# SIPPI

Ed. version 1.0

| COLLABORATORS | | | |
|---|---|---|---|
| | *TITLE* :<br><br>SIPPI | | |
| *ACTION* | *NAME* | *DATE* | *SIGNATURE* |
| WRITTEN BY | Thomas Mejer Hansen and Knud Skou Cordua | April 11, 2014 | |

| REVISION HISTORY | | | |
|---|---|---|---|
| NUMBER | DATE | DESCRIPTION | NAME |
| | | | |

# Contents

# About

SIPPI is a Matlab toolbox (compatible with GNU Octave) that allow sampling the solution of non-linear inverse problems with realistic a priori information.

In order to make use of SIPPI one has to

- Install and setup SIPPI
- Define the prior model, in form of the prior data structure
- Define the forward model, in form of the forward data structure, and the sippi_forward.m m-file
- Define the data and noise model, in form of the prior data structure
- Choose a method for sampling the a posteriori probability density.

# Chapter 1

# Installation

## 1.1 SIPPI

Download the latest version of SIPPI from http://sippi.sourceforge.net.

Unpack ZIPPI_1.0.zip somewhere, for example to 'c:\Users\tmh\SIPPI'. Then setup the Matlab path to point to the appropriate SIPPI directories:

```
addpath c:\Users\tmh\SIPPI
sippi_set_path
```

### 1.1.1 SGeMS (optional)

To make use of the SISIM and SNESIM type priori models SGeMS needs to be available.

Currently only SGeMS version 2.1 (download) for Windows is supported.

# Chapter 2

# Setting up SIPPI

## 2.1 The a priori model

### 2.1.1 Types of a priori models

#### 2.1.1.1 1D Generalized Gaussian

A 1D generalized Gaussian prior model can be specified using the 'gaussian' type prior model

```
prior{1}.type='gaussian';
```

A simple 1D Gaussian distribution with mean 10, and standard deviation 2, can be specified using

```
prior{1}.type='gaussian';
prior{1}.m0=10;
prior{1}.std=2;
```

The norm of a generalized Gaussian can be set using the 'norm' field. A generalized 1D Gaussian with mean 10, standard devation of 2, and a norm of 70, can be specified using (The norm is equivelent ot the beta factor referenced in Wikipedia:Generalized_normal_dist

```
prior{1}.type='gaussian';
prior{1}.m0=10;
prior{1}.std=2;
prior{1}.norm=70;
```

#### 2.1.1.2 VISIM

#### 2.1.1.3 FFTMA

#### 2.1.1.4 SISIM

#### 2.1.1.5 SNESIM

### 2.1.2 Sampling the prior

Once the prior data structure has been defined a sample from the prior distribution can be generated using

```
m=sippi_prior(prior);
```

'm' is a Matlab data structure of the same size as the 'prior' data structure. Thus, if two prior distributions have been defined in 'prior{1}' and 'prior{2}', then 'm{1}' will hold a realization of 'prior{1}', and 'm{2}' will hold a realization of 'prior{2}'.

Each time 'm=sippi_prior(prior)' is called, a new independant realization of the prior will be generated.

### 2.1.3 Sequential Gibbs sampling / Conditional Resampling

All the available a priori types available allow perturbing one realization of a prior into a new realization of prior, in the vicinity of the first one. To do this we make use of sequential Gibbs sampling [HCM12]. Sequential Gibbs in essence is a type of conditional resampling. From a current realization of a prior, a number of model parameters are discarded and treated as unknown, and the simulated conditional to the fixed values of the model parameters.

In order to generate a new realiztion 'm2' in the viciinity of the realization 'm1' use

```
m1=sippi_prior(prior);
[m2,prior]=sippi_prior(prior,m1);
```

If this process is iterated, then a random walk in the space of a priori acceptable models will be perform. And, the collection of realization obatined, will represent a sample from prior distribution.

