SpherWave: An R Package for Analyzing Scattered Spherical Data by Spherical Wavelets

by Hee-Seok Oh and Donghoh Kim

Introduction

Given scattered surface air temperatures observed on the globe, we would like to estimate the temperature field for every location on the globe. Since the temperature data have inherent multiscale characteristics, spherical wavelets with localization properties are particularly effective in representing multiscale structures. Spherical wavelets have been introduced in [Narcowich and Ward (1996)] and [Li (1999)]. A successful statistical application has been demonstrated in [Oh and Li (2004)].

*SpherWave* is an R package implementing the spherical wavelets (SWs) introduced by [Li (1999)] and the SW-based spatially adaptive methods proposed by [Oh and Li (2004)]. This article provides a general description of SWs and their statistical applications, and it explains the use of the *SpherWave* package through an example using real data.

Before explaining the algorithm in detail, we first consider the average surface air temperatures (in degrees Celsius) during the period from December 1967 to February 1968 observed at 939 weather stations, as illustrated in Figure 1.

Figure 1: Average surface air temperatures observed at 939 weather stations during the years 1967-1968.

In the *SpherWave* package, the data are obtained by

```R
> library(SpherWave)
> ### Temperature data from year 1961 to 1990
> ### list of year, grid, observation
> data(temperature)
> temp67 <- temperature$obs[temperature$year==1967]
> latlon <-
+   temperature$latlon[temperature$year==1967, ]
```

and Figure 1 is plotted by the following code.

```R
> sw.plot(z=temp67, latlon=latlon, type="obs",
+ xlab="", ylab="")
```

Similarly, various signals such as meteorological or geophysical signal in nature can be measured at scattered and unevenly distributed locations. However, inferring the substantial effect of such signals at an arbitrary location on the globe is a crucial task. The first objective of using SWs is to estimate the signal at an arbitrary location on the globe by extrapolating the scattered observations. An example is the representation in Figure 2, which is obtained by extrapolating the observations in Figure 1. This result can be obtained by simply executing the function `sbf()`. The details of its arguments will be presented later.

```R
> netlab <- network.design(latlon=latlon,
+ method="ModifyGottlemann", type="regular", x=5)
> eta <- eta.comp(netlab)$eta
> out.pls <- sbf(obs=temp67, latlon=latlon,
+ netlab=netlab, eta=eta, method="pls",
+ grid.size=c(100, 200), lambda=0.8)
> sw.plot(out.pls, type="field", xlab="Longitude",
+ ylab="Latitude")
```

Figure 2: An extrapolation for the observations in Figure 1.

Note that the representation in Figure 2 has inherent multiscale characteristics, which originate from the observations in Figure 1. For example, observe the global cold patterns near the north pole with local anomalies of the extreme cold in the central Canadian shield. Thus, classical methods such as spherical harmonics or smoothing splines are not very efficient in representing temperature data since they do not capture local properties. It is important to detect and explain local activities and variabilities as well as global trends. The second objective of using SWs is to decompose the signal properly according to spatial scales so as to capture the various activities...
of fields. Finally, SWs can be employed in developing a procedure to denoise the observations that are corrupted by noise. This article illustrates these procedures through an analysis of temperature data. In summary, the aim of this article is to explain how the SpherWave package is used in the following:

1) estimating the temperature field $T(x)$ for an arbitrary location $x$ on the globe, given the scattered observations $y_i, i = 1, \ldots, n$, from the model

$$y_i = T(x_i) + \epsilon_i, \quad i = 1, 2, \ldots, n,$$

(1)

where $x_i$ denote the locations of observations on the globe and $\epsilon_i$ are the measurement errors;

2) decomposing the signal by the multiscale analysis; and

3) obtaining a SW estimator using a thresholding approach.

As will be described in detail later, the multiscale analysis and SW estimators of the temperature field can be derived from the procedure termed multiscale spherical basis function (SBF) representation.

**Theory**

In this section, we summarize the SWs proposed by [Li (1999)] and its statistical applications proposed by [Oh and Li (2004)] for an understanding of the methodology and promoting the usage of the SpherWave package. [Narcowich and Ward (1996)] proposed a method to construct SWs for scattered data on a sphere. They proposed an SBF representation, which is a linear combination of localized SBFs centered at the locations of the observations. However, the Narcowich-Ward method suffers from a serious problem: the SWs have a constant spatial scale regardless of the intended multiscale decomposition. [Li (1999)] introduced a new multiscale SW method that overcomes the single-scale problem of the Narcowich-Ward method and truly represents spherical fields with multiscale structure.

