OpenDA

Deltares

Enabling Delta Life

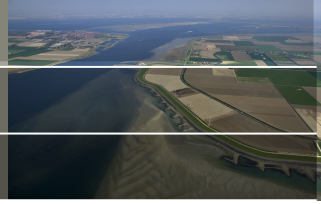
www.openda.org

Calibration in OpenDA

Martin Verlaan

Martin.Verlaan@deltares.nl

Outline

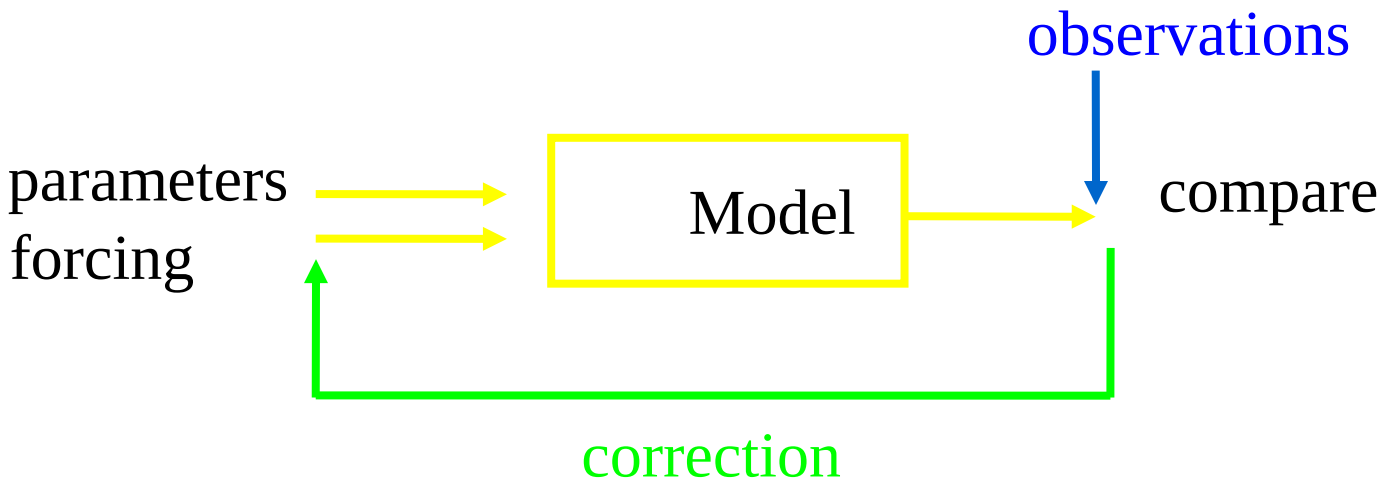


- Introduction
- Minimization of distance to observations
- Regularization
- Optimization
- Algorithms
- Example
- Options

OpenDA

Calibration

- **Many models contain uncertain parameters, often related to friction, boundary conditions or sub-soil material properties**
- **Model output can be validated against observations.**



Optimization of a costfunction

- calibration is defined as an optimization problem
- elaborate background in statistics, e.g. log-likelihood function

$$min_{p,x(0),w(t)} J = min_{p,x(0),w(t)} J_o + J_p + J_i + J_s$$

- measure distance or misfit of model to observations
- depends on uncertainty of observations and representativeness

$$J_o = \sum_t (y_o(t) - Hx(t))' R^{-1} (y_o(t) - Hx(t))$$

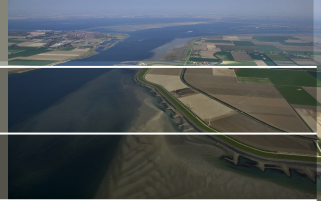
- for independent observation errors

$$J_p = \sum_t \frac{(y_o(t) - y_m(t))^2}{\sigma_o^2}$$

- may be ill posed!

OpenDA

Background terms



- often regularization is needed to obtain a well-posed problem
- a background error (Bayesian prior) solves this formally (and sometimes/often also practically)

$$J_p = (p - p_0)' P_p^{-1} (p - p_0)$$

- this denotes a distance to the prior estimate, e.g. for independent errors for the parameters

