**Instruction file**

The Matlab code presented below is a very simplified and schematic implementation of the Image Quilting (IQ) algorithm with Template Splitting (TS). It corresponds to the very first version of the IQ, but can be used for both 2D unconditional and conditional IQ simulations with both categorical and continuous variables. For instance, it uses square tilesizes having same dimensions in both x and y axis. Only the core idea of the IQ is implemented and other features are on their way (such as 3D IQ, multivariate simulation, multiscale analysis with kriging correction, IQ analysis with auxiliary variable, parallel computing, and so on).

Authors: Kashif Mahmud and Gregoire Mariethoz, 2013

Instructions:

1. Run the Matlab function “CIQ\_TS\_v3.m”
2. Read the input training image (TI) with the following line (line 20 in the code):

X1 = LoadGrid('ti\_lena\_crop.SGEMS');

1. Parameters input list with a set of example:

tilesize = 30; % Optimal size for the patch size is between 1/7 and 1/8 of the TI

tilesize\_vary = 0; % Randomize patch size (max +-10% of patch size)

overlap = 7; % Optimal size for the overlap region is between 1/3 and 1/4 of the tilesize

nbreplicates = 3; % Standard Number of replicas = 5 to 10 to avoid "Verbatim Copy"

cond = 1; % Define whether Conditional or Unconditional simulation (0 or 1)

c = 50; % No of conditioning points (Don’t need to be defined if you load your Data file in the code)

w = 0.9; % Conditioning weight (Range [0 1])

w\_v = 1; % Each variable's weight (For Multivariable IQ analysis)

% This is an univariate IQ version, Multivariate version will follow soon.

nbrealz = 1; % Number of realizations to be produced

temp\_split = 1; % temp\_split = 1: Perform template splitting and temp\_split = 0: Do not perform template splitting

do\_cut = 1; % do\_cut = 1: Perform optimum cutting and do\_cut = 0: Do not perform optimum cutting

Few important suggestions:

1. The optimum parameters will be different for different TIs and conditioning data, so one has to find out the best possible set by playing with these.
2. tilesize should be minimum to get better realizations/100% conditioning with dense data set.
3. In case of a dense data set one has to minimize the tilesize, overlapping area and number of replicates to initiate 100% conditioning. For this one has to investigate these parameters systematically because they will be different for each TI and they have an influence on the final realizations.
4. The most important parameter in case of dense data points is number of replicates (nbreplicates), one has to reduce this value even up to 1 (for severe cases). This would produce 100% conditioning but at the same time obviously ease the variability near the data points and might yield artifacts.
5. To increase the variability within the realizations one could shift the tilesize from one simulation to another (by using tilesize\_vary), which will randomize the tilesize (tilesize\_vary = max +-10% of tilesize).