Simstrat

1D k-epsilon lake model

Manual

eawag
aquatic research

May 2017

Swiss Federal Institute of
Aquatic Science and Technology

1. 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 parametrization 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:

• kepsmodel.exe	Model executable file
• [kepsilon].par	Simstrat parameter file (name can be given as argument), which configures the simulation (see Table 2)
• GetResults.R	R function for extraction of physical results

This document is a user manual. For an in-depth description of how the model works (governing equations, numerical schemes, parametrization, etc.), the reader is referenced 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.

2. Latest model changes

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 unstratified 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 unstratified 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 modelling of the thermocline seasonal behaviour. This setting can be enabled in the parameter file.
- Gravity-driven inflow is now available: 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.

3. Simstrat: code structure and installation

3.1. Code structure

The code of the model is organized as follows:

---

kepsmodel_2014.f90

program **keps**

Calls to **Initialization**, **Form** and **keps_simulation**

subroutine **keps_simulation**

Simulation main loop, call to all the functions

subroutine **Tridiagonal**

Tridiagonal matrix algorithm (solver)

subroutine **Coriolis**

Integration of the Coriolis force in the (x,y) velocities

subroutine **uvEquation**

Solving the advection-diffusion equations for the (x,y) velocities

subroutine **Buoyancy**

Calculation of the stratification coefficient

subroutine **Temperature**

Solving the temperature equation (transport, radiation, heat flux)

subroutine **TransportEquation**

Solving the compound equation (transport, source/sink)

subroutine **cmue_cn** and **cmue_qe**

Calculation of model parameters (cmue)

subroutine **StabilityFunctions**

Call to **cmue_cn** or **cmue_qe**

subroutine **Seiche**

Distributing the wind forcing into seiche energy

subroutine **Production**

Calculating shear stress and buoyancy energy production

subroutine **TKE**

Solving the turbulent kinetic energy (k) equation

subroutine **Dissipation**

Solving the TKE dissipation rate (ε) equation

subroutine **Advection**

Integration of the inflows and outflows to the model state subroutine **Form**

Calculation of the morphologic parameters of the basin

---

keps_initialization.f90

subroutine **Initialization**

Model initialization, calls to **ParameterList**, **Morph** and **InitCond**

subroutine **ParameterList**

Parameter initialization based on kepsilon.par

subroutine **Morph**

Morphology initialization based on the morphology file

subroutine **Grid**

Vertical grid initialization based on the grid file

subroutine **InitCond**

Applying the initial conditions based on the initial conditions file

subroutine **save_ini**

Preparation of the output grid based on the output depths and output times files

subroutine **check_advection**

Disabling of advection if all four inflow/outflow files are empty

---

keps_utilities.f90

subroutine **Interp**

Linear interpolation (nearest neighbour for values out of range)

subroutine **Interp_nan**

Linear interpolation (NaN for values out of range)

subroutine **Integrate**

Trapezoidal integration

subroutine **Forcing**

Calculation of all the relevant surface forcing parameters

subroutine **ReadForcing**

Reading the forcing file to obtain current values

subroutine **Absorption**

Reading the absorption file to obtain current values

subroutine **Lateral**

Reading the inflow/outflow files to obtain current values

subroutine **Lateral_rho**

Reading the inflow/outflow files to obtain current values, assuming that inflow placement is gravity-driven (not manual)

3.2. External files

During a simulation, the model opens several files for reading and writing. Until it is closed, each open file is associated to a user-defined file ID (unit). Direct output to the screen is associated to unit 6 on most platforms. A list of the used files is given in Table 1.

File ID	File description	Reference	Use
10	Parameter	First command argument, or 'kepsilon.par' by default	Read once
11	Morphology	Parameter file, line 4	Read once
12	Grid	Parameter file, line 3	Read once
13	Initial conditions	Parameter file, line 2	Read once
15	Output depths	Parameter file, line 8	Read once
15	Output times	Parameter file, line 9	Read once
20	Surface forcing	Parameter file, line 5	Read continuously
30	Light absorption	Parameter file, line 6	Read continuously
41	Water inflow	Parameter file, line 10	Read continuously
42	Water outflow	Parameter file, line 11	Read continuously
43	Temperature input	Parameter file, line 12	Read continuously
44	Salinity input	Parameter file, line 13	Read continuously
80	Physical output (binary)	Parameter file, line 7	Write continuously
81-93	Physical output (text files)	Path: parameter file, line 7	Write continuously

Table 1 – External files used by the model

3.3. Obtaining a FORTRAN compiler

The GNU Fortran compiler gfortran can be used. It can be obtained as part of the MinGW-w64 project at http://sourceforge.net/projects/mingw-w64/files/, through the installer executable. Once mingw64 is installed, the path to its "bin" directory (inside the installation folder) must be added to the PATH environment variable of the user.

