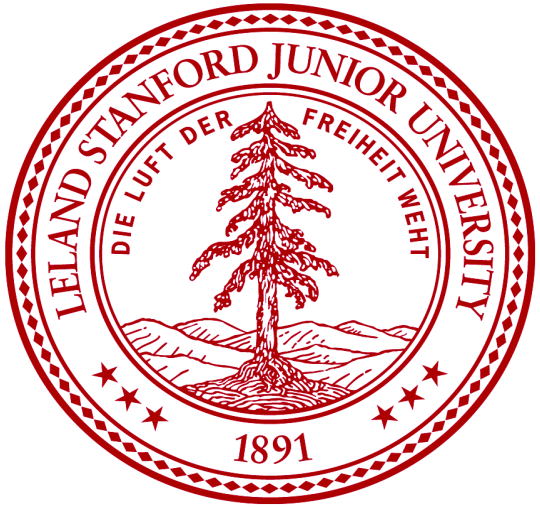
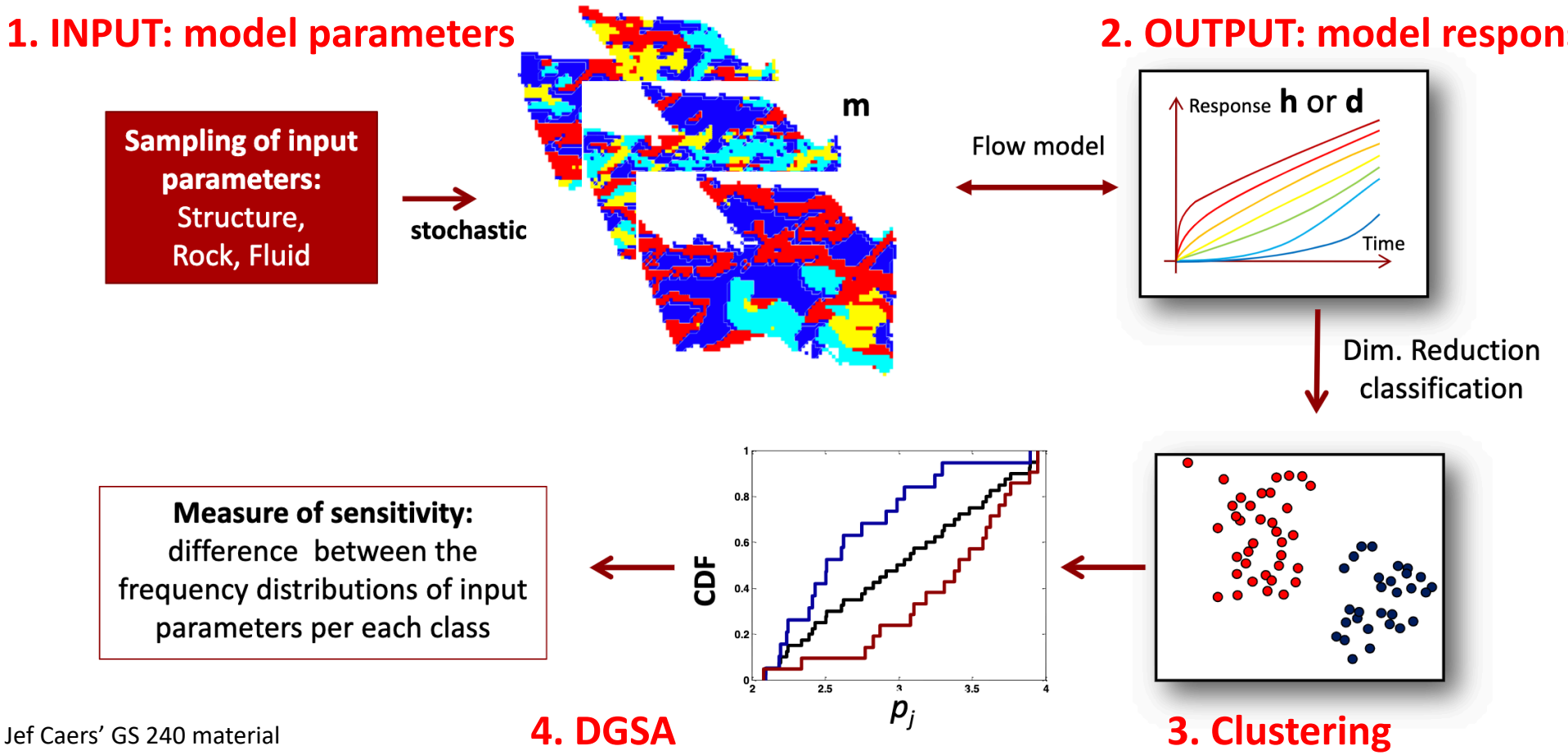
Sensitivity Analysis Tutorial

