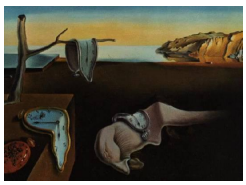

Journal of Statistical Software
August 2010, Volume 36, Issue 4. http://www.jstatsoft.org/

On Simulation of Manifold Indexed Fractional Gaussian Fields

Alexandre Brouste
Université du Maine

Jacques Istas
Université de Grenoble

Sophie Lambert-Lacroix
Université de Grenoble

Abstract

To simulate fractional Brownian motion indexed by a manifold poses serious numerical problems: storage, computing time and choice of an appropriate grid. We propose an effective and fast method, valid not only for fractional Brownian fields indexed by a manifold, but for any Gaussian fields indexed by a manifold. The performance of our method is illustrated with different manifolds (sphere, hyperboloid).

Keywords: manifold indexed fractional Brownian field, simulation.

1. Introduction

Rough phenomena arise in various fields (Frisch and Parisi 1985; Leland *et al.* 1994; Mandelbrot 1975; Peitgen and Saupe 1988; Pentland 1984): texture simulations and image processing, natural scenes (clouds, mountains) simulations, fluid mechanics, financial mathematics, ethernet traffic... Some phenomena, like ethernet traffic or financial data, are time-indexed. Other phenomena, like image processing, should be indexed by subsets of the Euclidean spaces $\mathbb{R}^2$ or $\mathbb{R}^3$. But, some other rough phenomena are not indexed by an Euclidean space, but by a manifold. Let us mention for instance the cosmic microwave background (Marinucci *et al.* 2007; National Aeronautics and Space Administration 2010) or solar data (Koenig and Chainais 2008), that lead to spherical data. Real-world textures are usually not supported by an Euclidean space, but by a surface. Applying an Euclidean indexed texture on, say, a sphere, always generates artefacts. Generating manifold indexed textures is therefore a challenge.

Fractional Gaussian fields, like fractional Brownian fields, are good candidates for modeling rough phenomena. Fractional Gaussian fields may be indexed by an Euclidean space or a manifold. The simulation of Euclidean indexed Gaussian fields poses a lot of problem, in particular storage and complexity. Recently, Brouste *et al.* (2007) propose a new and fast algorithm for simulating Euclidean indexed Gaussian fields. The aim of this paper is to extend

this algorithm to the manifold indexed case. Compared to the Euclidean case, two problems emerge. First, one has to choose a simulation grid. In the Euclidean case, the equidistributed grid is a standard choice. In the manifold case, there is no more an equidistributed grid. One therefore has to chose a grid and to adapt the algorithm to this non-equidistributed grid. Second, one has to chose a visualization grid for representing the manifold. That means that the algorithm must also adapt to this grid.

We propose to develop a function for which the user enters a grid of his/her choice and a distance to be used in order to determine the neighbors. Thus one can use our function for any manifold when having a covariance function, a grid and a distance. We give here two examples of manifolds: the sphere and the hyperboloid. For these two manifolds, there does not exist an equidistributed grid, but there exists a uniform probability measure. So we generate a set of random points following a uniform density. This set is separated into two subsets. On the first one, one generates a Gaussian random vector via an exact method. On the second subset, one only keeps the neighbors to simulate a random Gaussian vector. At the end, the points of the visualization grid are considered as neighbors of previous points and a last simulation is done.

The paper proceeds as follows. In Section 2, we present our function `fieldsim`. It is implemented in the R system for statistical computing (R Development Core Team 2010) and available from the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=FieldSim`. In Section 3, we present our method on spheres and hyperboloid. Technical tools are postponed to the appendix at the end of the paper.

2. Method

After introducing some notations, we recall the both (accurate and refined) steps of the function `fieldsim` proposed in Brouste *et al.* (2007). Then we present the extension of this method to the processes indexed by a manifold.

2.1. Notations and definitions

Let $(\mathcal{M}, g)$ be a C^∞-complete Riemannian manifold of dimension 2. Let $d_{\mathcal{M}}$ be its geodesic distance. Let $X(\cdot) = \{X(M), M \in \mathcal{M}\}$, be a real valued centered Gaussian field indexed by the manifold $\mathcal{M}$. When $(\mathcal{M}, g)$ is the usual Euclidean space, one speaks of X as an Euclidean random field.

