeatureFinder` for identifying connected components within a field, `centmatch`, `deltamm` and `minboundmatch` for pairing identified features within and across fields, as well as a host of functions for analyzing features either individually or for paired features: `FeatureAxis`, `FeatureComps`, `FeatureMatchAnalyzer`, `FeatureProps`, `FeatureTable`, and `interester`. The initial `FeatureFinder` function works on `"SpatialVx"` objects and outputs an object of class `features` while the pairing functions work on the output of this function which inherits some information from the original object. Subsequent analysis functions work on either the output from the pairing functions or `FeatureFinder` depending on whether they require features to be paired or not; e.g., `FeatureAxis` and `FeatureProps` work on `features` objects directly, whereas `FeatureComps`, `FeatureMatchAnalyzer`, and `FeatureTable` work on `"matched"` objects that are output by the pairing functions. The function `interester` follows the methodology of Davis, Brown, and Bullock (2006a) in pairing and summarizing features, so works on `"features"` objects.

3. The geometric indices proposed by AghaKouchak, Nasrollahi, Li, Imam, and Sorooshian (2011) are carried out by: `Cindex`, `Sindex`, `Aindex`. Because these indices work on individual fields, rather than comparing fields directly, they do not work on `"SpatialVx"` objects.

4. The spatial prediction comarison test proposed by Hering and Genton (2011) is carried out by: `lossdiff`, `empiricalVG` and `flossdiff`. These newer functions do not work on `"SpatialVx"` class objects.

2. Notation and main example data

To be consistent with the argument nomenclature of **SpatialVx** functions, let X denote the observed field and $\hat{X}$ that of the forecast. Because they are spatial fields, let $\mathbf{s} = (x, y) \in \mathcal{D}$, where $\mathcal{D}$ is the spatial domain and $\mathbf{s}$ represents the coordinates for a grid point. Often binary images need to be created from the $X(\mathbf{s})$ and $\hat{X}(\mathbf{s})$ fields. Denote the binary fields by $I_X(\mathbf{s})$ and $I_{\hat{X}}(\mathbf{s})$, respectively.

The shortest distance from a grid point $\mathbf{s}$ to the nearest point belonging to a set of points $A \subset \mathcal{D}$ is of particular interest and is denoted by $d(\mathbf{s}, A)$. A distance map provides $d(\mathbf{s}, A)$ at every point $\mathbf{s} \in \mathcal{D}$, and the existence of fast compuational algorithms allows these maps to be obtained quickly even for the very large domains encountered in weather forecast verification. All distance maps employed in **SpatialVx** use the function `distmap` from **spatstat**. Most of the methods in Section 5 are computed using these maps, as well as the minimum separation distance in Section 4.1.

One dataset from the ICP that remains included with **SpatialVx** is the 1 June 2005 case. This case was the primary case employed by many of the method developers. It consists of the Stage II analysis product as the "observed" field along with three models denoted `wrf4ncar`, `wrf4ncep` and `wrf2caps`. Several of the functions in **SpatialVx** operate on a special class of object that is created with the `make.SpatialVx` function.

```
R> data( "obs0601" );
R> data( "wrf4ncar0531" );
R> data( "wrf4ncep0531" );
R> data( "wrf2caps0531" );
R> data( "ICPg240Locs" );
R> loc <- ICPg240Locs;

R> hold <- make.SpatialVx( obs0601, list( wrf4ncar0531, wrf4ncep0531, wrf2caps0531 ),
+                          loc = loc, projection = TRUE, map = TRUE, loc.byrow = TRUE,
+                          field.type = "Precipitation", units = "mm/h",
+                          data.name = "ICP real case valid 1 June 2005 0 UTC",
+                          obs.name = "Stage II analysis",
+                          model.name = c( "Model 1", "Model 2", "Model 3" ) );

R> data( "obs0601" );
R> data( "wrf4ncar0531" );
R> data( "wrf4ncep0531" );
R> data( "wrf2caps0531" );
R> data( "ICPg240Locs" );
R> loc <- ICPg240Locs;

R> hold <- make.SpatialVx( obs0601, list( wrf4ncar0531, wrf4ncep0531, wrf2caps0531 ),
+                      thresholds = c( 0.1, 2.1, 5.1, 20.1 ),
+                 loc = loc, projection = TRUE, map = TRUE, loc.byrow = TRUE,
+                 field.type = "Precipitation", units = "mm/h",
+                 data.name = "ICP real case valid 1 June 2005 0 UTC",
+                 obs.name = "Stage II analysis",
+                 model.name = c( "Model 1", "Model 2", "Model 3" ) );
R>
R> lr <- apply( loc, 2, range );
R> l <- list( lon = matrix( loc[, 1], 601, 501, byrow = TRUE ),
+          lat = matrix( loc[, 2], 601, 501, byrow = TRUE ) );
R> zl <- range( c( c( obs0601 ), c( wrf4ncar0531 ), c( wrf4ncep0531 ),
+                c( wrf2caps0531 ) ), finite = TRUE );
```

```
R> library( "colorspaces" );
R> cl <- c( gray( seq( 1, 0,, 8 ) ), sequential_hcl(16, palette = "Hawaii") );
R> par( mfrow = c(2,2) );
R> map( type = "n", xlim = lr[,1], ylim = lr[,2], xlab = "", ylab = "" );
R> poly.image( l$lon, l$lat, obs0601, col = cl, zlim = zl,
+          xlab = "", ylab = "", axes = FALSE, add = TRUE );
R> map( database = "state", add = TRUE );
R> map.axes()

R> map( type = "n", xlim = lr[,1], ylim = lr[,2], xlab = "", ylab = "" );
R> poly.image( l$lon, l$lat, wrf4ncar0531, col = cl, zlim = zl,
+                     xlab = "", ylab = "", axes = FALSE, add = TRUE );
R> map( database = "state", add = TRUE );
R> map.axes()

R> map( type = "n", xlim = lr[,1], ylim = lr[,2], xlab = "", ylab = "" );
R> poly.image( l$lon, l$lat, wrf4ncep0531, col = cl, zlim = zl,
+                     xlab = "", ylab = "", axes = FALSE, add = TRUE );
R> map( database = "state", add = TRUE );
R> map.axes()

R> map( type = "n", xlim = lr[,1], ylim = lr[,2], xlab = "", ylab = "" );
R> poly.image( l$lon, l$lat, wrf2caps0531, col = cl, zlim = zl,
+                     xlab = "", ylab = "", axes = FALSE, add = TRUE );
R> map( database = "state", add = TRUE );
R> map.axes()
R> image.plot( obs0601, legend.only = TRUE, horizontal = TRUE, col = cl,
+          zlim = zl, legend.mar = 32 );
```

Figure 3 shows the resulting graphics displayed from the `poly.image` (from package **fields**; Nychka *et al.* 2017) and `map` (from package **maps**; Becker *et al.* 2018) lines above. Calling `map` first sets up the device so that the correct spatial aspect ratio will be graphed, `poly.image` allows for maintaining the grid projection (note the use of longitude and latitude coordinate matrices in its call) and `image.plot` adds the color legend bar. While none of these functions is from package **SpatialVx**, it is often useful to be able to make these types of graphs when evaluating high-resolution verification sets, and it is not obvious how to make them. Subjective evaluation by the human eye is culpable of biased judgment (see, e.g., the subjective evaluation of the nine ICP real cases in Ahijevych *et al.* 2009). Choice of color scheme (see `cl` assignment above) can greatly influence subjective assessments. Here, the **colorspace** package (Zeileis, Fisher, Hornik, Ihaka, McWhite, Murrell, Stauffer, and Wilke 2020) is used, which includes several color schemes that are designed to be easier for the human eye to differentiate (see Stauffer, Mayr, Dabernig, and Zeileis 2009, for information on color choices for meteorological variables). However, other factors beyond color can be important, as well. In particular, the range of values in this example is large, reaching around 120 mmh^{-1} but very few values exceed, e.g., 20 mmh^{-1}. The `zlim` argument can be adjusted to focus in on lower (or higher) values. However, in the present example, the spatial domain is very large and the precipitation

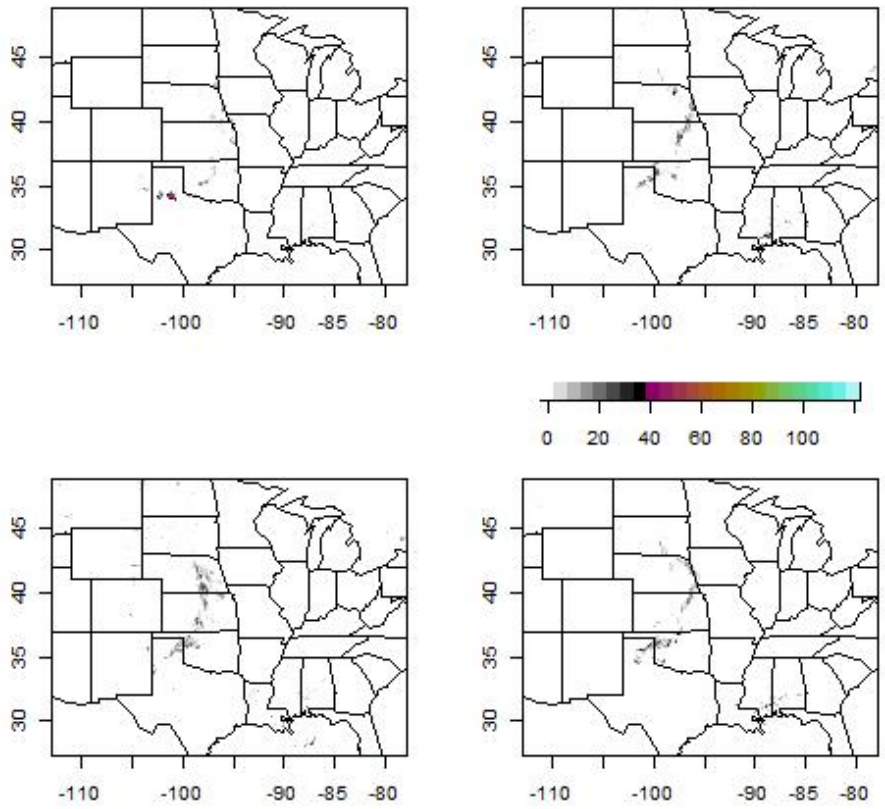

Figure 3: ICP real case for 1 June 2005. Top left is the Stage II analysis, top right is `wrf4ncar`, bottom left is `wrf4ncep` and bottom right is `wrf2caps`. Variable is precipitation (mmh^{-1}).

areas are relatively small, making it difficult to see these important features in the maps.

3. Field deformation

In terms of applying a grid-point-by-grid-point error summary that makes sense for high-resolution model verification, the most natural choice is to employ a field deformation approach. The methodology accounts for both intensity and displacement errors simultaneously, and information about the amount of deformation can supplement, or be included with, summaries of the intensity errors. The percent reduction in intensity error can also be of interest. Ebert and McBride (2000) proposed a particularly appealing breakdown of error into components pertaining to displacement, volume and shape, as well as total error.[5]

In fact, some of the earliest proposed methods belonged to the deformation category (e.g. Sampson and Guttorp 1992; Hoffman, Liu, Louis, and Grassoti 1995; Nehrkorn, Hoffman,

[5]Their method is a feature-based approach with a translation-only deformation on individual features matched between fields via a proximity rule.

Grassotti, and Louis 2003), although none of them explored the idea in any detail. Keil and Craig (2007, 2009) proposed a pyramid-scheme deformation approach, analogous to a wavelet-type decomposition, that results in simple summary measures that combines the amount of deformation with the intensity errors. **SpatialVx** includes only very limited functionality for performing deformation verification: the optical flow procedure[6] proposed by Marzban and Sandgathe (2010) and the image warping approach of Gilleland *et al.* (2010c). The former is only applicable when the forecast is nearly identical to the observation field. The latter requires numerically optimizing over many parameters and subsequently requires highly accurate initial values.

Only a brief example of image warping (Dryden and Mardia 1998) in **SpatialVx** is given in order to facilitate a comparison of the other methods discussed here because the optimization for this method using this package is difficult, so not generally recommended. The image warping model is given by $X(\mathbf{s}) = \hat{X}(\mathbf{W}(\mathbf{s})) + \varepsilon = \hat{X}(W_x(x,y), W_y(x,y)) + \varepsilon(\mathbf{s})$, where $\mathbf{W}$ represents the warping function whose parameters consist of control points (aka landmarks) and ε is additional error.[7] For the thin-plate-spline image warp procedure included with **SpatialVx**,

$$W_x(x,y) = a_{x,0} + a_{x,1} \cdot x + a_{x,2} \cdot y + \sum_{i=1}^{n_c} d_{x,i} \cdot U(\|\mathbf{p}_{X,i} - (x,y)\|), \tag{1}$$

where $\mathbf{p}_{X,i}$ is the i-th control point for $X(\mathbf{s})$, called the zero-energy control points, $U(r) = r^2 \log r$, $\sum_{i=1}^{n_c} d_{x,i} = \sum_{i=1}^{n_c} x \cdot d_{x,i} = \sum_{i=1}^{n_c} y \cdot d_{x,i} = 0$. The definition for $W_y(x,y)$ is similar. The control points for $\hat{X}(\mathbf{s})$ are the parameters and are given by $\mathbf{p}_{\hat{X},i}$, $i = 1, \ldots, n_c$, where n_c is the number of control points. The value of n_c must be chosen a priori by the user. The larger n_c is, the more intricate the warp will be, and the more difficult it will be to estimate. The estimates for $\mathbf{p}_{\hat{X},i}$, called the one-energy control points, are obtained through numerical optimization of a likelihood, or other objective function. The default objective function is a simple sum of squares but other choices include a likelihood that assumes a Gaussian distribution for $\varepsilon(\mathbf{s}) = \hat{X}(\mathbf{s}) - X(\mathbf{s})$, (use `lossfun = "Qgauss"` and `grlossfun = "defaultQgauss"`), as well as a Gaussian mixture as employed by Gilleland *et al.* (2010c) (use `loss = "Qnonzero"` and `grlossfun = "defaultQnonzero"`); more advanced users can write and use their own loss functions to use with the `lossfun` and `grlossfun` arguments. The parameters $a_{x,0}, a_{x,1}, a_{x,2}, d_{x,1}, \ldots, d_{x,n_c}, a_{y,0}, a_{y,1}, a_{y,2}, d_{y,1}, \ldots, d_{y,n_c}$ are derived from the resulting control points and bending energy matrix, from which the minimum bending energy can be derived (see Gilleland, Chen, DePersio, Do, Eilertson, Jin, Lang, Lindgren, Lindström, Smith, and Xia 2010b, for more details).[8]

In image analysis, these landmarks are readily identifiable. For example, a portrait of a person will include such landmarks as the point of the nose, the ends of the lips, the top of the forehead, etc. For a field consisting of an atmospheric variable, however, landmarks are not as easily come by. Therefore, they must be estimated and for a large field, typically $n_c \approx 200$ must be found (Gilleland *et al.* 2010c). Subsequently, it is a challenging optimization

[6] This code, called `OF`, was contributed to **SpatialVx** by Caren Marzban.

[7] The specific distribution for $\varepsilon(\mathbf{s})$ depends on the modeling assumptions, which will vary depending on the field variable of interest. This model is not the only possible choice but represents the basic idea in its simplest form.

[8] The bending energy is a measure of the amount of non-linear deformation required to map the zero-energy control points to the one-energy counterparts.

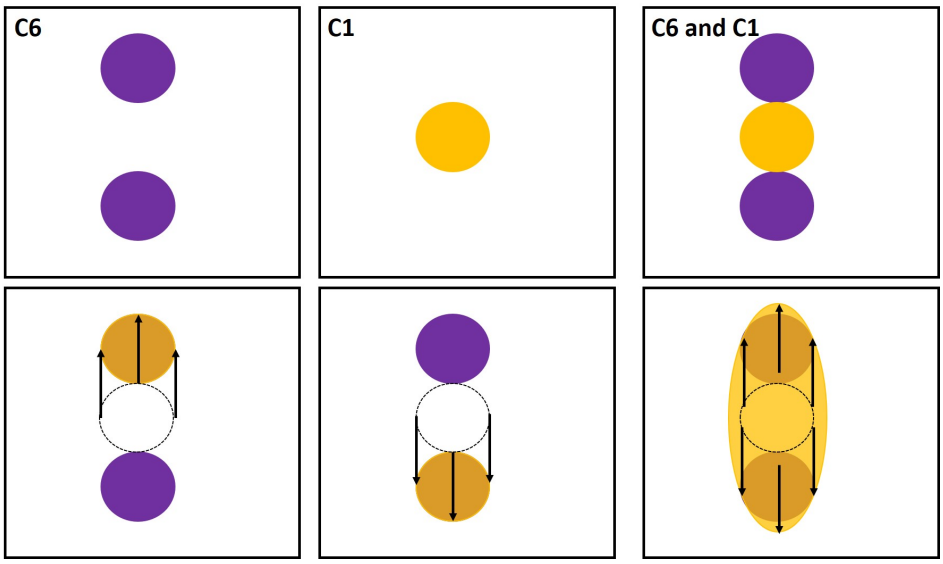

Figure 4: Illustration of the C1 and C6 comparison from Gilleland *et al.* (2020). Top left shows C6, top middle C1 and top right shows the two super-imposed on top of each other. The bottom row represents three sensible possibilities for deforming the C1 field to better match the C6 field. The first simply translates C1 up to be perfectly aligned with the top circle in C6 and the second translates down to match the lower circle of C6. Finally, the third scenario stretches C1 in order to overlap with both of the C6 circles; a more intricate warp might use additional control points to also try to squeeze the resulting ellipse in the middle so as to reduce the non-overlapped part between the two circles of C6.

problem and the function `warper`, which does the heavy lifting, requires very precise initial values.

Figure 4 illustrates one challenge with any displacement method applied to meteorological fields. In this scenario, the gold circle (C1) perfectly matches each of the two purple circles (C6) and each purple circle is translated by the same amount from the gold circle but in different directions. A simple translation up (bottom left) or down (bottom middle) make for two different deformations that result in the same amount of displacement and reduction in error so that there is not a unique deformation that is optimal.

On the bottom right of the figure, another possible deformation tries to overlap the gold circle with both purple circles. It might be argued that this deformation is less ideal than the first two but with more control points it migth be possible to better approximate the purple circles using only the one gold circle. The result could be a much better reduction in error but is at the expense of greater displacements and bending energy. Determining which of these, or other possible deformations, is ideal may not always be possible. Operationally, there would be no way to know if resulting deformations are sensible or not, or meet user needs. That is not to say that the method cannot be useful in this setting but these types of challenges must be understood.

Figures 5 and 6 demonstrate two different image warps applied to the ICP geometric observed case against the fourth forecast. The former re-scales the fourth forecast (the correct deformation) and the latter applies a rotation. Both result in about the same reduction in error

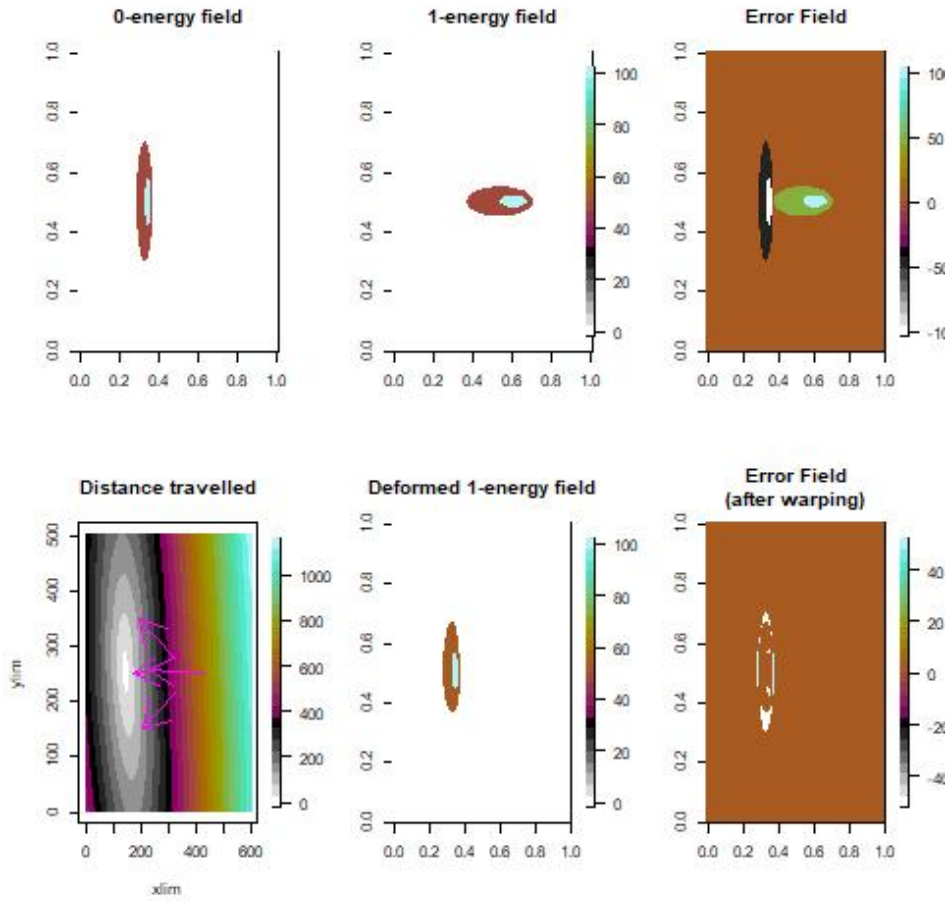

Figure 5: Example of an image warp applied to the verification set including the observed ICP geometric cases and its fourth "forecast." The transformation is a scale transformation along with a translation to the left.

($\approx$ 57% and 62%, respectively) and both have similar amounts of deformation. Note that these deformations involve only four control points that "control" the entire fields' deformation.

These figures can be approximately reproduced with the following code. As mentioned above, the main function for performing image warping, `warper`, requires a very good initial guess. Therefore, `iwarper` is used, which allows the user to interactively select the control points. The printout gives the estimates of the parameters in Eq (1) in addition to the original root-mean square error (RMSE_0) and the resulting RMSE_1 obtained after deforming the forecast field, as well as the percent reduction in error $(\text{RMSE}_0 - \text{RMSE}_1)/\text{RMSE}_0 \cdot 100\%$. The parameters $d_{x,1}, \ldots, d_{y,n_c}$ would be too numerous to display in the printout but are summarized through the minimum bending energy term that is displayed.

The following code can be used to approximately reproduce Figures 5 and 6. It is approximate because the user must select the control points by hand. That is, once `iwarper` is called, an image of the observed field will appear and the user is prompted to select the zero-energy control points, which is carried out by clicking on the image wherever a control point is desired.

