MATLAB code instruction of the Sparse Precision Selection (SPS) algorithm

http://arxiv.org/abs/1405.5576

Written by: Sam Davanloo Tajbakhsh

(https://sites.google.com/site/samdavanloo/home)

System Requirement

The code works fine on Macintosh and Windows operating systems. On Macintosh systems, it has been tested on OS X Mavericks, Yosemite, and El Capitan. On Windows systems, it has been tested on Windows 7 and 8. Furthermore, the code is tested on MATLAB 2014 and 2015b.

The SPS Package

The SPS package includes a script driver file *driverSPS.m*. All of the SPS package functions are available in the *SPS_V04* folder which should be added to the MATLAB path.

The Training Data

Let $X \in \mathbb{R}^{n \times d}$ be the matrix of input/independent variables and $Y \in \mathbb{R}^{n \times N}$ be N realizations of the Gaussian Process (GP) observed over n distinct points. X and Y are considered as the training data; so if cross-validation is required, it should be performed before this stage.

Main Variables

blk: A cell array containing the segmentation (blocking) information.
op1: A cell array containing the setting for the STAGE-I problem of the SPS algorithm.
hyper: A cell array containing the setting of the parameter of the covariance function.
process: A cell array containing the setting for computational capabilities.

How to Run the SPS Code

In this section, we briefly discuss how to run the SPS code. Open the *driverSPS.m* file. The lines with left arrow sign at the end are those that need to be tuned by the user. Below, we will go over these lines starting from the top:

- *blk.scheme* determines the segmentation scheme. User may chose either *'SS'* (Spatial Segmentation) or *'RS'* (Random Segmentation). Please refer to the paper for more information on these two blocking schemes.
- *uX* variable in the 'SS' segmentation scheme is a $d \times 1$ vector containing the number blocks along each coordinate. A vector of all ones basically impose no segmentation scheme which is used when n is not big.
- *blk.K* determines the number of blocks in the 'RS' segmentation scheme. *blk.K=1* defines only one block.
- *opt1.monitor* allows visual monitoring of the STAGE-I problem. If set to *'on'* shows information on iterations of the ADMM algorithm; otherwise, set it to *'off'*.
- *opt1.tol.primal* is the prima feasibility threshold for the ADMM algorithm. If *both* primal and dual feasibilities go below their thresholds, the algorithm will terminate.
- *opt1.tol.dual* is the dual feasibility threshold for the ADMM algorithm. If *both* primal and dual feasibilities go below their thresholds, the algorithm will terminate.
- *opt1.maxItr* is the maximum number of iterations of the ADMM algorithm and is another stopping criterion.
- *hyper.covFunc* determines the parametric covariance function of interest. User may select from the list below:
  - *SEiso*: Squared-Exponential (Isotropic)
  - *PEiso*: Powered-Exponential (Isotropic) - if selected, then *hyper.p* determines the power where it should be an integer number. Otherwise, set to [].
  - *Materniso*: Matern (Isotropic) - if selected, then *hyper.nu* determines the smoothness parameter which can be equal to 1/2, 3/2, or 5/2. Otherwise set it to [].
  - *Expiso*: Exponential (Isotropic)
- *hyper.nugget* If set to *'true'*, then it includes the nugget into the model; otherwise, user may set it to *'false'*.
- *process.type* determines if parallel processing capabilities are available or not. If so, set it to *'parallel'*; otherwise, set it to *'single'*.
- *process.nCores* determines the number of accessible processing cores and is considered only if *process.type* is *'parallel'*.

The *spsEstimatorHyper* calls the corresponding function and begins the SPS algorithm. It outputs the SPS parameter estimates which are available in the *hyper.param.val* as a vector.