In this paper we are only concerned with the second order properties of the field $X(\cdot)$. It is convenient to use a geometrical approach by considering the following Hilbert space $\mathcal{A}$, with the inner product $\langle U, V\rangle = E\{UV\} = Cov\{U, V\}$. The elements of $\mathcal{A}$ are the linear combinations, with real coefficients, of elements of $\{X(M), M \in \mathcal{M}\}$ and their limits for mean square convergence. So the covariance function $R(\cdot,\cdot)$ is defined by:

$$R(M, M') = \langle X(M), X(M')\rangle = Cov\{X(M), X(M')\} \quad M,\ M' \in \mathcal{M}.$$

This function is nonnegative definite (n.n.d.), that is for all $n \geq 1$, for all real scalars $\lambda_1, \ldots, \lambda_n$, and for all $M^1, \ldots, M^n \in \mathcal{M}$,

$$\sum_{i,j=1}^{n} \lambda_i \lambda_j R(M^i, M^j) \geq 0.$$

Conversely, for any n.n.d. function $R(\cdot,\cdot)$, there exists an unique centered Gaussian field of second order structure given by $R(\cdot,\cdot)$.

2.2. The function `fieldsim`

We give here a summary of the function `fieldsim` that allows to simulate an Euclidean random field. In this case $\mathcal{M}$ is a subspace like $[0,1]^2$ for instance with the usual Euclidean norm. This function yields discretization of sample path of the Gaussian field over a space discretization $\{\mathcal{S}_e, \mathcal{S}_r\}$ of $\mathcal{M}$, associated with any n.n.d. function $R(\cdot,\cdot)$. It is consisted of the two following steps.

Accurate simulation step. Given a space discretization $\mathcal{S}_e$, a sample of a centered Gaussian vector: $(X(M))_{M\in\mathcal{S}_e}$ with covariance matrix $\mathbf{R}$ given by $\mathbf{R}_{i,j} = R(M, M')$, $M, M' \in \mathcal{S}_e$, is simulated. This simulation is obtained by an algorithm based on Cholesky decomposition of the matrix $\mathbf{R}$.

Refined simulation step. Let $\mathcal{S}_r$ be the remaining space discretization. For each new point $M \in \mathcal{S}_r$ at which we want to simulate the field, $X(M)$ is generated by using only a set of neighbors instead of all the simulated components (as in the accurate simulation step). Precisely, let N_M be a neighbors set of M (for the Euclidean distance in $\mathbb{R}^2$) and $\mathcal{X}_{N_M}$ be the space generated by the variables $X(M')$, $M' \in N_M$. Let us remark that the neighbors set is defined with all the already simulated variables (in the accurate and refined simulation step). Let $X_{\mathcal{X}_{N_M}}(M)$ be the best linear combination of variables of $\mathcal{X}_{N_M}$ approximating $X(M)$ in the sense that the variance of the innovation $\varepsilon_{\mathcal{X}_{N_M}}(M) = X(M) - X_{\mathcal{X}_{N_M}}(M)$ is minimal. The new variable $X(M)$ is obtained by

$$X_{\mathcal{X}_{N_M}}(M) + \sqrt{Var(\varepsilon_{\mathcal{X}_{N_M}}(M))}U,$$

where U is a centered and reduced Gaussian variable independent of the already simulated components.

Note that the variable $X_{\mathcal{X}_{N_M}}(M)$ and the variance $Var(\varepsilon_{\mathcal{X}_{N_M}}(M))$ are completely determined by the covariance structure of the sequence $X(M)$, $X(M')$, $M' \in N_M$. For storage and computing time, the accurate simulation step must concern only a small variables number whereas the second step can relate a larger variables number. That leads to an effective and fast method to simulate any Gaussian field.

In Brouste *et al.* (2007), the procedure is implemented in the R package **FieldSim**. A natural discretization space is of the form $(k2^{-J}, l2^{-J})$, $k, l = 0, \ldots, 2^J$ for J a positive integer. The accurate step is applied for points $(k2^{-J_a}, l2^{-J_a})$, where J_a is a level to be chosen (in general $J_a = 1$ or 2). The refined step is applied for the remainder.

2.3. The function `fieldsim` adapted to manifolds

The previous procedure can be adapted to the case of fields indexed by a manifold. The main problem stands in the discretization grid choice. There is, in the case of the sphere for instance, no equidistributed grid and it is difficult to define a concept of finer grid such as in the case of field indexed by $[0,1]^2$. Moreover, this choice can be related to the software that one wishes to use to represent the manifold.

We propose to develop a function for which the user enters a grid of his/her choice for each step (accurate and refined) and a distance to be used in order to determine the neighbors.

Thus one can use our function for any manifold when having a covariance function, a grid and a distance. We give here two examples of manifolds: the sphere and the hyperboloid.