When a network of $n$ observation stations $N_1 := \{x_i\}_{i=1}^n$ is given, we can construct nested networks $N_1 \supset N_2 \supset \cdots \supset N_L$ for some $L$. We re-index the subscript of the location $x_i$ so that $x_i$ belongs to $N_l \setminus N_{l-1} = \{x_i\}_{i=1}^{M_l} (l = 1, \ldots, L; N_{L+1} := \emptyset)$, and use the convention that the scale moves from the finest to the smoothest as the resolution level index $l$ increases. The general principle of the multiscale SBF representation proposed by [Li (1999)] is to employ linear combinations of SBFs with various scale parameters to approximate the underlying field $T(x)$ of the model in equation (1). That is, for some $L$

$$T_l(x) = \sum_{i=1}^{L} \sum_{l=1}^{M_l} \beta_{li} \phi_{\eta_i}(\theta(x, x_i)),$$

(2)

where $\phi_{\eta_i}$ denotes SBFs with a scale parameter $\eta_i$ and $\theta(x, x_i)$ is the cosine of the angle between two location $x$ and $x_i$ represented by the spherical coordinate system. Thus geodetic distance is used for spherical wavelets, which is desirable for the data on the globe. An SBF $\phi(\theta(x, x_i))$ for a given spherical location $x_i$ is a spherical function of $x$ that peaks at $x = x_i$ and decays in magnitude as $x$ moves away from $x_i$. A typical example is the Poisson kernel used by [Narcowich and Ward (1996) and Li (1999)].

Now, let us describe a multiresolution analysis procedure that decomposes the SBF representation (2) into global and local components. As will be seen later, the networks $N_l$ can be arranged in such a manner that the sparseness of stations in $N_l$ increases as the index $l$ increases, and the bandwidth of $\phi$ can also be chosen to increase with $l$ to compensate for the sparseness of stations in $N_l$. By this construction, the index $l$ becomes a true scale parameter. Suppose $T_l, l = 1, \ldots, L$, belongs to the linear subspace of all SBFs that have scales greater than or equal to $l$. Then $T_l$ can be decomposed as

$$T_l(x) = T_{l+1}(x) + D_l(x),$$

where $T_{l+1}$ is the projection of $T_l$ onto the linear subspace of SBFs on the networks $N_{l+1}$, and $D_l$ is the orthogonal complement of $T_l$. Note that the field $D_l$ can be interpreted as the field containing the local information. This local information cannot be explained by the field $T_{l+1}$ which only contains the global trend extrapolated from the coarser network $N_{l+1}$. Therefore, $T_{l+1}$ is called the global component of scale $l+1$ and $D_l$ is called the local component of scale $l$. Thus, the field $T_l$ in its SBF representation (equation (2)) can be successively decomposed as

$$T_l(x) = T_L(x) + \sum_{l=1}^{L-1} D_l(x).$$

(3)

In general wavelet terminology, the coefficients of $T_L$ and $D_l$ of the SW representation in equation (3) can be considered as the smooth coefficients and detailed coefficients of scale $l$, respectively.

The extrapolated field may not be a stable estimator of the underlying field $T$ because of the noise in the data. To overcome this problem, [Oh and Li (2004)] propose the use of thresholding approach pioneered by [Donoho and Johnstone (1994)]. Typical thresholding types are hard and soft thresholding. By hard thresholding, small SW coefficients, considered as originating from the zero-mean noise, are set to zero while the other coefficients, considered as originating from the signal, are left unchanged. In soft thresholding, not only are the small coefficients set to zero but the large coefficients are also shrunk toward zero, based on the assumption that they are corrupted by additive noise. A reconstruction from these coefficients yields the SW estimators.
Network design and bandwidth selection

As mentioned previously, a judiciously designed network $N_l$ and properly chosen bandwidths for the SBFs are required for a stable multiscale SBF representation.

