3-day workshop on modelling lake ecosystems with WET:

Tutorial on sensitivity analysis with parsac

This tutorial introduces sensitivity analysis with the Python-based tool for sensitivity analysis and auto-calibration in parallel developed by Jorn Bruggeman of Bolding and Bruggeman ApS to a workable GOTM-WET model setup. On the use of parsac, this tutorial should be used as a supplement to the information provided on https://pypi.org/project/parsac/ and https://bolding-bruggeman.com/portfolio/parsac/. For sensitivity analysis, parsac relies on the Python package Sensitivity Analysis Library in Python (SALib) published by Herman & Usher (2017) and Iwanaga, Usher and Herman (2022).

This tutorial builds upon the lecture on sensitivity analysis (SA) of complex models that aims to provide an overview of best practices for SA with GOTM-WET. For a basic introduction to SA, we recommend SALib's guide. This tutorial requires a workable GOTM-WET model set-up and installed Python package parsac and SALib.

Note that SA is executed independently from the QWET interface. In practice, users can first create a complete QWET model project, then exit the QWET (and QGIS) interface, and then navigate to the model directory (and work directly in this, or make a copy of this into a new folder for calibration purposes): \WET_model_example\project_name\LAKEMODEL\Default

Tutorial materials

Workable GOTM-WET setup for Shahe Reservoir

For our case study, we will work with the Chinese Shahe reservoir. See the QWET tutorials and exercises on how to configure a GOTM-WET model to Shahe Reservoir with lake specific hypsography and inflow and weather forcing as well as in-lake observations and files with calibrated parameters.

parsac

parsac is a Python-based tool for sensitivity analysis and auto-calibration in parallel developed by Bolding&Bruggeman. It is designed for analysis of models that take significant time to run. To download parsac, you will need a Python working environment and can install via Python pip: “pip install parsac –user”. Parsac requires the Python package Parallel Python to parallelize the auto-calibration process. For more details on parsac and its installation, we recommend checking out its Github page: https://github.com/BoldingBruggeman/parsac.

For installing Python 3, parsac, and required packages (pp (parallel python) and SALib), see installation guide “Tutorial on parsac and Python 3 installer” provided on USB stick Day 2 or in WeChat group.

Sensitivity Analysis Library in Python (SALib)

SALib provides a range of different sensitivity methods to analyze mathematical models or systems. Currently, parsac supports several sensitivity methods which depends on their own or other sampling schemes (Table 1). For a thorough documentation of SALib, its sampling schemes and SA methods, we refer to SALib's readthedocs and their Github page.

In this tutorial we will focus on the Sobol and Delta Moment-Independent Measure as documented in Andersen et al. (2020).

Table 1. parsac supported SALib methods and their corresponding sampling schemes with their parsac names.

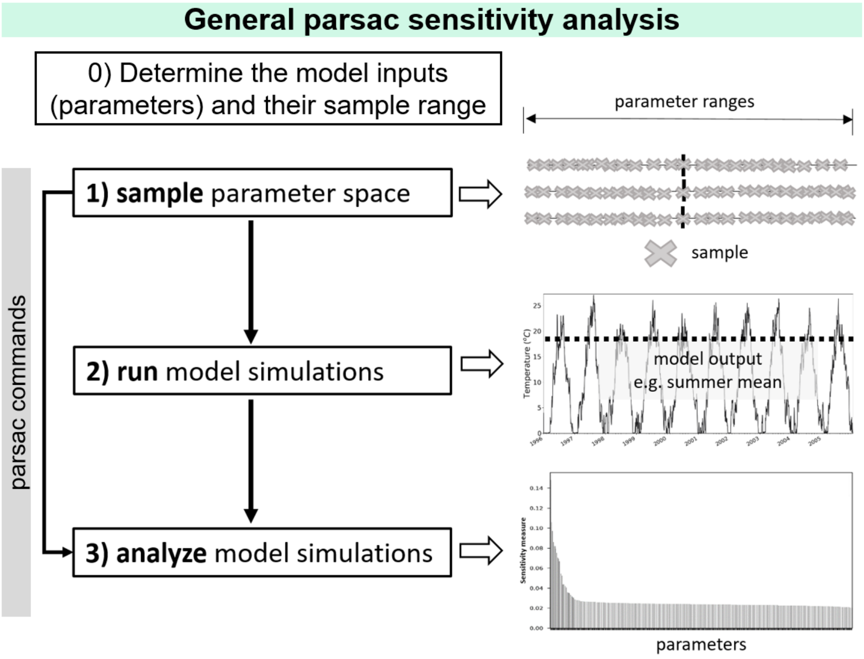
Sensitivity method	parsac name	**Sampling**	parsac name
Fourier Amplitude Sensitivity Test (FAST)	fast		fast
Random Balance Designs - Fourier Amplitude Sensitivity Test	rbd_fast	Latin hypercube sampling	latin
Method of Morris	morris		morris
Sobol Sensitivity Analysis	sobol	Saltelli's sampling	saltelli
Delta Moment-Independent Measure	delta	Latin hypercube sampling	latin
Derivative-based Global Sensitivity Measure	dgsm		
Fractional Factorial Sensitivity Analysis	ff		ff