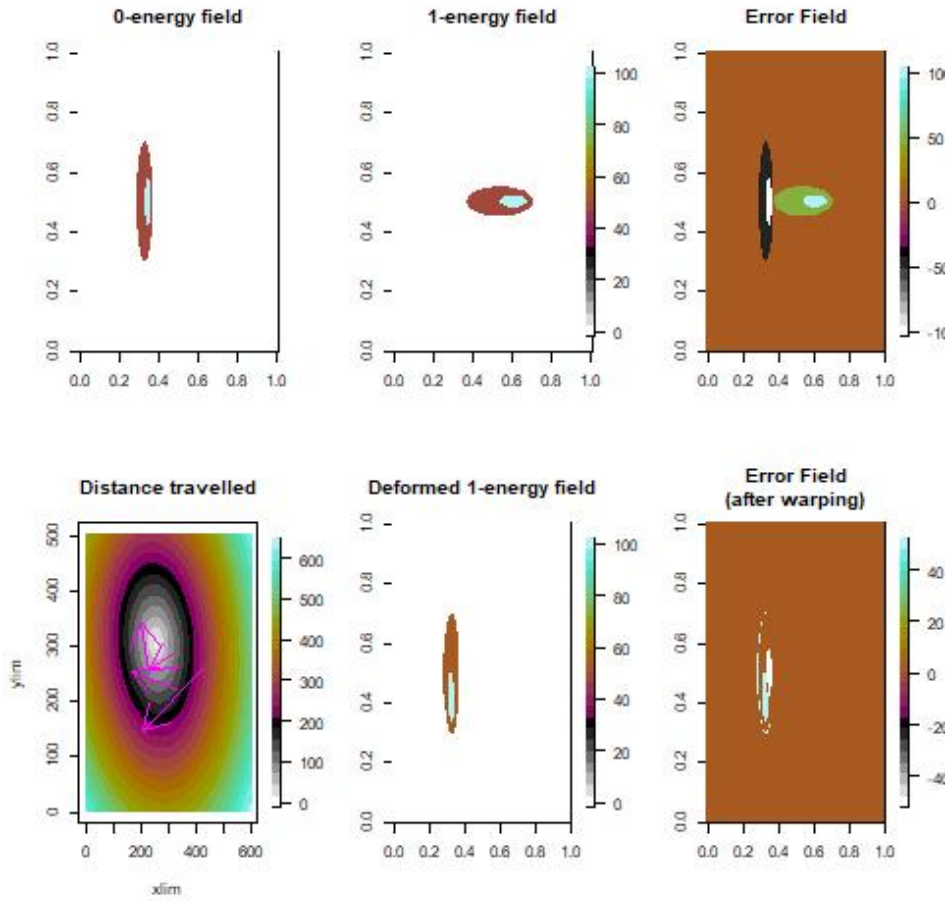

Figure 6: Same as Figure 5 except a different deformation is applied. In this case, the transformation is a rotation along with a translation to the left.

After `nc` points are selected (in this case, the default of four points), then the forecast image is drawn and the user is again prompted to select the one-energy control points. Once they are selected, the additional images are drawn. X is considered the zero-energy field because it is not deformed and $\hat{X}$ the one-energy field because it is deformed. Figure 7 demonstrates how the control points can be chosen to enact the scale change v. rotation transformations. Note that when the "observed" image is drawn a light dashed contour of the forecast image is also displayed to aid in choosing potential control points (similarly for when choosing the forecast control points).

```
R> library( "SpatialVx" );
R> library( "ICPtestcases" );

R> data( "geom000" );
R> data( "geom004" );
R> look <- iwarper( x0 = geom000, x1 = geom004 );
```

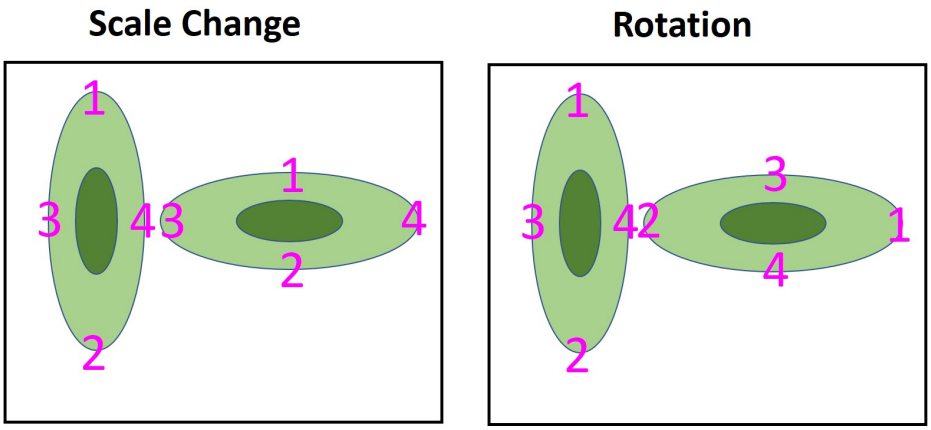

Figure 7: Illustration showing approximate chosen control points to enact the scale (cf. Figure 5) v. rotation (cf. Figure 6) transformations for the fourth ICP geometric case compared against the "observation." Left vertical ellipses represents the "observation" and the right horizontal represents the "forecast."

```
R> look <- warper( Im0 = geom000, Im1 = geom004, p0 = look$p0, init = look$p1,
+              s = cbind( rep( 1:601, 501 ), rep( 1:501, each = 601 ) ) );
R> plot( look, col = cl );
R> summary( look );

RMSE0             RMSE1 % RMSE Reduction      intercept(x)
     13.83465109        5.22889319       62.20437252     -424.73336838
          a_x(x)             a_x(y)      intercept(y)           a_y(x)
      3.30560675       -0.01051766      187.46472674       -0.11732001
          a_y(y)    Bending Energy
       0.34903801         0.04662763
```

Clearly, the image warping code available in **SpatialVx** is not highly evolved. However, there are situations when only a few fields need to be deformed and a by-hand approach can be taken, which along with some trial-and-error can lead to good deformations (see, e.g., Gilleland *et al.* 2016, who apply this by-hand approach on simulated fields from a propinquity model approach to compare future (modeled) climates to present (modeled) climates). Moreover, `iwarper` can be highly useful in understanding how different control point configurations lead to particular deformations and thereby lead to a greater understanding of the image warping method.

4. Feature-based Applications

Feature-based methods are closely linked with field deformation methods as both directly account for the spatial displacement, or misalignment, of the field variable. As already mentioned, the approach of Ebert and McBride (2000) applies field deformation to individual feature pairs.[9] The Structure Amplitude Location (SAL) approach introduced by Wernli,

[9]This method is not currently available in **SpatialVx**.

Paulat, Hagen, and Frei (2008) summarizes performance distributionally using individual features. The cluster analysis approaches of Marzban and Sandgathe (2006) and Marzban and Sandgathe (2008) can be considered to be feature-based methods.[10]

With any of the feature-based methods, features within the fields need to be identified. The method employed for identifying features in **SpatialVx** uses functions from **spatstat** to identify connected components. A connected component is derived from a binary field whereby a grid point is connected to another grid point if the two are adjacent and both have value one. They are sometimes called isolated clusters (e.g. AghaKouchak *et al.* 2011) and are characteristic of contiguous "blobs" within a field.

4.1. Method for Object-based Diagnostic Evaluation (MODE)

The Method for Object-based Diagnostic Evaluation (MODE; Davis *et al.* 2006a; Davis, Brown, and Bullock 2006b; Davis *et al.* 2009)[11] can be thought of as a brute-force field deformation approach because after identifying individual features in each field, a type of cluster analysis is performed in order to (i) possibly merge features within a field, and (ii) match features between the fields, and a fuzzy-logic algorithm then provides a summary of the amount of "deformation" and intensity errors. The intensity errors are determined, for example, using the upper quartile of intensities within features. The `interester` function from **SpatialVx** will carry out this fuzzy-logic algorithm.

There are three main steps involved with MODE and each step has possible alternative methods for carrying them out. **SpatialVx** is written to allow for considerable flexibility for these steps, and allows advanced users to easily write their own routines for a particular step and seemlessly incorporate them into the broader methodology. These main steps are:

1. Identify features within each of the fields, $X(\mathbf{s})$ and $\hat{X}(\mathbf{s})$.
2. Merge features within each of $X(\mathbf{s})$ and $\hat{X}(\mathbf{s})$ and/or match these features between the two fields.
3. Analyze the features either individually for each field or by comparing matched features.

Step 2 could be considered to be two steps, 2a (merge) and 2b (match), but both are optional depending on the user's goals, and in some cases the two are performed simultaneously, as with the method for merging and matching features proposed by Gilleland, Lee, Halley Gotway, Bullock, and Brown (2008). The last step again depends on the specific user's goals as to whether a direct comparison between the fields is desired or not, or whether there is any benefit to a distributional type of assessment of each field on its own.

There is one function, called `FeatureFinder`, for identifying features with **SpatialVx** but it is highly flexible and allows for employing the method introduced in Davis *et al.* (2006a), Wernli *et al.* (2008), Nachamkin (2009) and Lack *et al.* (2010). Additionally, it is possible to remove features that are too large, in case only small-scale features are of interest, or to separate out the small from the large features in an analysis.

[10] The function `clusterer` performs the method of Marzban and Sandgathe (2006) and `CSIsamples` performs the one from Marzban and Sandgathe (2008). The latter of which was contributed by Caren Marzban. Both utilize functions from **fastcluster** for computational efficiency.

[11] Note that the term "feature" is often referred to as "object" in the literature. In order to avoid confusion with an R object, the term "feature" is used here.

There are four functions available for step 2: (i) `centmatch`, which employs the centroid-distance procedure of Davis *et al.* (2006a), (ii) `deltamm`, which applies the procedure proposed by Gilleland *et al.* (2008), (iii) `minboundmatch`, which takes a nearest boundary approach as proposed by Ebert and McBride (2000), and (iv) `MergeForce` is not a merging function itself but solves a consistency issue with these other functions. That is, consistent with Davis *et al.* (2006a), `centmatch` only supplies implicit merges but does not actually merge the features within the resulting R object. In this case, `MergeForce` produces an R object with the features fully merged. For `deltamm`, the cluster-analysis type of approach originally propsed stops short of making a second pass for computational efficiency reasons. Here, `MergeForce` makes a second pass in order that features that are merged in one field can be matched with features that have been merged in the other.

There are six functions available for step 3. The first, `interester` mentioned above, performs the fuzzy-logic interest summary of MODE. `FeatureTable` uses features to produce contingency-table summaries along the traditional verification lines as proposed by Davis *et al.* (2009). `FeatureMatchAnalyzer` and `FeatureComps` compute several feature-comparison summaries for matched features, including: (i) centroid distance, (ii) difference in orientation angles, (iii) area ratio, (iv) intersection area, (v) bearing from the forecast feature to that of the observed, as well as (vi) several binary image metrics discussed in Section 5. `FeatureProps` calculates properties[12] of individual features, such as centroid, area, major axis angle (using **smatr**) and intensity.

Some **SpatialVx** functions utilize a special object class called `"SpatialVx"` which simplifies coding for the user in some cases; usually `x` is the main argument in this case. Other functions operate on individual matrices, when `X` and `Xhat` are the arguments. In some cases there is a method function for both possibilities. `FeatureComps` works on a specific type of list output from `solutionset` of **spatstat**, allowing users to work more directly with feature comparisons while `FeatureMatchAnalyzer` works on output from functions like `centmatch` and `deltamm`. Generally, functions in the first step work on `"SpatialVx"` objects, which can be created using `make.SpatialVx`, and return an object of class `"features"` that functions from subsequent steps can work on. In a similar fashion, functions from step 2 return objects of class `"matched"` that functions from step 3 can work on. This approach allows one function to take output from functions in the previous step without having to continually pass some of the arguments; these arguments are passed through the objects. This approach simplifies coding for the user and also allows avanced users to write alternative functions in any of the steps in such a way that the output can be used by existing functions in the other steps, provided the user is careful to output the right type of object(s) from their own function(s).

The following code demonstrates how to do MODE-type analyses in **SpatialVx** using the 1 June 2005 real case from the ICP.

```
R> look <- FeatureFinder( hold, smoothpar = 10.5, thresh = 5 );
R> plot( look );
```

Figure 8 displays the graphic produced from the above `plot` command.

```
R> summary( look );
```

[12] Davis *et al.* (2006a) refer to "properties" as "attributes." The latter term is avoided here in order not to confuse the reader with the R term "attributes," which refers to a specific mechanism used with an R object.

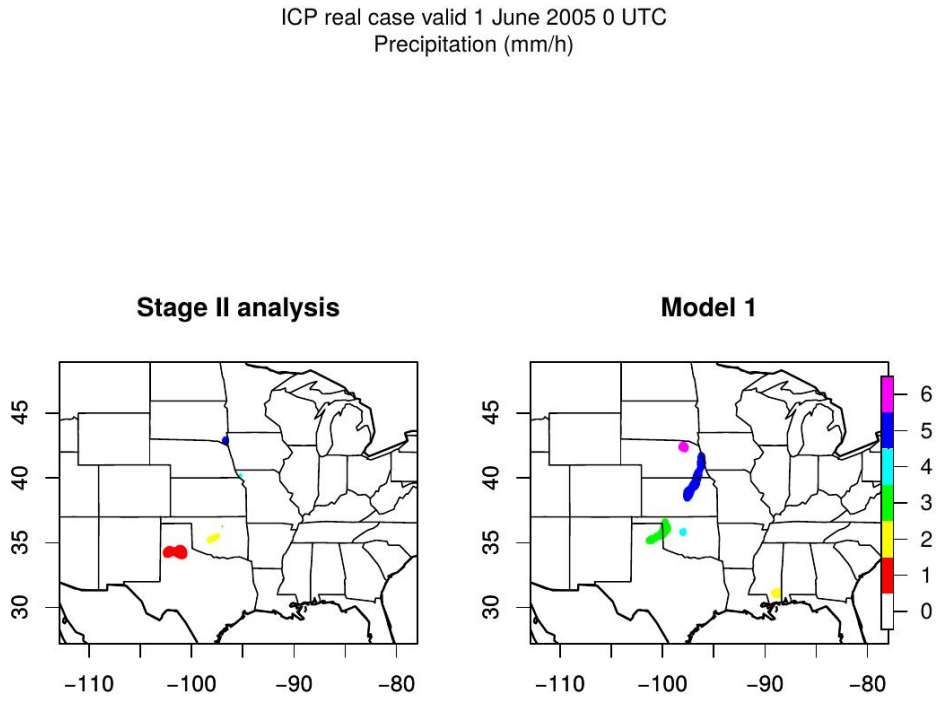

Figure 8: Features identified using a convolution (disc kernel) radius of 10.5 grid points with a threshold of 5 mmh^{-1}. Features are not merged or matched at this stage.

```
Verification field ( ) feature properties:
      centroidX centroidY area OrientationAngle AspectRatio Intensity0.25
[1,] -103.85129  39.82223 1312         63.64260   0.7393909         1.270
[2,] -101.66652  39.36492  390         51.09303   0.4864862         4.826
[3,] -100.16017  38.33127   12        138.25012   0.8471362         3.810
[4,]  -93.69013  38.67983   61        140.22244   0.8515019         5.080
[5,]  -90.58927  36.35705  186        138.09386   0.8732308         7.112
     Intensity0.9
[1,]      47.6758
[2,]      17.0434
[3,]       4.3180
[4,]      13.9700
[5,]      11.9380

 Forecast field ( ) feature properties:
      centroidX centroidY area OrientationAngle AspectRatio Intensity0.25
```

```
[1,] -107.61698  38.24863  145         45.38543   0.8634194         3.810
[2,] -105.96166  38.97449  498         53.92560   0.8274923         1.016
[3,] -101.36072  38.86168 1371         41.92198   0.4833802         3.556
[4,] -100.93799  39.12403  230         40.26762   0.9370622         4.318
[5,]  -94.52596  37.85652 1842         15.17204   0.2746186         3.810
[6,]  -90.65989  38.94111  338        135.11026   0.8789192         3.302
     Intensity0.9
[1,]      26.3144
[2,]      36.4744
[3,]      20.8280
[4,]      20.8280
[5,]      19.8120
[6,]      21.9964
```

```
R> look;
```

```
FeatureFinder(object = hold, smoothpar = 10.5, thresh = 5)
[1] "ICP real case valid 1 June 2005 0 UTC\nPrecipitation (mm/h)"
[1] "ICP real case valid 1 June 2005 0 UTC"
[2] "Stage II analysis"
[3] "Model 1"
[1] "convthresh"
[1] "Convolution Threshold"

 5   Stage II analysis  features identified.
6   Model 1  features identified.
```

The `summary` method function for `"features"` objects produces statistics on the individual features identified in each field using the terminology of "verification" field to mean the "observation" (in this case the Stage II analysis product). In particular, it provides the centroid coordinates, area (in grid points squared), orientation angle (in degrees), aspect ratio, and the lower quartile and 0.9 quantile of precipitation amount within each feature.

The next line of code calls the `print` method function for `"features"` objects. The output gives useful information about the object, including the original function call, the observation and forecast field names (as input to `make.SpatialVx`), the name for the feature identification method used (here, the default convolution threshold approach), as well as the number of features identified in each field. Five "observed" features are identified and six forecast features. The threshold choice of 5 mmh^{-1} leaves only fairly intense precipitation amounts so that the feature sizes are relatively small relative to the size of the domain but clearly from Figure 8, the first model mostly over forecast these areas of intense precipitation, particularly in eastern Kansas and Nebraska.

The next step of a MODE-type of analysis is to merge features within each field and/or match them between fields. The following code enacts the method proposed by Gilleland *et al.* (2008). The first step will take some time to compute so setting the `verbose` argument to `TRUE` will display its progress to the screen.

```
R> look2 <- deltamm( look, N = 701, verbose = TRUE );
R> look2 <- MergeForce( look2 );
R> plot(look2);
R> summary( look2 );
```

The above code results in one unmatched forecast feature and all of the observed features are matched (none are merged in this case). The next step is to make comparisons using the matched features. A host of plots will be produced with the `plot` command below, so it is recommended to send them to a file instead of the interactive device (e.g., using the `pdf` and `dev.off` functions, as commented out in the example below).

```
R> res <- FeatureMatchAnalyzer( look2 );
R> summary( res );
R> # pdf( "<path>/<filename>.pdf" );
R> plot( res );
R> # dev.off();
```

```
 Partial Hausdorff Distance:
[1] 358.3644 265.0113 115.4278 145.6779 117.6488

 Pratt's Figure of Merit:
[1] 9.234583e-05 1.261031e-04 3.399951e-05 1.543438e-04 1.613622e-03

 Minimum Separation Distance:
[1] 301.12463 237.30742  49.15403 125.98264  16.93164

 Centroid Distance:
[1] 4.081250 4.312849 1.312498 7.261459 4.212587

 Angle Difference:
[1]  18.25718   2.83257  96.32814  99.95482 122.92182

 Area Ratio:
[1] 0.110518293 0.783132530 0.008752735 0.265217391 0.100977199

 Intersection Area:
[1] 0 0 0 0 0

 Bearing:
[1]  -60.54385  -81.95897 -119.10315  -92.21930 -114.33202

 Baddeley's Delta Metric:
[1] 218.00033 179.49531  57.84966  89.90572  69.94918

 Hausdorff Distance:
[1] 358.9509 265.5978 116.0143 146.0927 118.4784
```

Hausdorff distance, partial Hausdorff distance, Baddeley's Δ, and Pratt's Figure of Merit (FoM) are discussed in Section 5. Minimum separation distance is the shortest distance between the two features. What is worth noting now is that the centroid distance between the first pair of matched features is only about four grid points, but the minimum separation distance and Baddeley's Δ metric are both very large and are also in units of grid points.

Next, perform the contingency-table approach suggested in Davis *et al.* (2009).

```
R> tres <- FeatureTable( look2 );
R> summary( tres );

Feature-based Contingency Table
             hits            misses      fales alarms correct negatives
                5                 0                 1                89
[1] "ICP real case valid 1 June 2005 0 UTC"
[2] "Stage II analysis"
[3] "Model 1"

[1] "Normal Approximation"

                 95% lower CI   Estimate 95% upper CI
GSS                0.76388367 0.82407407   0.88426448
POD                0.99996080 1.00000000   1.00003920
false alarm rate  -0.01054494 0.01111111   0.03276717
FAR                0.16665333 0.16666667   0.16668000
HSS                0.86737299 0.90355330   0.93973361
```

As can be seen, the heavy lifting is in identifying, and even moreso, in merging/matching features. Subsequent analyses are fairly simple to obtain from the objects output from `FeatureFinder` and the merge/match functions. The original MODE fuzzy-logic interest value analysis can be duplicated using the `interester` function. Similar to `deltamm`, it requires making comparisons across all of the combinations of features in order to identify the highest interest combinations. Traditionally, the entire output is given, so that is also done by the code below but is not reproduced here.

The interest values are caclualted through a weighted combination of interest functions. The interest function, $f(x)$, for all of the property comparisons, except for area ratio, intersection area and FoM is given by the piece-wise linear function

$$f(x) = \begin{cases} 1 & \text{if } x \leq b_1 \\ a_0 + a_1 \cdot x & \text{if } b_1 < x \leq b_2 \\ 0 & \text{if } x > b_2, \end{cases}$$

where $a_1 = -1/(b_2 - b_1)$, $a_0 = 1 - a_1 \cdot b_1$, and the user can select b_1 and b_2 through the `b1` and `b2` arguments. These arguments, as well as `weights`, are length-eleven numeric vectors that line up with the `properties` argument. A particular property may be excluded by setting the `weights` argument to zero for that property. For example, if centroid distance is not to

be included in the interest value, then set the first component of `weights` to zero. For area ratio and intersection area, the interest function is given by

$$f(x) = \begin{cases} 0 & \text{if } x < b_1 \\ a_0 + a_1 \cdot x & \text{if } b_1 \leq x < b_2 \\ 1 & \text{if } x \geq b_2, \end{cases}$$

where $a_1 = 1/(b_2 - b_1)$ and $a_0 = 1 - a_1 \cdot b_2$. Finally, FoM's interest function is given by $b_1 \exp\{-((x-1)/b_2)^4/2\}$.

```
R> ires <- interester( look2, verbose = TRUE );
R> summary( ires );
```

To remove the smallest sized features, the following modification to the call to `FeatureFinder` may be made. The following code results in only one observed feature and two forecast features.

```
look <- FeatureFinder( hold, smoothpar = 10.5, thresh = 5, min.size = 700 );
```

A function that is used by some of the functions already demonstrated is `FeatureAxis`. It can be a useful function by itself, however. In the following code, only the output for `summary( ang1 )` is shown for brevity.

```
R> ang1 <- FeatureAxis( look$X.feats[[ 1 ]] );
R> ang2 <- FeatureAxis( look$Y.feats[[ 1 ]] );
R> summary( ang1 );
R> summary( ang2 );

 Mid-point of Axis is at:
[1] "(125.454675192381, 205.600603829035)"

 Major Axis length =  55.54851

 Major Axis Angle =  63.6426  degrees

 Minor Axis length =  41.07206

 Minor Axis Angle =  153.6426  degrees

 Aspect ratio =  0.7393909


 sma fit summary (see help file for function sma from package smatr)

Call: sma(formula = y ~ x, data = data.frame(x = pts[, 1], y = pts[,
    2]))
```

```
Fit using Standardized Major Axis

------------------------------------------------------------
Coefficients:
Coefficients:
             elevation    slope
estimate     -47.59876 2.018254
lower limit -163.16280 1.302429
upper limit   67.96527 3.127501

H0 : variables uncorrelated
R-squared : 0.007336402
P-value : 0.69758

NULL


R> plot( ang1 );
R> plot( ang2 );
R> plot( ang1, zoom = TRUE );
R> plot( ang2, zoom = TRUE );
```

The last `plot` commands show the two features with their estimated (using **smatr** package) major and minor axes in their respective spatial locations within $\mathcal{D}$ (displayed in Figure 9 top row). The commands with `zoom=TRUE` show them blown up within a bounding box so that they can be more easily seen (Figure 9 bottom row). Notice the `look$X.feats[[ 1 ]]` and `look$Y.feats[[ 1 ]]` calls in the above code. These list components are not intended for the user to access, but they can nevertheless be obtained as they may be of interest for more advanced users. Here, the `X.feats` and `Y.feats` are the features for the observed and forecast fields, respectively, in list argument form where each feature is alone on the domain's grid as a binary matrix. It is a way to easily acces individual features but it is considered an advanced operation.