Lijing Wang



Methodology: DGSA

- Distance-Based Generalized Sensitivity Analysis, Fenwick, D., Scheidt, C. & Caers, J. Math Geosci (2014) 46: 493. https://doi.org/10.1007/s11004-014-9530-5



Adapted from Prof. Jef Caers' GS 240 material

Python Code

- Tutorial & code: https://github.com/lijingwang/DGSA_Light, forked from https://github.com/sdyinzhen/DGSA_Light

- Before we apply DGSA to calculate sensitivities:
  - 1&2 Monte Carlo: Multiple input and outputs -> on your own
  - 3 Clustering: K-medoids clustering on **Euclidean** distances between outputs -> code can take care
  - 4 DGSA: python code giving you the tornado chart -> code can take care

Input: Monte Carlo from prior distribution

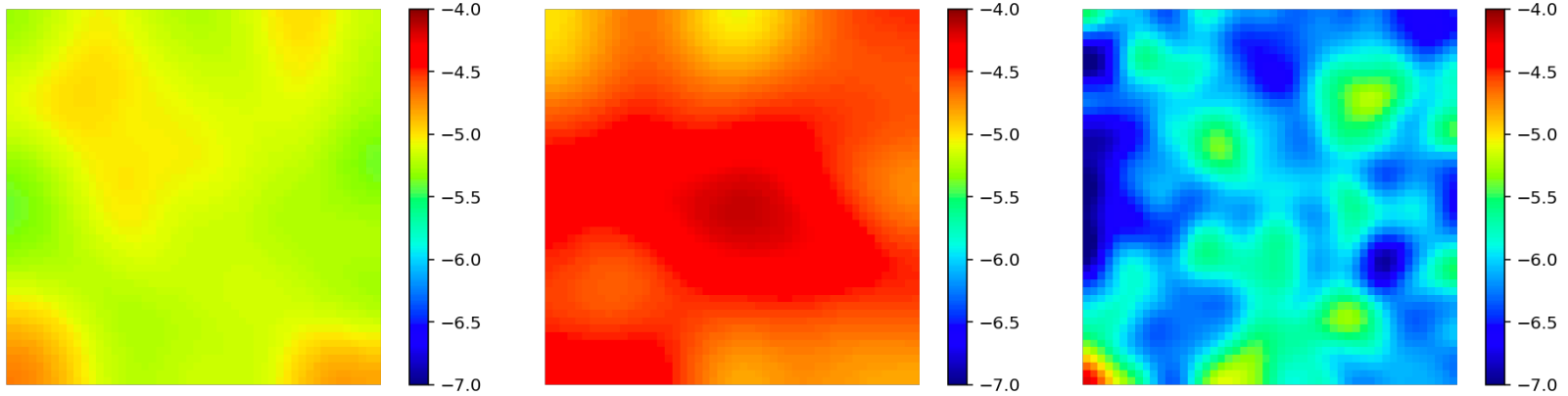
- Input, model parameters:
  - Global parameters: i.e. mean of log hydraulic conductivity
  - Sample: from a prior distribution, i.e. Gaussian or uniform: $U(-6, -4)$

log K -5.166 -4.559 -6.000 …

  - Spatial parameters: i.e. log hydraulic conductivity field
  - Sample: spatial distributed Gaussian fields

