Replies to Comments on the AM method of Manuscript bg-2017-41

I would like to express my sincere gratitude to Marko Laine, Jouni Susiluoto, Johanna Tamminen, and Heikki Haario for their insightful and constructive comments and suggestions. These are very useful comments to enhance the manuscript, and I will revise the manuscript accordingly. Below are my responses.

Comment 1:

In this comment we show that the authors may not have understood the AM algorithm properly or are using very non-optimal initial values and tuning parameters. In our tests, all the examples work without problems and out-of-the-box by the AM algorithm, and with only 10% of the simulations performed in the article. We do not claim any superiority of AM compared to other methods, but just want to point out that there might be some problems in the authors code. The DREAM algorithm may be very good algorithm for the purpose stated, but this demonstration does not necessarily have to depend on implementing some other method quite poorly.

Response:

I appreciate the reviewers' comments and suggestions. The AM code I used was downloaded from Dr. Heikki Haario's website about two years ago named mcmcrun.zip. When using the AM and DREAM algorithms to simulate the numerical examples in the manuscript, I initialized both algorithms in the same way, except that AM also needs the initialization of the Gaussian proposal distribution besides the initialization of the first point in the chain. More details please see the test examples below.

Comment 2:

The authors claim, that "AM can be unreliable for estimating the PDFs of the parameters that exhibit strong correlation". We have not seen such behavior, and would be happy to see test cases to support the claim. The authors may be mixing SCAM algorithm with AM, of which the first one as a single component method does indeed have a well-documented problem with strong correlations.

Response:

I had the claim based on the results of Case study I in the manuscript. The test function is a 50D Gaussian distribution function with correlation coefficients of 0.5, not very strong correlations between the parameters. But based on my test results, the standard deviations of and the covariance between some parameters were not able to converge to the correct values in 1e5 iterations as shown in the manuscript and the following Figure 1. In the test, the initial state of the chain was drawn from a uniform distribution with range [-50, 50]. When I started the chain from value of zero, the AM algorithm can converge to the correct statistics, as the reviewers presented in their comment file. On the other hand, I initialized DREAM algorithm from the same uniform distribution U[-50, 50], the DREAM can converge to the correct values within the same number of function evaluations.

Comment 3:

It seems, that the authors were not able to perform the relatively simple test cases with the AM algorithm properly. Because of this, there is no reason to believe that the results from the simulations with the DALEC model would appropriately portray the capabilities of AM. While the DREAM algorithm may be superior to AM/DRAM in some settings, we believe that the work presented here does not support such a claim.

Response:

I am not sure what the reviewers exactly meant of the word "properly". If the reviewers meant that we did not use a correct AM code, I would say that the AM code used for the manuscript was downloaded from Dr. Heikki Haario's website and it seems now it has an updated version. In the following I presented the AM results by using the AM code provided by the reviewers and the results were similar as those in the manuscript. If the reviewers meant that we did not implement the AM algorithm with suitable initialization of the start point and proposal distribution, I would say that in reality with little information about the underlying target distributions, it is very difficult to choose suitable initializations. In the numerical examples of the manuscript, both AM and DREAM algorithms used the same initialization of the start points.

Comment 4:

The presented examples are chosen to imitate the test cases in the manuscript. However, as the authors for some reason do not describe them fully, the functions are only our best guesses. In the manuscript, 10^6 MCMC iterations are used. Below, we use only 10^5, as we get convergence with it in all the test cases. We list the complete Matlab commands to generate the chains. A full code, including the plotting commands to generate the figures is included in the MCMC matlab toolbox cited above.

Response:

We used the MATLAB code provided by the reviewers at http://helios.fmi.fi/~lainema/mcmc/ to implement the three test cases and presented the results below along with the complete Matlab commands.