In the SpherWave package, we design a network for the observations in Figure 1 as

```r
> netlab <- network.design(latlon=latlon, + method="ModifyGottlemann", type="regular", x=5)
> sv.plot(z=netlab, latlon=latlon, type="network", + xlab="", ylab="", cex=0.6)
```

We then obtain the network in Figure 3 which consists of 6 subnetworks.

```r
> table(netlab)
netlab
   1  2  3  4  5  6
686 104 72 44 25  8
```

Note that the number of stations at each level decreases as the resolution level increases. The most detailed subnetwork 1 consists of 686 stations while the coarsest subnetwork 6 consists of 8 stations.

![Figure 3: Network Design](image)

The network design in the SpherWave package depends only on the location of the data and the template grid, which is predetermined without considering geophysical information. To be specific, given a template grid and a radius for the spherical cap, we can design a network satisfying two conditions for stations: 1) choose the stations closest to the template grid so that the stations could be distributed as uniformly as possible over the sphere, and 2) select stations between consecutive resolution levels so that the resulting stations between two levels are not too close for the minimum radius of the spherical cap. This scheme ensures that the density of $N_l$ decreases as the resolution level index $l$ increases. The function `network.design()` is performed by the following parameters: `latlon` denotes the matrix of grid points (latitude, longitude) of the observation locations. The SpherWave package uses the following convention. Latitude is the angular distance in degrees of a point north or south of the equator and North and South are represented by "+" and "−" signs, respectively. Longitude is the angular distance in degrees of a point east or west of the prime (Greenwich) meridian, and East and West are represented by "+" and "−" signs, respectively.

The function `network.design()` uses the following arguments.

- `obs`: the vector of observations
- `latlon`: the matrix of the grid points of observation sites in degree
- `netlab`: the index vector of the subnetwork level
- `eta`: the vector of spatial parameters according to the resolution level

Multi-scale SBF representation

Once the network and bandwidths are decided, the multi-scale SBF representation of equation (2) can be implemented by the function `sbf()`. This function is controlled by the following arguments.

- `obs`: the vector of observations
- `latlon`: the matrix of the grid points of observation sites in degree
- `netlab`: the index vector of the subnetwork level
- `eta`: the vector of spatial parameters according to the resolution level
method: the method for the calculation of coefficients of equation (2), "ls" or "pls"

approx: approx = TRUE will use the approximation matrix

grid.size: the size of the grid (latitude, longitude) of the extrapolation site

lambda: smoothing parameter for method = "pls".

Method has two options – "ls" and "pls". method = "ls" calculates the coefficients by the least squares method, and method = "pls" uses the penalized least squares method. Thus, the smoothing parameter lambda is required only when using method = "pls". approx = TRUE implies that we obtain the coefficients using m (< n) selected sites from among the n observation sites, while the interpolation method (approx = FALSE) uses all the observation sites. The function sbf() returns an object of class "sbf". See Oh and Kim (2006) for details. The following code performs the approximate multiscale SBF representation by the least squares method, and Figure 4 illustrates results.

```r
> out.ls <- sbf(obs=temp67, latlon=latlon, + netlab=netlab, eta=eta, + method="ls", approx=TRUE, grid.size=c(100, 200))
> sw.plot(out.ls, type="field", + xlab="Longitude", ylab="Latitude")
```

Figure 4: An approximate multiscale SBF representation for the observations in Figure 1.

As can be observed, the result in Figure 4 is different from that in Figure 2, which is performed by the penalized least squares interpolation method. Note that the value of the smoothing parameter lambda used in Figure 4 is chosen by generalized cross-validation. For the implementation, run the following procedure.

```r
> lam <- seq(0.1, 0.9, length=9)
> gcv <- NULL
> for(i in 1:length(lam))
+ gcv <- c(gcv, gcv.lambda(obs=temp67, + latlon=latlon, netlab=netlab, eta=eta, + lambda=lam[i])$gcv)
> lam[gcv == min(gcv)]
[1] 0.8
```

Multiresolution analysis

Here, we explain how to decompose the multiscale SBF representation into the global field of scale $l+1$, $T_{l+1}(x)$, and the local field of scale $l$, $D_l(x)$. Use the function swd() for this operation.

```r
> out.pls <- swd(out.pls)
```

The function swd() takes an object of class "sbf", performs decomposition with multiscale SWs, and returns an object of class "swd" (spherical wavelet decomposition). Refer to Oh and Kim (2006) for the detailed list of an object of class "swd". Among the components in the list are the smooth coefficients and detailed coefficients of the SW representation. The computed coefficients and decomposition results can be displayed by the function sw.plot() as shown in Figure 5 and Figure 6.

```r
> sw.plot(out.pls, type="swcoeff", pch=19, + cex=1.1)
> sw.plot(out.pls, type="decom")
```

Figure 5: Plot of SW smooth coefficients and detailed coefficients at different levels $l = 1, 2, 3, 4, 5$.