#### 2.1.3.1 Controlling sequential Gibbs sampling / Conditional Resampling

All properties related to sequential Gibbs sampling can be set in the 'seq_gibbs' data struture, for each prior type. The follwing two parameters determined how the a current model is perturbed

```
prior{m}.seq_gibbs.step=1;
prior{m}.seq_gibbs.type=2;
```

One can also optionally, as part of running the extended Metropolis sampler, automatically update the 'step'-length in order to ensure a specific approximate acceptance ratio 'P_target' of the Metropolios sampler. See [CHM12] for details.

The default parameters for adjusting the step length is givel below. These parameters will be set the fisrt time 'sippi_prior' is called with the 'prior' structure as output.The default parameters. [**?**, **?**]

```
prior{m}.seq_gibbs.step_min=0;
prior{m}.seq_gibbs.step_min=1;
prior{m}.seq_gibbs.i_update_step=50
prior{m}.seq_gibbs.i_update_step_max=1000
prior{m}.seq_gibbs.n_update_history=50
prior{m}.seq_gibbs.P_target=0.3000
```

By default adustment of the step length, in order to achieve an acceptance ratio of 0.3 ('prior{m}.seq_gibbs.P_target='), will be performed for every 50 ('prior{m}.seq_gibbs.i_update_step') iterations, using the acceptabce ratio observed in the last 50 ('prior{m}.seq_gibbs.i_update_history') iteration.

Adjustment of the step length will be performed only in the first 1000 ('prior{m}.seq_gibbs.i_update_step_max') iterations.

In order to disable automatiuc adjustment of the step length simply set

```
prior{m}.seq_gibbs.i_update_step_max=0; % disable automatic step length
```

## 2.2 The data and the noise

## 2.3 The forward model

# Chapter 3

# The a posteriori distribution

## 3.1 Sampling the a posteriori probability density

### 3.1.1 The rejection sampler

### 3.1.2 The extended Metropolis sampler

#### 3.1.2.1 The extended independent Metropolis sampler

### 3.1.3 linear least squares

## 3.2 Simulated Annealing

# Chapter 4

# Examples

## 4.1 Line fitting

The forward problem consists of computing the y-value as a function of the x-position of the data, and the polynomial coefficients determining the line. sippi_forward_linefit.m:

```
% sippi_forward_linefit Line fit forward solver for SIPPI
%
% [d,forward,prior,data]=sippi_forward_linefit(m,forward,prior,data);
%
function [d,forward,prior,data]=sippi_forward_linefit(m,forward,prior,data);

if length(m)==1;
    d{1}=forward.x*m{1};
elseif length(m)==2;
    d{1}=forward.x*m{1}+m{2};
else
    d{1}=forward.x.^2*m{1}+forward.x*m{2}+m{3};
end
```

## 4.2 Covariance model inference

## 4.3 Cross hole tomography

## 4.4 Reflection seismic inversion

# Chapter 5

# Bibliography


[CHM12] K. S. Cordua, T. M. Hansen, and K. Mosegaard, Monte Carlo full waveform inversion of crosshole GPR data using multiple-point geostatistical a priori information, H19--H31.

Geophysics, 77, 2012.

[HCLM13a] T.M. Hansen, K.S. Cordua, M.C. Looms, and K. Mosegaard, SIPPI: a Matlab toolbox for sampling the solution to inverse problems with complex prior information: Part 1, methodology, 470--480.

Computers & Geosciences, 52, 03 2013.

[HCLM13b] T.M. Hansen, K.S. Cordua, M.C. Looms, and K. Mosegaard, SIPPI: a Matlab toolbox for sampling the solution to inverse problems with complex prior information: Part 2, Application to cross hole GPR tomography, 481--492.

Computers & Geosciences, 52, 03 2013.

[HCM12] T. M. Hansen, K. C. Cordua, and K. Mosegaard, Inverse problems with non-trivial priors - efficient solution through sequential Gibbs sampling, 593--611.

Computational Geosciences, 16, 2012.

[HCM14] T. M. Hansen, K. S. Cordua, and K. Mosegaard, Accounting for imperfect forward modeling in geophysical inverse problems - exemplified for cross hole tomography, xx.

Accepted for publication in Geophysics, xx, 2014.