SIMSTRAT

1D k-epsilon lake model

**User Manual**

1. Introduction 2

2. Latest model changes up to version 2.0 2

Model set-up 3

2.1. Physical 3

3. Input files 4

3.1. Numerical 5

3.2. Physical 5

Morphology 5

Initial conditions 5

Forcing 6

Light attenuation 7

Inflow and outflow 7

3.3. Biochemical 9

Initial conditions 9

Inflow 10

4. Output 11

4.1. Physical 11

4.2. Biochemical 11

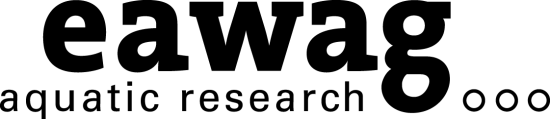
5. Parameter estimation 12

5.1. Introduction 12

5.2. Workflow 13

5.3. Set-up 13

5.4. Parallelization 14



Written by Adrien Gaudard in March 2015,

adapted to Version 2.0 by Fabian Bärenbold in September 2018

Swiss Federal Institute of  
Aquatic Science and Technology

# Introduction

SIMSTRAT is a model for the physical simulation of water reservoirs, including basin morphology, interaction with the atmosphere, inflow and outflow.

A reservoir is simulated as a 1D vertical water column that is horizontally averaged. The column is composed of a certain number of layers, the evolution of which is driven by the atmospheric forcing, allowing the parameterization of stratification, energy transfers, turbulence effects, seiches, etc. Physically, water velocities, turbulent kinetic energy and its dissipation rate (k-ε), temperature, salinity, seiche energy, stress and buoyancy are modeled.

The files being part of the model are the following:

* simstrat.exe Model executable file
* simstrat.par SIMSTRAT parameter file (name can be given as argument), which configures the simulation (see Table 1)

This document is a user manual. For an in-depth description of how the model works (governing equations, numerical schemes, parameterization, etc.), the reader is referred to the paper by Goudsmit, G‐H. et al. (2002): "Application of k‐ϵ turbulence models to enclosed basins: The role of internal seiches." in *Journal of Geophysical Research: Oceans (1978–2012)* 107.C12: 23-1. Further changes to the physical model (i.e. alterations that are not presented in this paper) are described in Chapter 2.

# Latest model changes up to version 2.0

After the publication of the above-referenced paper, a few modifications have been performed on the algorithms governing the physical model:

* The model tended to over-estimate wind-induced vertical mixing in winter (in non-stratified conditions), and to underestimate it during the stratified season. This is not surprising as one-dimensional models cannot account for horizontal gyres as well as two- or three-dimensional ones, and may give more energy to seiches than what really occurs in non-stratified conditions, when a basin is very difficult to excite vertically. It is now possible to feed the model with a time-series of pre-filtered wind, which will only be used for allotting seiche energy differently (equation (19) in the paper). For example, it has been found that reducing the wind when it is not sufficient to trigger seiches motion (the duration of the wind event is small when compared to oscillation period of the basin) helps towards better modeling of the thermocline seasonal behavior. This setting can be enabled in the parameter file.
* Improved parameterization of heat fluxes according to Schmid and Köster, 2016)
* Implementation of gravity driven inflows: one can let the inflow sink through the layers of the reservoir based on its density, entraining water with it and stopping when neutral buoyancy is reached. This can be particularly important because a one-dimensional model will first distribute an inflow across entire horizontal layers before it spreads vertically, therefore an arbitrary estimate of the inflow location can lead to great inaccuracies in compounds distribution and water column structure. This setting can be enabled in the parameter file.
* Implementation of surface bound in-/outflows: if the inflows are not gravity driven, on can either define them at a fixed spot in the morphology (i.e. subaquatic groundwater inflow) or let them vary with the water level (i.e. surface in- and outflows).
* Introduction of a ice/snow model by Love Raman Vinna (based on MyLake)

# Model set-up

## Physical

The model is run via its executable file, and is governed by a parameter file. The name of the parameter file can be given as first argument when calling the model executable; if nothing is given (or for example if the model is run with a double-click), then simstrat.par is the default (this will be the name used in the rest of this manual). This file specifies all input files, output locations, model settings and parameters. Table 1 shows an explanation of this file which is in JSON format. The model parameters (last part of Table 1) are better described in the above-referenced paper.