Folder setup for parsac

Within your working model folder, you will need to create a folder named *parsac*. In the *parsac* folder you need to include a xml file with the SA configurations. Optionally, you can include bat files to execute the different parsac SA steps (see below).

Sensitivity analysis routine with parsac

The SA routine consists of three actions in parsac: 0) Determine the model inputs (parameters) and their sample range, 1) Run the sample function to generate the model inputs, 2) Evaluate the model using the generated inputs, saving the model outputs, 3) Run the analyze function on the outputs to compute the sensitivity indices:


General parsac sensitivity analysis
0) Determine the model inputs (parameters) and their sample range
parameter ranges
1) sample parameter space
sample
2) run model simulations
model output
e.g. summer mean
Temperature (°C)
3) analyze model simulations
Sensitivity measure
parameters
parsac commands

The first step in the SA procedure "Determine the model inputs (parameters) and their sample range" involves configuring the xml file for parsac, while the last three steps are executed by parsac.

Configuring parsac xml file

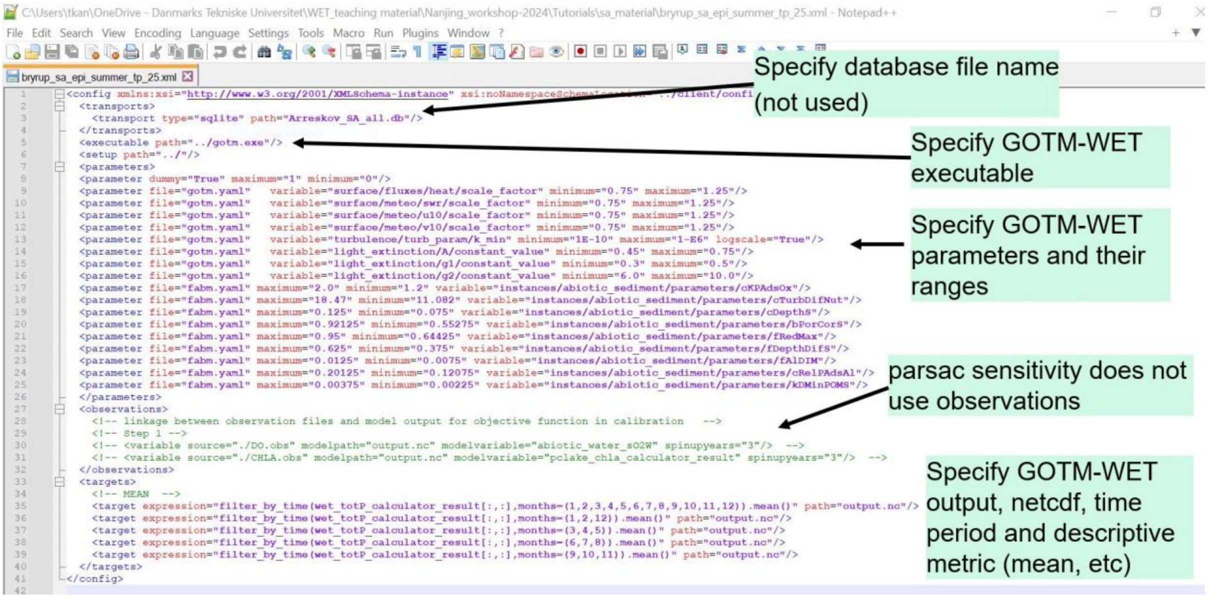
To determine which parameters are influential on specific model outputs, users must configure a xml file that provides a complete list of