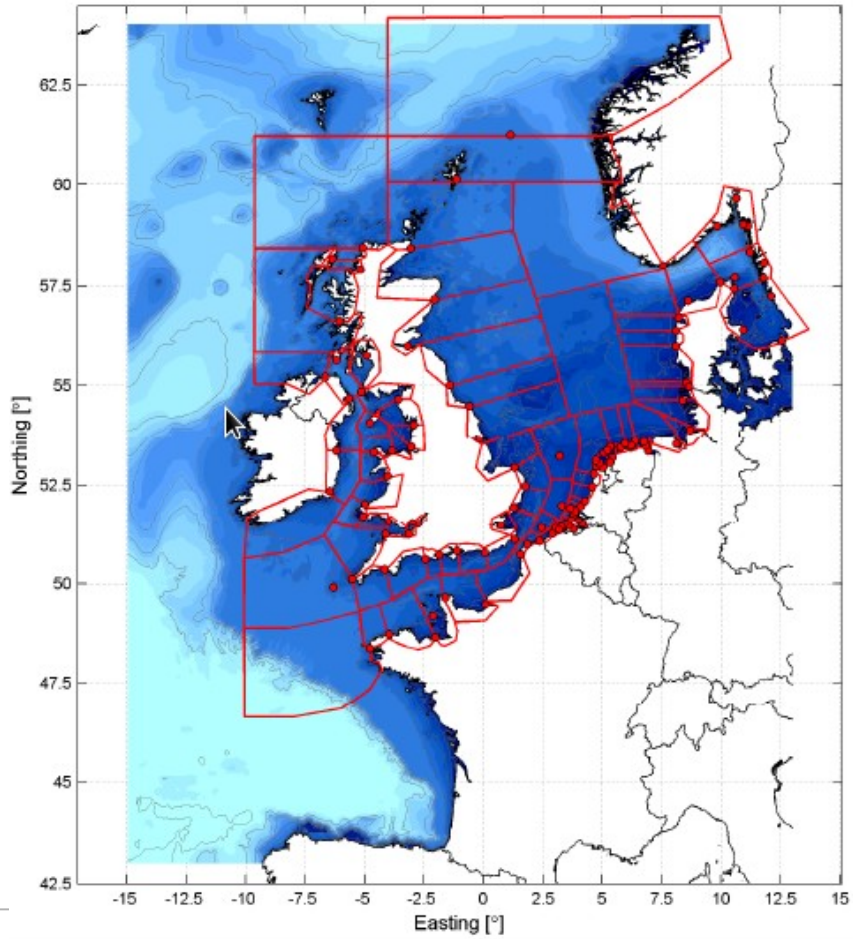
$$J_p = \frac{(p - p_0)^2}{\sigma_p^2}$$

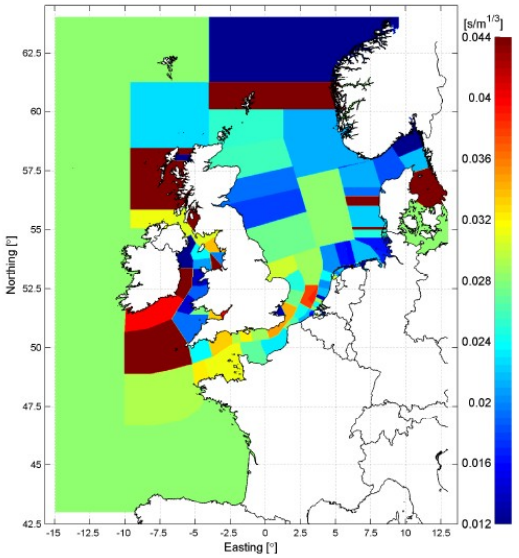
- can also be used as a constraint on smoothness
- there is a mathematical theorem relating smoothness of the solution to the covariance of the errors

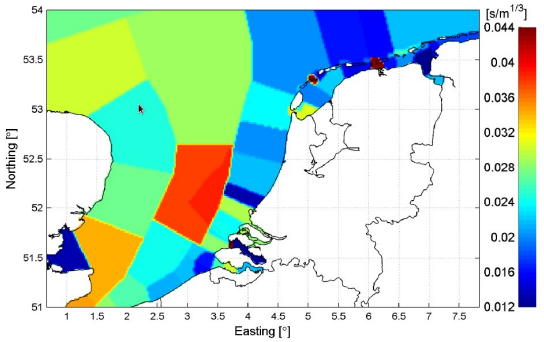
OpenDA

Reduction of number of parameters

- there is a limit to what can be extracted from observations
- remember : number of parameters < independent observations
- useful: common sense, knowledge about the application