|  |  |  |
| --- | --- | --- |
| **JSON key** | **Description** | **Typical value** |
| Input |  |  |
| Initial conditions | Path to initial conditions file |  |
| Grid | Path to grid file / vector of grid / grid resolution |  |
| Morphology | Path to morphology file |  |
| Forcing | Path to forcing file |  |
| Absorption | Path to light attenuation file |  |
| Inflow | Path to inflow file |  |
| Outflow | Path to outflow file |  |
| Inflow temperature | Path to temperature inflow file |  |
| inflow Salinity | Path to salinity inflow file |  |
| Output |  |  |
| Path | Path result folder (is created if non-existant) |  |
| OutputDepthReference | 1: Lake bottom, 2: Lake water table |  |
| Depths | Path to file / vector of depths / output depth resolution |  |
| Times | Path to file / vector of times / output time resolution |  |
| ModelConfig |  |  |
| MaxLengthInputData |  | 1000 |
| CoupleAED2 | Biogeochemistry model (0:off, 1:on) | 0 |
| TurbulenceModel | Turbulence model (1:k-ε, 2:M-Y) | 1 |
| StabilityFunction | Stability function (1:constant, 2:quasi-equilibrium) | 2 |
| FluxCondition | Flux condition (0:Dirichlet condition, 1:no-flux) | 1 |
| Forcing | Forcing (1:Wind+Temp+SolRad, 2:Wind+Temp+SolRad+VapP, 3:Wind+Temp+SolRad+VapP+Cloud, 4:Wind+HeatFlux+SolRad) | 3 |
| UseFilteredWind | Use filtered wind to compute seiche energy (if “true”, one more column is needed in forcing file) | false |
| SeicheNormalization | Seiche normalization (1:max N^2, 2:integral) | 2 |
| WindDragModel | Wind drag model (1:lazy (constant), 2:ocean (increasing), 3:lake (Wüest and Lorke 2003)) | 3 |
| InflowPlacement | Inflow placement (0/default:manual, 1:density-driven) | 0 |
| PressureGradients | Pressure gradients (0/default:off, 1: Svensson 1978, 2:?) | 0 |
| IceModel | 0: off, 1: on | 0 |
| SnowModel | 0: off, 1: on (needs an additional column in the forcing file: precipitation) | 0 |
| Simulation |  | 1 |
| Timestep s | Simulation timestep in seconds | 100 |
| Start d | Start time in days | 0 |
| End d | End time in days | 10000 |
| DisplaySimulation | Display in terminal (0: off, 1:when data is saved, 2: at every iteration | 1 |
| ModelParameters |  |  |
| lat | Latitude for Coriolis parameter [°] | 47 |
| p\_air | Air pressure [mbar] | 960 |
| a\_seiche | Fraction of wind energy to seiche energy [-] | 0.01 |
| q\_nn | Fit parameter for distribution of seiche energy [-] | 1.00 |
| f\_wind | Fraction of forcing wind to wind at 10m [-] | 1.00 |
| c10 | Wind drag coefficient (a physical constant around 0.001 if wind drag model is 1; a calibration parameter around 1 if wind drag model is 2 or 3) [-] | 0.001 / 1 |
| cd | Bottom drag coefficient [-] | 0.002 |
| hgeo | Geothermal heat flux [W/m2] | 0.10 |
| k\_min | Minimal value for TKE [J/kg] | 1e-15 |
| p\_radin | Fit parameter for absorption of IR radiation from sky [-] | 1.00 |
| p\_windf | Fit parameter for convective and latent heat fluxes [-] | 1.00 |
| beta\_sol | Fraction of short-wave radiation directly absorbed as heat by water [-] | 0.30 |
| beta\_snowice | Fraction of short-wave radiation directly absorbed as heat by snow and ice [-] |  |
| albsw | Albedo for reflection of short-wave radiation on water [-] | 0.08 |
| ice\_albedo | Albedo for reflection of short-wave radiation on ice [-] |  |
| snow\_albedo | Albedo for reflection of short-wave radiation on snow [-] |  |
| freez\_temp | Freezing temperature of water [°C] |  |

Table 1 – simstrat.par file

# Input files

The input files are opened and read by the model while it is running. For all these files, the given depths must be within the limits set in the lake morphology (depth is zero at the surface and negative as it decreases downwards), while the given times must fall in the frame set by the simulation start and end time. In files where a series of values is required, depths have to decrease monotonously while times have to increase monotonously.