Let us precise how we use it for this two manifolds. We denote by $\mathcal{S}_g$ the set of the point of $\mathcal{M}$ at which we want to generate the process (the visualization grid). This set choice can be induced for instance by the software that one wishes to use to represent the manifold (see Appendix C). Let us recall that we want to generate a path of field indexed by the manifold $\mathcal{M}$ (discretized at $\mathcal{S}_g$) and with covariance function $R(\cdot,\cdot)$. We need also to specify the concept of neighbors. A natural choice is to use the geodesic distance between two points. So the closest neighbors of some point M are the points closest to M according to this distance. In general, the cardinal of $\mathcal{S}_g$ is too large to use only the accurate simulation step. Moreover contrary to the grid chosen in the case of field indexed by $[0,1]^2$, one is not able in general to use any more a concept of finer grid. Indeed, for our sphere visualization grid for instance, using overlapping sub-grids (an atlas of 6 maps), each sub-grid have some parts of the sphere with very few points and there are some others with points accumulation (in particular on the six poles). That is why in this case, we propose to first simulate the process at uniform random points and next to simulate the process at the $\mathcal{S}_g$ points. The generation of uniform random points on the manifold as sphere or hyperboloid are given in Appendix B. Precisely, let $\mathcal{S}_u$ the uniform random points set. We propose to run the function `fieldsim` with the set $\{\mathcal{S}_u, \mathcal{S}_g\}$. This set $\{\mathcal{S}_u, \mathcal{S}_g\}$ is cut out into the set $\{\mathcal{S}_e, \mathcal{S}_r\}$.

2.4. How to chose the parameters?

This function requires in addition to $R(\cdot,\cdot)$, the integers N_e, the points number for the accurate step, and `nbNeighbor`, the neighbors number. Choosing these parameters is a delicate task. This problem can be related to the question of evaluating the simulation accuracy. One needs to define an appropriate accuracy criterion. One would wish to evaluate the "fractionality" of the simulated field. That is what was proposed in Brouste *et al.* (2007) by estimating the fractional index. This is only partially satisfactory. A second, and more classical, way is to estimate the mean square error between the simulated sample path and the true one. A third one consists in evaluating a distance between the covariance of the simulated field and the true covariance. Second, one needs the effective computation of the error. We have investigated the mean square error in a theoretical way. Apart obvious results (i.e. the number of neighbors used in the refined grid tends to infinity), the computations are so intricate that we did not succeed. The same occurs for studying theoretically a distance between the covariance of the simulated field and the true covariance.

In this subsection, we propose two protocols to help the user in the error control and the parameters choice. To do that, we propose to estimate the distance between the covariance of the simulated field and the true covariance since in our context this function is known. Firstly, one can simulate several paths and estimate this error on the part of the process simulated in an exact way and that of the process simulated in an refined way. The first error only corresponds to the sampling error. The second one contains in more the approximation error due to the use of some neighbors instead of all the already simulated variables. Thus for a covariance function and a given grid, one compares these two kinds of error. In particular one uses this protocol to chose the neighbors number that results from a compromise between computational times and error loss control. Table 1 gives the results for different values of number of neighbors obtained for $n_s = 10,000$ sample paths simulated over the same

Number of neighbors	Accurate error	Refined error	Error ratio	CPU time
4	0.00013	0.00828	62.342	0.264
6	0.00027	0.00192	6.975	0.321
8	0.00010	0.00110	10.757	0.381
12	0.00023	0.00074	3.226	0.524
15	0.00028	0.00050	1.827	0.678
20	0.00036	0.00042	1.153	1.039
40	0.00064	0.00143	2.250	3.217
50	0.00016	0.00022	1.392	4.889

Table 1: Simulation results for $H = 0.25$, $n_s = 10,000$, $N_e = 50$, $N_r = 50$.

Cardinal of $\mathcal{S}_u$	Visualization error	CPU time
36	0.00070	2.350
81	0.00077	3.047
169	0.00115	4.496
361	0.00117	8.114

Table 2: Simulation results for $H = 0.25$, $n_s = 10,000$, $N_e = 50$, `nbNeighbor` $= 15$.

100 uniform random points. The points number for the accurate step is $N_e = 50$, so the points number for the refined step is $N_r = 50$. The covariance function is given by $R_{\mathcal{S}_1}(\cdot,\cdot)$ defined at the Section 3.1 (with $H = 0.25$). The distance between the covariance of the simulated field and the true covariance is estimated by the mean ℓ_2-distance between the estimated covariance function coefficients and the true ones. Firstly we remark that the ratio of the errors (accurate and refined) goes to 1 as the neighbors number increases. For this example, one also observes that there is an important gain while passing from 8 to 12 neighbors. But it seems unnecessary to increase this number of neighbors to some value greater than 15.