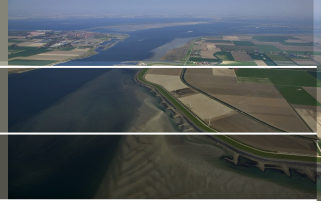






OpenDA

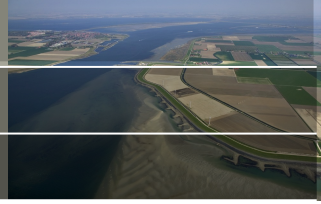
Optimization algorithms



- Dud
  - Efficient for nearly linear models
- Sparse Dud
  - More efficient than Dud when one parameter influences only few observations
- Simplex
  - Robust generic method
- Powell
  - Can be efficient for finding last decimals of solution
- Gridded Full Search
  - Useful for testing
- GLUE, SCE
  - Can handle local optima
- Conjugate gradient
- LBFGS
- SPSA (under construction)

OpenDA

DUD algorithm

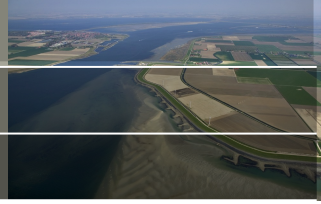


Does not Use Derivatives (DUD)

- **Start running first guess and modifying each parameter**
- **Linearize the model around these values**
- **Solve linear problem**
- **IF**
  this is an improvement update linearization with new point (remove worst estimate from the list)
- **ELSE**
  do a line-search (only until there is improvement)

OpenDA

DUD input file

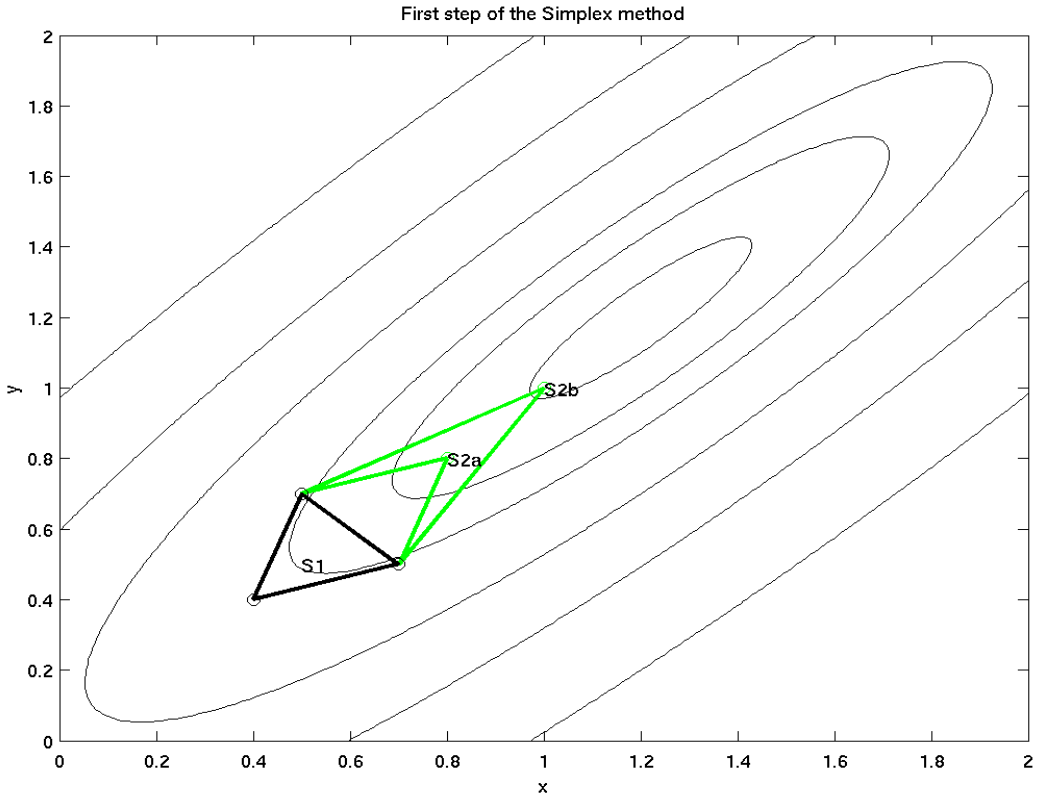


```
<?xml version="1.0" encoding="UTF-8"?>
<DudConfig xmlns="http://www.openda.org" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openda.org
	<costFunction weakParameterConstraint="false" class="org.openda.algorithms.SimulationKwadraticCostFunction" factor="1.0"/>
	<outerLoop maxIterations="10" absTolerance="0.01" relTolerance="0.01" relToleranceLinearCost="0.001"/>
	<lineSearch maxIterations="5" maxRelStepSize="10.0">
		<backTracking shorteningFactor="0.5" startIterationNegativeLook="3"/>
	</lineSearch>
	<!-- optional additional criteria for stopping the calibration -->
	<stopCriteria>
		<stopCriterion class="org.openda.algorithms.AbsoluteAveragePerLocationStopCriterion" threshold="1.0E-5"/>
	</stopCriteria>
</DudConfig>
```