4.2. Structure Amplitude Location (SAL)

The Structure, Amplitude and Location (SAL) method acts on objects of class `"features"` as returned by `FeatureFinder`. The method provides distributional summaries of forecast performance that consider displacement errors, intensity errors and a spatial texture summary. Note that this approach is best applied on relatively small domains, much smaller than that of the example provided here.

The structure component is given by

$$S = 2 \cdot \frac{V(\hat{X}) - V(X)}{V(\hat{X}) + V(X)},$$

where

$$V(X) = \frac{\sum_{i=1}^{k} Z_i \cdot V(Z_i)}{\sum_{i=1}^{k} Z_i},$$

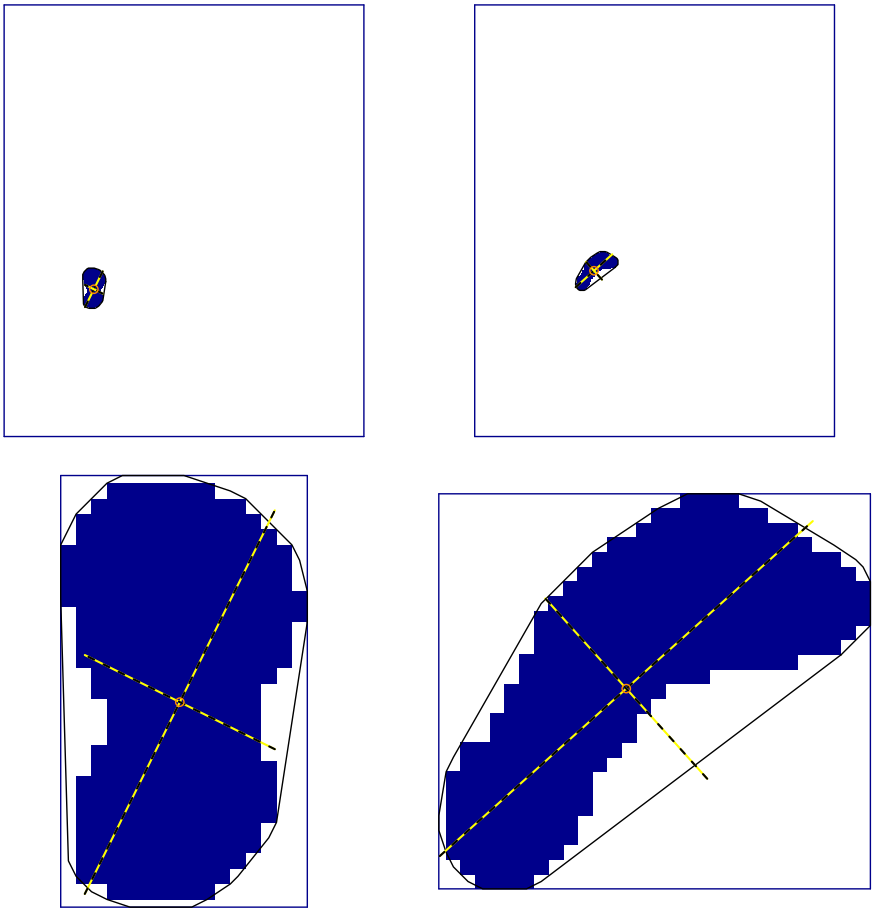

Figure 9: A pair of matched features shown in their respective locations in space along with their major and minor axes and convex hull (top row). Each is also shown within their own bounding box (bottom row) in order to better see their shapes and orientations.

with Z_i, $i = 1, \ldots, k$ denoting the set of grid points for individual features within the field and

$$V(Z_i) = \frac{1}{\max_{\mathbf{s}\in Z_i}\{X(\mathbf{s})\}} \sum_{\mathbf{s}\in Z_i} X(\mathbf{s}) = \frac{n_i}{\max_{\mathbf{s}\in Z_i}\{X(\mathbf{s})\}} \bar{X}_{Z_i},$$

where X is replaced with $\hat{X}$ when applied to the forecast field and n_i is the number of grid points in feature i. Note that the features within each field are neither merged within a field nor matched between fields. The larger the value of S, the worse the forecast predicts. Negative values for S are indicative of forecast features that are too small or incorrectly shaped.

The amplitude component is a simple comparison of the domain-averaged values given by

$$A = 2 \cdot \frac{\frac{1}{N}\sum_{\mathbf{s}\in\mathcal{D}} X(\mathbf{s}) - \frac{1}{N}\sum_{\mathbf{s}\in\mathcal{D}} \hat{X}(\mathbf{s})}{\frac{1}{N}\sum_{\mathbf{s}\in\mathcal{D}} X(\mathbf{s}) + \frac{1}{N}\sum_{\mathbf{s}\in\mathcal{D}} \hat{X}(\mathbf{s})} = 2 \cdot \frac{\bar{X} - \hat{\bar{X}}}{\bar{X} + \hat{\bar{X}}}.$$

This component has a range in $[-2, 2]$ with zero indicating a perfect forecast. $A = 1$ means

that the forecast is biased by a factor of three compared to the domain averaged observed values, and $A = -1$ indicates a low bias by a factor of three.

The location component is the sum of two types of location components, $L = L_1 + L_2$ where L_1 is the centroid distance between $X(\mathbf{s})$ and $\hat{X}(\mathbf{s})$ taken over the entire domain, $\mathcal{D}$, normalized by the largest distance between two boundary points of $\mathcal{D}$, denote this distance by D. The second component involves the centers of mass of the individual features within each field and is given by

$$L_2 = 2 \cdot \frac{|\mu(\hat{X}) - \mu(X)|}{D},$$

using X and $\hat{X}$ for brevity with

$$\mu(X) = \frac{\sum_{i=1}^{k} \nu_i \cdot \bar{X}_{Z_i}}{\sum_{i=1}^{k} \bar{X}_{Z_i}},$$

where ν_i is the centroid distance of the entire field and the individual feature i. The range of values that L can realize are in $[0, 2]$ with zero a perfect score.

See Wernli *et al.* (2008, 2009) for details about this approach. The following code will perform the SAL method.

```
R> look <- FeatureFinder( hold, smoothpar = 10.5, thresh = 5 );
R> saller( look );

ICP real case valid 1 June 2005 0 UTC
Precipitation (mm/h)
[1] "ICP real case valid 1 June 2005 0 UTC"
[2] "Stage II analysis"
[3] "Model 1"

         S          A          L
0.68897269 0.09949281 0.22615191
```

The value returned by `saller` is an object of class `"saller"` and the L_1 and L_2 components can be accessed, if assigned to an object. For example, if the last line above is replaced by `tmp <- saller( look )`, then L_1 can be obtained using `tmp$L1`.

4.3. Feature composites

Another distributional approach available in **SpatialVx** is similar in spirit to the composite method proposed by Nachamkin (2004, 2009); Nachamkin, Chen, and Schmidt (2005). The idea is to center all of the features on top of each other (for each field individually) to form a composite that can be graphed to compare the distribution of features. Using the `look` object from the previous example in Section 4.2:

```
R> look3 <- compositer( look );
R> plot( look3, col = cl );
```

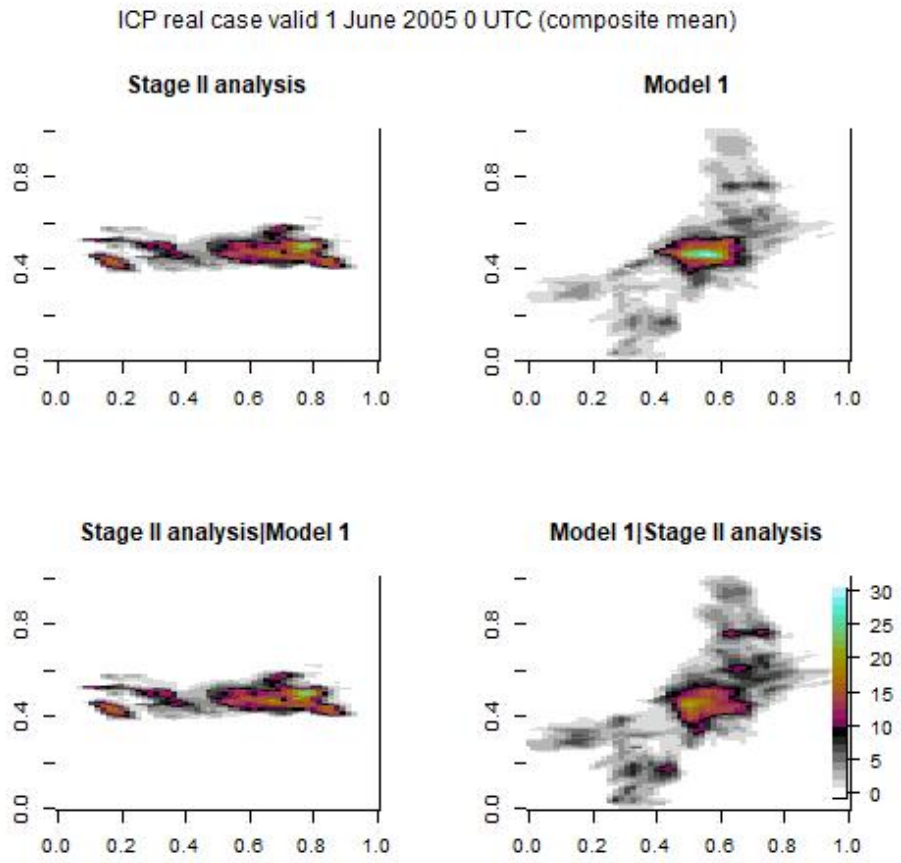

Figure 10: Composite features for the 1 June 2005 case using a threshold of 5 mmh^{-1}. Model 1 is `wrf4ncar`. The bottom panels show the conditional composites, which restrict summing of feature existence to features that overlap with features from the other field.

The resulting graphic from the `plot` command above is shown in Figure 10. All of the features from the analysis (top left) and model (top right) fields are centered on top of each other and the sum is taken resulting in the "density" plots shown. The bottom row is similar except that the densities are conditioned on the presence of the features from the other field.

From this graphic, it is easy to see that the features from the forecast field tend to be too too small. The model also has more low-intensity "scatter" that is oriented incorrectly from the overall orientation tendencey of the observation for this time point.

Features can be combined across `"features"` (or `"matched"`) objects in order to obtain a distributional picture of features over, e.g. many snapshots in time, by using the `combiner` function.

4.4. Shape analysis

Shape analysis (Dryden and Mardia 1998) is a closely related topic to image warping. Micheas, Fox, Lack, and Wikle (2007) proposed using shape analysis within a feature-based approach

to verify quantitative precipitation field ensembles, and Lack *et al.* (2010) studied the idea further. The idea is to identify landmark points (analogous to control points in image warping) and subsequently distill the information from the features down to comparisons based on the positioning of these points. Micheas *et al.* (2007) and Lack *et al.* (2010) additionally summarize the comparison using intensity information.

```
R> holdg0 <- make.SpatialVx( geom000, geom004, thresholds = c(0.01, 50.01),
+                    projection = TRUE, map = TRUE, loc = ICPg240Locs,
+                    loc.byrow = TRUE,
+                    field.type = "Geometric Objects Pretending to be Precipitation",
+                    units = "mm/h", data.name = "ICP Geometric Cases",
+                    obs.name = "geom000", model.name = "geom004" );

R> lookg0 <- FeatureFinder(hold, do.smooth = FALSE, thresh = 2, min.size = 200);
R> lookg0 <- hiw( lookg0 );
R> summary( lookg0 );

       [,1]
SSloc 15625
SSavg     0
SSmin     0
SSmax     0

R> par(mfrow=c(1,2));
R> plot(lookg0);
R> plot(lookg0, which = "Xhat");
```

Following shape analysis for images, the features are first rotated to have the same orientation and then landmarks are paired according to the respective angles from this re-orientation. The `summary` output gives information about the sum of squares translation errors (taken over each pair of landmark points), as well as the average, minimum and maximum for each pair of features. Because the only difference between the re-oriented shapes is a translation error, only `SSloc` is positive and the rest are zero.

Figure 11 shows the result of the above `plot` command. Because the shapes are perfectly concave, the landmarks are easily found. Unfortunately, in real meteorological fields, individual features are rarely perfectly concave (cf. Figure 12) making it difficult to identify landmark points in a meaningful way. One solution is to heavily smooth the features prior to identifying them but it will still not guarantee that the shapes are concave. The main function, `hiw`, for identifying landmarks makes a reasonable attempt to account for holes within the features as well as other situations where the features diverge from concavity. Nevertheless, even if the landmarks can reasonably be attained, it is subsequently difficult to make valid comparisons between such landmark points. Therefore, it is best to perform shape analysis on fields that naturally render concave features.

Figure 13 depicts one alternative way that boundary points might be selected for convex shapes. The idea is to first deform the shape into a perfect circle with the same centroid and area as the original shape. The circle, of course is a simple concave shape whose boundary

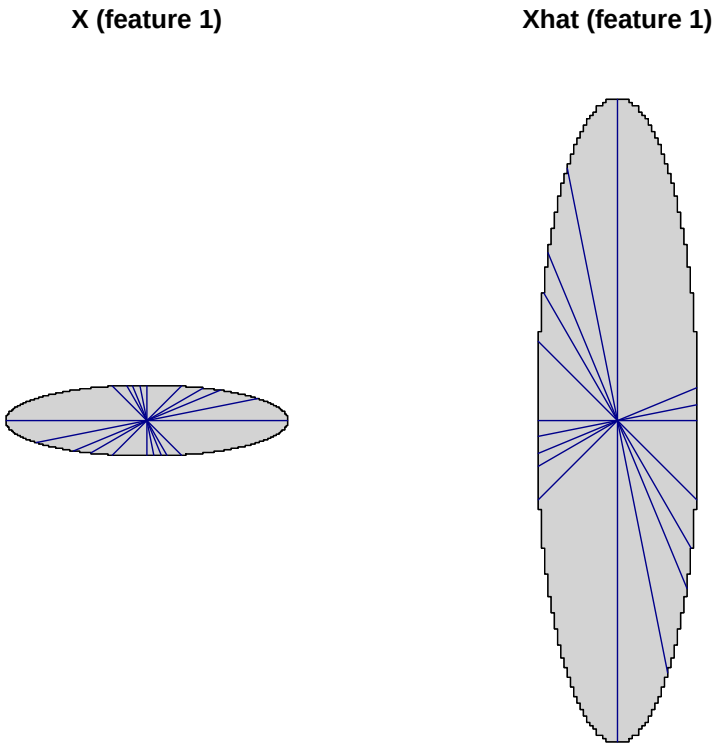

Figure 11: Example of how landmark points are determined for a concave shape. The two ovals from `geom000` and `geom004` are identical apart from a re-scaling and translation. Rays from the centroid of each oval are extended to the boundary at pre-determined angles. Where the rays intersect with the boundary determines the landmarks.

points are easily identified. Using these boundary points, perform the inverse warp transform to determine their positions on the original image. Although this method generally produces shape points that better depict the convex shape, analyzing the result remains a challenge and the idea is not carried out any further here.

5. Spatial alignment summary measures

All of the methods for making non-distributional comparisons between two fields that have been discussed so far, image warping and MODE, are relatively complicated methods. They provide useful diagnostic information but also require expertise to be able to analyze the results effectively. This section deals with very simple summaries of spatial alignment errors for the entire domain. Of course, all of the summaries discussed in this section can be applied to individual features, as is carried out in the MODE examples of Section 4.1. In fact, some of them are much better suited to the task than the commonly used centroid distance. The

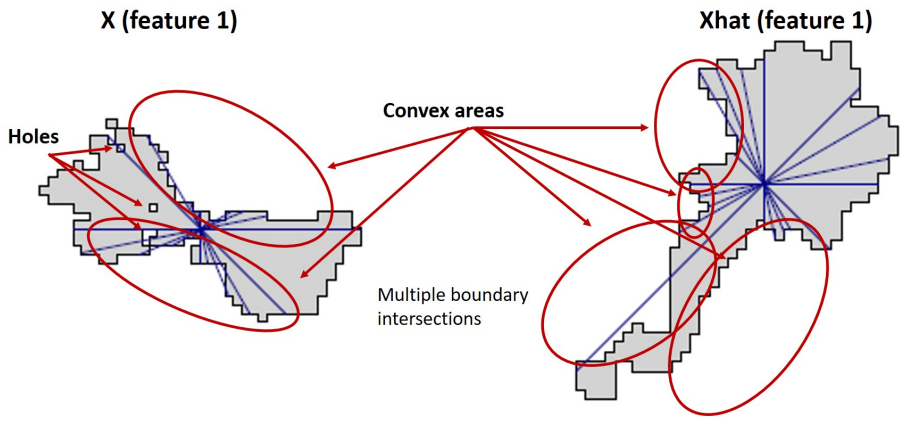

Figure 12: Similar to Figure 11 but for features within a (heavily smoothed) real precipitation field. Note the holes and lack of concavity, which make determining the landmarks difficult.

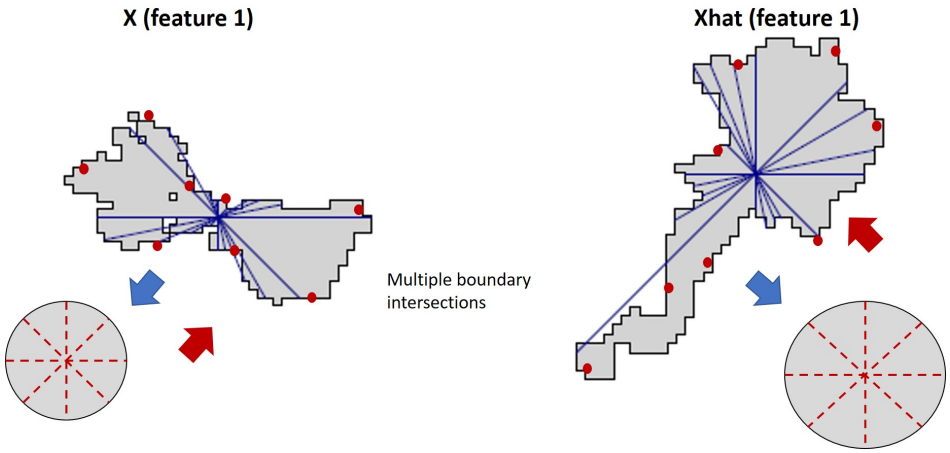

Figure 13: Illustration (not to scale) of a possible method to identify boundary points on a convex shape. The shape is first image warped to a circle (with the same centroid and area). Rays are extended from the circle's center to its boundary to find boundary points for the circle. These points can then be deformed back to the original shape.

newest summaries available with **SpatialVx** are those introduced by Gilleland (2021) so are not included in the feature-based functions.

The formulas and background on the measures employed in this section can be found in Baddeley (1992a,b); Gilleland *et al.* (2008); Gilleland (2011, 2017); Gilleland *et al.* (2020) and Gilleland (2021). Gilleland *et al.* (2020) provided a rigorous evaluation of several of these measures using challenging test cases developed therein and the main results are summarized in Gilleland (2021) along with additional evaluation for FoM and the new measures, G, G_β, etc., proposed therein. As all of these measures can be computed efficiently using distance maps, it is useful to give some background on them first.

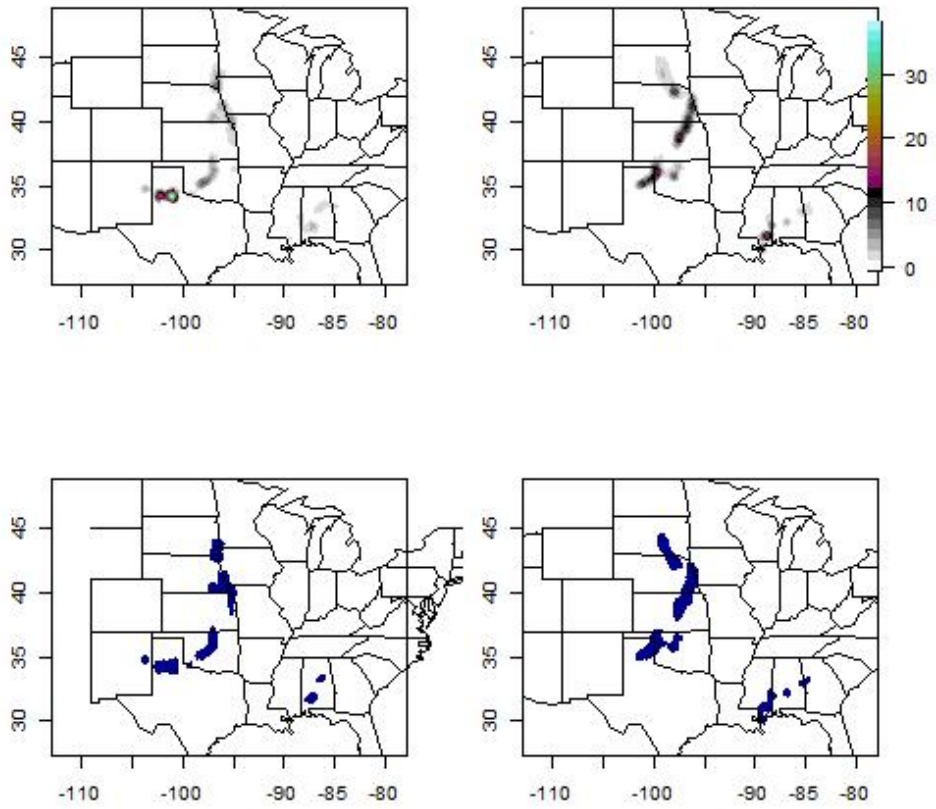

Figure 14: Top shows image plots for the convolution-radius filtered fields (using a radius of ten grid points) from the top row of Figure 3. Bottom row is the binary field created from the images in the top row using a threshold of 2.1 mmh^{-1}.