Secondly, concerning the choice of the size of the random grid with respect to the size of S_g, one uses the following protocol. For several grid size values (here N_e fixed to 50), one estimates the distance between the covariance of the simulated field and the true covariance (over several paths). Next one retains the value of the grid size for which this error is smallest. Table 2 gives the results for different sizes of random grid $\mathcal{S}_u$ obtained for $n_s = 10,000$ sample paths simulated. The size of the visualization grid is equal to 192. For this example, we observe that we do not need to increase the size or the random grid beyond 81.

3. Numerical results

In this section, we illustrate the method proposed here through simulations for two manifolds: sphere and hyperboloid.

3.1. Some examples of sphere indexed fractional fields

Let $\mathcal{S}$ be the unit sphere of $\mathbb{R}^3$, and let $d_{\mathcal{S}}$ be its geodesic distance. We give hereafter several examples of fields indexed by $\mathcal{S}$.

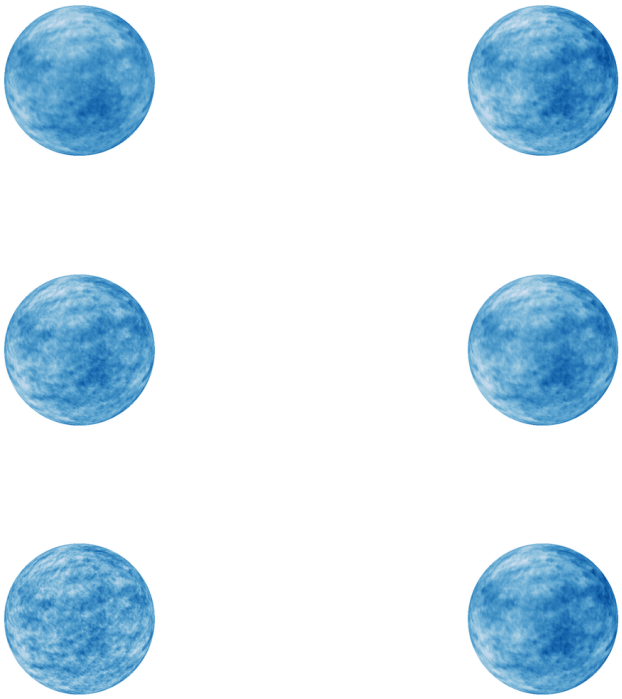

Figure 1: On the left, sphere indexed fractional fields with covariance function R_{S1} and Hölder index $H = 0.45$ (top), $H = 0.3$ (middle) and $H = 0.15$ (bottom). On the right, sphere indexed fractional fields of Hölder index $H = 0.45$ with covariance function R_{S2} (top), R_{S3} (middle) and R_{S4} (bottom). All simulations are done with $N_e = 100$, $N_r = 900$, $N_g = 100$ and `nbNeighbor` $= 15$.

The spherical fractional Brownian fields were introduced by Istas (2005) in the following way. There exists a centered Gaussian field called spherical fractional Brownian field (sfBf) whose covariance function is given by

$$R_{\mathcal{S}_1}(M, M') = \frac{1}{2}\left\{d_{\mathcal{S}}^{2H}(O, M) + d_{\mathcal{S}}^{2H}(O, M') - d_{\mathcal{S}}^{2H}(M, M')\right\},$$

where O is any given point of $\mathcal{S}$, if and only if $H \in (0, 1/2]$.

We can show (see Istas (2009)) that the following functions

$$\begin{aligned} R_{\mathcal{S}_2}(M, M') &= \exp\left(-d_{\mathcal{S}}^{2H}(M, M')\right), \\ R_{\mathcal{S}_3}(M, M') &= \ln\left(1 + d_{\mathcal{S}}^{2H}(O, M)\right) + \ln\left(1 + d_{\mathcal{S}}^{2H}(O, M')\right) - \ln\left(1 + d_{\mathcal{S}}^{2H}(M, M')\right), \\ R_{\mathcal{S}_4}(M, M') &= \frac{1}{1 + d_{\mathcal{S}}^{2H}(M, M')}, \end{aligned}$$

are covariance functions for $H \in (0, 1/2]$.