Throughout the simulation, the given values will be linearly interpolated (in depth and time) to obtain values at the coordinates needed by the model. If these coordinates are outside the given range, the value of the nearest neighbour is used. The model does not tolerate missing values. The files can have an arbitrary extension but must be text files.

## Numerical

The entry given to to the json key “Input.Grid” can either be a string (path to a file), a vector containing the grid points (meaning the borders or faces of the grid layers) or a value specifying the number of grid points. If a path is given, the file can contain again either all the values which define the model grid (mostly used for variable grid spacing) or a number specifying the number of grid points. If the grid points are specified, one needs to make sure to include the top and bottom values as defined in the morphology file otherwise an error occurs and the simulation aborts.

The Output.Depths key specifies at which depths the model results will be written. It can either be a string (path to a file), a vector containing the output depths in [m] or a value specifying output resolution in [m]. If a path to a file is given, this file can again contain either all output depths in [m] or an output resolution in [m]. The key “Output.OutputDepthReference” indicates whether the output depths should be interpreted as absolute height above sediment (value is 1) or as depth below water level (value is 2). If the reference is 1, depths have to be given as negative depths below water table. Conversely, if it is 2, depths have to be given as positive depths above sediment.

The Output.Times key specifies at which times the model results will be written. It can either be a string (path to a file), a vector containing the output times in [days] or a value specifying output time resolution in [5 min] units. If a path to a file is given, this file can again contain either all output times in [days] or the resolution.

## Physical

### Morphology

The key “Input.Morphology” specifies the shape of the basin by giving its surface area (positive) at various depths. The values should cover at least the entire depth range of the reservoir: from the initial surface (0 m depth) to bottom (ideally 0 m2 surface area). During the simulation, water level will not be allowed to rise above the depth of the first given value which can be 0 or any positive number for which one knows the surface area.

The first line of the file is a header, the next lines are the input: depths [m] in the first column, surface areas [m2] in the second column. An example of this file:

z [m] Area [m2]

0 500000

-5 450000

-10 410000

-20 332000

-40 175000

-50 100000

-60 0

### Initial conditions

The “Input.Initial conditions” key specifies the state of the water column at simulation start time. Depth-dependent values for several variables can be given. Having initial conditions that are close to reality help the model to reach a physically consistent state faster. The depth values in the first column refer to the depth values in the morphology file. The first depth value is taken to be the initial water level of the reservoir (i.e. if -3 is chosen, the initial water level is set 3 meters below the 0 in the morphology file). The initial data values are extrapolated to the maximum depth in the morphology file in case not all the depths are given in the initial data file.

The first line of the file is a header, the next lines are the input: depths [m] in the first column, initial conditions in columns 2 to 7 (horizontal velocity East U [m/s], horizontal velocity North V [m/s], temperature T [°C], salinity S [‰], turbulent kinetic energy k [J/kg] and its dissipation rate ε [W/kg]). An example of this file:

z [m] U [m/s] V [m/s] T [°C] S [‰] k [J/kg] eps [W/kg]

0 0.00 0.00 9.3 0.13 3e-06 5e-10

-5 0.00 0.00 9.2 0.13 3e-06 5e-10

-10 0.00 0.00 7.2 0.13 3e-06 5e-10

-15 0.00 0.00 5.2 0.13 3e-06 5e-10

-20 0.00 0.00 5.2 0.14 3e-06 5e-10

-40 0.00 0.00 5.1 0.14 3e-06 5e-10

-60 0.00 0.00 4.9 0.14 3e-06 5e-10

-100 0.00 0.00 4.7 0.15 3e-06 5e-10

### Forcing

The forcing file specifies the atmospheric conditions to be applied at the reservoir surface throughout the simulation. At various times (in days), several parameters are specified, depending on the forcing mode chosen by the key “ModelConfig.Forcing”.

