User Guide for Hermir-Fields version 0.1: Toolbox for Synthesis of Stationary Gaussian Fields

Hannes Helgason, Vladas Pipiras, and Patrice Abry*

April 2, 2012

Contents

*H. Helgason is affiliated with KTH, Stockholm (Sweden); V. Pipiras is with UNC, Chapel Hill (USA); P. Abry is with ENS de Lyon, Lyon (France)

Hermir-Fields is a collection of MATLAB routines which implements methods for the synthesis of stationary Gaussian fields. Hermir-Fields is freeware and the most current version is available for download at [1]. Feel welcome to send comments and questions to `hannes.helgason@ee.kth.se`.

1 Organization of document

This is document is organized into the following sections:

2. Introduction	– the synthesis problem and methodology used in the toolbox.
3. Getting started	– installation instructions
4. Brief introduction to code	– examples and notes on how to use the code

2 Introduction

2.1 Synthesis of stationary Gaussian fields in Hermir-Fields

The software's purpose is to provide routines for numerically synthesizing *stationary Gaussian fields*. A stationary Gaussian field $X = \{X_n, n \in \mathbb{Z}^2\}$ is a collection of multivariate Gaussian random variables for which

- *the mean of X is constant*: $EX_n = \mu \in \mathbb{R}$, for all $n = (n_1, n_2) \in \mathbb{Z}^2$
- *the covariance between X_n and $X_{n'}$ depends only on the difference $n' - n$, for all $n, n' \in \mathbb{Z}^2$*

In this document and in the code we assume that $\mu = 0$ which does not imply any restrictions – a nonconstant mean can always be added to the random variables afterwards. Assuming $EX = 0$, this means that

$$EX_n X_{n+h} = EX_0 X_h =: r(h) \tag{1}$$

for all $n = (n_1, n_2)$ and $h = (h_1, h_2)$ in $\mathbb{Z}^2$; here we have denoted the covariance of the field by $r(h)$. The current version of Hermir-Fields is used to generate stationary Gaussian fields X on *square* grids

$$G(N) = \{n \in \mathbb{Z}^d : 0 \le n_1, n_2 \le N - 1\}; \tag{2}$$

that is, it outputs realizations of $X_n = X_{(n_1,n_2)}$ for $n \in G(N)$.

2.2 Methodology

For details about the methodology underlying the synthesis methods implemented in Hermir-Fields, see [2].

3 Getting started

3.1 Installation instructions

1. *Requirements:* The software requirements for Hermir-Fields are

- Utility for extracting compressed archives stored in ZIP format
- MATLAB (code tested on version 7.10 but should work on other versions)

2. *Unpacking the archive:* HERMIR-FIELDS is distributed as a compressed tar or zip file. To install,

   (a) Download the compressed archive.

   (b) Uncompress it at the desired location using `zip` or some other ZIP-compatible utility.

   This will create a directory tree rooted at `HermirFields0_1` containing MATLAB functions and some demos.

3. For a quick look at HERMIR-FIELDS, start MATLAB, and run the demo script `DemoSynth.m`

4 Brief introduction to code

4.1 Overview

Field synthesis using HERMIR-FIELDS is comprised of the following three steps:

1. Define covariance structure (see Section 4.2).
2. Embed covariance using smoothing embedding using either *nonoverlapping* or *overlapping* windows.
3. Synthesize Gaussian field using embedded covariance.

The last step only works as long as the embedding results in a valid covariance (see [2] for discussion). The three main MATLAB functions for field synthesis using HERMIR-FIELDS are:

- `CEMWindow.m` – performs smoothing embedding using *nonoverlapping* windows
- `CEMOverlapWindow.m` – performs smoothing embedding using *overlapping* windows
- `FieldSynth.m` – 2D field synthesis given embedded covariance

Based on the results in [2], we recommend using `CEMOverlapWindow.m` instead of `CEMWindow.m`, as it should result in more efficient embedding. For demonstrations, see Section 4.3.