Spherical wavelet estimators

We now discuss the statistical techniques of smoothing based on SWs. The theoretical background is based on the works of Donoho and Johnstone (1994) and Oh and Li (2004). The thresholding function swthresh() for SW estimators is

```r
> swthresh(swd, policy, by.level, type, nthresh, + value=0.1, Q=0.05)
```

This function swthresh() thresholds or shrinks detailed coefficients stored in an swd object, and returns the thresholded detailed coefficients in a modified swd object. The thresh.info list of an swd object has the thresholding information. The available policies are "universal", "sure", "fdr", "probability", and "Lorentz". For the first three thresholding policies, see Donoho and Johnstone (1994, 1995) and Abramovich and Benjamini (1996).
Figure 6: Multiresolution analysis of the multiscale SBF representation $T_1(x)$ in Figure 2. Note that the field $T_1(x)$ is decomposed as $T_1(x) = T_6(x) + D_1(x) + D_2(x) + D_3(x) + D_4(x) + D_5(x)$.

Figure 7: Thresholding result obtained by using the FDR policy

$q$ specifies the false discovery rate (FDR) of the FDR policy. `policy = "probability"` performs thresholding using the user supplied threshold represented by a quantile value. In this case, the quantile value is supplied by `value`. The Lorentz policy takes the thresholding parameter $\lambda$ as the mean sum of squares of the detailed coefficients.
by level controls the methods estimating noise variance. In practice, we assume that the noise variances are globally the same or level-dependent. by.level = TRUE estimates the noise variance at each level $l$. Only for the universal, Lorentz, and FDR policies, a level-dependent thresholding is provided. The two approaches, hard and soft thresholding can be specified by type. In addition, the Lorentz type $q(t, \lambda) := \text{sign}(t) \sqrt{t^2 - \lambda^2} I(|t| > \lambda)$ is supplied. Note that only soft type thresholding is appropriate for the SURE policy. By providing the number of resolution levels to be thresholded by nthresh, we can also specify the truncation parameter.

The following procedures perform thresholding using the FDR policy and the reconstruction. Comparing Figure 6 with Figure 7, we can observe that the local components of resolution level 1, 2, and 3 of Figure 7 are shrunk so that its reconstruction (Figure 8) illustrates a smoothed temperature field. For the reconstruction, the function swr() is used on an object of class "swd".

```r
> ### Thresholding
> out.fdr <- swthresh(out.dpls, policy="fdr", + by.level=FALSE, type="soft", nthresh=3, Q=0.05)
> sw.plot(out.fdr, type = "decom")
> ### Reconstruction
> out.reconfdr <- swr(out.fdr)
> sw.plot(z=out.reconfdr, type="recon", + xlab="Longitude", ylab="Latitude")
```

![Figure 8: Reconstruction](image)

We repeatedly use sw.plot() for display. To summarize its usage, the function sw.plot() displays the observation, network design, SBF representation, SW coefficients, decomposition result or reconstruction result, as specified by type = "obs", "network", "field", "swcoeff", "decom" or "recon", respectively. Either argument sw or z specifies the object to be plotted. z is used for observations, subnetwork labels and reconstruction result and sw is used for an sbf or swd object.

**Conclusion remarks**

We introduce SpherWave, an R package implementing SWs. In this article, we analyze surface air temperature data using SpherWave and obtain meaningful and promising results; furthermore provide a step-by-step tutorial introduction for wide potential applicability of SWs. Our hope is that SpherWave makes SW methodology practical, and encourages interested readers to apply the SWs for real world applications.

**Acknowledgements**

This research was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (KRF-2005-070-C00021).

**Bibliography**


Hee-Seok Oh  
Seoul National University, Korea  
E-mail: heeseok@stats.snu.ac.kr

Donghoh Kim  
Sejong University, Korea  
E-mail: donghohkim@sejong.ac.kr