Test I: 50 dimensional Gaussian

In the manuscript, we explicitly described the 50D Gaussian function as follows: "The test function is a 50 dimensional (50 parameters) multivariate Gaussian distribution with the mean at the zeros. The covariance matrix was constructed such that the variance of the *i*th dimension is equal to $0.1\times i\times i$ and the covariance of *i*th and *j*th variables is calculated as $0.05\times i \times j$. Both the AM and DREAM located the initial states of the chains from a uniform distribution $x_0 \in U[-50, 50]^{50}$."

We used the following implementation and got the Figure 1.

############## Implementation of Test I ##############

```
nsimu = 100000;    % number of simulations
npar = 50;
% Construct the covariance matrix
```

```
A = 0.5 * eye(npar) + 0.5 * ones(npar);
for i = 1:npar
      for j = 1:npar
            Sig(i,j) = A(i,j) * (0.1*i * j);
      end
end
ci = inv(Sig);
model.ssfun = @(x,d) x(:)'*ci*x(:);
% Initial value is from U[-50, 50]
a=-50; b=50;
par0 = a + (b-a).*rand(1,npar);
options.nsimu = nsimu;
options.method = 'am';
options.qcov = eye(npar)/npar*2.4^2.;
options.adaptint = 1000;
for i=1:npar, params{i} = {sprintf('x_{%d}',i), par0(i)}; end
[results,chain] = mcmcrun(model,[],params,options);
```

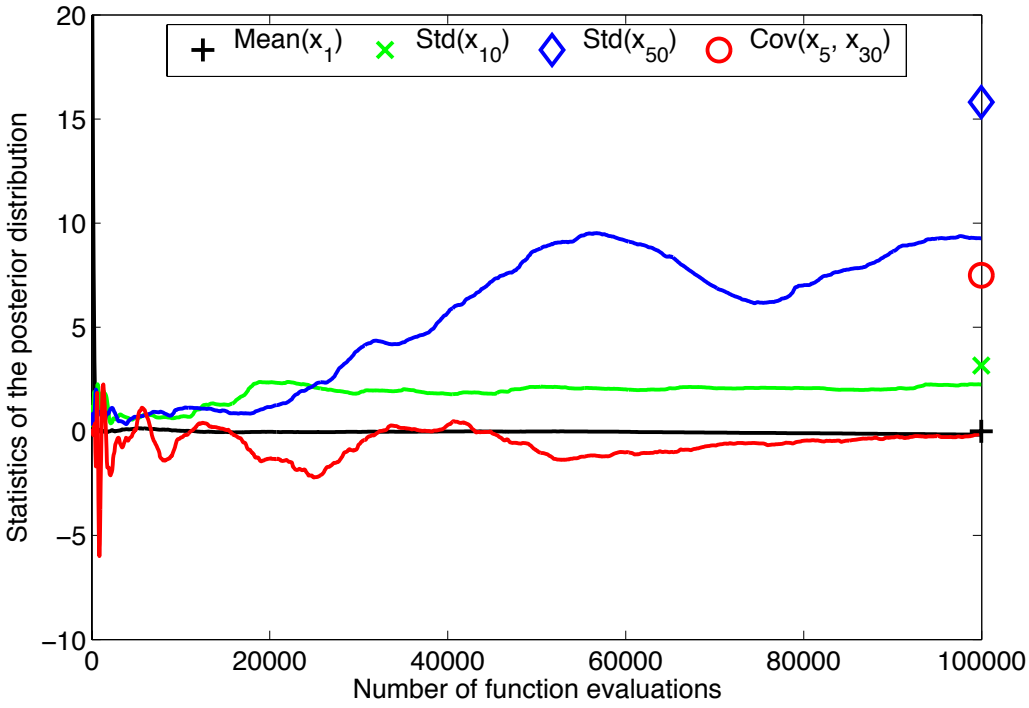

Figure 1: Convergence of 50D Gaussian distribution based on the AM algorithm. The true values of the four statistics are indicated at the right hand side of the figure.