- Outer iterations can stop on various criteria
- WeakParameterConstraint adds term for difference to initial pars
- LineSearch starts when linear fit fails (look at output!)
- Use documentation or xml-editor that supports xml-schemas for assistance

OpenDA

Optimization methods

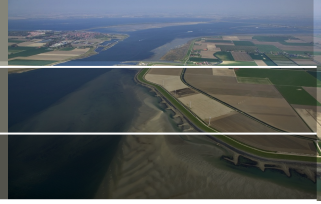


Simplex

- **try mirror point with largest value**
- **try to extend**
- **...**

OpenDA

Simplex input

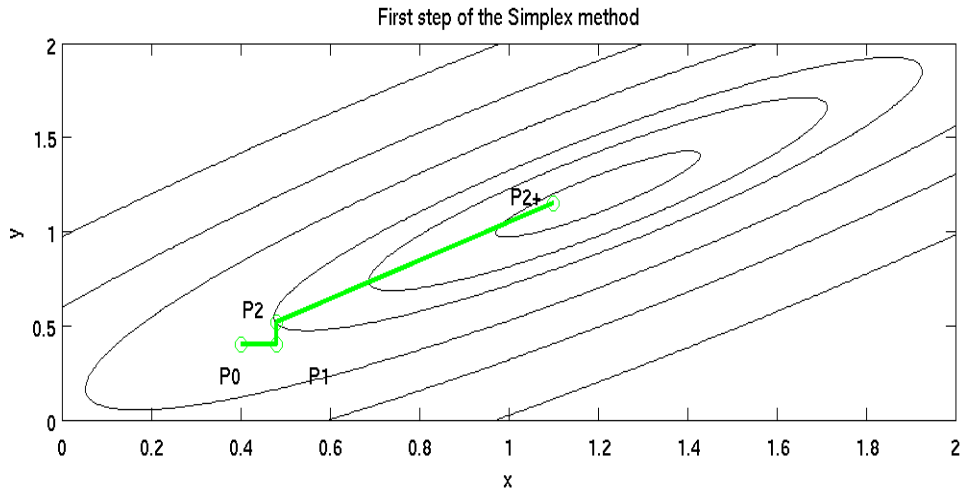


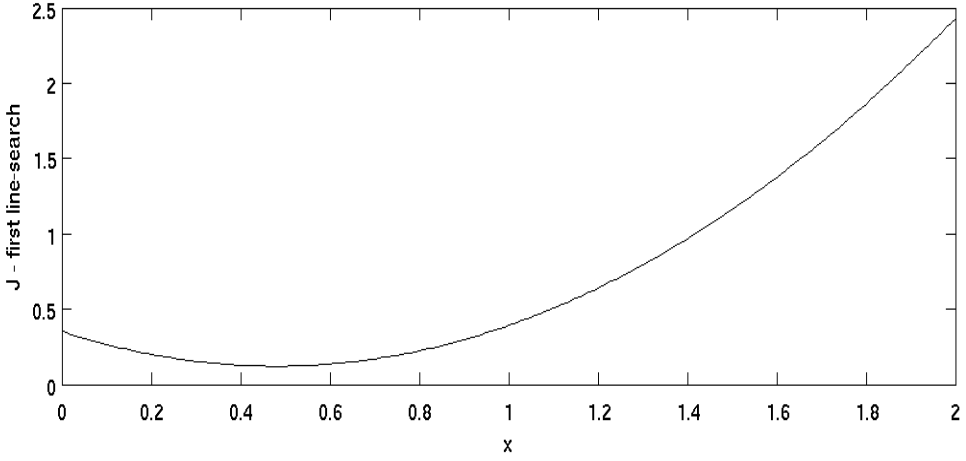
```
<SimplexConfig xmlns="http://www.openda.org" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openda
	<costFunction weakParameterConstraint="false" class="org.openda.algorithms.SimulationKwadraticCostFunction" factor="0.5"/>
	<outerLoop maxIterations="20" absTolerance="0.01" relTolerance="0.01"/>
	<!-- optional additional criteria for stopping the calibration -->
	<stopCriteria>
		<stopCriterion class="org.openda.algorithms.AbsoluteAveragePerLocationStopCriterion" threshold="1.0E-5"/>
	</stopCriteria>
</SimplexConfig>
```

- Simplex has only outer iterations
- Standard stopCriteria test for changes to cost value