Possibly require dimension reduction

Spatial log K



Forward function: from input to output

- MODFLOW, CrunchTope …

log K

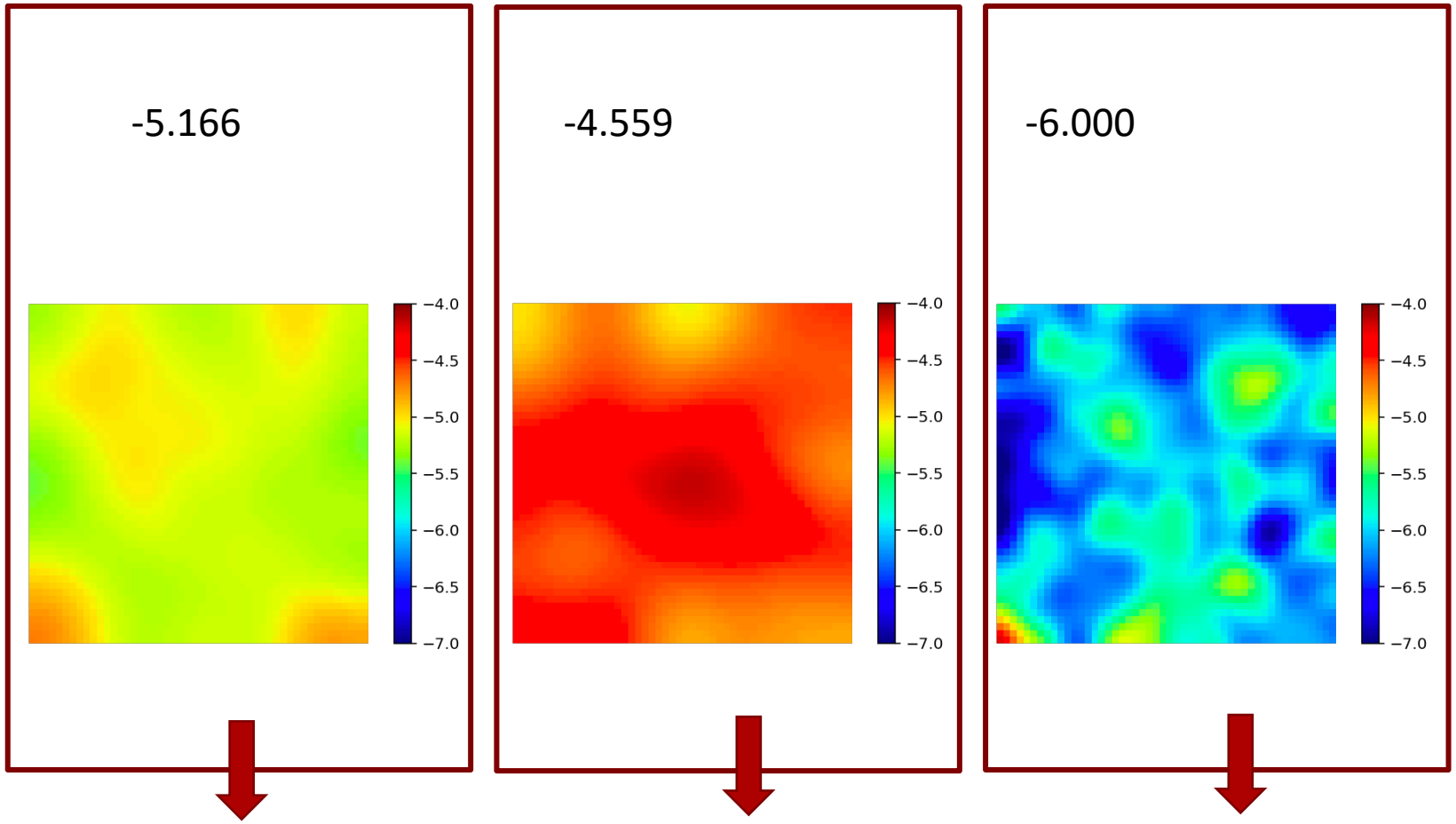
-5.166

-4.559

-6.000

…

Spatial log K