The results in Figure 1 were similar to those presented in the manuscript. Please note that when I used the same Matlab commands in the reviewers' comment file, I can get the similar figure as the one of the reviewers. But the different results between the reviewers and the manuscript were from different AM implementations of a different test function: (1) the reviewers used a different Gaussian distribution; and (2) the reviewers initialized the AM algorithm from a different start point.

Test II: 10 dimensional Cauchy distribution

I was afraid that the 10D Cauchy function implemented in the reviewers MCMC simulation was not right (could be a typo). Based on Wikipedia, the -2*log of the 10D Cauchy function is proportional to $(1+10)*\log(1+(x-u)^T E^{-1}(x-u))$, here we took the mean u=0, and the covariance matrix E as the identity matrix, so it can be simplified as (1+10)*log(1+sum(x.^2)), not the 2*sum(log(1+x.^2) in the reviewers' comment file.

We used the following implementation and got the Figure 2.

############### Implementation of Test II ###############

```
nsimu = 100000;    % number of simulations
npar = 10;
model.ssfun = @(x,d) (1+10)*log(1+sum(x.^2));
a=-10; b=10;
par0 = a + (b-a).*rand(1,npar);
options.nsimu = nsimu;
options.method = 'am';
for i=1:npar, params{i} = {sprintf('x_{%d}',i), par0(i)}; end
[results,chain] = mcmcrun(model,[],params,options);
```

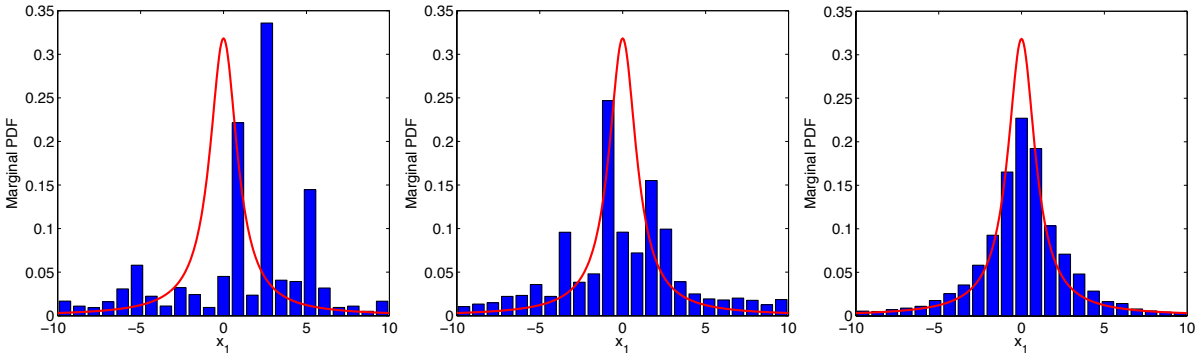

Figure 2: Approximated (histogram) and actual (red curve) marginal posterior distributions of one dimension (x_1) of the 10D Cauchy function based on three AM independent runs.

I implemented three AM runs using the above same Matlab commands and got three different results as presented in Figure 2, where the first plot was similar to the one in the manuscript, and the third plot was similar to the one in the reviewers' comment file. Base on the test results, it seems that AM can sample the 10D Cauchy function, but not always, as sometimes it may produce the incorrect results as shown in the first two plots in Figure 2.

Test III: 3 modes 4 dimensional mixed Gaussian

I agree with the reviewers that no method will find all the modes if they are too distant and there is no prior information about the locations. And I also agree with the reviewers that DRAM and AM can be made to sample multi-modal distributions with *suitable* initial values. The incorrect sampling results of AM in the manuscript may be caused by the unsuitable initialization, but in reality we have little information about the suitable initialization for a specific problem, as discussed in the manuscript.