1. Parameters and their value ranges, and
2. GOTM-WET output

Some ideas to determine parameter ranges used in the SA:

- % ranges from default values
- % ranges from calibrated values
- Min and max based on literature

Be aware that parsac or GOTM-WET does not check if parameter values are within theoretical and technical ranges. So, users can specify parameter values that are not valid.
As available sensitivity methods cannot handle time series model output, all specified GOTM-WET output must be reduced to a scalar value (for instance average, minimum or maximum values) across depth and time in the specified netcdf output.


C:\Users\tkan\OneDrive - Danmarks Tekniske Universitet\WET_teaching material\Nanjing_workshop-2024\Tutorials\sa_material\bryrup_sa_epi_summer_tp_25.xml - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
bryrup_sa_epi_summer_tp_25.xml

```
<config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="../client/confi
  <transports>
    <transport type="sqlite" path="Arreskov_SA_all.db"/>
  </transports>
  <executable path="../gotm.exe"/>
  <setup path="../"/>
  <parameters>
  <parameter dummy="True" maximum="1" minimum="0"/>
  <parameter file="gotm.yaml"   variable="surface/fluxes/heat/scale_factor" minimum="0.75" maximum="1.25"/>
  <parameter file="gotm.yaml"   variable="surface/meteo/swr/scale_factor" minimum="0.75" maximum="1.25"/>
  <parameter file="gotm.yaml"   variable="surface/meteo/u10/scale_factor" minimum="0.75" maximum="1.25"/>
  <parameter file="gotm.yaml"   variable="surface/meteo/v10/scale_factor" minimum="0.75" maximum="1.25"/>
  <parameter file="gotm.yaml"   variable="turbulence/turb_param/k_min" minimum="1E-10" maximum="1-E6" logscale="True"/>
  <parameter file="gotm.yaml"   variable="light_extinction/A/constant_value" minimum="0.45" maximum="0.75"/>
  <parameter file="gotm.yaml"   variable="light_extinction/g1/constant_value" minimum="0.3" maximum="0.5"/>
  <parameter file="gotm.yaml"   variable="light_extinction/g2/constant_value" minimum="6.0" maximum="10.0"/>
  <parameter file="fabm.yaml" maximum="2.0" minimum="1.2" variable="instances/abiotic_sediment/parameters/cKPAdsOx"/>
  <parameter file="fabm.yaml" maximum="18.47" minimum="11.082" variable="instances/abiotic_sediment/parameters/cTurbDifNut"/>
  <parameter file="fabm.yaml" maximum="0.125" minimum="0.075" variable="instances/abiotic_sediment/parameters/cDepthS"/>
  <parameter file="fabm.yaml" maximum="0.92125" minimum="0.55275" variable="instances/abiotic_sediment/parameters/bPorCorS"/>
  <parameter file="fabm.yaml" maximum="0.95" minimum="0.64425" variable="instances/abiotic_sediment/parameters/fRedMax"/>
  <parameter file="fabm.yaml" maximum="0.625" minimum="0.375" variable="instances/abiotic_sediment/parameters/fDepthDifS"/>
  <parameter file="fabm.yaml" maximum="0.0125" minimum="0.0075" variable="instances/abiotic_sediment/parameters/fAlDIM"/>
  <parameter file="fabm.yaml" maximum="0.20125" minimum="0.12075" variable="instances/abiotic_sediment/parameters/cRelPAdsAl"/>
  <parameter file="fabm.yaml" maximum="0.00375" minimum="0.00225" variable="instances/abiotic_sediment/parameters/kDMinPOMS"/>
  </parameters>
  <observations>
    <!-- linkage between observation files and model output for objective function in calibration   -->
    <!-- Step 1 -->
    <!-- <variable source="./DO.obs" modelpath="output.nc" modelvariable="abiotic_water_sO2W" spinupyears="3"/>  -->
    <!-- <variable source="./CHLA.obs" modelpath="output.nc" modelvariable="pclake_chla_calculator_result" spinupyears="3"/>  -->
  </observations>
  <targets>
    <!-- MEAN  -->
    <target expression="filter_by_time(wet_totP_calculator_result[:,:],months=(1,2,3,4,5,6,7,8,9,10,11,12)).mean()" path="output.nc"/>
    <target expression="filter_by_time(wet_totP_calculator_result[:,:],months=(1,2,12)).mean()" path="output.nc"/>
    <target expression="filter_by_time(wet_totP_calculator_result[:,:],months=(3,4,5)).mean()" path="output.nc"/>
    <target expression="filter_by_time(wet_totP_calculator_result[:,:],months=(6,7,8)).mean()" path="output.nc"/>
    <target expression="filter_by_time(wet_totP_calculator_result[:,:],months=(9,10,11)).mean()" path="output.nc"/>
  </targets>
</config>
```