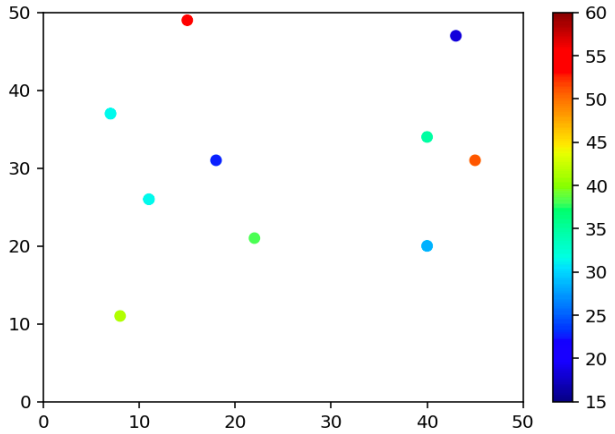


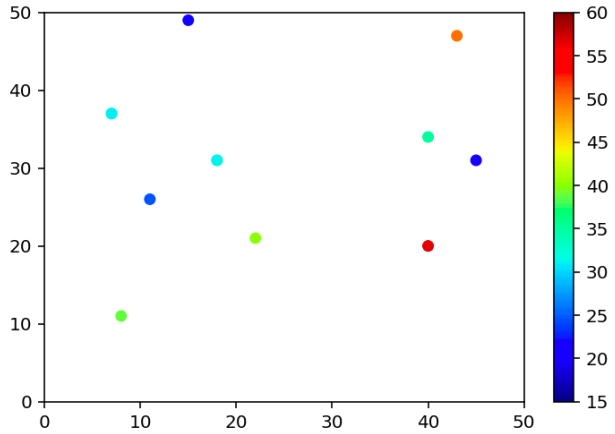
…

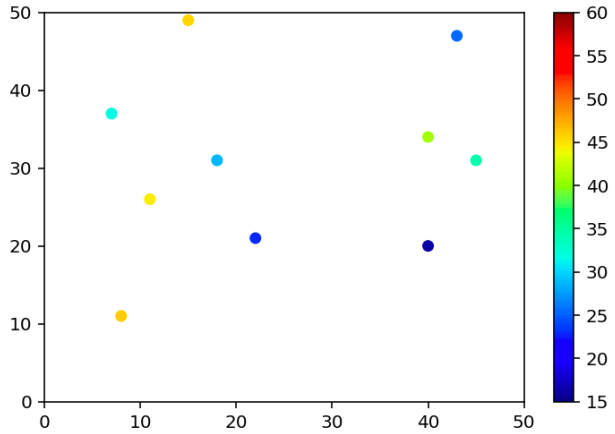
Forward functions

Output

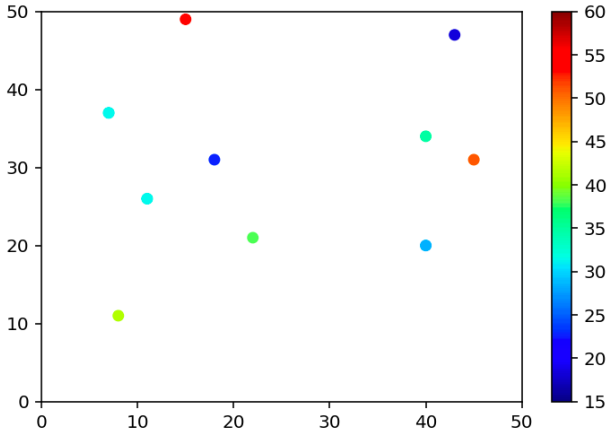
- Output, model responses, i.e. head maps