The first line of the file is a header, the next lines are the input: the structure of the columns is shown in Table 3.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Forcing mode | Column | | | | | | | | | |
| 1 | 2 | 3 | | 4 | | 5 | 6 | 7 | 8 |
| **1** | Time [d] | Wind speed East [m/s] | | Wind speed North [m/s] | | Water surface temperature [oC] | Solar radiation [W/m2] |  |  |  |
| **2** | Time [d] | Wind speed East [m/s] | Wind speed North [m/s] | | Air temperature [oC] | | Solar radiation [W/m2] | Vapor pressure [mbar] |  | (Precipitation [m/h]) |
| **3** | Time [d] | Wind speed East [m/s] | Wind speed North [m/s] | | Air temperature [oC] | | Solar radiation [W/m2] | Vapor pressure [mbar] | Cloud cover [-] | (Precipitation [m/h]) |
| **4** | Time [d] | Wind speed East [m/s] | Wind speed North [m/s] | | Heat flux [W/m2] | | Solar radiation [W/m2] |  |  |  |

Table 3 – Structure for forcing file

If the use of filtered wind is enabled, one more column has to be added after the standard ones. It contains the filtered wind speed [m/s] (norm value). If the snow module is enabled (not necessary for ice!), precipitation data has to be added at the end (only possible for forcing modes 2 and 3).

An example of this file (with forcing mode “3” and without filtered wind or snow):

t [d] U [m/s] V [m/s] T [°C] Sol [W/m2] Vap [mbar] Cloud [-]

36556.0000 -0.87 -1.69 7.00 0.00 7.10 0.80

36556.0417 -0.98 2.41 7.20 0.00 7.10 0.46

36556.0833 3.80 -0.17 7.40 1.00 7.10 0.46

36556.1250 3.04 -2.90 7.50 0.00 7.10 0.46

36556.1667 5.20 2.99 7.40 0.00 7.10 0.40

36556.2083 3.47 1.99 6.90 0.00 7.10 0.65

36556.2500 -1.83 3.22 6.90 0.00 7.10 0.65

36556.2917 -0.91 3.79 7.00 0.00 7.00 0.55

36556.3333 -2.05 -2.84 7.30 26.00 7.00 0.20

36556.3750 4.76 1.16 8.20 189.00 6.80 0.10

### Light attenuation

The light absorption file specifies the attenuation coefficient of solar radiation as a function of depth and time. Here, the zero depth always represents the water surface (even if its absolute position varies during the simulation).

The first line of the file is a header, the second line gives the number of depths for which the attenuation coefficient is specified (say n), the third line represents these depths (with the first number being a dummy value for display), the next lines are the input: times [d] in the first column, attenuation coefficients [m-1] in columns 2 to n+1. An example of this file:

t (1.column) z (1.row) Abs [m-1]

2

-1 0 -5

0 0.200 0.300

2130 0.212 0.331

2260 0.177 0.198

2390 0.667 0.668

10000 0.700 0.750

In this example, the light absorption coefficient on day 0 would be 0.2 m-1 at the surface, then linearly increase to 0.3 m-1 at 5 m depth, and remain constant below this depth.

### Inflow and outflow

Four files define the flows entering and coming out of the simulated reservoir, as a function of depth and time: water inflow, water outflow, temperature input and salinity input. Their contents represent a different physical quantity, but their structure is similar.

Depending on the setting for the key “ModelConfig.InflowPlacement” inflow placement the files will be read differently:

* Manual inflow placement of deep and surface in-/outflow

The values in the files must be given for a range of depths on a per-meter basis (Q/h), as they will be integrated over depth by the model. Water inflow values must be positive, water outflow values must be negative. Temperature and salinity input can be either, as it can be used as an independent source or sink. In order to specify temperature (resp. salinity) of the inflowing water, the given values must be the product of the water inflow (as in the water inflow file) and the inflow temperature (resp. salinity), and thus be positive. In addition, the depths and times must match.

The first line of the file is a header, the second line gives the number of deep inflows (the ones that don’t move with the water level) and surface inflows (the ones that move with the water level). The third line represents these depths (with the first number being a dummy value for display), the next lines are the input: times [d] in the first column, values (water inflow [m2/s], water outflow [m2/s], temperature input [°Cm2/s] or salinity input [‰m2/s]) in columns 2 to nval+1.

* Density-driven inflow placement

Only the first two columns (starting on line 4) of the inflow files will be read. For the outflow file, there is no difference to the case of manual inflow placement. The first three lines of the file are ignored, the next lines are the input: times [d] in the first column, values (water inflow [m3/s], water outflow [m2/s], inflow temperature [°C] or inflow salinity [‰]) in the second column.