OpenDA

Optimization methods



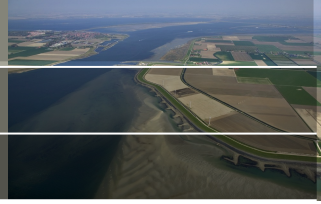


Powell

- **one line-search per parameter each outer iteration**
- **update seach directions**

OpenDA

Powell input

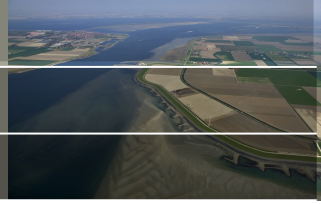


```
<PowellConfig xmlns="http://www.openda.org" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openda.
    <costFunction weakParameterConstraint="false" class="org.openda.algorithms.SimulationKwadraticCostFunction" factor="0.5"/>
    <outerLoop maxIterations="10" absTolerance="0.01" relTolerance="0.01"/>
    <lineSearch type="brent" maxIterations="10" relTolerance="0.01" maxRelStepSize="100.0">
        <brent startBracketValue="1.0"/>
    </lineSearch>
    <!-- optional additional criteria for stopping the calibration -->
    <stopCriteria>
        <stopCriterion class="org.openda.algorithms.AbsoluteAveragePerLocationStopCriterion" threshold="1.0E-5"/>
    </stopCriteria>
</PowellConfig>
```

- Powell performs a series of line-searches
- Each line-search:
- Bracket one lower cost between two larger ones
- Followed by a series of parabolic fits
- Initial line-searches change one parameter
- Later line-searches use an estimate of the Hessian (Quadratic fit)

OpenDA

Gridded Full Search

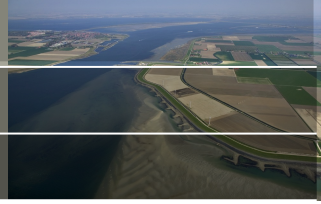


```
<?xml version="1.0" encoding="UTF-8"?>
<GriddedFullSearchConfig xmlns="http://www.openda.org" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://ww
	<costFunction weakParameterConstraint="false" class="org.openda.algorithms.SimulationKwadraticCostFunction"/>
	<!-- for each parameter a regular grid is defined. The computations cover all possible combinations of these values.
	Note that the number of model runs increases quickly with the number of parameters, eg 10 steps for 3 parameters give 10x10x10=100
	<gridRange min="[7.0,1.5]" max="[9.0,1.8]" step="[0.2,0.05]"/>
	<!-- optional additional criteria for stopping the calibration -->
	<stopCriteria>
		<stopCriterion class="org.openda.algorithms.AbsoluteAveragePerLocationStopCriterion" threshold="1.0E-3"/>
	</stopCriteria>
</GriddedFullSearchConfig>
```

- Test all combinations of parameters on a regular pattern
- Value with lowest cost is returned
- Parameters are in same order as in StochModel

OpenDA

DUD Output



```
% ======================================================
% DUD outer iteration no.1
% ======================================================
% ------------------------------------------------------
costs{1}        =[0.46925713409755426,35.512591737499974,37.30276589788343];
% ------------------------------------------------------
% Start search until improvement,
% Next try p=    [8.663,1.71]
%   delta_p=     [0.663,0.014]
% ======================================================================
% prepare no 4
% configstring = OscillatorStochModel.xml
% opening :/home/verlaanm/deltares/src/openda_20101025/public/tests/simple_oscillator/./model/OscillatorStochModel.xml
% simulationTimespan =[0.0,0.05,10.0]
% parameters@names =t_damp,omega parameters=[8.0,1.5708]
% parameterUncertainty@names =t_damp,omega parameterUncertainty=[1.0,0.1257]
% systemNoise = {[0.0,0.0],[0.3,0.3]}
% initialState = [0.8,0.0]
% initialStateUncertainty = [0.8,0.8]
evaluatedParameters{4} =[8.663227729470336,1.710474267579495];
% ======================================================================
% evaluate no. 4
predicted{4}    =[-0.04871533545298321,-0.623329637530464,0.18902524808841653,0.4492358459183649,-0.25841736087328293,-0.2942934600416863,0.
observed{4}     =[-0.04313443918374465,-0.6309221702813459,0.1759519292051853,0.4657659569580843,-0.2452863996167396,-0.3179787785265744,0.26
residuals{4}    =[0.005580896269238564,-0.007592532750881897,-0.013073318883231239,0.01653011103971941,0.013130961256543339,-0.023685318484888
obs_stdev{4}    =[0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1];
%
% ----------------------------------------------------------------------
% RESIDUAL STATISTICS:
% ----------------------------------------------------------------------------------------------------------------------------------
% ObsID       Num of data       Min       Max       Bias       STD       RMS
% ----------------------------------------------------------------------------------------------------------------------------------
% 0.0         10      -0.02802444432221536    0.027600887633858262    -0.0016453322976147937  0.017827003466233938    0.016992026532941254
% ----------------------------------------------------------------------------------------------------------------------------------
```