5.1. The distance map and its role in calculating the measures

Recall that $d(\mathbf{s}, A)$ represents the shortest distance from grid point $\mathbf{s}$ to the nearest grid point in the set A and let I_X and $I_{\hat{X}}$ be the set of one-valued grid points in the binary "observed" and forecast fields, respectively. All of the summary measures described in this section can be calculated efficiently using distance maps applied to the binary fields. As an example, Figure 14 shows the binary field obtained from applying the convolution-radius filter (with a radius of ten grid points) used by MODE.[13]

The following code will reproduce Figures 14 and 15. `kernel2dsmooth` from **smoothie** performs the convolution-radius filtering while `binarizer` from **SpatialVx** employs functions from **spatstat** to create the binary fields as class `"owin"` objects that can be used with `distmap` from **spatstat**. Note that `binarizer` can also return regular matrices but in the code below,

[13]The smoothed fields are used for this example in order to make crisper-looking distance maps for illustrative purposes. The measures presented in this section do not apply any smoothing, although applying the summaries to fields that have been smoothed over different sized "neighborhoods" could be useful.

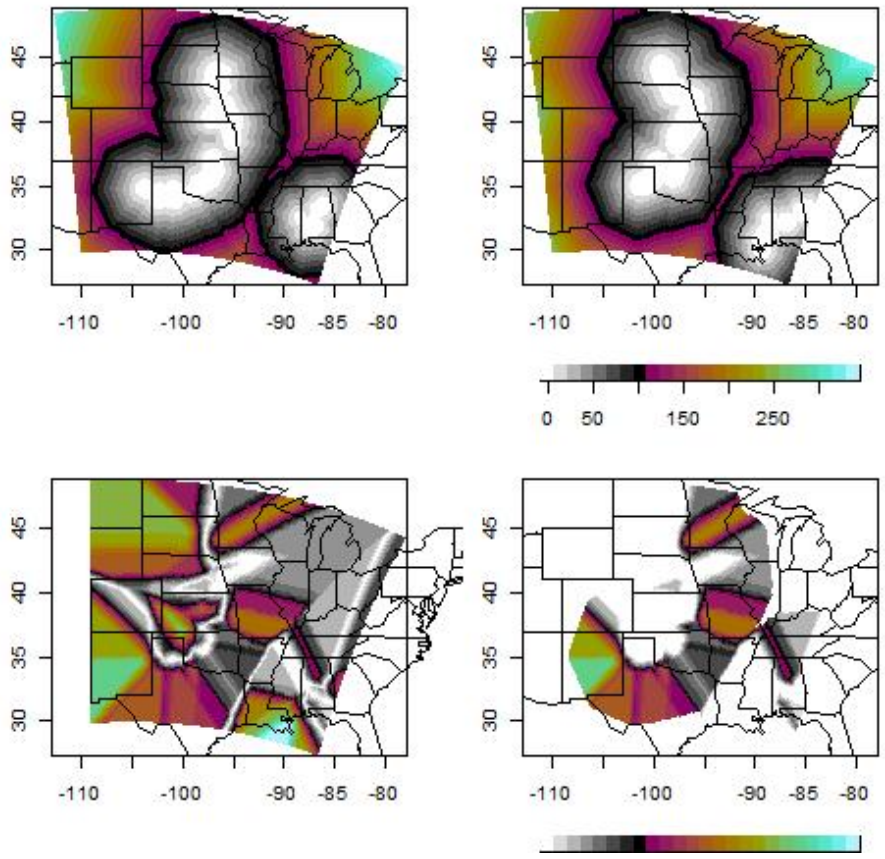

Figure 15: Top row shows the distance maps for the binary fields in the bottom row of Figure 14. Bottom left shows the magnitude difference between the two fields in the top row. Bottom right first applies the cut-off function $\omega(x) = \max\{x, 150\}$ to each field in the top row before taking the magnitude difference between them.

the regular matrices are converted from the `"owin"` objects.

```
R> Xsm <- kernel2dsmooth( x = obs0601, kernel.type = "disk", r = 10 );
R> Xhatsm <- kernel2dsmooth( x = wrf4ncar0531, kernel.type = "disk", r = 10 );
R> zlsm <- range( c( c( Xsm ), c( Xhatsm ) ), finite = TRUE );

R> par( mfrow = c( 2, 2 ) );
R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, Xsm, col = cl, zlim = zlsm, add = TRUE,
+             xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();

R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
```

```
R> poly.image( l$lon, l$lat, Xhatsm, col = cl, zlim = zlsm, add = TRUE,
+           xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();
R> image.plot( Xsm, zlim = zlsm, col = cl, legend.only = TRUE );

R> # Create binary fields using a threshold of 0.1 mm/h.
R> Z <- binarizer( X = Xsm, Xhat = Xhatsm, threshold = 2.1, value = "owin" );
R> XsmBin <- as.matrix( Z[[ 1 ]] ) * 1; XhatsmBin <- as.matrix( Z[[ 2 ]] ) * 1;
R> XsmDM <- distmap( Z[[ 1 ]] ); XhatsmDM <- distmap( Z[[ 2 ]] );
R> XsmDM <- as.matrix( XsmDM ); XhatsmDM <- as.matrix( XhatsmDM );

R> # Magnitude difference in distance maps.
R> dDM <- abs( XhatsmDM - XsmDM );
R> tmp1 <- XhatsmDM; tmp2 <- XsmDM;
R> tmp1[ tmp1 > 150 ] <- 0; tmp2[ tmp2 > 150 ] <- 0;
R> dDM2 <- tmp1 - tmp2;

R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, XsmBin, col = c( "white", "darkblue" ),
+         add = TRUE, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();

R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, XhatsmBin, col = c( "white", "darkblue" ),
+         add = TRUE, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();

R> # Distance map figure.
R> zldm <- range( c( c( XsmDM ), c( XhatsmDM ) ), finite = TRUE );
R> zldDM <- range( c( dDM ), finite = TRUE );

R> par( mfrow = c( 2, 2 ) );
R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, XsmDM, col = cl, add = TRUE,
+         zlim = zldm, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();

R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, XhatsmDM, col = cl, add = TRUE,
+         zlim = zldm, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();
R> image.plot( XsmDM, zlim = zldm, col = cl, legend.only = TRUE,
```

```
+         horizontal = TRUE, legend.mar = 0.01 );

R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, dDM, col = cl, add = TRUE,
+         zlim = zldDM, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();

R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, dDM2, col = cl, add = TRUE,
+         zlim = zldDM, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();

R> image.plot( dDM, zlim = zldDM, col = cl, legend.only = TRUE,
+         horizontal = TRUE, legend.mar = 0.01 );
```

Baddeley's Δ (Baddeley 1992a,b) is given by the L_p norm of $\tau = |\omega(d(\mathbf{s}, I_{\hat{X}}) - \omega(d(\mathbf{s}, I_X))|$ taken over all $\mathbf{s} \in \mathcal{D}$, where ω is any concave function so that $\omega(x + y) \leq \omega(x) + \omega(y)$. The function `deltametric` from **spatstat** is used to compute Δ where $\omega(x) = \max\{x, c\}$ for c a user-chosen constant. The role of ω is to ensure that the alteration of a fixed number of grid points in the set has a bounded effect on the measure. It helps to mitigate edge effects that result from the fact that its calculation is over the entire domain (see, e.g., Schwedler and Baldwin 2011, who demonstrate this issue in the spatial verification context). Note that τ is just the magnitude difference between the distance maps defined by $d(\mathbf{s}, I_{\hat{X}})$ and $d(\mathbf{s}, I_X)$, meaning that Δ can be computed efficiently.[14] In particular, Baddeley's Δ with $c = \infty$ is the L_p norm of the field in the bottom left panel of Figure 15 and for $c = 150$, it is the L_p norm of the field in the bottom right panel of the figure.

The Hausdorff distance, for the finite domains of the verification application is the maximum of $d(\mathbf{s}, I_{\hat{X}})$ over all $\mathbf{s} \in I_X$ and $d(\mathbf{s}, I_X)$ over all $\mathbf{s} \in I_{\hat{X}}$, and can be obtained as the limit when p goes to infinity in the L_p norm of τ so that it is a special case of Baddeley's Δ.[15] That is, the Hausdorff distance is the maximum value of the field in the bottom left panel of Figure 15.

It can be useful to restrict attention to the sets of grid points with value one in the binary fields. Figure 16 shows the same distance maps from Figure 15 masked out using the binary fields from Figure 14. The left panel is the distance map from the top right of Figure 15 masked out using the binary field from the bottom left of Figure 14, and analogously for the right panel. The following code will reproduce Figure 16.

```
R> Z1 <- XhatsmDM; Z1[ XsmBin == 0 ] <- NA;
R> Z2 <- XsmDM; Z2[ XhatsmBin == 0 ] <- NA;
```

[14] `deltamm` does not use `deltametric` when computing Δ in order to minimize having to re-compute the distance maps multiple times.

[15] The partial Hausdorff measure takes the maximum of the k-th largest order statistic, instead of the maximum, of each of these quantities.

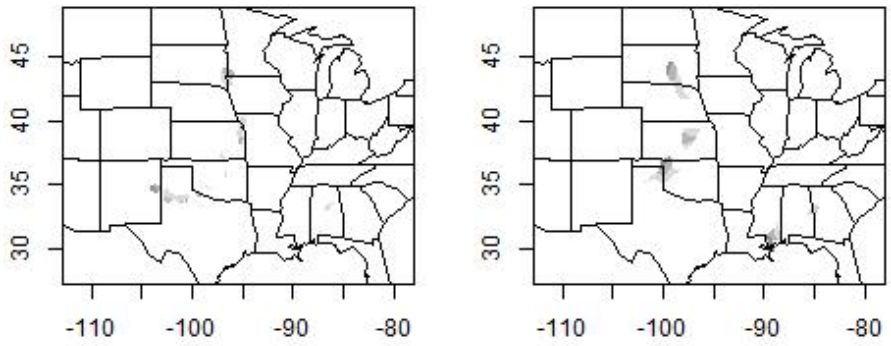

Figure 16: The distance maps from the top row of Figure 15 conditioned on the other field. Left is the `wrf4ncar0531` distance map only displayed where there exists an "observed" feature (i.e., all points outside the one-valued set of grid points in Figure 14 bottom left are masked out). Right is the distance map for the "observed" field masked using the bottom right binary field of Figure 14.

```
R> par( mfrow = c( 1, 2 ) );
R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, Z1, col = cl, add = TRUE,
+      zlim = zldm, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();

R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, Z2, col = cl, add = TRUE,
+      zlim = zldm, xlab = "", ylab = "", axes = FALSE );
R> map( database = "state", add = TRUE );
R> map.axes();
```

The mean-error distance between $I_{\hat{X}}(\mathbf{s})$ and $I_X(\mathbf{s})$, denoted by $\text{MED}(I_{\hat{X}}(\mathbf{s}), I_X(\mathbf{s}))$, is the average value of $d(\mathbf{s}, I_{\hat{X}})$ over all $\mathbf{s} \in I_X$. Note that MED is not symmetric because generally $\text{MED}(I_{\hat{X}}(\mathbf{s}), I_X(\mathbf{s})) \neq \text{MED}(I_X(\mathbf{s}), I_{\hat{X}}(\mathbf{s}))$. Gilleland (2017) argued that the lack of symmetry can be exploited in the forecast verification domain because $\text{MED}(I_{\hat{X}}(\mathbf{s}), I_X(\mathbf{s}))$ informs about misses and $\text{MED}(I_X(\mathbf{s}), I_{\hat{X}}(\mathbf{s}))$ informs about false alarms in a spatial distance manner. $\text{MED}(I_{\hat{X}}(\mathbf{s}), I_X(\mathbf{s}))$ is the average of $d(\mathbf{s}, I_{\hat{X}}(\mathbf{s}))$ over the set $I_X \subset \mathcal{D}$ of the left panel in Figure 16, while $\text{MED}(I_X(\mathbf{s}), I_{\hat{X}}(\mathbf{s}))$ is the average of $d(\mathbf{s}, I_X(\mathbf{s}))$ over those grid points in $I_{\hat{X}} \subset \mathcal{D}$ of the right panel of the figure.

Although included in this software, FoM suffers from many drawbacks and is not discussed further here. The new measures involve the product of two terms: the area of symmetric difference between $I_{\hat{X}}$ and I_X, denoted by y_1, and a weighted sum of the MED taken in both directions, denoted by y_2. Specifically, $y_2 = n_{I_X} \cdot \text{MED}(I_{\hat{X}}(\mathbf{s}), I_X(\mathbf{s})) + n_{I_{\hat{X}}} \cdot \text{MED}(I_X(\mathbf{s}), I_{\hat{X}}(\mathbf{s}))$, where n_{I_X} and $n_{I_{\hat{X}}}$ are the number of grid points in the sets I_X and $I_{\hat{X}}$, respectively. The first measure is $G(I_{\hat{X}}(\mathbf{s}), I_X(\mathbf{s})) = (y_1 y_2)^{1/3}$ and the second is $G_\beta(I_{\hat{X}}(\mathbf{s}), I_X(\mathbf{s})) = \max\{1 - y_1 y_2/\beta, 0\}$; both are symmetric. Gilleland (2021) suggested, but did not analyze, an asymmetric version of these measues by defining $y_1(A, B) = n_B - n_{AB}$ and $y_2(A, B) = \text{MED}(A, B) \cdot n_B$ which show up in the printout. The rationale given in Gilleland (2021) for the cubed root in the definition of G was that an area (units of grid points squared) is multiplied by a distance (units of grid points) so that the cubed root would give units of grid points, akin to the other measures. However, in fact, the y_2 term involves the sum of distances over an area so that the units for y_2 are actually grid points to the fourth power, and thus the product term y is in units of grid points to the sixth power. Therefore, it is $\sqrt{G}$ that is in units of grid points.

All of the measures, except FoM and G_β, are zero when the two fields match perfectly and increasing values imply decreasing forecast performance. FoM and G_β are scaled to fall between zero and one with one representing a perfect match and zero a poor match. See Gilleland (2021) to learn how to choose β.

The manner by which **SpatialVx** carelessly assumes a regular grid of points for geo-referenced data introduces an inaccuracy that cannot be avoided in calculating the distance maps; particularly for locations near the poles. Nevertheless, the measures have been shown to be useful despite this sacrifice for computational efficiency as has been demonstrated by, e.g., Gilleland (2011, 2017, 2021), Brunet and Sills (2016), Schwedler and Baldwin (2011) and Gilleland *et al.* (2020).

5.2. Calculating the measures with SpatialVx

The following code will calculate all of the various measures described above except the new ones.

```
R> locmeasures2d( hold, k = 4 );

[1] "ICP real case valid 1 June 2005 0 UTC\nPrecipitation (mm/h)"
Comparison for:
[1] "ICP real case valid 1 June 2005 0 UTC"
[2] "Stage II analysis"
[3] "Model 1"
[1] "ICP real case valid 1 June 2005 0 UTC\nPrecipitation (mm/h)"
```

```
 Comparison for:
[1] "ICP real case valid 1 June 2005 0 UTC"
[2] "Stage II analysis"
[3] "Model 1"
Threshold(s) is (are):
[1] "0.1"  "2.1"  "5.1"  "20.1"
Baddeley Delta Metric with c =  Inf
             0.1      2.1      5.1     20.1
p = 2;  26.58215 31.76413 51.00272 81.36362

 Hausdorff distance
          0.1      2.1      5.1     20.1
[1,] 89.73287 121.5368 344.4186 311.4498

 Quantile (if k in (0,1) or k-th highest (if k = 1, 2, ...) difference in distance maps.
             0.1      2.1      5.1     20.1
k = 4;  89.73287 121.5368 342.5874 310.6195

 Mean error distance (Miss)
          0.1      2.1      5.1     20.1
[1,] 5.241503 11.54831 14.58828 26.51177

 Mean error distance (FalseAlarm)
          0.1      2.1      5.1     20.1
[1,] 5.026305 15.99586 21.85267 102.7902

 Mean square error distance (Miss)
          0.1      2.1      5.1    20.1
[1,] 120.9354 287.6795 341.6212 784.217

 Mean square error distance (FalseAlarm)
          0.1     2.1      5.1     20.1
[1,] 84.88085 490.693 1362.729 20954.02

 Partial Hausdorff distance
        0.1 2.1 5.1 20.1
k = 4;   NA  NA  NA   NA

 Pratt's figure of merit (FOM)
                     0.1       2.1       5.1       20.1
alpha = 0.1;  0.6377748 0.3089145 0.1135902 0.01969333
NULL
```

The results above are applied to `wrf4ncar` (Model 1). To apply them to Model 2 (`wrf4ncep`), use the `model` argument: `locmeasures2d( hold, k = 4, model = 2 );`.

Because of their recent development, G and G_β are not included with `locmeasures2d` nor

do they have a method function for `"SpatialVx"` objects. Instead, they must be called individually and work directly on the matrices of interest.

```
R> Gbeta( X = obs0601, Xhat = wrf4ncar0531, threshold = 2.1, beta = 601 * 501 / 2 );

Observation (A) =  obs0601
Model (B) =  wrf4ncar0531
Threshold and rule: 2.1 ( > )
beta =  150550.5
Gbeta(A,B) =  0.8584995


 Component parts and asymmetric Gbeta:

           nA             nB            nAB nA + nB - 2nAB          medAB
 7.717000e+03   8.488000e+03   9.720000e+02   1.426100e+04   1.599586e+01
        medBA     medAB * nB     medBA * nA    asymGbetaAB    asymGbetaBA
 1.154831e+01   1.357729e+05   8.911833e+04   9.549769e-01   9.734793e-01
```

The print method for the returned `"Gbeta"` class object shows the original call to the function and shows that the binary field created from `X` and `Xhat` was made by assigning one to values greater than (the `rule=">"` argument) 2.1, along with the value of β chosen. The value of the measure is about 0.86, meaning that for this choice of β the forecast predicted values above 2.1 mmh^{-1} reasonably well, although there may still be room for improvement.[16] The printout also provides information about the components that make up G_β, as well as the asymmetric versions.

Measures like G_β are particularly well suited for ranking different models against the same observation.

```
R> c( Gbeta( X = obs0601, Xhat = wrf4ncar0531, threshold = 2.1,
+                beta = 601 * 501 / 2 ) );
R> c( Gbeta( X = obs0601, Xhat = wrf4ncep0531, threshold = 2.1,
+               beta = 601 * 501 / 2 ) );
R> c( Gbeta( X = obs0601, Xhat = wrf2caps0531, threshold = 2.1,
+               beta = 601 * 501 / 2 ) );

[1] 0.8584995
[1] 0.7322819
[1] 0.8592137
```

According to G_β, `wrf2caps` is the best model for this case with `wrf4ncar` a close second and `wrf4ncep` a distant third. Note that use of `c()` strips the attributes of the `"Gbeta"` object so that only the value of G_β is displayed.

To calculate G, the relevant function is called `TheBigG` and otherwise works analogously as `Gbeta`.

[16] Values closer to one are better and those closer to zero are worse, but the precise meaning depends on the scale of features in the field relative to the domain size and the choice of β. Therefore, this measure is perhaps best suited to making comparisons against competing models or over different forecast cycles, lead times, etc.

```
R> TheBigG( X = obs0601, Xhat = wrf4ncar0531, threshold = 2.1 );

Observation (A) =  obs0601
Model (B) =  wrf4ncar0531
Threshold and rule: 2.1 ( > )
G(A,B) =  1474.713


 Component parts and asymmetric G:

           nA            nB           nAB nA + nB - 2nAB         medAB
 7.717000e+03  8.488000e+03  9.720000e+02  1.426100e+04  1.599586e+01
        medBA    medAB * nB    medBA * nA       asymG.AB       asymG.BA
 1.154831e+01  1.357729e+05  8.911833e+04  1.020469e+09  6.011031e+08
```

6. Field qualities

Besides the more direct comparison of spatial alignments and/or intensity errors, it can be useful to analyze distributional aspects of individual fields, possibly also comparing these qualities.

6.1. Structure function

Harris, Foufoula-Georgiou, Droegemeier, and Levit (2001) proposed use of the structure function in order to evaluate the model's ability to reproduce the variability observed, in their case, of precipitation. The structure function is given by

$$S_q(\mathbf{h}) = \frac{1}{n_h} \sum_{\mathbf{s} \in \mathcal{N}_h} |X(\mathbf{s}+\mathbf{h}) - X(\mathbf{s})|^q \,, \tag{2}$$

where $\mathbf{h} = (h_x, h_y)$ is a spatial lag in each coordinate, $\mathcal{N}_h$ is the set of pairs of grid points separated by $\mathbf{h}$, n_h is the number of points in $\mathcal{N}_h$, and q is a user-chosen parameter.[17] When $q = 2$, Eq (2) is equivalent to the better-known semi-variogram. Use of the variogram for evaluating high-resolution forecast models was proposed by Marzban and Sandgathe (2009).

Unlike most of the functions in **SpatialVx**, $S_q(\mathbf{h})$ can be easily applied to spatial data that are not on a regular grid of points. There are two functions for computing the structure function depending on the type of data (regular grid or irregular set of points). The function `structurogram` works on irregular data sets and shoud only be used for smaller data sets. The functions `structurogram.matrix` and `variogram.matrix` are relatively minor modifications[18] of the **fields** function `vgram.matrix` and can be used on the larger spatial fields; provided they are on a regular grid.

```
R> Slook <- structurogram.matrix( obs0601, R = 20, zero.out = TRUE );
R> plot( Slook, col = cl );
```

[17] Harris *et al.* (2001) compute the average only over the nonzero grid points in $\mathcal{N}_h$.

[18] `structurogram.matrix` and `variogram.matrix` differ from `vgram.matrix` in that they allow missing values, can be applied only to the nonzero grid pionts, and do not include the robust version of the variogram.

```
R> Slook2 <- structurogram.matrix( obs0601, R = 20 );
R> plot( Slook2, col = cl );
```

Figures 17 and 18 show the plots resulting from the above code. The value `R=20` limits the extent of lags $\mathbf{h}$ and the higher this number, the less computationally efficient the process. The graph in the first figure is calculated only over the nonzero grid points and the second over all of them. In the first example, the plot on the right of the figure suggests that there may be some anisotropy when caclulating over only the nonzero grid points because of the break from perfect semi-circles about the origin. This anisotropy does not show up in the latter figure. In either case, the plot on the left suggests that the spatial range of dependence seems to be somewhere around the 20 grid points. The larger variability in the left panel of Figure 17 when compared against that of Figure 18 suggests that there is a relatively rough texture when only considering the nonzero grid points but that the texture is relatively fine when considering all grid points. Because most grid points are zero, this result makes sense because where there is precipitation, the amounts of precipitation are highly variable, but there is no variability where there is no precipitation. Inclusion of the zero values also considerably decreases the value of $S_q(\mathbf{h})$ over all lags $\mathbf{h}$ as evidenced by the range of values on the ordinate axis of the left panels of these figures, as well as the color legend on the right panels. Again, this result makes sense because of the large number of zeros in the latter case that lower the average difference in lagged quantities.