3.4. Compiling Simstrat

Using gfortran, Simstrat can be compiled at the command line.

As an alternative, Code::Blocks is a text-editor and IDE compatible that supports Fortran. It can be obtained at http://www.codeblocks.org/downloads/26#windows (installer executable). When Code::Blocks is opened for the first time, you have to select the compiler you want to use; choose gfortran.exe in the "bin" directory of the mingw64 installation folder. Open the main source file "kepsmodel_2016.f90" from the Simstrat source code. Clicking "build" compiles the current code and generates the model executable "kepsmodel_2016.exe".

4. Model set-up

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 kepsilon.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 2 shows an explanation of this file; its lines should be written in the same structure and order as given. The model parameters (last part of Table 2) are better described in the above-referenced paper.

Line in kepsilon.par	**Description**	**Typical value**
1	Comment line	
2	Path to initial conditions file	
3	Path to grid file	
4	Path to morphology file	
5	Path to forcing file	
6	Path to light attenuation file	
7	Path to physical output file or directory	
8	Path to output depths file	
9	Path to output times file	
10	Path to inflow file	
11	Path to outflow file	
12	Path to temperature inflow file	
13	Path to salinity inflow file	
14	(not read)	
15	(not read)	
16	Comment line	
17	Timestep [s]	300
18	Initial simulation time [d]	0
19	Final simulation time [d]	10000
20	Comment line	
21	Turbulence model (1:k-ε, 2:M-Y)	1
22	(not read)	
23	Stability function (1:constant, 2:quasi-equilibrium)	2
24	Flux condition (0:Dirichlet condition, 1:no-flux)	1
25	Forcing (1:Wind+Temp+SolRad, 2:Wind+Temp+SolRad+VapP, 3:Wind+Temp+SolRad+VapP+Cloud, 4:Wind+HeatFlux+SolRad)	3
26	Use filtered wind to compute seiche energy (0:off, 1:on) (if 1:on, one more column is needed in forcing file)	0
27	Seiche normalization (1:max N^2, 2:integral)	2
28	Wind drag model (1:lazy (constant), 2:ocean (increasing), 3:lake (Wüest and Lorke 2003))	3
29	Inflow placement (0/default:manual, 1:density-driven)	0
30	Pressure gradients (0/default:off, 1: Svensson 1978, 2:?)	0
31	Enable salinity transport (0:off, 1:on)	1
32	Display simulation (0:off, 1:when data is saved)	1
33	Display diagnose (0:off, 1:all settings, 2:all settings and forcing each iteration)	0
34	Averaging data	10
35	Comment line	
36	Latitude for Coriolis parameter [°]	47

37	Air pressure [mbar]	960
38	Fraction of wind energy to seiche energy [-]	0.01
39	Fit parameter for distribution of seiche energy [-]	1.00
40	Fraction of forcing wind to wind at 10m [-]	1.00
41	Wind drag coefficient (used if wind drag model is 1) [-]	0.001
42	Bottom drag coefficient [-]	0.002
43	Geothermal heat flux [W/m2]	0.10
44	Minimal value for TKE [J/kg]	1e-15
45	Fit parameter for absorption of IR radiation from sky [-]	1.00
46	Fit parameter for convective and latent heat fluxes [-]	1.00
47	Fraction of short-wave radiation directly absorbed as heat [-]	0.30
48	Albedo for reflection of short-wave radiation [-]	0.08

Table 2 – kepsilon.par file

5. 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 (lines 18-19 in kepsilon.par). In files where a series of values is required, depths have to decrease monotonically while times have to increase monotonically.

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.

5.1. Numerical

The grid file (given in kepsilon.par, line 3) specifies the location of the grid points for the numerical discretization of the vertical water column. If only one value is given, this value is taken to be the maximal number of grid points, which will then be linearly distributed from the maximum to the minimum depth. If several values are given, these values are taken to be the absolute depths (in meters) at which grid points are set. Surface and bottom points are included in the grid by default.

The output depths file (given in kepsilon.par, line 8) specifies at which depths the model results will be written. If only one value is given (say n), the results will be written every n depth point, starting from the uppermost point. The output depths then depend on the vertical discretization defined in the grid file. If several values are given, these values are taken to be the absolute depths (in meters) at which the results have to be written.

The output times file (given in kepsilon.par, line 9) specifies at which times the model results will be written. If only one value is given (say n), the results will be written every n timestep, starting from the simulation start time. The output depths then depend on the time settings defined in kepsilon.par, lines 17-18. If several values are given, these values are taken to be the absolute times (in days) at which the results have to be written.