4.2 Defining and configuring covariance structures

The code uses MATLAB *function handles* to pass covariance structures in calls to embedding routines. To introduce new covariance structures to the code, one simply has to write a MATLAB function of the following generic form:

```
function val = mycovar(n1,n2,param)
% Write short description of function here
%
% Usage:
%   val = mycovar(n1,n2,param)
```

```
% Inputs
%   n1         matrix of indices for 1st coordinate
%   n2         matrix of indices for 2nd coordinate
%              NOTE: n1 and n2 have to be of equal dimensions
%              where entry n2(k,j) corresponds to entry n1(k,j)
%   param      this is optional and more parameters can be included
% Output
%   val        matrix with values of covariance at the indices given
%              in n1 and n2
%              val is of the same dimensions as n1 and n2 where
%              entry val(k,j) corresponds to the pair (n1(k,j),n2(k,j))

WRITE YOUR CODE HERE

%
```

Note that the embedding routines will only use the entries `n1` and `n2` when calling the covariance function; the user is therefore free to define a covariance function which uses more than one parameter variable, for example:

```
function val = mycovar(n1,n2,param1,param2,param3)
```

would be perfectly alright.

4.2.1 Example: Definition and configuration of a covariance structure

HERMIR-FIELDS includes a file `poweredexpW.m` which implements the function

$$r(n_1, n_2) = \exp\left(-(\theta \|n\|_W)^\alpha\right), \tag{3}$$

where W is a symmetric and positive definite 2×2 matrix, $n = (n_1\ n_2)^T$, and $\|n\|_W := \sqrt{n^T W n}$. To ensure that (3) is a covariance function, one requires $0 < \alpha \leq 2$ and $\theta > 0$.

The code in `poweredexpW.m` looks as follows:

```
function val = poweredexpW(n1,n2,theta,alpha,W)
% Anisotropic covariance in 2D with exponential-type decay
%   r(n) = exp(-alpha ||n||_W^beta)
% where W symmetric and positive definite,
% n=(n1,n2), ||n||_W^2 = n W n^T,
% alpha>0, and 0 < beta<=2
%
% Usage:
%   val = poweredexpW(n1,n2,theta,alpha,W)
% Inputs
%   n1         matrix of indices for 1st coordinate
%   n2         matrix of indices for 2nd coordinate
%              NOTE: n1 and n2 have to be of equal dimensions
```

```
%              where entry n2(k,j) corresponds to entry n1(k,j)
%   theta      positive scalar
%   alpha      scalar satisfying 0 < beta <= 2
%   W          symmetric and positive definite 2-by-2 matrix
% Output
%   val        matrix with values of covariance at the indices given
%              in n1 and n2
%              val is of the same dimensions as n1 and n2 where
%              entry val(k,j) corresponds to the pair (n1(k,j),n2(k,j))

% Checking arguments:
if (alpha <= 0) || (alpha > 2),
    error('Input argument "alpha" has to satisfy 0 < alpha <= 2.')
end
if theta <= 0,
    error('Input argument "theta" has to satisfy theta > 0.')
end

tmp = W(1,1)*n1.^2 + W(2,2)*n2.^2 + 2*W(1,2)*n1.*n2;
weigthedNorm = sqrt(tmp);
val = exp(-(theta*weigthedNorm).^alpha);

% Copyright (c), Hannes Helgason, KTH, 2011
```

Next we show an example for how to create a function handle for the covariance function defined in `expdecaycovW.m` with the parameter configuration $W = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix}$, $\alpha = 0.01$, $\beta = 1$:

```
% CONFIGURE PARAMETERS
alpha = 0.01;
beta = 1;
W = [1 1;1 2];
% CREATE FUNCTION HANDLE
funch = @(n1,n2) expdecaycovW(n1,n2,alpha,beta,W);
```

4.3 Demos

The files `DemoSynth.m` and `DemoFindMaxSize.m` provide documented demos for the code.

5 Acknowledgements

Contributors to Hermir-Fields are Hannes Helgason (KTH, Stockholm, Sweden), Vladas Pipiras (UNC, Chapel Hill, USA), and Patrice Abry (ENS de Lyon, Lyon, France).

6 Copyright and warranty

Hermir-Fields is copyrighted material. There is no warranty attached to Hermir-Fields. For copying permissions and warranty notice, see the documents `COPYING.txt` and `WARRANTY.txt` located in the folder `Documentation` in the Hermir-Fields distribution.

References

[1] Hermir. Website: `http://www.hermir.org`.

[2] H. Helgason, V. Pipiras, and P. Abry. Smoothing windows for the synthesis of Gaussian stationary random fields using circulant matrix embedding, 2012. Submitted manuscript.