Next, it is of interest to also check the structure function for the forecast, in this case `wrf4ncar0531`.

```
R> Slook <- structurogram.matrix( wrf4ncar0531, R = 20, zero.out = TRUE );
R> plot( Slook, col = cl );
```

Figure 19 shows the results from the above code. This structure function is taken over only the nonzero grid points. Compared aginst Figure 17, it is clear that the model does not capture the variability (or texture) of the precipitation areas that are observed in the Stage II analysis field for this valid time. Also, the spatial range of dependence appears to be much shorter and the isotropy assumption appears to be more valid.

6.2. Geometric indices

The amount of dispersion in a spatial field is quantified by the variogram and/or structure function. Perhaps more simply, an overall summary of a field's dispersion can be obtained through a couple of geometric indices; although they do not inform about different scales the way a variogram does. AghaKouchak *et al.* (2011) suggested three geometric indices, connectivity (C_{index}), complexity (or area, A_{index}) and shape (S_{index}). The latter two are very similar to one another. Applied to binary fields, A_{index} is the area of the set of nonzero grid points, call this set $\mathcal{Z}$ and let $n_{\mathcal{Z}}$ denote the size of $\mathcal{Z}$, divided by the convex hull surrounding $\mathcal{Z}$. The area index was also used within the MODE framework (cf. Davis *et al.* 2009). The shape index is the shortest possible perimeter with $n_{\mathcal{Z}}$ grid points in $\mathcal{Z}$ divided by an estimate of the actual perimeter around $\mathcal{Z}$. The connectivity index is defined by

$$C_{\text{index}} = 1 - \frac{n_c - 1}{\sqrt{n_{\mathcal{Z}}} + n_c},$$

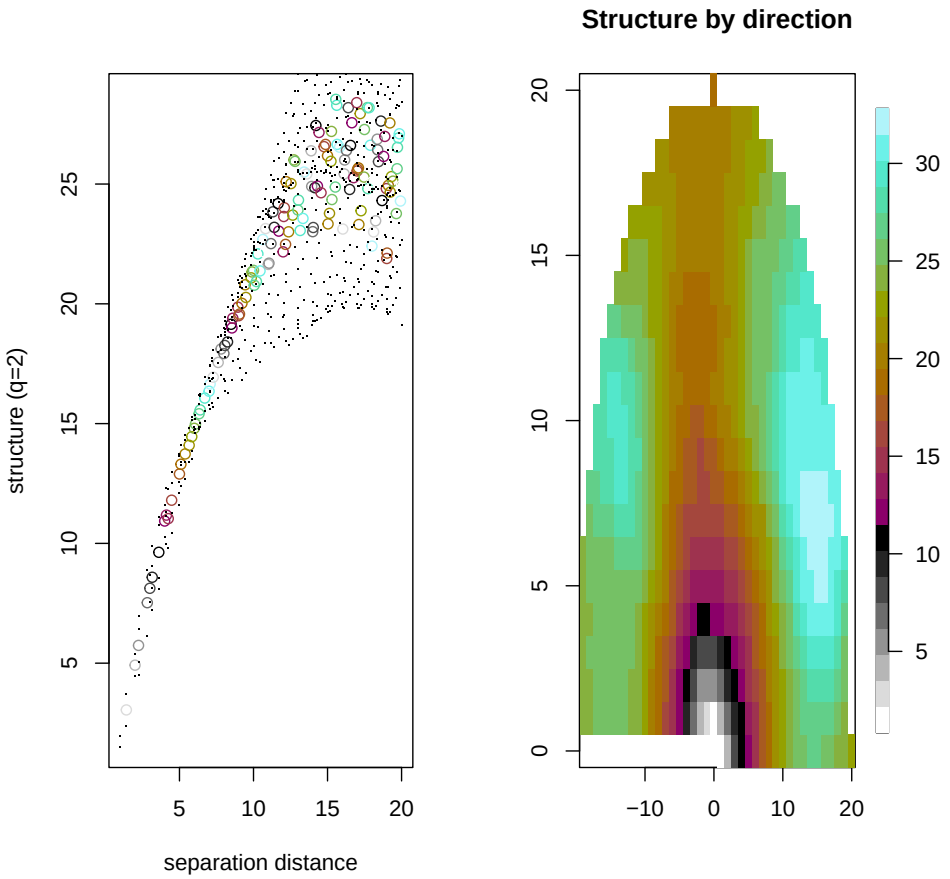

Figure 17: Structure function with $p = 2$ (semi-variogram), calculated only on the nonzero grid points, on the left and the same by angle on the right. Circles in the right panel indicate the isotropic structure function (ignoring direction) while the dots are further distinguished by different directions. The directional components are better visualized in the right panel.

where n_c is the number of isolated clusters (i.e., connected components).

The geometric indices range between zero and one. Figure 20 illustrates how the gemoetric indices vary depending on the field properties. The shape index measures how circular the set $\mathcal{Z}$ is in shape so that values near one are more circular and values near zero are not at all circular in shape. The connectivity index will always be one if there is only one isolated cluster; the area index will always be one if the only isolated cluster is concave in shape.

Because these indices operate on binary fields, the binary fields `XsmBin` and `XhatsmBin` assigned in the code in Section 4.3 will be used to demonstrate the relevant functions in **SpatialVx** for computing these indices.

```
R> round( c( Cindex( XsmBin ), Aindex( XsmBin ), Sindex( XsmBin ) ), digits = 2 );
R> round( c( Cindex( XhatsmBin ), Aindex( XhatsmBin ), Sindex( XhatsmBin ) ),
+         digits = 2 );
```

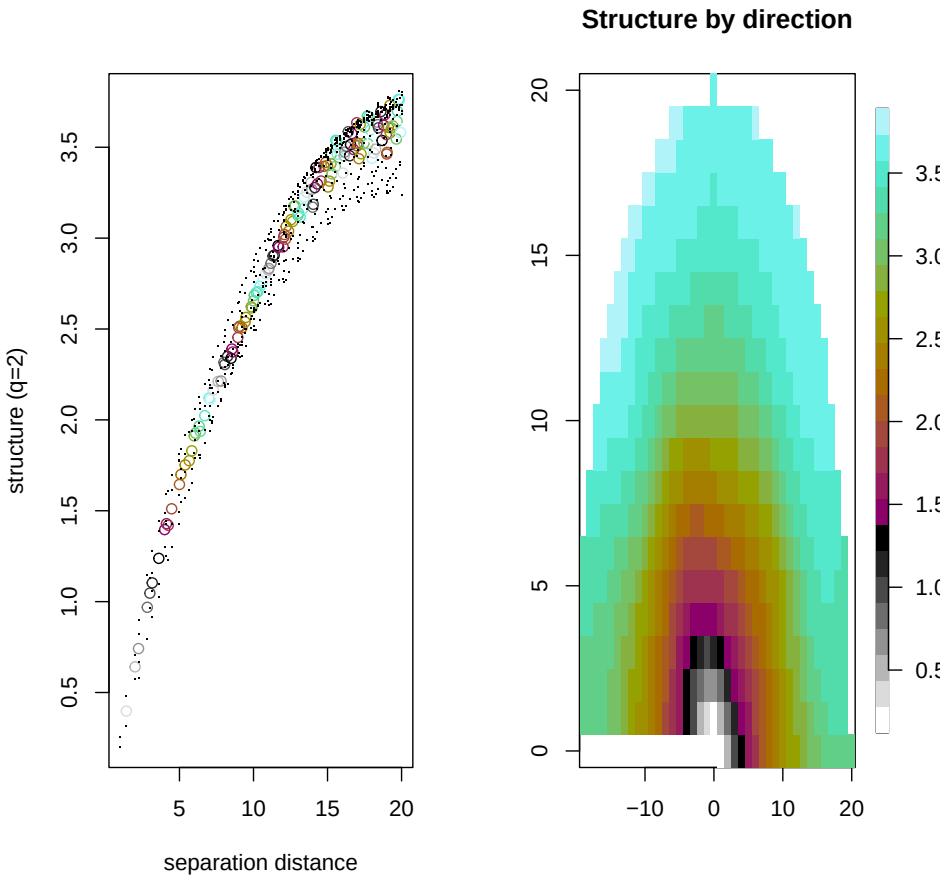

Figure 18: Structure function with $p = 2$ (semi-variogram), calculated on all of the grid points, on the left and the same by angle on the right.

```
        Aindex        A   Aconvex       dx       dy   Sindex     Pmin
  0.92    0.11  8240.00 75094.50     1.00     1.00     0.25   364.00
     P
1430.00

        Aindex        A  Aconvex       dx       dy   Sindex     Pmin
  0.95    0.13 10275.00 80576.00     1.00     1.00     0.27   406.00
     P
1518.00
```

The `dx` and `dy` arguments of the `Aindex` function are used if the grid points should have different lengths depending on the x- or y- directions, but this correction is only important for the individual numerator and denominator components that go into the area index, which are also reported in the output. In each case, the connectivity index is less than one, though near one, indicating that there is more than one connected component in the field but that there are relatively few such isolated clusters. Both the area and shape indices are closer to

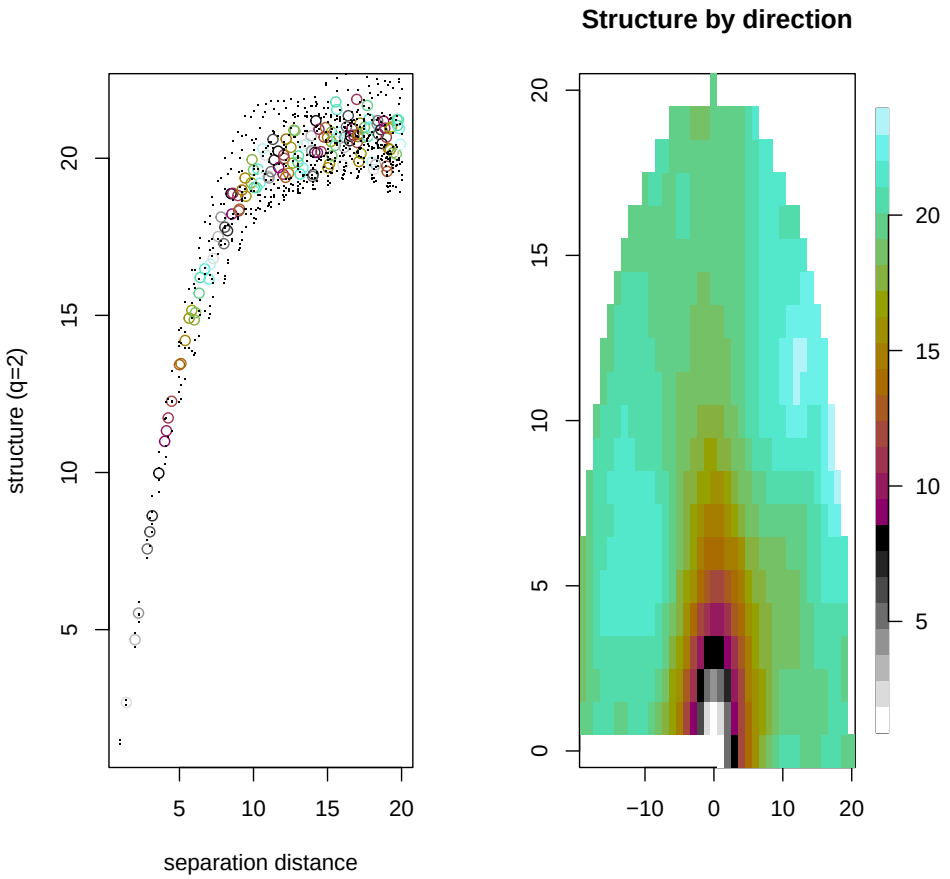

Figure 19: Same as Figure 17 except for `wrf4ncar0531`.

zero than one indicating that the clusters are relatively disperse.

7. Competing forecast verification

A gaping hole that has been in the spatial verification literature and methodologies is a lack of uncertainty information. The image warping procedure from Section 3 is an exception in that the modeling framework lends itself naturally to statistical inference. The challenge, however, is the sheer number of parameters (the control points) needed for most reasonable warps. Moreover, the CI's for this approach tend to be too narrow.

One method introduced by Hering and Genton (2011), the spatial prediction comparison test (SPCT), does provide a statistical inference framework in the realm of competing forecast verification. The procedure begins by estimating the loss function for each of the two forecasts that are to be compared. Any loss function may be applied but Gilleland (2013a) found that the absolute-error loss seemed to work best with the ICP precipitation fields. Let $\hat{X}_1(\mathbf{s})$ denote the first forecast and $\hat{X}_2(\mathbf{s})$ denote the competing second forecast, then let $g_1(X(\mathbf{s}), \hat{X}_1(\mathbf{s})) = g_1(\mathbf{s}) = |\hat{X}_1(\mathbf{s}) - X(\mathbf{s})|$ and, similarly, $g_2(\mathbf{s}) = |\hat{X}_2(\mathbf{s}) - X(\mathbf{s})|$. $g_1(\mathbf{s})$ and $g_2(\mathbf{s})$ are the loss

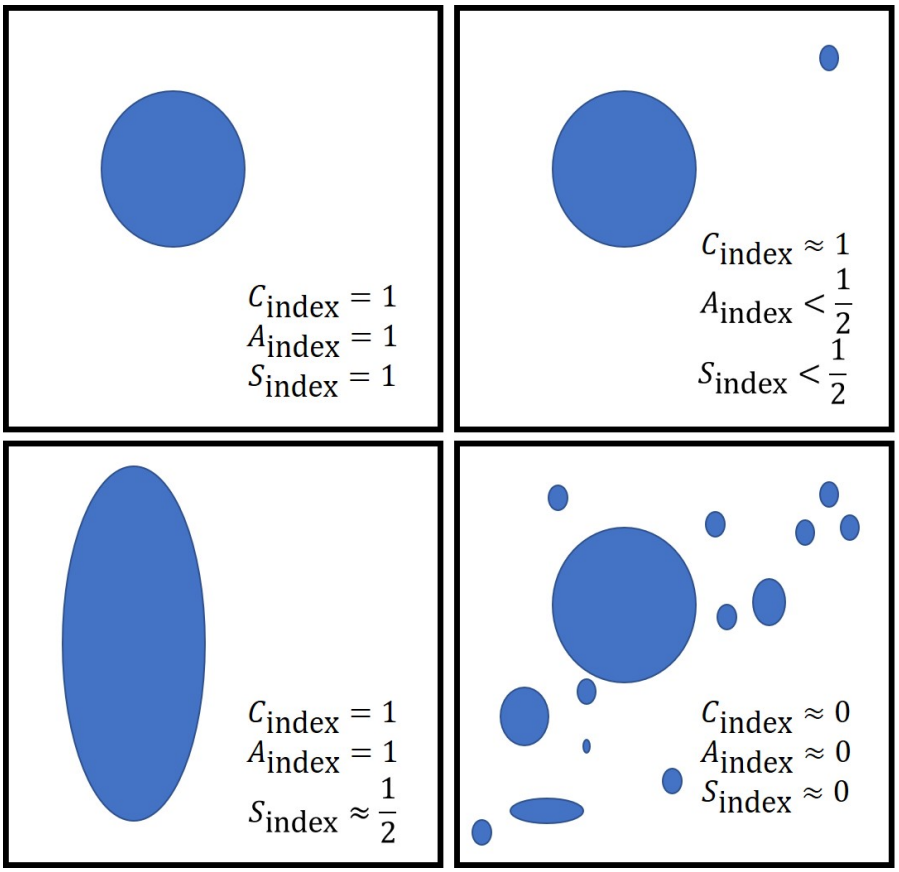

Figure 20: Illustration of how the geometric indices vary according to field properties.

functions, here absolute-error loss, for each forecast against the observed series. Interest is then in the loss differential field given by $\psi(\mathbf{s}) = g_1(\mathbf{s}) - g_2(\mathbf{s})$. Note that $\psi(\mathbf{s})$ is itself a spatial field.

The null hypothesis for the SPCT is $\mathcal{H}_0 : E[\psi(\mathbf{s})] = 0$. The trick is to estimate a variance for $\psi(\mathbf{s})$ that accounts not only for the spatial dependence in the loss differential field but also is robust against the contemporaneous correlation between $g_1(\mathbf{s})$ and $g_2(\mathbf{s})$.[19] Hering and Genton (2011) proposed estimating the variance using the spectral density similarly as proposed for statistical time series forecasts by Diebold and Mariano (1995). First, a parametric varigoram model is fit to the empirical variogram typically using all lags up to half the maximum possible lag. The variance is then estimated through a weighted average over all spatial lags of the fitted parametric variogram model. Hering and Genton (2011) showed that the SPCT is both accurate and powerful, and robust to contemporaneous correlation using empirical tests for size and power.

The SPCT can be applied to either irregularly spaced grids or to regular grids. The function `spct` can be used for irregularly spaced data and smaller regular grids. For the large-sized grids such as the ICP test cases employed here, the following code will perform the test.

[19]It is likely that the contemporaneous correlation is strong because both $\hat{X}_1(\mathbf{s})$ and $\hat{X}_2(\mathbf{s})$ are attempting to predict the same observation. However, even if these two forecasts were independent of each other, both loss functions involve the same $X(\mathbf{s})$, so contemporaneous correlation must be considered here.

```
R> hgres <- lossdiff( x = obs0601, xhat1 = wrf4ncar0531, xhat2 = wrf4ncep0531,
+                     lossfun = "abserrloss" );
R> hgres <- empiricalVG.lossdiff( hgres, maxrad = 8 );
R> hgres <- flossdiff( hgres );
R> hgres <- summary( hgres );
R> plot( hgres, icol = cl );

[1] "abserrloss: wrf4ncar0531 vs wrf4ncep0531 (against verification: obs0601)"
Fitted trend information (note: not used by subsequent functions, information only):


Call:
lm(formula = y ~ x1 + x2, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-68.813   0.048   0.091   0.147  72.174

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.760e-01  1.031e-02 -17.068   <2e-16 ***
x1           2.670e-04  2.243e-05  11.903   <2e-16 ***
x2          -1.597e-05  2.690e-05  -0.594    0.553
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.135 on 301098 degrees of freedom
Multiple R-squared:  0.0004715, Adjusted R-squared:  0.0004649
F-statistic: 71.02 on 2 and 301098 DF,  p-value: < 2.2e-16


 number of non-zero loss differential points is:  301101

 Mean Loss Differential:

 -0.09964427

 Summary of empirical variogram values:
                    vgram         d        ind    d.full vgram.full
N              29.0000000 29.000000 196.000000 98.000000 98.0000000
mean            2.5232000  5.053909   1.867347  5.307258  2.6423543
Std.Dev.        0.8559370  2.020273   3.514476  1.849032  0.7723522
min             0.4037322  1.000000  -7.000000  1.000000  0.3719266
Q1              2.0856273  3.605551   0.000000  4.123106  2.2099495
median          2.8040068  5.385165   2.000000  5.656854  2.8512320
Q3              3.2010362  6.708204   5.000000  7.000000  3.2060091
max             3.4839552  8.000000   8.000000  8.000000  3.7116931
```

```
missing values  0.0000000  0.000000   0.000000  0.000000  0.0000000
               robust.vgram          N dx dy
N              9.800000e+01     98.000  1  1
mean           1.880931e-08 297400.163  1  1
Std.Dev.       7.104522e-09   1377.313 NA NA
min            1.427967e-09 295020.000  1  1
Q1             1.449733e-08 296332.500  1  1
median         2.028027e-08 297256.000  1  1
Q3             2.437813e-08 298377.000  1  1
max            2.945054e-08 300600.000  1  1
missing values 0.000000e+00      0.000  0  0
Test Statistic for null hypothesis of equal predictive ability on average
[1] -0.04215516
p-value for two-sided alternative hypothesis is:  0.966375
p-value for (one-sided) alternative hypothesis that mu(D) < 0 is:  0.4831875
p-value for (one-sided) alternative hypothesis that mu(D) > 0 is:  0.5168125
```

Note that a test for spatial trend is performed with the results displayed in the output. The display is for information only and is not employed within the SPCT procedure by these functions. If a trend is present, however, it should be taken into account before applying the SPCT as results will be erroneous if a trend is present. A trend can be subtracted out of the field before applying the test. In this case, it does appear that a trend may be present but it seems to be only very slight (≈ 0.0003 in the x-direction and -0.00002 in the y-direction), if statistically significant in the x-direction (it is not significant in the y-direction). The result of this test for comparing `wrf4ncar0531` against `wrf4ncep0531` relative to `obs0601` under absolute-error loss is to fail to reject $\mathcal{H}_0$.

Of course, the SPCT does not account for spatial alignment errors and therefore does not circumvent the usual issues with traditional forecast verification applied to high-resolution verification sets. Gilleland (2013a) proposed two loss functions that can be used with the SPCT in order to adjust for displacement errors. One uses a weighted average of absolute-error (or other) loss with the difference in distance maps as in the lower left panel of Figure 15. Unfortunately, the distance maps have rather strong spatial dependence rendering such a test useless. The other employs a similar weighting but instead of difference in distance maps, it is a difference in the distance each grid point "travelled" to obtain the deformed image. The loss function for the intensities is then applied to the deformed forecast field, rather than the orignal one. This latter approach showed promising results. The tools to perform this test are available in **SpatialVx** but require a fairly advanced user to implement. Moreover, the uncertainty in the deformations should be considered in the final result, which adds considerable complexity to the test.