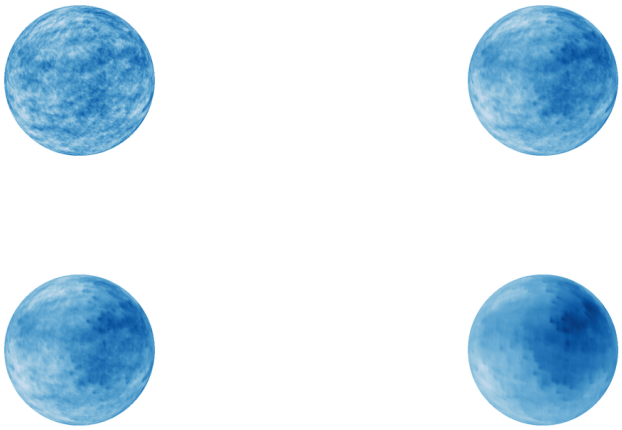

Figure 2: Sphere indexed fractional fields with covariance function R_{S5} and Hölder index $H = 0.2$ (top left), $H = 0.45$ (top right), $H = 0.5$ (bottom left) and $H = 0.8$ (bottom right). All simulations are done with $N_e = 100$, $N_r = 900$, $N_g = 100$ and `nbNeighbor` $= 15$.

Finally we can restrict the fractional Brownian fields indexed by $\mathbb{R}^3$ to the sphere. We obtain the following covariance function:

$$R_{\mathcal{S}_5}(M, M') = 1 - 2^{2H-1} \left(\sin \left(\frac{d_{\mathcal{S}}(M, M')}{2} \right) \right)^{2H},$$

where $H \in (0, 1)$.

Figures 1 and 2 gives some representations of such fields. For each sphere, we have chosen $N_e = 100$, $N_r = 900$ and `nbNeighbor` $= 15$. The discretized grid $\mathcal{S}_g$ is given as in Appendix C with $N_g = 100$.

3.2. Some examples of hyperboloid indexed fractional fields

Let $\mathcal{H}$ be the hyperboloid: $\mathcal{H} = \{x^2 + y^2 - z^2 = -1, \ z \geq 1\}$ with its geodesic distance $d_{\mathcal{H}}$. Since $\mathcal{H}$ is unbounded, we will simulate the field on a hyperbolic cap. We give hereafter several examples of fields indexed by $\mathcal{H}$.

The hyperbolic fractional Brownian fields (hfBf) introduced by Istas (2005) have covariance function given by

$$R_{\mathcal{H}_1}(M, M') = \frac{1}{2} \left\{ d_{\mathcal{H}}^{2H}(O, M) + d_{\mathcal{H}}^{2H}(O, M') - d_{\mathcal{H}}^{2H}(M, M') \right\},$$

where O is any given point of $\mathcal{H}$ and $H \in (0, 1/2]$.

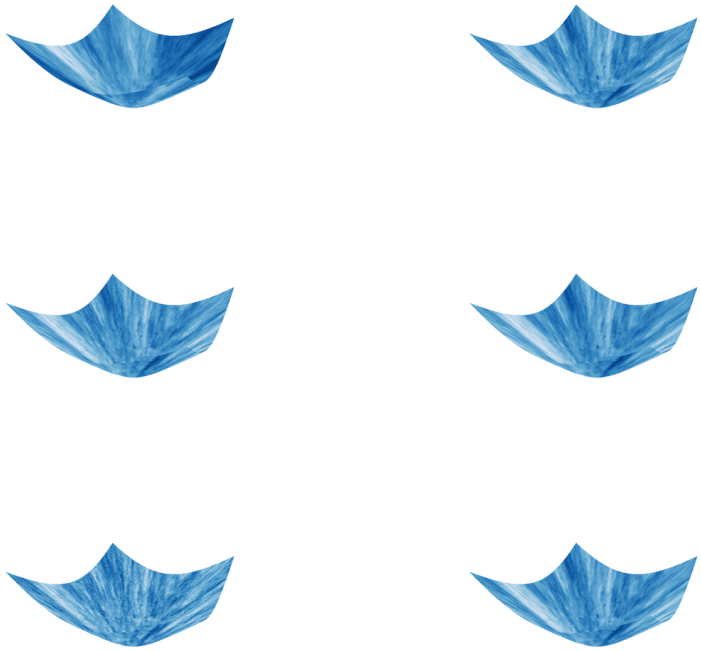

Figure 3: On the left, hyperboloid indexed fractional fields with covariance function R_{H1} and index $H = 0.45$ (top), $H = 0.3$ (middle) and $H = 0.15$ (bottom). On the right, hyperboloid indexed fractional fields of index $H = 0.45$ with covariance function R_{H2} (top), R_{H3} (middle) and R_{H4} (bottom). All simulations are done with $N_e = 100$, $N_r = 1000$, $N_g = 150$ and `nbNeighbor` $= 15$.