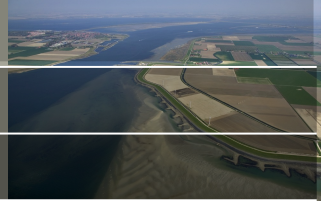
Estimated parameters from linear fit

Try full model with these pars

Results of full run

OpenDA

DUD output



```
% Error estimate for this outer iteration
parameterErrorEstimateStd{3}    =[1.574360951603447,0.020367356266850184];
parameterErrorCorrelations{3}   =[1.0,-0.11806660474049042;-0.11806660474049056,0.9999999999999998];
% stop criterion 1, imain > maxit:       3 < 10
% stop criterion 2, diff < abstol:       0.0017994068709786448 > 0.01
% stop criterion 3, relDiff < reltol:    236.0017006534848 > 0.01
% stop criterion 4, linearized cost relative error: 0.004193543182922271 < 0.0010
% stop criterion AbsoluteAveragePerLocationStopCriterion: 1.0622549942029128E-5 > 1.0E-5
%       satisfied at NO locations!
% ================================================================
% SimulationKwadraticCostfunction: optimal results
%     number of evaluations: 6
%     all cost values:
%         [35.513,37.303,0.469,0.289,1.807E-3,7.625E-6]
%     all parameter values
%         [8,9,8,8.663,9.057,9.004;1.571,1.571,1.696,1.71,1.699,1.7]
%     number of observations: 10
%     best cost:
%         cost = 7.625E-6
%     best parameters:
%                 [9.004,1.7]
% ================================================================
% Application Done
```

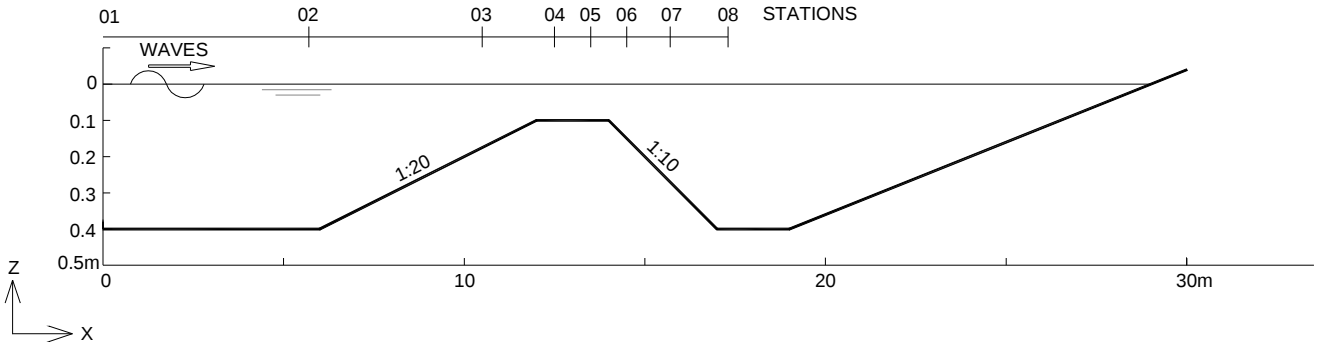
- Estimated errors (from accuracies in input and linear fit)
- Summary of calibration