8. Bearings and circle histograms

The material in this section is largely new in that it has not, to the best of this author's knowledge, been proposed in the literature. It also shows how to get more in depth with the feature-based objects in Section 8.1. Section 8.2 shows one way that point observations might be compared with a gridded forecast. The methods in this section are aimed at more

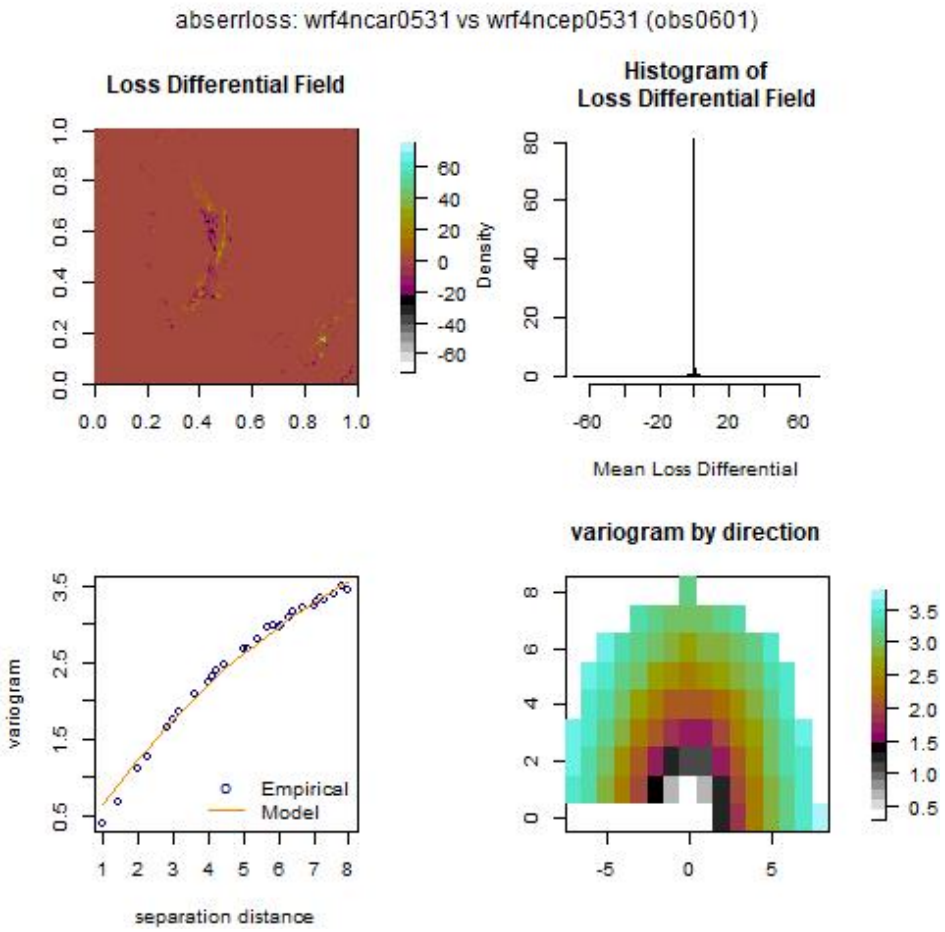

Figure 21: Upper left panel is the loss differential field under absolute-error loss (i.e., $|\hat{X}(\mathbf{s}) - X(\mathbf{s})|$). Upper right is a histogram of the field in the upper left (across space). Lower left is the empirical varigoram of the loss differential field along with the fitted exponential variogram, and bottom right is the same variogram but displayed by direction.

advanced R users.

8.1. Feature-based application

It can often be useful to understand if a forecast has any directional bias (e.g., tends to predict rain too far to the east and north). For this purpose, the `bearing` and `CircleHistogram`[20] functions from **SpatialVx** can be useful.

The bearing is the angle from one point in space to another relative to a reference direction. The reference direction typically used, and the one available with `bearing`, is north. Figure 22 illustrates the meaning of the bearing. Notice that it is not symmetric in that the bearing from a point $\mathbf{s}_0$ to another point $\mathbf{s}_1$ is not the same as the bearing form $\mathbf{s}_1$ to $\mathbf{s}_0$.

The following code illustrates how to find the bearing between the centroids of two matched

[20] `bearing` was adapted from code by Randy G. Bullock and `CircleHistogram` was contributed by Matthew Pocernich.

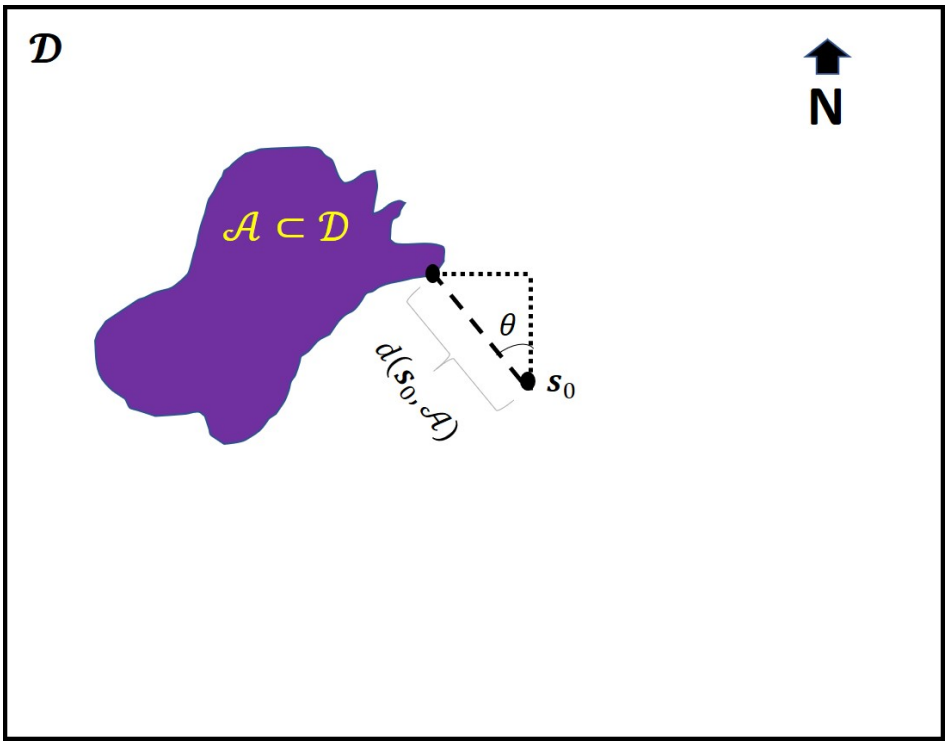

Figure 22: Illustration of the meaning of the bearing from a (point) observation station, $\mathbf{s}_0$, to the nearest point within a gridded (forecast) set of points, labeled as $\mathcal{A}$. Here, θ represents the bearing with north as the reference.

features. It uses the `look2` object returned by `deltamm` from Section 4.1. In this case, the fourth forecast feature is matched by `deltamm` to the second observed feature. Note that `look2$unmatched` will reveal that all observed features had a match but that the first forecast feature was not matched to any observed features.

```
R> look2$matches

     Forecast Observed
[1,]        4        2
[2,]        6        5
[3,]        3        1
[4,]        5        4
[5,]        2        3

R> Xftr <- look2$X.feats[[ 2 ]];
R> plot( Xftr );
R> Yftr <- look2$Y.feats[[ 4 ]];
R> plot( Yftr );
R> Xprops <- FeatureProps( Xftr );
R> Yprops <- FeatureProps( Yftr );
R> Xcen <- c( Xprops$centroid$x, Xprops$centroid$y );
```

```
R> Ycen <- c( Yprops$centroid$x, Yprops$centroid$y );
Xcen;
Ycen;

[1] 288.3359 162.1359
[1] 283.9174 174.8739

Xprops$area;
Yprops$area;

[1] 390
[1] 230

bearing( Xcen, Ycen );

[1] -19.2617

Gbeta( Xftr, Yftr, beta = 3900 );

Observation (A) =  Xftr
Model (B) =  Yftr
beta =  3900
Gbeta(A,B) =  0.8245017


 Component parts and asymmetric Gbeta:

           nA            nB           nAB nA + nB - 2nAB          medAB
  390.0000000   230.0000000    21.0000000   578.0000000      7.0515849
        medBA    medAB * nB    medBA * nA   asymGbetaAB   asymGbetaBA
    7.6829529  1621.8645321  2996.3516446     0.9777140     0.9273074
```

The bearing between the centroids of these two matched features is about -19.16 degrees. The larger of the two areas is 390 and supposing a displacement error of ten grid points were considered to be too much error, G_β is calculated using $\beta = 390 \cdot 10 = 3900$. The resulting value of $G_\beta \approx 0.82$ suggests that the match is good but not great. Notice that `class( Xftr )` is `"owin"` and that `Gbeta` works on this class of object seemlessly and without providing a threshold (it assumes the object is a binary field returned, for example, by `solutionset`). The `"owin"` objects from `look2`, defining the features, are already binary so no threshold is necessary, and `Gbeta` can be called directly on them.

There are not enough features within these two fields to warrant using a circle histogram. Therefore, it will be necessary to load additional data sets. Here, the remaining eight ICP real test cases for `wrf4ncar` are loaded from **ICPtestcases**. G from Section 5 is calculated for the matched pairs of features but any other summary measure could be substituted. The circle histogram re-scales these values to be $\sqrt{G}$ as suggested in Section 5.1, which is displayed in Figure 23. Centroid distance is used for the matching, in part for the interest of

computational efficiency, and in part to demonstrate the `centmatch` function. In the analysis below, the different initialization times are entered as different "models" into the `"SpatialVx"` object `hold2`, which is simply for convenience. It is possible to enter them as an array with the different initialization times in the third dimension. In this way, other actual models could be included (also as arrays) in the list object.

```
R> data( "wrf4ncar0425" );
R> data( "wrf4ncar0512" );
R> data( "wrf4ncar0513" );
R> data( "wrf4ncar0517" );
R> data( "wrf4ncar0518" );
R> data( "wrf4ncar0524" );
R> data( "wrf4ncar0602" );
R> data( "wrf4ncar0603" );

R> hold2 <- make.SpatialVx( obs0601, list( wrf4ncar0425, wrf4ncar0512, wrf4ncar0513,
R>                                         wrf4ncar0517, wrf4ncar0518, wrf4ncar0524,
R>                                         wrf4ncar0531, wrf4ncar0602, wrf4ncar0603 ),
R>                 thresholds = c( 0.1, 2.1, 5.1, 20.1 ),
R>                 loc = loc, projection = TRUE, map = TRUE, loc.byrow = TRUE,
R>                 field.type = "Precipitation", units = "mm/h",
R>                 data.name = "ICP real cases (wrf4ncar)",
R>                 obs.name = "Stage II analysis",
R>                 model.name = c( "wrf4ncar0425", "wrf4ncar0512", "wrf4ncar0513",
R>                                 "wrf4ncar0517", "wrf4ncar0518", "wrf4ncar0524",
R>                                 "wrf4ncar0531", "wrf4ncar0602", "wrf4ncar0603" ) );
R> bgres <- numeric( 0 );
R> for( i in 1:9 ) {
R>
R>      cat( i, "\n" );
R>
R>      looki <- FeatureFinder( hold2, smoothpar = 5, thresh = 2.1, model = i );
R>      looki <- centmatch( looki );
R>      looki <- MergeForce( looki );
R>
R>      nm <- dim( looki$matches );
R>
R>      # Make sure there is at least one match.
R>      if( any( nm == 0 ) ) next
R>
R>      for( j in 1:nm[ 1 ] ) {
R>
R>              cat( j, " " );
R>
R>              Xftr <- looki$X.feats[[ looki$matches[ j, 2 ] ]];
R>              Yftr <- looki$Y.feats[[ looki$matches[ j, 1 ] ]];
R>
```

```
R>              Xprops <- FeatureProps( Xftr );
R>              Yprops <- FeatureProps( Yftr );
R>
R>              Xcen <- c( Xprops$centroid$x, Xprops$centroid$y );
R>              Ycen <- c( Yprops$centroid$x, Yprops$centroid$y );
R>
R>              bval <- bearing( Xcen, Ycen );
R>              gval <- TheBigG( Xftr, Yftr );
R>
R>              bgres <- rbind( bgres, c( bval, c( gval ) ) );
R>
R>      } # end of inner 'j' loop.
R>
R>      cat( "\n" );
R>
R> } # end of outer 'i' loop.
R>
R> CircleHistogram( sqrt( bgres[,2] ), bgres[,1],
R>    COLS = sequential_hcl(4, palette = "Hawaii"),
R>                leg = TRUE, main = "", units = "g.p." );
```

Inspection of Figure 23 suggests that forecast features have centroids that do not tend to fall in any particular direction from the observed features more than another, although the worst matches (with $\sqrt{G} > 36.99$) are all directly south of the observed feature. There is a lack of bearings between about 180 and 275 degrees.

Of course, similar issues as demonstrated in, for example, Figure 4 for image warping must be considered. For example, should C1 be matched to the upper or lower circle from C6, or should the two circles of C6 be "merged" and considered a single feature? If not merged, then the choice of matching C1 to either of the two C6 circles is arbitrary and will depend on the matching algorithm used. Because each circle is in the opposite direction, this choice may affect the outcome of the circle histogram. In fact, if the two circles of C6 are merged, the same problem is apparent because an arbitrary choice must be made in terms of deciding which nearest point of C6 should the bearing be calculated from C1.[21] Such considerations should be taken into account when drawing conclusions from the circle histogram, but it does nevertheless provide useful information; if only as a starting point to suggest where to look for sources of error.

The feature matches represented in Figure 23 are only those features that have been matched, and it says nothing about those features that were not matched. Nevertheless, it represents pairs of features that are chosen by some measure, in this case centroid distance, to be well forecasted features. The fact that there are many cases with large values of G provides evidence that centroid distance might not be such a good measure of forecast accuracy in terms of spatial alignments. Gilleland *et al.* (2020) provide numerous contrived examples

[21] Note that the centroid of the merged C6 example is the same as the centroid of C1 so that the centroid distance is zero (a perfect score!) so the merged C6 would certainly be matched with C1 if using this measure to define matches.

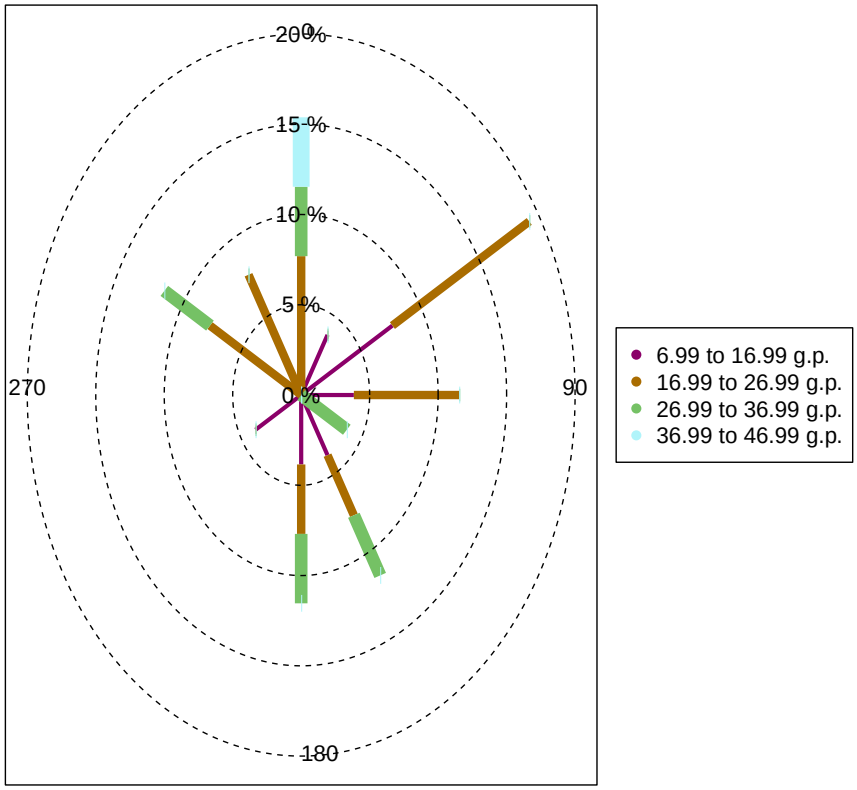

Figure 23: Circle histogram of matched features for the nine ICP real cases using a smoothing parameter of 5 grid points (abbreviated g.p.), threshold of 5.1 mmh^{-1}, and centroid distance for matching features. Histograms on the petals are for $\sqrt{G}$ (see Section 5).

where the centroid distance does not give useful information about forecast accuracy, so Figure 23 provides evidence that those types of scenarios are not unrealistic. This result does not mean that centroid distance is not useful for matching features, however. There are many instances where the only two features in a region are wildly different (e.g., they might have large area differences) but being the only two features, it seems reasonable to compare them further rather than declare them to be misses and false alarms.

The `FeatureAxis` function could be used on individual features in order to make circle histograms of any individual-feature property (e.g., A_{index}), using the orientation angle as the petal angles, for the forecast and observed fields separately. This analysis is not carried out here.

8.2. Point observations v. gridded forecast

Observations often come from point locations in space, rather than on a grid. Such observations may be used to inform an analysis that can be put on the same grid as the forecast, as

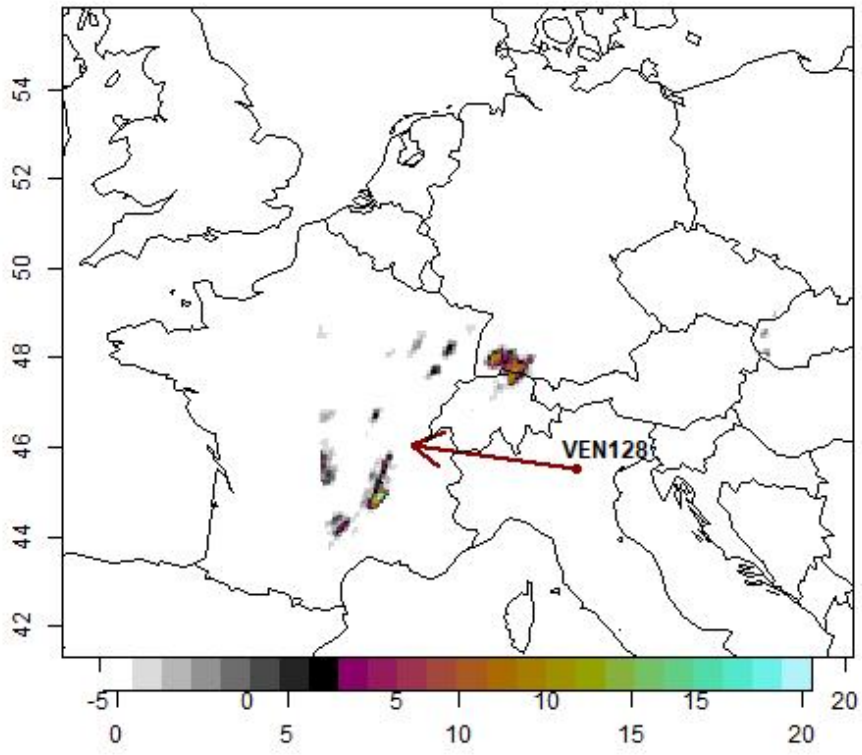

Figure 24: One example from the MesoVICT core case. Precipitation is in mmh^{-1} and the forecast is valid on 6 June 2007 at 0 UTC. Also shown is the VEN128 observing station obtained from the JDC data network (Dorninger *et al.* 2013, 2018). The arrow points to the nearest forecast grid point with heavy precipitation (> 5.1 mmh^{-1}).

in the examples of this paper, but it may also be of interest to compare the point location data directly to the forecast values on their grid.

The approach described here is similar in spirit to that of Brunet, Sills, and Casati (2018). The idea, here, is to calculate the shortest distance and bearing from the observing station to the nearest "event" (e.g., where the variable exceeds a high threshold). Such information can be aggregated over many cases and displayed in a circle histogram as in the previous section. The same uniqueness issue as before applies in that there may be more than one such nearest location so that the choice of which to use may be arbitrary. Nevertheless, the information provided should prove useful in diagnosing a forecasting system.

Figure 24 shows an example from the MesoVICT core case along with a station location and an arrow displaying its bearing and distance to the nearest forecast precipitation rate exceeding 5.1 mmh^{-1}. Figure 25 shows the corresponding distance map (grid points; left), as well as the distances (km) from this station to each individual grid point.

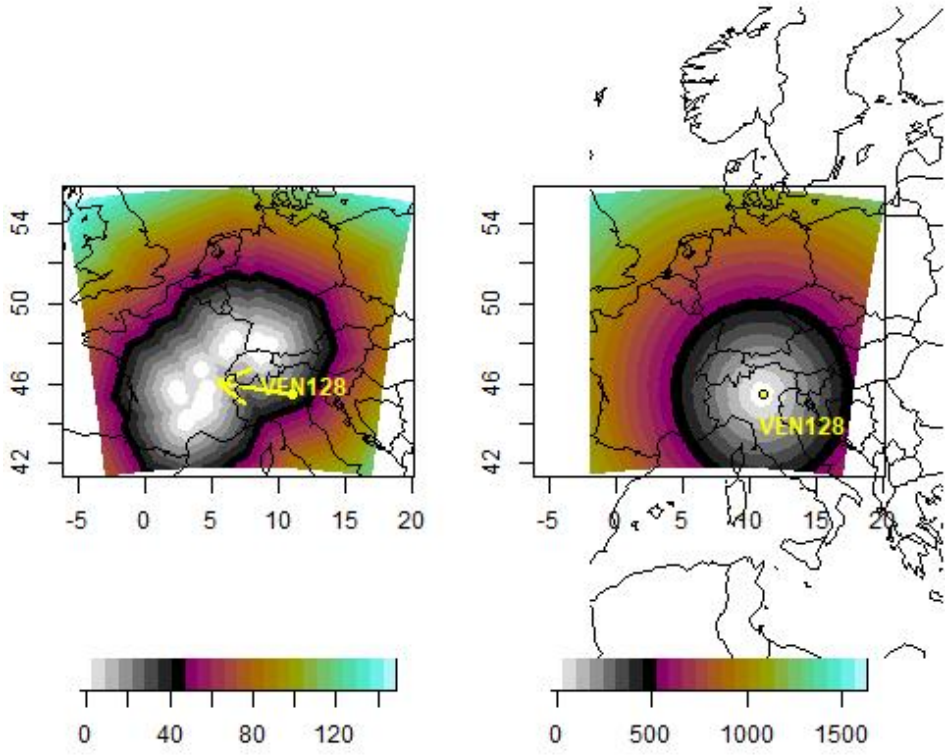

Figure 25: Distance map (grid squares) for the field in Figure 24 using a threshold of 5.1 mmh^{-1} (left) and the distance (km) between the VEN128 station and each forecast grid point.