We can show (see Istas (2009)) that the following functions

$$\begin{aligned} R_{\mathcal{H}_2}(M, M') &= \exp\left(-d^{2H}_{\mathcal{H}}(M, M')\right), \\ R_{\mathcal{H}_3}(M, M') &= \ln\left(1 + d^{2H}_{\mathcal{H}}(O, M)\right) + \ln\left(1 + d^{2H}_{\mathcal{H}}(O, M')\right) - \ln\left(1 + d^{2H}_{\mathcal{H}}(M, M')\right), \\ R_{\mathcal{H}_4}(M, M') &= \frac{1}{1 + d^{2H}_{\mathcal{H}}(M, M')}, \end{aligned}$$

are covariance functions for $H \in (0, 1/2]$. Figure 3 gives some representations of such fields. For each example, we have chosen $N_e = 100$, $N_r = 900$ and `nbNeighbor` $= 15$. The discretized grid $\mathcal{S}_g$ is given as in Appendix C with $N_g = 150$.

Acknowledgments

The authors would like to thank the both referees for their relevant comments. Part of this work was supported by the Interuniversity Attraction Pole (IAP) research network in Statistics P5/24 and ANR-GDSA project.

References

Adler D, Murdoch D (2010). ***rgl**: 3D Visualization Device System (OpenGL).* R package version 0.91, URL http://CRAN.R-project.org/package=rgl.

Brouste A, Istas J, Lambert-Lacroix S (2007). "On Fractional Gaussian Random Fields Simulations." *Journal of Statistical Software*, **23**(1), 1–23. URL http://www.jstatsoft.org/v23/i01/.

Frisch U, Parisi G (1985). "Turbulence and Predictability in Geophysical Fluid Dynamics and Climate Dynamics." In M Ghil, R Benzi, G Parisi (eds.), *Proceedings of the International School of Physics E. Fermi*, pp. 84–88. North-Holland.

Istas J (2005). "Spherical and Hyperbolic Fractional Brownian Motion." *Electronic Communications of Probability*, **10**, 254–262.

Istas J (2009). "Manifold Indexed Fractional Fields." Submitted.

Koenig E, Chainais P (2008). "Multifractal Analysis on the Sphere." In *Proceedings of the 3rd International Conference on Image and Signal Processing*, volume 5099 of *Lecture Notes In Computer Science*, pp. 613–621. Springer-Verlag, Berlin.

Leland W, Taqqu M, Willinger W, Wilson V (1994). "On the Self-Similar Nature of Ethernet Traffic (Extended Version)." *IEEE/ACM Transactions on Networking*, **2**(1), 1–15.

Mandelbrot B (1975). "Stochastic Models for the Earth's Relief, the Shape and the Fractal Dimension of the Coastlines, and the Number-Area Rule for Islands." In *Proceedings of the National Academy of Sciences*, 10, pp. 3825–3828.

Marinucci D, Pietrobon D, Balbi A, Baldi P, Cabella P, Kerkyacharian G, Natoli P, Picard D, Vottorio N (2007). "Spherical Needlets for Cosmic Microwave Background Data Analysis." *Monthly Notices of the Royal Astronomical Society*, **383**, 539–545.

National Aeronautics and Space Administration (2010). "LAMBDA: Legacy Archive for Microwave Background Data Analysis." URL http://lambda.gsfc.nasa.gov/product/map/dr4/m_products.cfm.

Neuwirth E (2007). ***RColorBrewer**: **ColorBrewer** Palettes.* R package version 1.0-2, URL http://CRAN.R-project.org/package=RColorBrewer.

Peitgen H, Saupe D (1988). *The Science of Fractal Images.* Springer-Verlag, New York.

Pentland AP (1984). "Fractal-Based Description of Natural Scenes." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6**, 661–674.

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

A. Using FieldSim

The new version of **FieldSim** is a set of R functions that allows performing simulations of manifold indexed fractional Gaussian fields with known covariance function. Three classes of functions are implemented:

- Three building functions: `setManifold` permits the user to create an object of class `manifold`; `constructcovf` that produces covariance functions; `constructgrid` that produces several grids for different manifolds.
- Simulation function `fieldsim` that performs simulations of the path of the manifold indexed Gaussian fields.
- Print function `visualize` that gives graphical representation of path of some manifold indexed Gaussian fields.