OpenDA

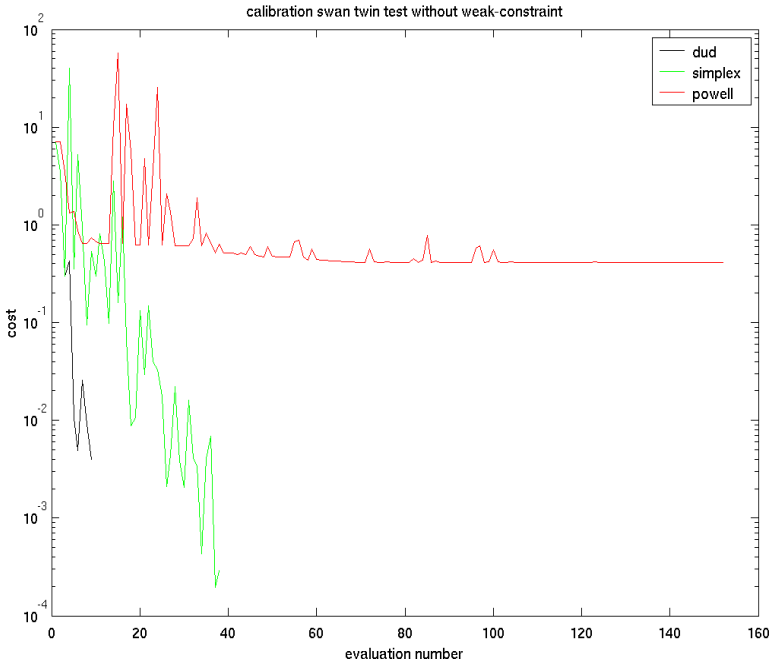
SWAN: 3rd generation wave model

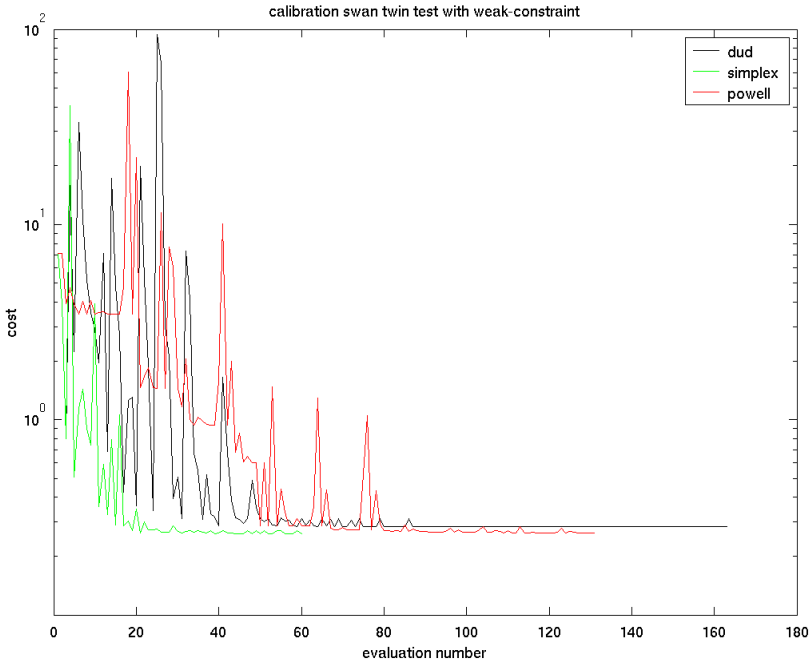
Wave breaking and interaction over a bar

S_{tot} =wind input+ non-linear interactions (quadruplets &/+triads)+ whitecapping + bottom friction+ depth induced wave breaking