The following code demonstrates how to find the shortest distance from an observing station location to the nearest forecast grid point where, in this case precipitation rate, exceeds a threshold (here 5.1 mmh^{-1}), as well as the bearing from this station to this grid point. Data are obtained from https://projects.ral.ucar.edu/icp/ under the "MesoVICT Cases" tab (last downloaded on 3 April 2021). The `[path]` must be replaced with the specific path where the data are downloaded to your own computer.

```
R> xdat <- read.table( "[path]/200706062000_ngts.txt" );
R> colnames( xdat ) <- c( "stationID", "lat", "lon", "pressureHGT",
+                         "stationHGT", "visibility", "windDIR",  "windSPD",
+                         "airTemp", "dewptTemp", "minairTemp",  "maxairTemp",
+                         "pressureStationHGT",  "pressureSeaLvl", "airpresstend",
+                         "presstraceform", "pcprate", "pcpaccumperiod", "presentWx",
+                         "mostsigWx", "pennultimatesigWx", "totcloudCVR",
+                         "lowlvlcloudCVR",  "cloudBSA", "lowcloudTYP",  "medcloudTYP",
```

```
+                    "highcloudTYP", "gustWSP" );
R> xdat[ xdat == -9990 ] <- NA;
R> xdat[ , "pcprate" ] <- xdat[ , "pcprate" ] * 10;

R> mod <- read.table( "[path]/cmh_8km_2007062100+18.dat", skip = 46 );
R> mod <- mod[, -c(3, 4, 10, 11, 14:17) ];
R> colnames( mod ) <- c( "x-coordinate (km, distance from origin)",
+         "y-coordinate (km, distance from origin)",
+         "precipitation (mm/h)", "10m u-wind (m/s)",
+         "10m v-wind (m/s)", "2m potential temperature (deg C)",
+         "2m equivalent potential temperature (deg C)",
+         "msl - pressure (hPa) reduced with model formulae",
+         "msl - pressure (hPa) reduced with std pressure reduction formulae",
+         "mixing ratio ( kg / kg * 10#(-3), post processing)",
+         "moisture flux divergence (kg / (kg * s#(-1)* 10#(-4)), post processing)" );

R> mod <- matrix( mod[,3], 209, 193, byrow = TRUE );
R> mod[ mod == 9999 ] <- NA;

R> mloc <- read.table( "[path]/VERA_8km_coordinates_lam_phi.txt" );
R> ml <- list( lon = matrix( mloc[,1], 209, 193, byrow = TRUE ),
+          lat = matrix( mloc[,2], 209, 193, byrow = TRUE ) );
R> mlr <- apply( mloc, 2, range, finite = TRUE );

R> testsite <- xdat[ xdat[,"stationID"] == "VEN128", ];

R> Zmod <- im( mod ); Zmod <- solutionset( Zmod > 5.1 );
R> ZmodDM <- distmap( Zmod ); ZmodDM <- as.matrix( ZmodDM );

R> dmod <- rdist.earth( rbind( c( 11.01372, 45.50916 ) ), mloc, miles = FALSE );
R> dmodMAT <- matrix( dmod, 209, 193, byrow = TRUE );

R> IDz <- c( ZmodDM == 0 );
R> ds <- c( ml$lon[ dmodMAT  == min( c( dmod )[ IDz ] ) ],
+                c( ml$lat[ dmodMAT == min( c( dmod )[ IDz ] ) ] ) );

R> bearing( rbind( c( 11.01372, 45.50916 ) ), ds );
R> rdist.earth( rbind( c( 11.01372, 45.50916 ) ), rbind( ds ), miles = FALSE );

R> par( oma = c( 2.1, 0, 0, 0 ) );
R> map( type = "n", xlim = mlr[,1], ylim = mlr[,2] );
R> poly.image( ml$lon, ml$lat, mod, col = cl, axes = FALSE, add = TRUE,
+              xlab = "", ylab = "" );
R> arrows( x0 = 11.01372, y0 = 45.50916, x1 = ds[ 1 ], y1 = ds[ 2 ],
+           col = "darkred", lwd = 2 );
R> map( add = TRUE );
R> map.axes()
```

```
R> image.plot( mod, col = cl, legend.only = TRUE, horizontal = TRUE );
R> points( testsite[,"lon"], testsite[,"lat"], pch = 21,
+             col = "darkred", bg = "darkred" );
R> text( 12, 46.0, labels = "VEN128", font = 2 );

R> par( oma = c( 2.1, 0, 0, 0 ) );
R> map( type = "n", xlim = mlr[,1], ylim = mlr[,2] );
R> poly.image( ml$lon, ml$lat, ZmodDM, col = cl, axes = FALSE, add = TRUE,
+               xlab = "", ylab = "" );
R> arrows( x0 = 11.01372, y0 = 45.50916, x1 = ds[ 1 ], y1 = ds[ 2 ],
+           col = "yellow", lwd = 2 );
R> map( add = TRUE );
R> map.axes()
R> image.plot( ZmodDM, col = cl, legend.only = TRUE, horizontal = TRUE );
R> points( testsite[,"lon"], testsite[,"lat"], pch = 21,
+                col = "yellow", bg = "yellow" );
R> text( 12, 46.0, labels = "VEN128", font = 2, col = "yellow" );

R> par( oma = c( 2.1, 0, 0, 0 ) );
R> map( type = "n", xlim = mlr[,1], ylim = mlr[,2] );
R> poly.image( ml$lon, ml$lat, dmodMAT, col = cl, axes = FALSE, add = TRUE,
+               xlab = "", ylab = "" );
R> map( add = TRUE );
R> map.axes()
R> image.plot( dmodMAT, col = cl, legend.only = TRUE, horizontal = TRUE );
R> points( testsite[,"lon"], testsite[,"lat"], pch = 21,
+                col = "black", bg = "yellow" );
R> text( 12, 46.0, labels = "VEN128", font = 2, col = "yellow" );
```

9. Summary and Discussion

This document provides background on high-resolution spatial weather forecast verification limited to the analysis of a single snapshot in time of space. Some major methods were described in detail, such as the feature-based and spatial alignment summary measures. Some newer or even untested methods were presented in order to help uncover some of the challenges inherent with evaluating a high-resolution forecast product. It is hoped that the in-depth look at some of the existing methods may spur new ideas from readers who might wish to pursue methodological advances in this domain. This document also serves as an introduction to the R software package **SpatialVx**. Many of the utilities described have a broader application reach than forecast verification.

A complete weather forecast system includes many components, such as the data assimilation system, physics package, boundary conditions, and statistical post-processing. The methods discussed here could be applied to several of these components. Moreover, a forecast system is generally run over many time points with differing lead and cycle times (e.g., one forecast cycle might be initialized at 0 UTC and another at 12 UTC). A forecast product might

have unique error sources depending on the lead and cycle times and should be considered separately. Aggregating evaluative methods over multiple cases can be challenging.

The feature-based methods provide a reasonably exhaustive set of tools for analyzing every aspect of forecast performance and are widely used in practice. However, they also require the user to make many decisions about how to carry out the analysis, and each decision can have a large impact on the results. The wealth of information provided by these methods may have to be summarized over the numerous cycles and lead times mentioned above, and much information may be lost in this process.

If the desired outcome of a verification analysis is to rank competing forecasts, then spatial alignment summary measures are highly useful. These methods do not suffer from the complexity issue of the feature-based approaches but a single summary measure will not provide all of the infomation about how one forecast performed better than another. That is, diagnostic information about forecast performance is scarce. These simple summaries provide complementary tools to the more complex feature-based approaches.

9.1. Other appication areas and future directions

Several avenues of research in the realm of spatial verification exist and are worth exploring. Only some such issues are briefly mentioned here. Although most methods were developed with precipitation forecasts in mind, they generally apply to a wide array of possible variables. In fact, they have been applied in many different domains, such as hydrology (e.g., Koch, Jensen, and Stisen 2015; Koch, Siemann, Stisen, and Sheffield 2016; Koch, Demirel, and Stisen 2018; Vereecken, Pachepsky, Simmer, Rihani, Kunoth, Korres, Graf, Franssen, Thiele-Eich, and Shao 2016; Courdent, Grum, and Steen Mikkelsen 2018, where new methods have also been developed), the Madden-Julian Oscillation (MJO) initialization (Kerns and Chen 2014), fire spread (e.g., Duff, Chong, Taylor, and Tolhurst 2012), pollutant fields (e.g., Farchi, Bocquet, Roustan, Mathieu, and Quérel 2016), large-scale indicators for severe weather (Gilleland *et al.* 2016), and synoptic-scale Rossby waveguides (e.g., Giannakaki and Martius 2016). Nevertheless, precipitation has been the application area where the methods are used most often, leaving room for modifications of the techniques to suit other variables.

Many forecast systems provide probabilistic forecast information, often via ensemble forecasts. Examples from the literature include Basu, S. and Dodov, B. and Foufoula-Georgiou, E. (2003); Ben Bouallègue and Theis (2014); Gallus (2010); Vich, Romero, and Homar (2011-11-01); Duc, Saito, and Seko (2013); Dey, Leoncini, Roberts, Plant, and Migliorini (2014); Fox, Micheas, and Peng (2016); Mittermaier and Csima (2017) and Radanovics *et al.* (2018). Climate is an arena that is particularly well suited for spatial-verification use because the numbers of models to be evaluated is typically fewer than in weather forecast verification, although this point may be changing, as for example, with some of the new model intercomparison projects. Examples of spatial verification use in climate include: De Sales and Xue (2010); Moise and Delage (2011); Kwiotkowski *et al.* (2014); Lockhoff, Zolina, Simmer, and Schulz (2014); Siongco *et al.* (2014); Yorgun and Rood (2014); De Haan *et al.* (2015); Hobæk Haff *et al.* (2015); Konca-Kedzierska (2015) and Gilleland *et al.* (2016).

Some discussion of comparing point observations with a gridded forecast was given here but generally, **SpatialVx** is aimed at verification sets (i.e., a forecast product and an observation whereby the forecast is valid at the same time as the observation occurs) where both the forecast and observation are gridded on the same regular grid. Tustison, Harris, and

Foufoula-Georgiou (2001) discuss the issue of representativeness error between forecasts and observations.

Related to the issue of having both the observed and forecast fields on the same regular grid concerns the type of grid itself. As forecast systems move to a more natural mesh, some of the computational tricks used with **SpatialVx** may no longer apply, or may need to be updated.

As mentioned above, a forecast system might cover multiple points in time. One way to handle this situation is to employ boxplots to show the distribution of spatial summary measures across time. Another solution is to aggregate the information across time, which may lead to loss of information. A third, more difficult solution, is to incorporate both space and time into the evaluation. A few examples in the literature include, MODE Time-Domain (e.g., Clark, Bullock, Jensen, Xue, and Kong 2014), which treats forecast features as three-dimensional geometric shapes through space and time, and image warping (e.g., Gilleland *et al.* 2010b), which is a natural extension of the image warping procedure from two-dimensional space to three-dimensional space. Beyond employing boxplots, the latter two approaches are ripe for further research.

Inference for spatial verification

As mentioned in Section 7, there is a distinct lack of inference methodology associated with most of the various spatial verification approaches. Image warping (as well as the closely related shape analysis) is an exception in that it is set up in a classical statistical modeling framework that is similar to a regression model. Therefore, it is possible to derive parametric inferential procedures based on distributional assumptions. Penalized likelihood or Bayesian estimation are the most natural choices for estimating the model because of the need to penalize non-physical deformations. However, for most other methods, there is no elegant statistical model from which to base inference. Spatial fields in meteorological data sets tend to be very large so that any inferential procedure must be computable in an efficient manner, and although many such methods exist (see Heaton, Datta, Finley, Furrer, Guiness, Guhaniyogi, Gerber, Gramacy, Hammerling, Katzfuss, Lindgren, Nychka, Sun, and Zammit-Mangion 2019, for a thorough review) for Gaussian distributed data, many of the important meteorological fields do not readily present themselves for such treatment.[22] Because many procedures involve binary fields, a spatial Poisson process (or similar) may be appropriate as a starting point.

Surrogate fields are often used in image analysis and computer vision and could be an avenue for obtaining uncertainty information for the various spatial verification methods as they reproduce the spatial distribution of the spatial field. Venugopal, Basu, and Foufoula-Georgiou (2005) use them to normalize their foreast quality index (FQI) by a representative measure.[23] Despite that this approach produces realistic spatial fields with reasonable distributional properties, they do not typically make physical sense.

In particular, Figure 26 shows one realization of a surrogate field obtained from the `obs0601` field shown in the top left panel of Figure 3. Clearly, there is little resemblance between this field and that of `obs0601`. The large amount of small-intensity noise can be removed, but

[22] Gilleland *et al.* (2010c) employ a Markov random field approach to the estimation of the deformations in their image warping procedure, but stopped short of carrying out any inference.

[23] The FQI measure introduced by Venugopal *et al.* (2005) incorporates intensity and a spatial displacement summary. The FQI is not discussed in this document but is available with **SpatialVx** through the function `FQI`. Its help file should be sufficient to learn how to apply the function.

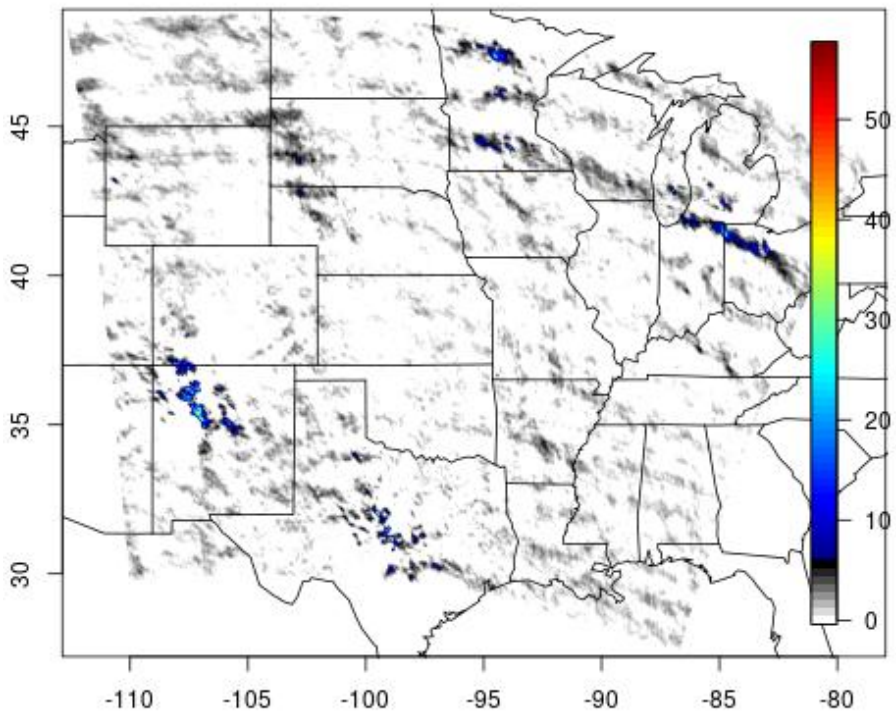

Figure 26: Example of one surrogate realization from the 1 June 2005 analysis example shown in the top left panel of Figure 3.

resulting features will not only be more numerous and in very different locations but may not resemble the original properties enough so as to provide meaningful uncertainty estimates.

The code to produce surrogate fields using **SpatialVx** is shown below. The returned object, `Z`, is an array with dimensions $601 \times 501 \times 10$ where the third dimension gives the ten realizations (the default value for argument `n` is ten).

```
R> Z <- surrogater2d( Im = obs0601 );
R> map( type = "n", xlim = lr[,1], ylim = lr[,2] );
R> poly.image( l$lon, l$lat, Z[,,1], col = cl,
+        xlab = "", ylab = "", axes = FALSE, add = TRUE );
R> map( add = TRUE, database = "state" );
R> map.axes()
R> image.plot( Z[,,1], col = cl, legend.only = TRUE );
```

Acknowledgments

This work was sponsored in part by the National Center for Atmospheric Research (NCAR), which is a major facility sponsored by the National Science Foundation under Cooperative Agreement No. 1852977.

References

AghaKouchak A, Mehran A (2013). "Extended contingency table: Performance metrics for satellite observations and climate model simulations." *Water Resources Research*, **49**(10), 7144–7149.

AghaKouchak A, Nasrollahi N, Li J, Imam B, Sorooshian S (2011). "Geometrical Characterization of Precipitation Patterns." *Journal of Hydrometeorology*, **12**(2), 274–285.

Ahijevych D, Gilleland E, Brown BG, Ebert EE (2009). "Application of Spatial Verification Methods to Idealized and NWP-Gridded Precipitation Forecasts." *Weather Forecast.*, **24**(6), 1485–1497.

Baddeley A, Rubak E, Turner R (2015). *Spatial Point Patterns: Methodology and Applications with R*. Chapman and Hall/CRC Press, London, U.K., 828 pp. URL http://www.crcpress.com/Spatial-Point-Patterns-Methodology-and-Applications-with-R/Baddeley-Rubak-Turner/9781482210200.

Baddeley AJ (1992a). "Errors in binary images and an Lp version of the Hausdorff metric." *Nieuw Arch. Wiskunde*, **10**, 157–183.

Baddeley AJ (1992b). *Robust Computer Vision Algorithms*, chapter An error metric for binary images, pp. 59–78. W. Forstner and S. Ruwiedel, Eds., Wichmann, 402 pp.

Baddeley AJ, Turner R (2005). "Spatstat: an R package for analyzing spatial point patterns." *Journal of Statistical Software*, **12**(6), 1–42.

Baldwin ME, Kain JS (2006). "Sensitivity of Several Performance Measures to Displacement Error, Bias, and Event Frequency." *Weather Forecast.*, **21**(4), 636–648.

Basu, S and Dodov, B and Foufoula-Georgiou, E (2003). "A Novel Measure for QPF Verification and its Usefulness in Multimodel Ensemble Forecasting." In *EGS - AGU - EUG Joint Assembly*, volume 5, p. 4323. Provided by the SAO/NASA Astrophysics Data System, http://adsabs.harvard.edu/abs/2003EAEJA.....4323B.

Becker RA, Wilks A, Brownrigg R, Minka TP, Deckmyn A (2018). *maps: Draw Geographical Maps*. R package version 3.3.0, URL https://CRAN.R-project.org/package=maps.

Ben Bouallègue Z, Theis SE (2014). "Spatial techniques applied to precipitation ensemble forecasts: from verification results to probabilistic products." *Meteorological Applications*, **21**(4), 922–929. ISSN 1469-8080.

Bivand RS, Pebesma EJ, Gomez-Rubio V (2013). *Applied Spatial Data Analysis with R*. Second Edition, Springer, New York, NY, 405pp.

Bobb JF, Varadhan R (2020). *turboEM: A Suite of Convergence Acceleration Schemes for EM, MM and Other Fixed-Point Algorithms*. R package version 2020.1, URL https://CRAN.R-project.org/package=turboEM.

Briggs WM, Levine RA (1997). "Wavelets and Field Forecast Verification." *Monthly Weather Review*, **125**(6), 1329–1341.

Brill KF, Mesinger F (2009). "Applying a General Analytic Method for Assessing Bias Sensitivity to Bias-Adjusted Threat and Equitable Threat Scores." *Weather Forecast.*, **24**(6), 1748–1754.

Brown BG, Jensen TG, Halley Gotway J, Bullock RG, Gilleland E, Fowler T, Newman K, Adriaansen D, Blank L, Burek T, Harrold M, Hertneky T, Kalb C, Kucera P, Nance L, Wolff J (2021). "The Model Evaluation Tools (MET): More than a decade of community-supported forecast verification." *Bulletin of the American Meteorological Society*. doi:10.1175/BAMS-D-19-0093.1.

Brune S, Buschow S, Friederichs P (2021). "The local wavelet-based organization index–Quantification, localization and classification of convective organization from radar and satellite data." *Quarterly Journal of the Royal Meteorological Society (in press)*. doi:10.1002/qj.3998.

Brunet D, Sills D (2016). "A Generalized Distance Transform: Theory and Applications to Weather Analysis and Forecasting." *IEEE Transactions on Geoscience and Remote Sensing*, **55**(3), 1752–1764. doi:10.1109/TGRS.2016.2632042.

Brunet D, Sills D, Casati B (2018). "A spatio-temporal user-centric distance for forecast verification." *Meterol. Zeit.*, **27**(6), 441–453. doi:10.1127/metz/2018/0883.

Buschow S, Friederichs P (2021). "SAD: Verifying the scale, anisotropy and direction of precipitation forecasts." *Quarterly Journal of the Royal Meteorological Society*, **147**(735), 1150–1169. doi:10.1002/qj.3964.

Buschow S, Pidstrigach J, Friederichs P (2019). "Assessment of wavelet-based spatial verification by means of a stochastic precipitation model (wv_verif v0.1.0)." *Geoscientific Model Development*, **12**(8), 3401–3418. doi:10.5194/gmd-12-3401-2019.

Canty A, Ripley B (2020). "boot: Bootstrap R (S-Plus) Functions." R package version 1.3-25.

Casati B (2010). "New Developments of the Intensity-Scale Technique within the Spatial Verification Methods Intercomparison Project." *Weather Forecast.*, **25**(1), 113–143.

Casati B, Ross G, Stephenson DB (2004). "A new intensity-scale approach for the verification of spatial precipitation forecasts." *Meteorological Applications*, **11**(2), 141–154.

Clark AJ, Bullock RG, Jensen TL, Xue M, Kong F (2014). "Application of Object-Based Time-Domain Diagnostics for Tracking Precipitation Systems in Convection-Allowing Models." *Weather and Forecasting*, **29**(3), 517–542.

Courdent V, Grum M, Steen Mikkelsen P (2018). "Distinguishing high and low flow domains in urban drainage systems 2 days ahead using numerical weather prediction ensembles." *Journal of Hydrology*, **556**, 1013–1025.

Davis C, Brown B, Bullock R (2006a). "Object-Based Verification of Precipitation Forecasts. Part I: Methodology and Application to Mesoscale Rain Areas." *Monthly Weather Review*, **134**(7), 1772–1784.

Davis C, Brown B, Bullock R (2006b). "Object-Based Verification of Precipitation Forecasts. Part II: Application to Convective Rain Systems." *Monthly Weather Review*, **134**(7), 1785–1795.

Davis CA, Brown BG, Bullock R, Halley Gotway J (2009). "The Method for Object-Based Diagnostic Evaluation (MODE) Applied to Numerical Forecasts from the 2005 NSSL/SPC Spring Program." *Weather Forecast.*, **24**(5), 1252–1267.

Davison A, Hinkley DV (1997). *Bootstrap Methods and Their Applications*. Cambridge University Press, Cambridge, U.K. ISBN 0-521-57391-2.

De Haan L, Kanamitsu LM, De Sales F, Sun L (2015). "An evaluation of the seasonal added value of downscaling over the United States using new verification measures." *Theoretical and Applied Climatology*, **122**(1), 47–57. doi:10.1007/s00704-014-1278-9.

De Sales F, Xue Y (2010). "Assessing the dynamic-downscaling ability over South America using the intensity-scale verification technique." *International Journal of Climatology*, **31**(8), 1205–1221. doi:10.1002/joc.2139.

Dey SRA, Leoncini G, Roberts NM, Plant RS, Migliorini S (2014). "A spatial view of ensemble spread in convection permitting ensembles." *Monthly Weather Review*.

Diebold FX, Mariano RS (1995). "Comparing predictive accuracy." *J. Bus. Econ. Stat.*, **13**, 253–263.

Diggle PJ, Ribeiro Jr PJ (2006). *Model-based Geostatistics*. Springer, New York, NY, 230pp.

Dorninger M, Gilleland E, Casati B, Mittermaier MP, Ebert EE, Brown BG, Wilson LJ (2018). "The set-up of the Mesoscale Verification Inter-Comparison over Complex Terrain project." *Bulletin of the American Meteorological Society*, **99**(9), 1887–1906. doi:10.1175/BAMS-D-17-0164.1.