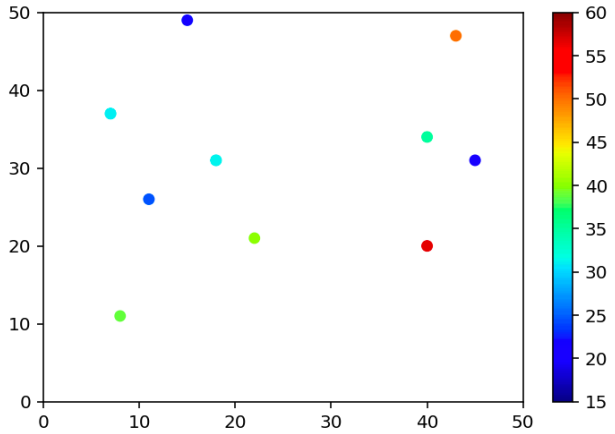


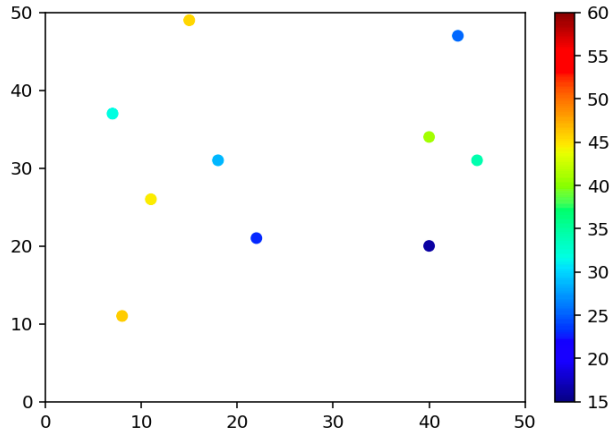




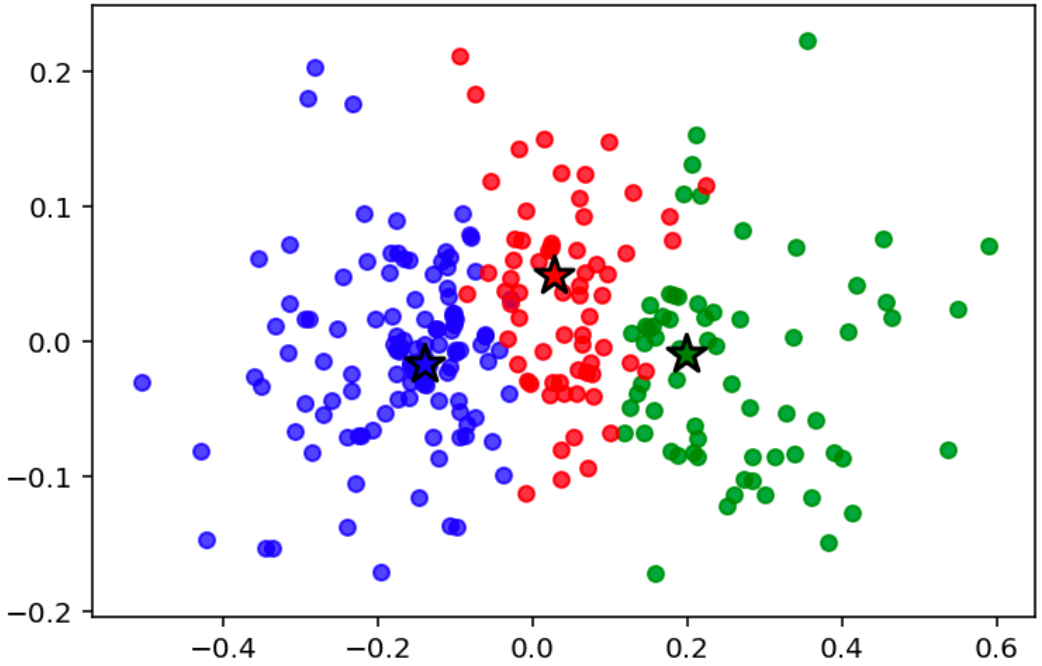
Clustering: K-medoid







- Euclidean distances: K-medoid.



jupyter Tutorial Last Checkpoint: a minute ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

In [1]:
```
import numpy as np
from DGSA_light import DGSA_light
from gsa_pareto_plt import gsa_pareto_plt
import pandas as pd
%matplotlib inline
%config InlineBackend.figure_format = 'retina'
```

Step 1, input

In [2]:
```
# 1 input, model_parameters:  L x N_input  dimension
model_parameters = pd.read_csv('./tutorial_data/m_parameters.csv')
```

Step 2, output

In [3]:
```
# 2 output, model_responses: L x N_output dimension
model_responses = pd.read_csv('./tutorial_data/m_responses.csv')
```

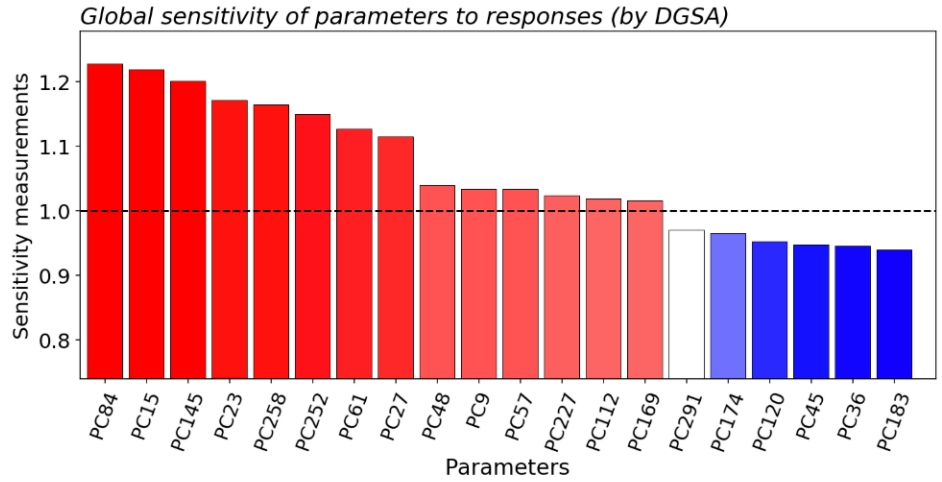
Step 3 & 4, Clustering + DGSA

In [4]:
```
# Call DGSA
# 3 clusutering by K-medoids, Euclidean distance
# 4 DGSA: calculating sensitivities
dgsa_measures = DGSA_light(model_parameters.values, model_responses.values,model_parameters.columns)
```