5.2. Physical

Morphology

The morphology file (given in kepsilon.par, line 4) 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 surface (0 m depth) to bottom (ideally 0 m^2 surface area). During the simulation, water level will not be allowed to rise above the depth of the first given value (overflow).

The first line of the file is a header, the next lines are the input: depths [m] in the first column, surface areas [m^2] 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 initial conditions file (given in kepsilon.par, line 2) 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 helps the model to reach a physically consistent state faster. The first depth value is taken to be the initial water level of the reservoir.

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 (given in kepsilon.par, line 5) 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 in kepsilon.par, lines 25-26.

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.

Line in kepsilon.par	Column						
	1	2	3	4	5	6	7
1	Time [d]	Wind speed East [m/s]	Wind speed North [m/s]	Water surface temperature [^{o}C]	Solar radiation [W/m^2]		
2	Time [d]	Wind speed East [m/s]	Wind speed North [m/s]	Air temperature [^{o}C]	Solar radiation [W/m^2]	Vapor pressure [mbar]	
3	Time [d]	Wind speed East [m/s]	Wind speed North [m/s]	Air temperature [^{o}C]	Solar radiation [W/m^2]	Vapor pressure [mbar]	Cloud cover [-]
4	Time [d]	Wind speed East [m/s]	Wind speed North [m/s]	Heat flux [W/m^2]	Solar radiation [W/m^2]		

Table 3 – Structure for forcing file

If the use of filtered wind is enabled (line 26 in kepsilon.par), one more column has to be added after the standard ones. It contains the filtered wind speed [m/s] (norm value).

An example of this file (with, in kepsilon.par, '3' on line 25 and '0' on line 26):

```
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 attenuation file (given in kepsilon.par, line 6) 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 attenuation coefficent 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 (given in this order in kepsilon.par, lines 10-13). Their contents represent a different physical quantity, but their structure is similar.

Depending on the setting for inflow placement (given in kepsilon.par, line 29), the files will be read differently:

- Manual inflow placement
  The values in the files must be given for a range of depths on a per-meter basis, 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 depths for which the values are 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, values (water inflow [m^2/s], water outflow [m^2/s], temperature input [°Cm^2/s] or salinity input [‰m^2/s]) in columns 2 to n+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 [m^3/s], water outflow [m^2/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
-1      -10.0   -5.0    0.0
3084    0.000   0.425   0.425
3098    0.000   0.515   0.515
3112    0.000   0.280   0.280
```

```
t [d]   InflowQ [m3/s]
1
-1      -1
3084    3.1875
3098    3.8625
3112    2.1000
```

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
-1      -10.0   -5.0    0.0
3084    0.000   -0.425  -0.425
3098    0.000   -0.515  -0.515
3112    0.000   -0.280  -0.280
```

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

```
t (1.column)    z (1.row)       InflowT [°C*m2/s]
3
-1      -10.0   -5.0    0.0
3084    0.000   2.125   2.125
3098    0.000   2.575   2.575
3112    0.000   1.400   1.400
```

```
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‰ with the inflow given above:

```
t (1.column)   z (1.row)        InflowS [‰*m2/s]
3
-1        -10.0    -5.0     0.0
3084      0.000    0.085    0.085
3098      0.000    0.103    0.103
3112      0.000    0.056    0.056
```

```
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.

6. Output

The model can write the physical results either in a single binary file, or in separate text files. If the output location (given in kepsilon.par, line 7) is a file, then the model will write it in binary form, whereas if it is a directory, the model will write a text file per variable there. In the latter case, 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	Units
U	Water velocity (East direction)	m/s
V	Water velocity (North direction)	m/s
T	Temperature	°C
S	Salinity	‰
Qv	Vertical advection	m^3/s
k	Turbulent kinetic energy	J/kg
eps	Dissipation rate of turbulent kinetic energy	W/kg
nuh	Turbulent diffusivity	J·s/kg
N2	Brunt-Väisälä frequency (stratification coefficient)	s^{-2}
B	Production rate of buoyancy	W/kg
P	Production rate of shear stress	W/kg
Ps	Production rate of seiche energy	W/kg
Es	Seiche energy (one value for the whole water column)	J

Table 4 – Simstrat physical variables

Calling the R function or GetResults.R, the results can be extracted in data matrices and plotted for a given time period and depth points.

7. Parameter estimation

7.1. Introduction

Parameter estimation is performed through the software package PEST[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 R script Control.R 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 R script OutputInstructions.R 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.

7.2. 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 R script OutputInstructions.R 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.

[1] http://www.pesthomepage.org/

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).

7.3. 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 R script Control.R is used, one first has to provide measurement (field) data in the first lines of the R script Control.R, 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 R script OutputInstructions.R 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.

7.4. 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.