Specify database file name (not used)

Specify GOTM-WET executable

Specify GOTM-WET parameters and their ranges

parsac sensitivity does not use observations

Specify GOTM-WET output, netcdf, time period and descriptive metric (mean, etc)

parsac SA commands
The corresponding parsac command lines, arguments and options are:

1. **"parsac sensitivity sample xmlfile info {fast,latin,morris,saltelli,ff} samplesize"**
   xmlfile	XML formatted configuration file (xml file)
   info	Path to output of the "sample" step (pickle file, .pkl)
   samplesize	Number of samples generated by sampling scheme

2. **"parsac sensitivity run info"**
   info	Path to output of the "sample" step (pickle file, .pkl)

3. **"parsac sensitivity analyze [-h] [--print_to_console]**
   [--select SELECT SELECT] [--pickle PICKLE] info
   {fast,rbd_fast,morris,sobol,delta,dgsm,ff}
   --num_resamples RESAMPLES"
   info	Path to output of the "sample" step (pickle file, .pkl)
   --print_to_console	Print results directly to console
   --select SELECT SELECT	This requires two values: N OUTPUTXML. Selects the N most sensitive parameters for a calibation run and save it to OUTPUTXML
   --pickle PICKLE	Path of pickle file to write with analysis results
   --num_resamples	The number of resamples when computing confidence intervals

Some considerations before running a SA

For SA, parsac functions as a wrapper and relies on the Python package SALib to sample and perform SA methods. So, we recommend users to familiarize themselves with best practices for SALib, which can be found on their website and publications. As SALib is a Python package, users can read, interact and manipulate SA results file created by parsac (.pkl file) via Python and thereby utilize other SALib functions if necessary.

1 - parsac sensitivity sample

Users specify sampling scheme corresponding to chosen SA method and points to the xml file with determined model parameters and ranges. As all forcing (i.e. weather and inflow) in GOTM can be controlled by a scale factor (for instance for wind speed), it is possible to include weather and inflow forcing in the SA.

Users must also specify sample size. Andersen et al. (2020) showed reliable SA results for a screening setting with Delta method for three different lakes with a sample size of 100. SA results are sensitive to sample size, therefore confidence intervals on calculated sensitivity indices should be assessed.

This step creates a .pkl file to store an array of sampled model parameterization with dimensions sample size rows and D columns, where D is the number of parameters, as well as a dictionary with arguments (for instance sample size, sample method and xml file path).

2 - parsac sensitivity run

In this step you run all model simulations with the model parameterizations specified in the pkl file. parsac appends an array with simple size for rows and model output for columns to the pkl file.

If you have installed parallel python package, parsac will execute model simulations in parallel.

3 - parsac sensitivity analyze

In this step, parsac computes the sensitivity indices by analyzing the sampled model parameterization and generated model output with the specified SA method.

We highly recommend including the argument “--pickle PICKLE”, so parsac will create a .pkl file with computed sensitivity indices. For example, a Delta SA result file will include a dictionary of all specified model outputs and their respective sensitivity indices and additional information:

parsac name		Type
delta	Delta indices	List
delta_conf	Confidence intervals on Delta indices	List
S1	Sobol first-order indices	List
S1_conf	Confidence intervals on Sobol first-order indices	List
names	Parameter string, as specified in xml	Tuples
Parameter	Parameter string, as specified in xml	Array

All with number of parameters as dimension and containing the indices in the same order as the parameter file (and names and Parameters).

Completed SA

After running “parsac sensitivity analyze” and have output sensitivity metrics to console or pkl file, it is now your job to understand the SA results and its importance for you lake model case study.