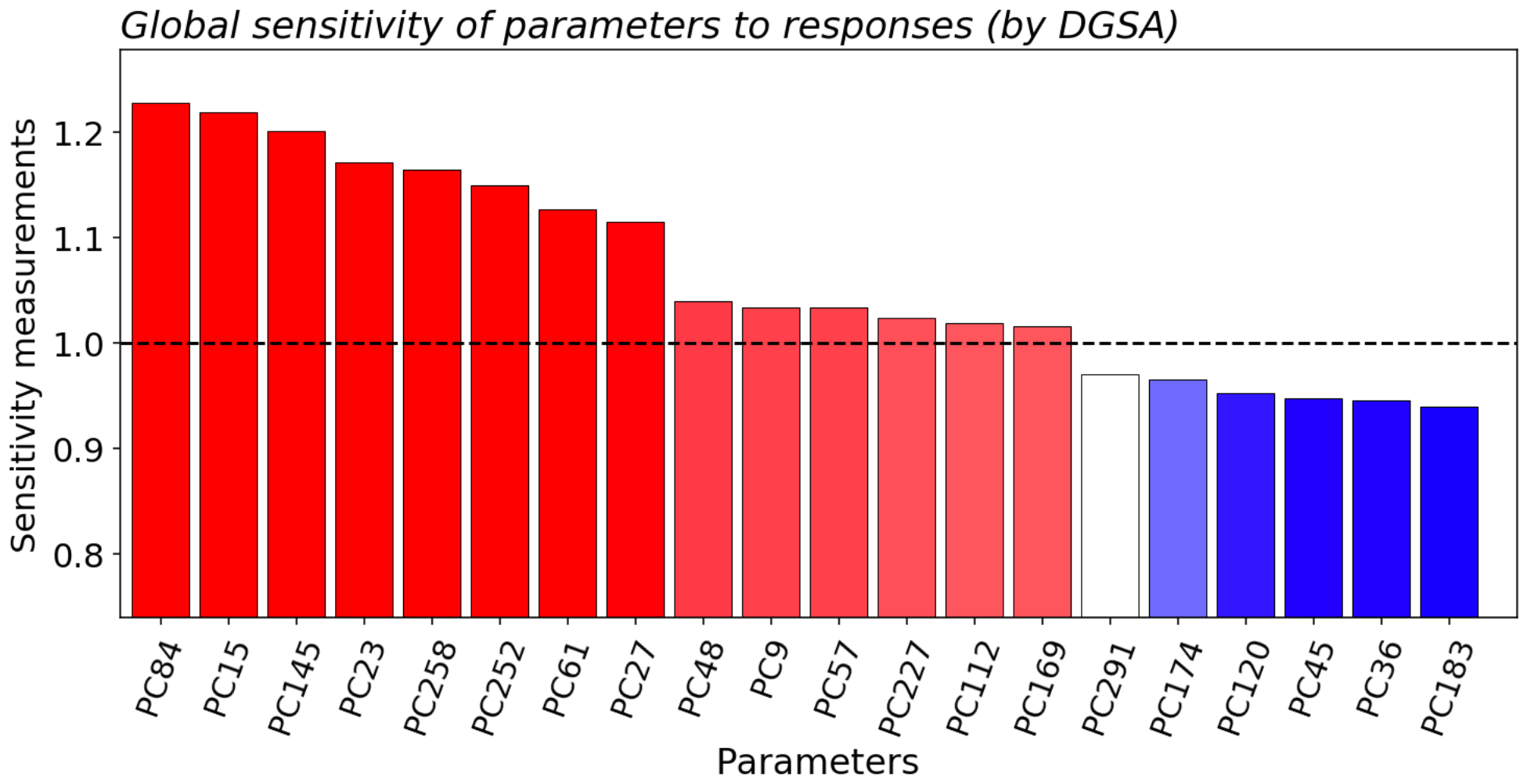
```
100%|██████████| 3000/3000 [00:32<00:00, 93.41it/s]
```

Visualization: the Pareto plot

In [5]:
```
gsa_pareto_plt(dgsa_measures)
```



DGSA result


Global sensitivity of parameters to responses (by DGSA)
Sensitivity measurements
1.2
1.1
1.0
0.9
0.8
PC84
PC15
PC145
PC23
PC258
PC252
PC61
PC27
PC48
PC9
PC57
PC227
PC112
PC169
PC291
PC174
PC120
PC45
PC36
PC183
Parameters