Dorninger M, Mittermaier MP, Gilleland E, Ebert EE, Brown BG, Wilson LJ (2013). "MesoVICT: Mesoscale Verification Inter-Comparison over Complex Terrain." *Technical Report NCAR/TN-505+STR*, NCAR.

Dryden IL, Mardia KV (1998). *Statistical Shape Analysis*. John Wiley and Sons, Inc., New York, NY, U.S.A., 347 pp.

Duc L, Saito K, Seko H (2013). "Spatial-temporal fractions verification for high-resolution ensemble forecasts." *Tellus A*, **65**(18171).

Duff TJ, Chong DM, Taylor P, Tolhurst KG (2012). "Procrustes based metrics for spatial validation and calibration of two-dimensional perimeter spread models: A case study considering fire." *Agricultural and Forest Meteorology*, **160**(0), 110 – 117. ISSN 0168-1923.

Ebert EE (2008). "Fuzzy verification of high-resolution gridded forecasts: a review and proposed framework." *Meteorological Applications*, **15**(1), 51–64.

Ebert EE (2009). "Neighborhood Verification: A Strategy for Rewarding Close Forecasts." *Weather Forecast.*, **24**(6), 1498–1510.

Ebert EE, Gallus WA (2009). "Toward Better Understanding of the Contiguous Rain Area (CRA) Method for Spatial Forecast Verification." *Weather Forecast.*, **24**(5), 1401–1415.

Ebert EE, McBride JL (2000). "Verification of precipitation in weather systems: determination of systematic errors." *Journal of Hydrology*, **239**(1–4), 179–202.

Farchi A, Bocquet M, Roustan Y, Mathieu A, Quérel A (2016). "Using the Wasserstein distance to compare fields of pollutants: application to the radionuclide atmospheric dispersion of the Fukushima-Daiichi accident." *Tellus B*, **68**(0). ISSN 1600-0889.

Finley AO, Banerjee S, Carlin BP (2007). "spBayes: An R Package for Univariate and Multivariate Hierarchical Point-Referenced Spatial Models." *Journal of Statistical Software*, **19**(4), 1–24. URL https://www.jstatsoft.org/article/view/v019i04.

Finley AO, Banerjee S, EGelfand A (2015). "spBayes for Large Univariate and Multivariate Point-Referenced Spatio-Temporal Data Models." *Journal of Statistical Software*, **63**(13), 1–28. URL https://www.jstatsoft.org/article/view/v063i13.

Fox NI, Micheas AC, Peng Y (2016). "Applications of Bayesian Procrustes shape analysis to ensemble radar reflectivity nowcast verification." *Atmospheric Research*, **176–177**, 75–86.

Gallus WA (2010). "Application of Object-Based Verification Techniques to Ensemble Precipitation Forecasts." *Weather Forecast.*, **25**(1), 144–158.

Giannakaki P, Martius O (2016). "An Object-Based Forecast Verification Tool for Synoptic-Scale Rossby Waveguides." *Weather Forecast.*, **31**(3), 937–946.

Gilleland E (2011). "Spatial Forecast Verification: Baddeley's Delta Metric Applied to the ICP Test Cases." *Weather Forecast.*, **26**(3), 409–415.

Gilleland E (2013a). "Testing Competing Precipitation Forecasts Accurately and Efficiently: The Spatial Prediction Comparison Test." *Monthly Weather Review*, **141**(1), 340–355.

Gilleland E (2013b). "Two-dimensional kernel smoothing: Using the R package smoothie." *Technical report*, NCAR Technical Note, TN-502+STR, 17pp. URL http://opensky.library.ucar.edu/collections/TECH-NOTE-000-000-000-869.

Gilleland E (2017). "A New Characterization within the Spatial Verification Framework for False Alarms, Misses, and Overall Patterns." *Weather and Forecasting*, **32**(1), 187–198.

Gilleland E (2020a). "Bootstrap methods for statistical inference. Part I: Comparative forecast verification for continuous variables." *Journal of Atmospheric and Oceanic Technology*, **37**(11), 2117–2134. doi:10.1175/JTECH-D-20-0069.1.

Gilleland E (2020b). "Bootstrap methods for statistical inference. Part II: Extreme-value analysis." *Journal of Atmospheric and Oceanic Technology*, **37**(11), 2135–2144. doi:10.1175/JTECH-D-20-0070.1.

Gilleland E (2021). "Novel measures for summarizing high-resolution forecast performance." *Advances in Statistical Climatology, Meteorology and Oceanography*, **7**(1), 13–34. doi:10.5194/ascmo-7-13-2021.

Gilleland E, Ahijevych D, Brown BG, Casati B, Ebert EE (2009). "Intercomparison of Spatial Forecast Verification Methods." *Weather Forecast.*, **24**(5), 1416–1430.

Gilleland E, Ahijevych DA, Brown BG, Ebert EE (2010a). "Verifying Forecasts Spatially." *Bulletin of the American Meteorological Society*, **91**(10), 1365–1373.

Gilleland E, Brown BG, Ammann CM (2013). "Spatial extreme value analysis to project extremes of large-scale indicators for severe weather." *Environmetrics*, **24**(6), 418–432. doi:10.1002/env.2234.

Gilleland E, Bukovsky M, Williams CL, McGinnis S, Ammann CM, Brown BG, Mearns LO (2016). "Evaluating NARCCAP model performance for frequencies of severe-storm environments." *Advances in Statistical Climatology, Meteorology and Oceanography*, **2**(2), 137–153.

Gilleland E, Chen L, DePersio M, Do G, Eilertson K, Jin Y, Lang E, Lindgren F, Lindström J, Smith R, Xia C (2010b). "Spatial forecast verification: image warping." *Technical Report NCAR/TN-482+STR*, NCAR.

Gilleland E, Lee TCM, Halley Gotway J, Bullock RG, Brown BG (2008). "Computationally Efficient Spatial Forecast Verification Using Baddeley's Delta Image Metric." *Monthly Weather Review*, **136**(5), 1747–1757.

Gilleland E, Lindström J, Lindgren F (2010c). "Analyzing the image warp forecast verification method on precipitation fields from the ICP." *Weather Forecast.*, **25**(4), 1249–1262.

Gilleland E, Skok G, Brown BG, Casati B, Dorninger M, Mittermaier MP, Roberts N, Wilson LJ (2020). "A novel set of verification test fields with application to distance measures." *Monthly Weather Review*, **148**(4), 1653–1673. doi:10.1175/MWR-D-19-0256.1.

Gräler B, Pebesma EJ, Heuvelink G (2016). "Spatio-Temporal Interpolation using gstat." *The R Journal*, **8**, 204–218. doi:10.32614/RJ-2016-014.

Harris D, Foufoula-Georgiou E, Droegemeier KK, Levit JJ (2001). "Multiscale Statistical Properties of a High-Resolution Precipitation Forecast." *Journal of Hydrometeorology*, **2**(4), 406–418.

Heaton MJ, Datta A, Finley AO, Furrer R, Guiness J, Guhaniyogi R, Gerber F, Gramacy RB, Hammerling D, Katzfuss M, Lindgren F, Nychka DW, Sun F, Zammit-Mangion A (2019). "ACase Study Competition among Methods for Analyzing Large Spatial Data." *Journal of Agricultural, Biological and Environmental Statistics*, **24**, 398–425. doi:10.1007/s13253-018-00348-w.

Hering AS, Genton MG (2011). "Comparing spatial predictions." *Technometrics*, **53**, 414–425.

Hijmans RJ (2021). *raster: Geographic Data Analysis and Modeling*. R package version 3.5-11, URL https://CRAN.R-project.org/package=raster.

Hobæk Haff I, Frigessi A, Maraun D (2015). "How well do regional climate models simulate the spatial dependence of precipitation? An application of pair-copula constructions." *Journal of Geophysical Research: Atmospheres*, **120**(7), 2624–2646. ISSN 2169-8996. 2014JD022748.

Hoffman RN, Liu Z, Louis JF, Grassoti C (1995). "Distortion Representation of Forecast Errors." *Monthly Weather Review*, **123**(9), 2758–2770.

Jun M, Knutti R, Nychka DW (2008). "Local eigenvalue analysis of CMIP3 climate model errors." *Tellus*, **60A**, 992–1000. doi:10.1111/j.1600-0870.2008.00356.x.

Kain JS, Weiss SJ, Bright DR, Baldwin ME, Levit JJ, Carbin GW, Schwartz CS, Weisman ML, Droegemeier KK, Weber DB, Thomas KW (2008). "Some practical considerations regarding horizontal resolution in the first generation of operational convection-allowing NWP." *Weather Forecast.*, **23**, 931–952.

Keil C, Craig GC (2007). "A Displacement-Based Error Measure Applied in a Regional Ensemble Forecasting System." *Monthly Weather Review*, **135**(9), 3248–3259.

Keil C, Craig GC (2009). "A Displacement and Amplitude Score Employing an Optical Flow Technique." *Weather Forecast.*, **24**(5), 1297–1308.

Kerns BW, Chen SS (2014). "ECMWF and GFS model forecast verification during DYNAMO: Multiscale variability in MJO initiation over the equatorial Indian Ocean." *Journal of Geophysical Research: Atmospheres*, **119**(7), 3736–3755. ISSN 2169-8996.

Koch J, Demirel MC, Stisen S (2018). "The SPAtial EFficiency metric (SPAEF): multiple-component evaluation of spatial patterns for optimization of hydrological models." *Geosci. Model Dev.*, **11**, 1873–1886. doi:10.5194/gmd-11-1873-2018.

Koch J, Jensen KH, Stisen S (2015). "Toward a true spatial model evaluation in distributed hydrological modeling: Kappa statistics, Fuzzy theory, and EOF-analysis benchmarked by the human perception and evaluated against a modeling case study." *Water Resources Research*, **51**(2), 1225–1246. ISSN 1944-7973.

Koch J, Siemann A, Stisen S, Sheffield J (2016). "Spatial validation of large-scale land surface models against monthly land surface temperature patterns using innovative performance metrics." *Journal of Geophysical Research: Atmospheres*, **121**(10), 5430–5452. ISSN 2169-8996. 2015JD024482.

Konca-Kedzierska K (2015). "Comparison of selected methods of analysis for reconstructed fields of precipitation in climate scenarios over Poland." pp. 1–9.

Kwiotkowski L, Halloran PR, Mumby PJ, Stephenson DB (2014). "What spatial scales are believable for climate projections of sea surface temperature?" *Climate Dynamics*, **43**(5), 1483–1496. doi:10.1007/s00382-013-1967-6.

Lack SA, Limpert GL, Fox NI (2010). "An Object-Oriented Multiscale Verification Scheme." *Weather Forecast.*, **25**(1), 79–92.

Lakshmanan V, Kain JS (2010). "A Gaussian Mixture Model Approach to Forecast Verification." *Weather Forecast.*, **25**(3), 908–920.

Levy AAL, Ingram W, Jenkinson M, Huntingford C, Hugo Lambert F, Allen M (2013). "Can correcting feature location in simulated mean climate improve agreement on projected changes?" *Geophysical Research Letters*, **40**(2), 354–358. ISSN 1944-8007.

Lindgren F, Rue H (2015). "Bayesian spatial modelling with R-INLA." *Journal of Statistical Software*, **63**(19), 1–25. doi:10.18637/jss.v063.i19.

Lindgren F, Rue H, Lindström J (2011). "An explicit link between Gaussian fields and Gaussian Markov Random Fields: The stochastic partial differential equation approach (with discussion)." *Journal of the Royal Statistical Society B*, **73**(4), 423–498. doi:10.1111/j.1467-9868.2011.00777.x.

Livina VN, Edwards NR, Goswami S, Lenton TM (2008). "A wavelet-coefficient score for comparison of two-dimensional climatic-data fields." *Q. J. R. Meteorol. Soc.*, **134**(633), 941–955. ISSN 1477-870X.

Lockhoff M, Zolina O, Simmer C, Schulz J (2014). "Evaluation of satellite-retrieved extreme precipitation over Europe using gauge observations." *J. Climate*, **27**, 607–623.

Lund U, Agostinelli C (2018). *CircStats: Circular Statistics, from "Topics in Circular Statistics" (2001)*. R package version 0.2-6, URL https://CRAN.R-project.org/package=CircStats.

Mariani S, Casaioli M (2018). "Effects of model domain extent and horizontal grid size on contiguous rain area (CRA) analysis: A MesoVICT study." *Meteorologische Zeitschrift*, **27**(6), 481–502. doi:10.1127/metz/2018/0897.

Marzban C, Sandgathe S (2006). "Cluster Analysis for Verification of Precipitation Fields." *Weather Forecast.*, **21**(5), 824–838.

Marzban C, Sandgathe S (2008). "Cluster Analysis for Object-Oriented Verification of Fields: A Variation." *Monthly Weather Review*, **136**(3), 1013–1025.

Marzban C, Sandgathe S (2009). "Verification with Variograms." *Weather Forecast.*, **24**(4), 1102–1120.

Marzban C, Sandgathe S (2010). "Optical Flow for Verification." *Weather Forecast.*, **25**(5), 1479–1494.

Marzban C, Sandgathe S, Lyons H, Lederer N (2009). "Three Spatial Verification Techniques: Cluster Analysis, Variogram, and Optical Flow." *Weather Forecast.*, **24**(6), 1457–1471.

Mass CF, Ovens D, Westrick K, Colle BA (2002). "Does Increasing Horizontal Resolution Produce More Skillful Forecasts?" *Bulletin of the American Meteorological Society*, **83**(3), 407–430.

Melo C, Santacruz A, Melo O (2012). *geospt: An R package for spatial statistics*. R package version 1.0-0, URL geospt.r-forge.r-project.org/.

Micheas AC, Fox NI, Lack SA, Wikle CK (2007). "Cell identification and verification of QPF ensembles using shape analysis techniques." *Journal of Hydrology*, **343**(3–4), 105–116.

Mittermaier MP, Csima G (2017). "Ensemble versus deterministic performance at the kilometer scale." *Weather and Forecasting*, **32**, 1697–1709. doi:10.1175/WAF-D-16-0164.1.

Mittermaier MP, Roberts N (2010). "Intercomparison of Spatial Forecast Verification Methods: Identifying Skillful Spatial Scales Using the Fractions Skill Score." *Weather Forecast.*, **25**(1), 343–354.

Moise AF, Delage FP (2011). "New climate model metrics based on object-orientated pattern matching of rainfall." *J. Geophys. Res.*, **116**(D12).

Müllner D (2013). "fastcluster: Fast Hierarchical, Agglomerative Clustering Routines for R and Python." *Journal of Statistical Software*, **53**(9), 1–18. URL http://www.jstatsoft.org/v53/i09/.

Nachamkin JE (2004). "Mesoscale Verification Using Meteorological Composites." *Monthly Weather Review*, **132**(4), 941–955.

Nachamkin JE (2009). "Application of the Composite Method to the Spatial Forecast Verification Methods Intercomparison Dataset." *Weather Forecast.*, **24**(5), 1390–1400.

Nachamkin JE, Chen S, Schmidt J (2005). "Evaluation of Heavy Precipitation Forecasts Using Composite-Based Methods: A Distributions-Oriented Approach." *Monthly Weather Review*, **133**(8), 2163–2177.

Nehrkorn T, Hoffman RN, Grassotti C, Louis JF (2003). "Feature calibration and alignment to represent model forecast errors: Empirical regularization." *Quarterly Journal of the Royal Meteorological Society*, **129**(587), 195–218.

Nychka DW, Furrer R, Paige J, Sain S (2017). "fields: Tools for spatial data." doi:10.5065/D6W957CT. R package version 10.3, URL https://github.com/NCAR/Fields.

Padoan SA, Bevilacqua M (2015). "Analysis of Random Fields Using CompRandFld." *Journal of Statistical Software*, **63**(9), 1–27. URL http://www.jstatsoft.org/v63/i09/.

Pebesma EJ (2012). "spacetime: Spatio-Temporal Data in R." *Journal of Statistical Software*, **51**(7), 1–30. doi:10.18637/jss.v051.i07.

Pebesma EJ (2018). "Simple features for R: Standardized support for spatial vecto data." *The R Journal*, **10**(1). doi:10.32614/RJ-2018-009.

Pebesma EJ, Bivand RS (2005). "Classes and methods for spatial data in R." *R News*, **5**. URL https://cran.r-project.org/doc/Rnews.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Radanovics S, Vidal JP, Sauquet E (2018). "Spatial Verification of Ensemble Precipitation: An Ensemble Version of SAL." *Weather and Forecasting*, **33**(4), 1001–1020.

Ribatet M (2020). *SpatialExtremes: Modelling Spatial Extremes*. R package version 2.0-9, URL https://CRAN.R-project.org/package=SpatialExtremes.

Rue H, Martino S, Chopin N (2009). "Approximate Bayesian inference for latent Gaussian models using integrated nested Laplace approximations (with discussion)." *Journal of the Royal Statistical Society B*, **71**(2), 319–392. doi:10.1111/j.1467-9868.2008.00700.x.

Sainsbury-Dale M, Zammit-Mangion A, Cressie N (2021). "Modelling, fitting, and prediction with non-Gaussian spatial and spatio-temporal data using FRK." *arXiv:2110.02507*.

Sampson PD, Guttorp P (1992). "Operational evaluation of air quality models." *Technical report*, NRCSE Technical Report, NRCSE-TRS 018, National Research Center for Statistics and the Environment, Seattle, Washington, 22pp.

Schlather M, Malinowski A, Menck PJ, Oesting M, Strokorb K (2015). "Analysis, Simulation and Prediction of Multivariate Random Fields with Package RandomFields." *Journal of Statistical Software*, **63**(8), 1–25. URL https://www.jstatsoft.org/v63/i08/.

Schlather M, Malinowski A, Oesting M, Boecker D, Strokorb K, Engelke S, Martini J, Ballani F, Moreva O, Auel J, Menck PJ, Gross S, Ober U, Ribeiro P, Ripley BD, Singleton R, Pfaff B, R Core Team (2022). *RandomFields: Simulation and Analysis of Random Fields*. R package version 3.3.14, URL https://cran.r-project.org/package=RandomFields.

Schwedler BRJ, Baldwin ME (2011). "Diagnosing the Sensitivity of Binary Image Measures to Bias, Location, and Event Frequency within a Forecast Verification Framework." *Weather Forecast.*, **26**(6), 1032–1044.

Siongco AC, Hohenegger C, Stevens B (2014). "The Atlantic ITCZ bias in CMIP5 models." *Climate Dynamics*.

Skoien JO, Bloschl G, Laaha G, Pebesma E, Parajka J, Viglione A (2014). "Rtop: An R package for interpolation of data with a variable spatial support, with an example from river networks." *Computers & Geosciences*.

Skok G, Hladnik V (2017). "Verification of Gridded Wind Forecasts in Complex Alpine Terrain: A New Wind Verification Methodology Based on the Neighborhood Approach." *Monthly Weather Review*, **146**(1), 63–75. doi:10.1175/MWR-D-16-0471.1. URL https://doi.org/10.1175/MWR-D-16-0471.1.

Smith BJ, Yan J, Cowles MK (2008). "Unified Geostatistical Modeling for Data Fusion and Spatial Heteroskedasticity with R Package ramps." *Journal of Statistical Software*, **25(10)**, 1–21.

Stauffer R, Mayr GJ, Dabernig M, Zeileis A (2009). "Somewhere over the rainbow: How to make effective use of colors in meteorological visualizations." *Bulletin of the American Meteorological Society*, **96**(2), 203–216. doi:10.1175/BAMS-D-13-00155.1.

Tustison B, Harris D, Foufoula-Georgiou E (2001). "Scale issues in verification of precipitation forecasts." *J. Geophys. Res.*, **106**(D11), 11775–11784.

Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. Fourth edition, Springer, New York, NY, 506 pp.

Venugopal V, Basu S, Foufoula-Georgiou E (2005). "A new metric for comparing precipitation patterns with an application to ensemble forecasts." *J. Geophys. Res.*, **110**(D8).

Vereecken H, Pachepsky Y, Simmer C, Rihani J, Kunoth A, Korres W, Graf A, Franssen HJH, Thiele-Eich I, Shao Y (2016). "On the role of patterns in understanding the functioning of soil-vegetation-atmosphere systems." *Journal of Hydrology*, **542**, 63–86.

Vich M, Romero R, Homar V (2011-11-01). "Ensemble prediction of Mediterranean high-impact events using potential vorticity perturbations. Part II: Adjoint-derived sensitivity zones." *Atmospheric Research*, **102**(3), 311(9).

Warton DI, Duursma RA, Falster DS, Taskinen S (2012). "smatr 3 - an R package for estimation and inference about allometric lines." *Methods in Ecology and Evolution*, **3**, 257–259.

Wealands SR, Grayson RB, Walker JP (2005). "Quantitative comparison of spatial fields for hydrological model assessment—-some promising approaches." *Advances in Water Resources*, **28**(1), 15–32.

Weniger M, Kapp F, Friederichs P (2016). "Spatial Verification Using Wavelet Transforms: A Review." *Q. J. R. Meteorol. Soc.*, **143**(702), 120–136.

Wernli H, Hofmann C, Zimmer M (2009). "Spatial Forecast Verification Methods Intercomparison Project: Application of the SAL Technique." *Weather Forecast.*, **24**(6), 1472–1484.

Wernli H, Paulat M, Hagen M, Frei C (2008). "SAL–A Novel Quality Measure for the Verification of Quantitative Precipitation Forecasts." *Monthly Weather Review*, **136**(11), 4470–4487.

Whitcher B (2020). *waveslim: Basic Wavelet Routines for One-, Two-, and Three-Dimensional Signal Processing*. R package version 1.8.2, URL https://CRAN.R-project.org/package=waveslim.

Wikle CK, Zammit-Mangion A, Cressie N (2019). *Spatio-Temporal Statistics with R*. Chapman & Hall/CRC, Boca Raton, Florida, U.S.A., 396pp. URL https://spacetimewithr.org.

Yorgun MS, Rood RB (2014). "An object-based approach for quantification of GCM biases of the simulation of orographic precipitation. Part I: Idealized simulations." *Journal of Climate*, **27**(24), 9139–9154.

Zammit-Mangion A, Cressie N (2021). "FRK: an R package for spatial and spatio-temporal prediction with large datasets." *Journal of Statistical Software*, **98**(4), 1–48.

Zeileis A, Fisher JC, Hornik K, Ihaka R, McWhite CD, Murrell P, Stauffer R, Wilke CO (2020). "Escaping RGBland: Selecting Colors for Statistical Graphics." *Computational Statistics & Data Analysis*, **53**(9), 3259–3270. doi:10.1016/j.csda.2008.11.033.

Affiliation:

Eric Gilleland
Research Applications Laboratory
National Center for Atmospheric Research
P.O. Box 3000
Boulder, Colorado 80307
E-mail: EricG@ucar.edu
URL: https://staff.ral.ucar.edu/ericg/