The R environment is the only user interface returned to R. In order to make it easier for the reader not familiar to R language, we detail the call to functions and the command used to produce graphical outputs.

A.1. Setting manifold

A manifold is an object of the S4 class `manifold` with four slots which are:

- `name` which is the name of the manifold we consider (`character`);
- `atlas` which the union of discretized domains that covers the manifold (must be a matrix where the number of line is the dimension of the space where the manifold lives);
- `distance` which is the distance considered on the manifold (`function`);
- `origin` which is the origin considered on the manifold (must be a point on the manifold).

The setter `setManifold` permits the user to create an object of class `manifold` with all its slots. We give below the example of the field indexed by some interval.

```
R> library("FieldSim")
R> name1 <- "plane1"
R> mesh <- seq(from = 0, to = 1, length = 16)
R> atlas1 <- rbind(rep(mesh, each = 16), rep(mesh, 16))
R> d1 <- function(xi, xj) {
+    return(sqrt(t(xi - xj) %*% (xi - xj)))
+ }
R> origin1 <- rbind(0, 0)
R> manifold1 <- setManifold(name = name1, atlas = atlas1, distance = d1,
+    origin = origin1)
R> str(manifold1)
```

```
Formal class 'manifold' [package "FieldSim"] with 4 slots
  ..@ name    : chr "plane1"
  ..@ atlas   : num [1:2, 1:256] 0 0 0 0.0667 0 ...
  ..@ distance:function (xi, xj)
  ..@ origin  : num [1:2, 1] 0 0
```

All slots have to be set so as to create the object. If the slot is of wrong type, a error message appears. In order to access, for instance, to the slot `name` of the manifold `manifold`, use `manifold@name`. Some of standard manifolds have already been implemented such as `plane`, `sphere` and `hyperboloid`.

A.2. Building grids

Some grids can be constructed with the function `constructgrid(manifold, typegrid, Ng)`. It has, three argument, the manifold on which the grid is settled, the type of grid the user want to construct and the number of points of this grid. There are four types of grid: `regular`, `finer`, `random` and `visualization`. Some informations on the `random` grid and on the `visualization` grid for the sphere and the hyperboloid manifolds are postponed in Appendixs B and C.

A.3. Building covariance function

In order to simulate further Gaussian processes or fields, we have to define a covariance function. In fact the user can construct himself this covariance function (semi-definite positive on the manifold). In order to treat more easily some particular case, a function have been implemented that constructs for known manifolds some particular covariance functions. Let us examine the call of this function `constructcovf(manifold, "fBm", H = 0.6)`. The first argument is a manifold object, the second parameter is the type of covariance (in this case, fBm covariance function) and the third argument (depending of the type) is parameter of this covariance (for the fBm, it is the Hurst parameter H). For the moment, the user can choose between two types, namely: `"fBm"` (fractional Brownian field) and `"mBm"` (multifractional Brownian field).

A.4. Sphere indexed fractional fields

The function `fieldsim` takes as arguments a manifold, and a covariance function. It returns the values of the field on the atlas of the manifold computed by our method.

Defining the manifold

Let us defined the sphere manifold with

```
R> sphere <- setManifold("sphere")
R> str(sphere)

Formal class 'manifold' [package "FieldSim"] with 4 slots
  ..@ name    : chr "sphere"
  ..@ atlas   : num [1:3, 1:864] -0.75 -0.75 NaN -0.75 -0.614 ...
```

```
  ..@ distance:function (xi, xj)
  ..@ origin  : num [1:3, 1] 1 0 0
```

In this case, the slots `atlas`, `distance` and `origin` are fixed to default values, namely a visualization grid, the usual geodesic distance on the sphere

```
R> dS <- function(xi, xj){
+    u <- sum(xi * xj)
+    if (u < -1) u <- -1
+    if (u > 1) u <- 1
+    return(acos(u))
+  }
```

and an origin fixed to `rbind(1, 0, 0)`. As explained in Appendix C, it is natural that `NaN` values exist in the sphere default visualization atlas.