SWAN calibration : costfunction

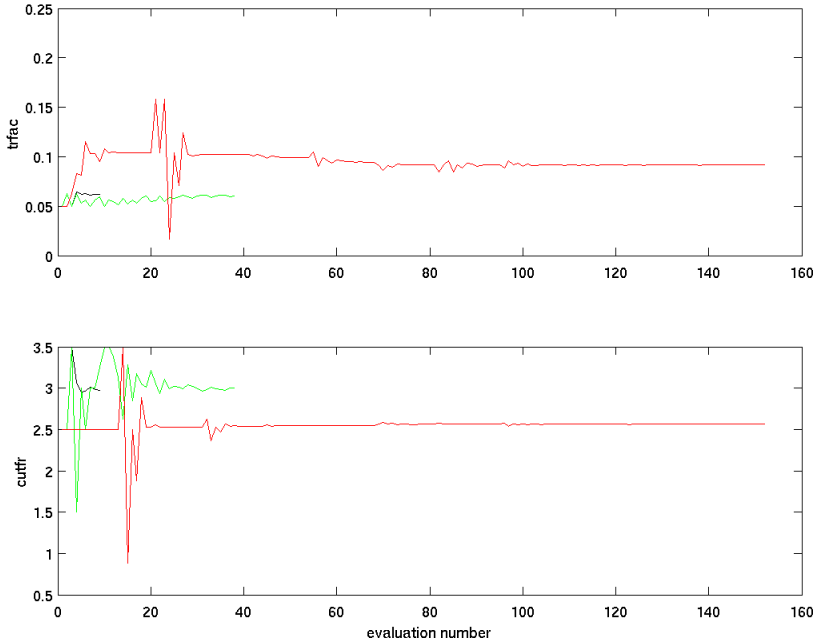


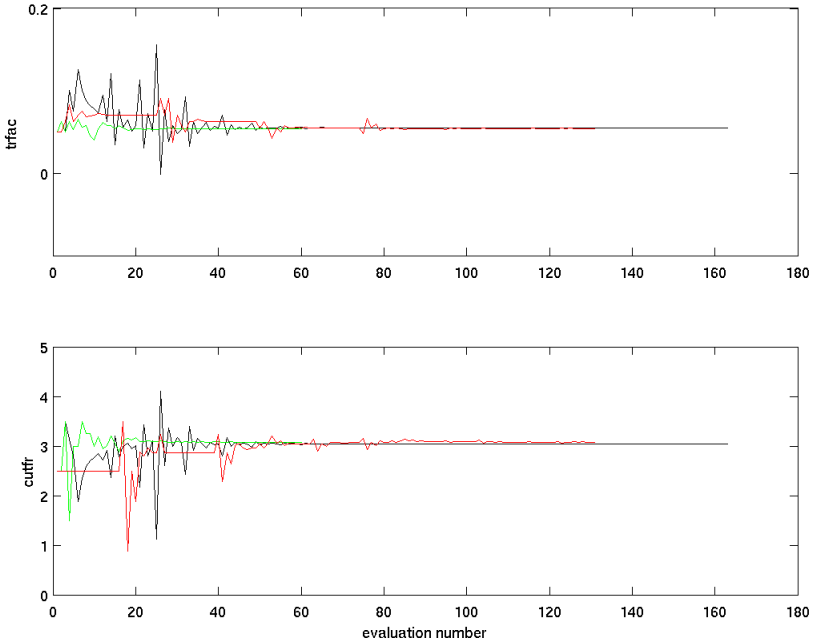


Local optimum: Powell does not converge here
Weak constraint helps here: larger cost values

OpenDA

SWAN calibration : parameter estimates

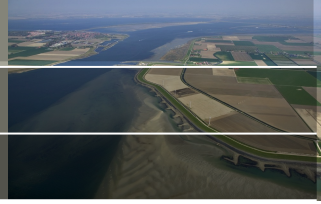




Weak constraint: closer to initial parameters

OpenDA

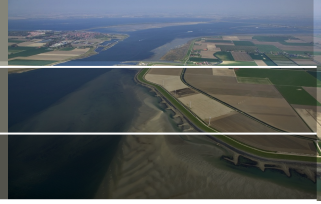
Options



- Restarts
  - After a crash, when you need more iterations
- Parallel computing and scheduling
  - Modify blackbox wrapper
- Data selection
  - Obervations are truncated in time based on simulation interval and options in StochObserver

OpenDA

Restart



Create restart files

```
<?xml version="1.0" encoding="UTF-8"?>
<openDaApplication>
<stochObserver className="org.openda.utils.CsvStochObserver">
<workingDirectory>./stochobserver</workingDirectory>
<configFile>observations_oscillator_generated_for_calibration.csv</configFile>
</stochObserver>
<restartOutFilePrefix>restart_</restartOutFilePrefix>
<stochModelFactory
className="org.openda.models.oscillator.OscillatorStochModelFactory">
…
```

Use restart file

```
<restartInFile>restart_1.xml</restartInFile>
```

OpenDA

Parallel computing/schedule runs

Input file of Model wrapper:

```
<!-- computation actions -->
<!-- define actions relative to working dir. -->
<computeActions>
    <action workingDirectory="%instanceDir%"  linuxExe="%OPENDADIR%/delftFlowRunQsub.sh" >
        <arg>%runid%</arg>
        <arg>%runid%_%instanceNumber%</arg>
     </action>
</computeActions>
<additionalComputeActions>
    <action workingDirectory="%instanceDir%"  linuxExe="%OPENDADIR%/delftFlowRunWait.sh" >
        <arg>%runid%</arg>
        <arg>%runid%_%instanceNumber%</arg>
     </action>
</additionalComputeActions>
```

Use scripts to start and wait for runs.

OpenDA

Time selection of observations

- Eg to exclude spin-up of model

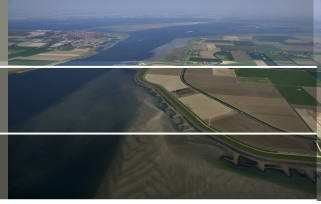
Blackbox model config file:

```
<timeInfo start="2009-01-01T00:00:00" end="2009-04-01T00:00:00"/>
```

Noos StochObserver:

```
<timeSeries status="validate" minDateTime="200901040000"
 MaxDateTime="200904010000" minValue="-3.0" maxValue="3.0" >
   aberdeen_waterlevel_astro.noos
 </timeSeries>
```

Final remarks



- Calibration tools only try to minimize the cost function. You have to provide the problem such that the results are meaningful.
- Look carefully at output files. Many things can go different than expected.

OpenDA