Using easyGSA

Basic syntax of easyGSA

Typing easyGSA into command line should return a welcome text and ensure that the tool is available for use.

```
easyGSA
```

```
Copyright 2018-2019, Resul Al and Gurkan Sin, all rights reserved.
This is easyGSA, version 1.2.0

easyGSA is a global sensitivity analysis (GSA) tool performing
variance decomposition-based Sobol sensitivity analysis using machine
learning algorithms, such as Gaussian process regression (GPR) and
artificial neural networks (ANN). Main (Si) and total (STi) effects
are returned.

To cite easyGSA in publications, please use:
 Al, R., Behera, C.R., Zubov, A., Gernaey, K. V., Sin, G., 2019.
 Meta-modeling based efficient global sensitivity analysis for
 wastewater treatment plants - An application to the BSM2 model.
 Comput. Chem. Eng. 127, 233-246.
 https://doi.org/10.1016/j.compchemeng.2019.05.015

To request special permissions or suggest improvements, please contact:
- Resul Al (resal@kt.dtu.dk) and Gurkan Sin (gsi@kt.dtu.dk)
- Technical University of Denmark
```

Computing the first (Si) and the total (STi) order Sobol sensitivity indices

Step 1: Model is defined as a Matlab function handle f. The size of the sampling matrices is defined as N.

```
% Model: Ishigami function [https://www.sfu.ca/~ssurjano/ishigami.html]
f = @(x) sin(x(:,1)) + 7.*sin(x(:,2)).^2 + 0.1.*x(:,3).^4.*sin(x(:,1));
N = 1e4; % Number of MC samples. Minimum recommended: 1e3
```

Step 2: Input space is defined with a uniform distribution between lower and upper bounds of parameters that are subject to GSA.

```
% Uniform Input Space
pars = strseq('x',1:3); % input parameter names
lbs  = -pi.*ones(1,3);  % lower bounds of input parameters
ubs  =  pi.*ones(1,3);  % upper bounds of input parameters
InputSpace = {'ParNames',pars,'LowerBounds',lbs,'UpperBounds',ubs};
```

Step 3: Calling the easyGSA tool with its basic syntax.

```
% call easyGSA tool to perform Sobol sensitivity analysis with MC approach
[Si,STi] = easyGSA(f,N,InputSpace{:})
```

```
Copyright 2018-2019, Resul Al and Gurkan Sin, all rights reserved.
This is easyGSA, version 1.2.0

easyGSA is a global sensitivity analysis (GSA) tool performing
variance decomposition-based Sobol sensitivity analysis using machine
learning algorithms, such as Gaussian process regression (GPR) and
```

```
 artificial neural networks (ANN). Main (Si) and total (STi) effects
 are returned.

 To cite easyGSA in publications, please use:
  Al, R., Behera, C.R., Zubov, A., Gernaey, K. V., Sin, G., 2019.
  Meta-modeling based efficient global sensitivity analysis for
  wastewater treatment plants - An application to the BSM2 model.
  Comput. Chem. Eng. 127, 233-246.
  https://doi.org/10.1016/j.compchemeng.2019.05.015

 To request special permissions or suggest improvements, please contact:
 - Resul Al (resal@kt.dtu.dk) and Gurkan Sin (gsi@kt.dtu.dk)
 - Technical University of Denmark

3 model inputs are found.
Sobol sampling is used to generate input matrices.
1 model output is found.
Your model is already vectorized.
Matrices are being filled with @(x)sin(x(:,1))+7.*sin(x(:,2)).^2+0.1.*x(:,3).^4.*sin(x(:,1))
Matrix A is filled.
Matrix B is filled.
There are 3 inputs.
Matrix for input number 1 is filled.
Matrix for input number 2 is filled.
Matrix for input number 3 is filled.
All matrices in yAB are filled.

  First Order Indices:
          Output_1
          _________

    x1      0.33489
    x2      0.44526
    x3    0.0091184


  Total Order Indices:
          Output_1
          ________

    x1    0.54871
    x2    0.43734
    x3    0.23538
```

```
Si =
      0.33489
      0.44526
    0.0091184
STi =
      0.54871
      0.43734
      0.23538
```

More detailed syntax

More detailed results of the Sobol sensitivity analysis can also be extracted as the following.

```
[Si,STi,results] = easyGSA(f,N,InputSpace{:},...
                              'SamplingMethod','LHS',... % default: 'Sobol'
                              'Estimator','Saltelli',... % default: 'Jansen'
                              'UseSurrogate','GPR',...   % options: 'GPR', 'ANN'
                              'UseParallel',true,...     % default:  false
                              'Verbose',false)           % defualt:  true
```

```
 Copyright 2018-2019, Resul Al and Gurkan Sin, all rights reserved.
 This is easyGSA, version 1.2.0

 easyGSA is a global sensitivity analysis (GSA) tool performing
 variance decomposition-based Sobol sensitivity analysis using machine
 learning algorithms, such as Gaussian process regression (GPR) and
 artificial neural networks (ANN). Main (Si) and total (STi) effects
 are returned.

 To cite easyGSA in publications, please use:
  Al, R., Behera, C.R., Zubov, A., Gernaey, K. V., Sin, G., 2019.
  Meta-modeling based efficient global sensitivity analysis for
  wastewater treatment plants - An application to the BSM2 model.
  Comput. Chem. Eng. 127, 233-246.
  https://doi.org/10.1016/j.compchemeng.2019.05.015

 To request special permissions or suggest improvements, please contact:
 - Resul Al (resal@kt.dtu.dk) and Gurkan Sin (gsi@kt.dtu.dk)
 - Technical University of Denmark
Si =
      0.31103
      0.45826
   -0.0035807
STi =
      0.54134
      0.46899
      0.22556
results = struct with fields:
          GPRmodel: [1×1 RegressionGP]
          GPRstats: [1×1 struct]
                 A: [10000×3 double]
                AB: {[10000×3 double]  [10000×3 double]  [10000×3 double]}
                 B: [10000×3 double]
         Estimator: 'Saltelli'
       LowerBounds: [-3.1416 -3.1416 -3.1416]
             Model: @(x)sin(x(:,1))+7.*sin(x(:,2)).^2+0.1.*x(:,3).^4.*sin(x(:,1))
                 N: 10000
          ParNames: {3×1 cell}
               STi: [3×1 double]
    SamplingMethod: 'LHS'
                Si: [3×1 double]
       UpperBounds: [3.1416 3.1416 3.1416]
       UseParallel: 1
      UseSurrogate: 'GPR'
          UserData: [1×1 struct]
                 X: [250×3 double]
            gprMdl: [1×1 RegressionGP]
          gprStats: [1×1 struct]
         n_outputs: 1
           results: [1×1 struct]
           verbose: 0
                 y: [250×1 double]
                yA: [10000×1 double]
               yAB: {[10000×1 double]  [10000×1 double]  [10000×1 double]}
                yB: [10000×1 double]
```