Defining the covariance function

In order to compute sample paths of the spherical fractional Brownian field, it is possible to use the function

```
R> H <- 0.4
R> R.S.1 <- constructcovf(sphere, "fBm", H = H)
```

to compute its covariance function. Other type of covariance functions have been defined in Section 3.1 which use the manifold distance. For instance, let us construct the second example

```
R> dS <- sphere@distance
R> R.S.2 <- function(xi, xj) exp(-dS(xi, xj)^(2 * H))
```

Defining the simulation grid

As mentioned in this paper, we construct a simulation grid composed of two subsets

```
R> S.u <- constructgrid(sphere, "random", 10)
R> S.g <- constructgrid(sphere, "visualization", 12)
R> simulationgrid <- cbind(S.u, S.g)
```

Simulating via the FieldSim method

The commands (the first one to integrate the simulation grid to the manifold object and the second to compute the sample path)

```
R> sphere@atlas <- simulationgrid
R> resS <- fieldsim(sphere, R.S.1, Ne = 100, nbNeighbor = 15)
```

simulate a spherical fBm on the mesh given by the concatenation of the random grid and the visualization grid. The quantity `Ne` is the number of points of the grid to be simulated in the accurate step and `nbNeighbor` the number of neighbors used in the refined step.

Visualizing the sphere indexed fractional field

Note that the function `visualize` needs the packages **rgl** (Adler and Murdoch 2010) and **RColorBrewer** (Neuwirth 2007) that load automatically with the **FieldSim** package. Before using the function `visualize`, the user has to defined the visualization atlas of the manifold object and the proper sample path associated with this atlas

```
R> sphere@atlas <- S.g
R> res <- resS[(dim(S.u)[2] + 1):length(resS)]
```

After that, the command

```
R> visualize(sphere, res)
```

allows us to visualize the sphere indexed fractional field.

A.5. Hyperboloid indexed fractional fields

To obtain hyperboloid indexed fractional fields, similar functions can be used by starting with

```
R> hyper <- setManifold("hyperboloid")
```

B. Uniform random generator on manifold

B.1. Spherical uniform random generator

We parameterize the unit sphere by the polar coordinate system

$$x = \cos\theta \sin\phi, \quad y = \sin\theta \sin\phi, \quad z = \cos\phi,$$

where $\phi \in [0, \pi]$ and $\theta \in [0, 2\pi]$. To generate uniform random points on the unit sphere, we can use the following way. Firstly we generate θ with an uniform distribution on $[0, 2\pi]$. Secondly (and independently of the first sample), we generate ϕ as a random variable on $[0, \pi]$ with density $\sin(\phi)/2$.

B.2. Hyperboloid uniform random generator

We parameterize the hyperboloid by the hyperbolic coordinate system

$$x = \cos\theta \sinh\phi, \quad y = \sin\theta \sinh\phi, \quad z = \cosh\phi,$$

where $\phi \geq 0$ and $\theta \in [0, 2\pi]$. Let us remark that there does not exist uniform random generator on hyperboloid since this space is not bounded. So one considers the hyperbolic cap defined by $\{x^2 + y^2 - z^2 = -1,\ 1 \leq z \leq Z\}$, where Z is some real in $(1, \infty)$. To generate uniform random points on the hyperbolic cap, we can use the following way. Firstly we generate θ with an uniform distribution on $[0, 2\pi]$. Secondly (and independently of the first sample), we generate ϕ as a random variable on $[0, \text{arccosh}Z]$ with density $\sinh(\phi/(Z-1))$.

C. Visualization grids

C.1. Spherical visualization grid

The sphere-visualization grid on the sphere of size $6 \times N_g^2$, is given by the union of the 6 domains centered around one of the 6 triply orthogonal poles. Each domain are composed of the heights on the sphere (when they exists) corresponding to the regular mesh (of length N_g) $[-3/4, 3/4]^2$ of the others two cartesian coordinates.

C.2. Hyperboloid visualization grid

The hyperboloid-visualization grid on the hyperboloid of size N_g^2, is composed of the positive heights on the hyperboloid corresponding to the regular mesh (of length N_g) $[-3, 3]^2$ of the others two cartesian coordinates.

Affiliation:

Alexandre Brouste
Laboratoire de Statistique et Processus
Avenue Olivier Messiaen
72000 Le Mans cedex 9, France
E-mail: Alexandre.Brouste@univ-lemans.fr
URL: http://www.univ-lemans.fr/sciences/statist/pages_persos/Brouste/

Jacques Istas
LJK, BP 53
38041 Grenoble cedex 9, France
E-mail: Jacques.Istas@imag.fr
URL: http://ljk.imag.fr/membres/Jacques.Istas/

Sophie Lambert-Lacroix
LJK, BP 53
38041 Grenoble cedex 9, France
E-mail: Sophie.Lambert@imag.fr
URL: http://ljk.imag.fr/membres/Sophie.Lambert/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 36, Issue 4 *Submitted:* 2009-10-02
August 2010 *Accepted:* 2010-06-06