An example of the water inflow file (left: manual inflow placement, right: density-driven inflow placement) for equal total inflow:

t (1.column) z (1.row) InflowQ [m2/s]

3 2

-1 -10.0 -5.0 0.0 -2.0 0.0

3084 0.000 0.425 0.425 1 1

3098 0.000 0.515 0.515 1 1

3112 0.000 0.280 0.280 1 1

t [d] InflowQ [m3/s]

1

-1 -1

3084 5.1875

3098 5.8625

3112 4.1000

An example of the water outflow file (for a neutral water balance with the inflow given above):

t (1.column) z (1.row) OutflowQ [m2/s]

3 2

-1 -10.0 -5.0 0.0 -2.0 -0

3084 0.000 -0.425 -0.425 -1 -1

3098 0.000 -0.515 -0.515 -1 -1

3112 0.000 -0.280 -0.280 -1 -1

An example of the temperature input file (left: manual inflow placement, right: density-driven inflow placement), for a deep inflow at a temperature of 5°C and a surface inflow at 10°C with the inflow given above:

t (1.column) z (1.row) InflowT [°C\*m2/s]

3 2

-1 -10.0 -5.0 0.0 -2.0 -0.0

3084 0.000 2.125 2.125 10 10

3098 0.000 2.575 2.575 10 10

3112 0.000 1.400 1.400 10 10

t [d] InflowT [°C]

1

-1 -1

3084 5

3112 5

An example of the water inflow file (left: manual inflow placement, right: density-driven inflow placement), for an inflow at a salinity of 0.2‰ (both deep and surface inflows) with the inflow given above:

t (1.column) z (1.row) InflowS [‰\*m2/s]

3 2

-1 -10.0 -5.0 0.0 -2.0 -0.0

3084 0.000 0.085 0.085 0.2 0.2

3098 0.000 0.103 0.103 0.2 0.2

3112 0.000 0.056 0.056 0.2 0.2

t [d] InflowS [‰]

1

-1 -1

3084 0.2

3112 0.2

If the four files are empty, the calculation of vertical advection is deactivated. This is in particular useful in case inflows are negligible for the dynamics of the reservoir.

# Output

## Physical

The output is written to a separate text file for each output variable and stored in the location defined by “Output.Path”. If the output folder does not exist, it will be created automatically. The files are named according to the variable they contain var\_out.dat, where var is the short name of the variable (see Table 4).

|  |  |  |  |
| --- | --- | --- | --- |
| Short name | Description | Grid | Units |
| U | Water velocity (East direction) | Volume | m/s |
| V | Water velocity (North direction) | Volume | m/s |
| T | Temperature | Volume | °C |
| S | Salinity | Volume | ‰ |
| Qv | Vertical advection | Face | m3/s |
| k | Turbulent kinetic energy | Face | J/kg |
| eps | Dissipation rate of turbulent kinetic energy | Face | W/kg |
| nuh | Turbulent diffusivity | Face | J·s/kg |
| N2 | Brunt-Väisälä frequency (stratification coefficient) | Face | s-2 |
| B | Production rate of buoyancy | Face | W/kg |
| P | Production rate of shear stress | Face | W/kg |
| Ps | Production rate of seiche energy | Face | W/kg |
| H\_A | Long-wave radiation from sky | - | W/m2 |
| H\_W | Long-wave radiation from water | - | W/m2 |
| H\_K | Sensible heat flux | - | W/m2 |
| H\_V | Latent heat flux | - | W/m2 |
| Rad0 | Solar radiation penetrating lake | - | W/m2 |
| Ice\_h | Ice thickness on lake | - | m |
| Snow\_h | Snow height above ice | - | m |
| Water\_depth | Water depth (positive height above sediment) | - | m |

Table 4 – current Simstrat Output variables

# Parameter estimation

## Introduction

Parameter estimation is performed through the software package PEST[[1]](#footnote-1), which allows state-of-the-art model calibration and uncertainty analysis. More information about how the software works can be found in the PEST User Manual. In order to install PEST, one first has to download the archive containing all required files, and unzip it. The path to this directory must then be added to the PATH environment variable.

PEST requires several inputs that configure the parameter estimation for SIMSTRAT:

* A control file (keps\_calib.pst) that specifies the parameter estimation setup: optimization settings, parameter values and ranges, field data, references to the other files, etc.
* A template file (keps\_par.tpl) that mimics the kepsilon.par file, but with parameter names instead of values. Throughout the optimization process, PEST will fill it in with the values it wants and provide this new file to the model.
* A batch file (model.bat) which takes care of the execution of the model, including necessary preparation and post-processing.
* Field data that can be used to calibrate the model (e.g. temperature profiles).
* Model-generated data that can be directly related to the field data (i.e. at the same times and depths). At each iteration, PEST executes a batch file that will run the model and write a file with the required data in a single text file (ModelObs.dat), for subsequent comparison with field data.
* An instruction file (keps\_obs.ins) that tells PEST how to read the text file of model-generated data.

