matk Documentation

Release 0

Dylan R. Harp

October 10, 2013

CONTENTS

Contents:

class `matk.matk`(***kwargs*)

Class for Model Analysis ToolKit module

`add_ins`(*insfilenm*, *model_outfile*)

Add an instruction file to problem

`add_obs`(*name*, ***kwargs*)

Add observation to problem

`add_par`(*name*, ***kwargs*)

Add parameter to problem

`add_tpl`(*tplfilenm*, *model_infile*)

Add a template file to problem

`calibrate`()

Calibrate pymads problem model

`forward`(*workdir=None*)

Run pymads problem forward model using current values

`get_obs_names`()

Get observation names

`get_obs_values`()

Get observation values

`get_par_dist_pars`()

Get parameters needed by parameter distributions

`get_par_dists`()

Get parameter probabilistic distributions

`get_par_maxs`()

Get parameter lower bounds

`get_par_mins`()

Get parameter lower bounds

`get_par_names`()

Get parameter names

`get_par_values`()

Get parameter values

`get_residuals`()

Get least squares values

`get_samples`(*siz=None*, *noCorrRestr=False*, *corrmat=None*, *outfile=None*, *seed=None*)

Draw lhs samples from scipy.stats module distribution

siz [int] number of samples to generate, ignored if samples are provided

noCorrRestr: bool if True, correlation structure is not enforced on sample

corrmat [matrix] correlation matrix

outfile [string] name of file to output samples to

seed [int] random seed to allow replication of samples

samples [ndarray] Parameter samples

outfile [string] name of file to write samples in. If outfile=None, no file is written.

`get_sims()`
Get the current simulated values

`model`
Python function or system command to run model

`ncpus`
Set number of cpus to use for concurrent model evaluations

`parameters_file`
Set the name of the parameters_file for parallel runs

`read_model_files`(*workdir=None*)
Write model files with current parameters

`results_file`
Set the name of the results_file for parallel runs

`run_parallel()`
Run models concurrently on multiprocessor machine

`run_samples`(*siz=None*, *noCorrRestr=False*, *corrmat=None*, *samples=None*, *outfile=None*, *parallel=False*, *ncpus=1*, *templatedir=None*, *workdir_base=None*, *seed=None*, *save_dirs=True*)
Use or generate samples and run models First argument (optional) is an array of samples

siz [int] number of samples to generate, ignored if samples are provided

noCorrRestr: bool if True, correlation structure is not enforced on sample

corrmat [matrix] correlation matrix

samples [ndarray] matrix of samples, npar columns by siz rows

outfile [string] name of file to write samples and responses in. If outfile=None, no file is written.

parallel [bool] if True, models run concurrently with 'ncpus' cpus

ncpus [int] number of cpus to use to run models concurrently

templatedir [string] name of folder including files needed to run model (e.g. template files, instruction files, executables, etc.)

workdir_base [string] base name for model run folders, run index is appended to workdir_base

seed [int] random seed to allow replication of samples

save [bool] if True, working directories during parallel model execution will not be deleted

responses [ndarray] Responses from model runs

samples [ndarray] Parameter samples, same as input samples if provided

`seed`
Set the name of the templatedir for parallel runs

`set_obs_values`(**args*, ***kwargs*)
Set simulated values using a dictionary or keyword arguments

`set_par_values`(**args*, ***kwargs*)
Set parameters using values in first argument

templatedir

Set the name of the templatedir for parallel runs

workdir

Set the base name for parallel working directories

workdir_base

Set the base name for parallel working directories

workdir_index

Set the working directory index for parallel runs

write_model_files(*workdir=None*)

Write model files with current parameters

CHAPTER

ONE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

INDEX