I agree with Dr. Jasper Vrugt in his comment that AM algorithm may have difficulty in sampling multimodal distributions with far-disconnected modes and the problem could become more difficult with increasing dimensionality. Dr. Vrugt gave a one-dimensional, two-modal distribution example from Figure 5 of Vrugt (2016). Below I showed a 10-dimensional, three-modal example. The example is the same with the case study III in the manuscript but has 10 dimensions. The AM was implemented with the following commands:

############### Implementation of Test III ###############

```
nsimu = 100000;    % number of simulations
npar = 10;
mu1 = -8*ones(1,npar);
mu2 = zeros(1,npar);
mu3 = 8*ones(1,npar);
sigs = ones(1,npar);
w = [0.1, 0.3, 0.6];
model.ssfun= @(x,d) -2*log(w(1)*mvnorpf(x,mu1,sigs) + w(2)*mvnorpf(x,mu2,sigs) + ...
w(3)*mvnorpf(x,mu3,sigs));
options.nsimu = nsimu;
options.method = 'dram';
options.qcov = eye(npar)*5^2;
options.adascale = 2.4 / sqrt(npar) * 5;
options.drscale = 5;
% 3 AM runs with initial value of -5,0,5, respectively
for i=1:npar, params{i} = {sprintf('x_{%d}',i), 0}; end
[results,chain] = mcmcrun(model,[],params,options);
```

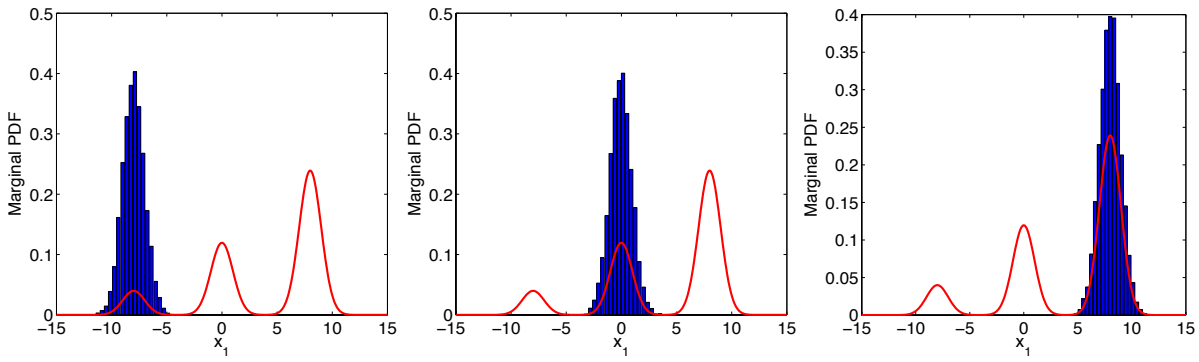

Figure 3. Approximated (histogram) and actual (red curve) marginal posterior distributions of one dimension (x_1) of the 10D, three-modal Gaussian function based on three AM runs.

I implemented three AM runs with initial values of -5, 0, 5, respectively and got three different results as presented in Figure 3. The results indicated that AM may have some difficulties in sampling the multi-modes for a relatively high-dimensional problem. On the other hand, Laloy and Vrugt (2012) demonstrated that DREAM can successfully sample the 25-dimensional, three-modal Gaussian function with the two neighboring modes having a separation of 10 units.

Comment 5:

We also noticed that in the manuscript there are some mix-ups with the references related to various versions of adaptive MCMC algorithms. Below are the correct references and other potentially interesting references.

Response:

I appreciate the reviewers for providing the very useful and helpful references. The provided references will be added in the revised manuscript and given the right citations.

Laloy, E., and J.A. Vrugt: High-dimensional posterior exploration of hudrologic models using multiply-try DREAM(zs) and high-performance computing, Water resour. Res., 48, W01526, 2012.

Vrugt, J.A.: Markov chain Monte Carlo simulation using the DREAM software package: Theory, concepts, and MATLAB implementation, Environmental modeling & Software, 75, 273-316, 2016.

Once again, I appreciate the reviewers for the excellent comments and suggestions to improve the manuscript; I will revise the manuscript accordingly based on the suggestions.

Sincerely yours,
Dan Lu