The MATLAB script Control.m writes the control file, based on given matrices (time and depth) of measurements and based on a set of parameters and corresponding properties (group, type, initial value, minimum and maximum). The control file can then be edited manually, but permanent changes should be made in the script.

The MATLAB script OutputInstructions.m writes the text file of model-generated data, based on given grid points (time and depth) that correspond to those of the actual measurements. It also writes the corresponding instruction file.

Typically, all these files can be located in a PEST working directory, which also contains a location for the model to write the results throughout parameter estimation. PEST will then create many other files.

## Workflow

When running, PEST uses the template file keps\_par.tpl to create the configuration file kepsilon\_PEST.par (filled with the parameters it wants to use) and copies the latter to the appropriate given location. It then launches the model batch file model.bat. This file will make sure the model results are written within the current PEST directory (instead of overwriting other files), and then run the kepsilon model with the previously-created configuration file. When completed, it will call the MATLAB script OutputInstructions.m to rewrite the relevant results (i.e. those that can be compared to field data) into a single file ModelObs.dat, and to write an instruction file keps\_obs.ins that explains how to read this. PEST then compares field and model observations to calculate the objective function, and then iterates to attempt minimizing this objective function (towards a better fit between model and reality).

## Set-up

It is recommended that parameter estimation should be completely independent of simple model runs, and thus use different configuration and output files (ideally at different locations). All PEST-related files, including the configuration file ‘kepsilon.par’ that PEST will be modifying and feeding the model, can then be for example placed in a sub-directory of the model, which could also contain a place for model results. This logic has been applied in the given file structure.

In order to set-up a parameter estimation procedure, the first step is to write a control file. The PEST user manual provides a thorough description of all the available settings.

If the MATLAB script Control.m is used, one first has to provide measurement (field) data in the first lines of the MATLAB script Control.m, in the same way as exemplified. The script will then identify each measurement with a unique name and write them line by line. Thereafter, in the following lines, the properties of the parameter set must be specified. Each parameter has a name, type (‘none’ if to be optimized, ‘fixed’ otherwise), limit type (‘factor’ or ‘relative’), initial value, minimum, maximum and group (‘none’ if the parameter should never be optimized). The script should be run with no error or warning.

Secondly, the template file keps\_par.tpl must be present and accessible to PEST, as well as the model batch file model.bat (both are referenced near the end of the control file).

Optionally, it can be verified that the text file of model-generated data and the corresponding instruction file get written correctly. For that, the MATLAB script OutputInstructions.m can be run once (no change should be made), assuming that the configuration file kepsilon\_PEST.par is already available (as referenced near the end of the control file) and that complete result files are present (at the location specified in this configuration file).

Finally, the “pestchek” command can be run, with the name of the control file as argument, to check for mistakes in the PEST set-up.

In order to launch parameter estimation, one has to run the “pest” command, with the control file as argument. For example, if the command is run in the same directory as the control file keps\_calib.pst: “pest keps\_calib”. PEST will then start running the model several times and attempt optimization with the provided settings. Once optimization is complete, the optimized parameter set can be found in the same folder in the file with a “.par” extension.

## Parallelization

PEST can run in parallelized mode and operate much faster (using several CPUs, or several computers). In addition to all the files specified in the previous section, this requires a run management file keps\_calib.rmf (same name as the control file, but different extension) which specifies the available threads (or slaves) and the working directories of each of them. Every additional slave can for example be assigned a sub-directory of the main PEST directory (which also hosts a slave). They should then use independent configuration files, simulation batch files, model observation files and result directories.

In order to launch parameter estimation in parallelized mode, one first has to run the command “pslave” in the working directory of each of the slaves that should be used, and specify the model batch file when prompted. Then, one has to run the “ppest” command, with the control file as argument, in the main PEST directory. For example, if the command is run in the same directory as the run management file keps\_calib.rmf and control file keps\_calib.pst: “ppest keps\_calib”. PEST will then start running the model several times and attempt optimization with the provided settings. Once optimization is complete, the optimized parameter set can be found in the same folder in the file with a “.par” extension.

1. http://www.pesthomepage.org/ [↑](#footnote-ref-1)