MARIGOLD

Release 0.0.1

adix

Apr 05, 2024

CONTENTS:

*M*ultiphase *A*nalysis of *R*aw *I*nformation for *G*lobal *O*r *L*ocal *D*ata

MARIGOLD.Condition(jgref, jgloc, jf, theta, ...)	Class to handle the local probe data
MARIGOLD.extracts_and_loads	

CHAPTER ONE

MARIGOLD.CONDITION

class MARIGOLD.**Condition**(*jgref: float*, *jgloc: float*, *jf: float*, *theta: int*, *port: str*, *database: str*)

Class to handle the local probe data

Data is stored in the Condition.phi property. It's actually 3 layers of dictionary phi [angle] gives a dictionary with the various r/R phi [angle][r/R] gives a dictionary with the MIDAS output The MIDAS output is itself a dictionary, with the keys listed in the "tab_keys" array So phi[angle][r/R]['alpha'] should give you the void fraction at r/R for phi = angle This structure is initialized with zeros for the MIDAS output at the pipe center and wall

__init__(*jgref: float*, *jgloc: float*, *jf: float*, *theta: int*, *port: str*, *database: str*) → None

Methods

`TD_FR_ID`()	
`__init__`(jgref, jgloc, jf, theta, port, database)	
`approx_vf`([n])	Method for approximating vf with power-law relation.
`approx_vf_Kong`([n])	Method for approximating vf from Kong.
`area_avg`(param[, even_opt, recalc])	Method for calculating the area-average of a parameter, "param"
`calc_W`()	Calculates the wake deficit function, W, from the experimental data
`calc_avg_lat_sep`()	Calculates average lateral separation distance between bubbles
`calc_cd`([method, rho_f, vr_cheat, mu_f])	Method for calculating drag coefficient
`calc_dpdz`([method, rho_f, rho_g, mu_f, ...])	Calculates the pressure gradient, dp/dz, according to various methods.
`calc_errors`(param1, param2)	Calculates the errors, , between two parameters (param1 - param2) in midas_dict
`calc_grad`(param[, recalc])	Calculates gradient of param based on the data in self.
`calc_linear_interp`(param)	Makes a LinearNDInterpolator for the given param.
`calc_linear_xy_interp`(param)	Makes a LinearNDInterpolator for the given param in x y coords
`calc_mu3_alpha`()	Calculates the third moment of alpha
`calc_mu_eff`([method, mu_f, mu_g, alpha_max])	Method for calculating effective viscosity.
`calc_sigma_alpha`()	Calculates the second moment of alpha
`calc_vgj`([warn_approx])	Method for calculating Vgj, by doing

continues on next page

Table 1 – continued from previous page

calc_vgj_model()	Method for calculating Vgj based on models
calc_void_cov()	Calculates the void covariance
calc_vr([warn_approx])	Method for calculating relative velocity.
calc_vr_model([method, c3, n, iterate_cd, quiet])	Method for calculating relative velocity based on models
calc_vwvg()	Calculates void weighted Vgj
circ_segment_area_avg(param, hstar[, ...])	Method for calculating the area-average of a parameter, "param" over the circular segment defined by h
circ_segment_void_area_avg(param, hstar[, ...])	Method for calculating the void-weighted area-average of a parameter over the circular segment defined by h
find_hstar_pos([method, void_criteria])	Returns the vertical distance from the top of the pipe to the bubble layer interface
fit_spline(param)	Fits a RectBivariateSpline for the given param.
interp_area_avg(param[, interp_type])	Function to area-average param, using the spline interpolation of param
line_avg(param, phi_angle[, even_opt])	Line average of param over line defined by phi_angle
line_avg_dev(param, phi_angle[, even_opt])	Second moment of param over line defined by phi_angle
max(param[, recalc])	Return maximum value of param in the Condition
max_line(param, angle)	Return maximum value of param at a given angle
max_line_loc(param, angle)	Return r/R location of maximum value of param at a given angle
max_loc(param)	Return location of maximum param in the Condition
min(param[, recalc, nonzero])	Return minimum value of param in the Condition
min_loc(param)	Return minimum value of param in the Condition
mirror([sym90, axisym, uniform_rmesh, ...])	Mirrors data, so we have data for every angle
plot_contour(param[, save_dir, show, ...])	Method to plot contour of a given param
plot_isoline(param, iso_axis, iso_val[, ...])	Plot profiles of param over iso_axis at iso_val
plot_profiles(param[, save_dir, show, ...])	Plot profiles of param over x_axis, for const_to_plot, i.e. over r/R for = [90, 67.5 .
plot_spline_contour(param[, save_dir, show, ...])	Plots a contour from a spline interpolation
plot_surface(param[, save_dir, show, ...])	Method to plot a surface of a given param
pretty_print([print_to_file, FID, mirror])	Prints out all the information in a Condition in a structured way
rough_FR_ID()	Identifies the flow regime for the given condition, by some rough methods
spline_circ_seg_area_avg(param, hstar[, int_err])	Function to area-average over a circular segment defined by h, using the spline interpolation of param
spline_void_area_avg(param)	Function to void-weighted area-average param over a circular segment defined by h, using the spline interpolation of param
top_bottom(param[, even_opt])	Honestly, I forgot what this does
void_area_avg(param[, even_opt])	Method for calculating the void-weighted area-average of a parameter

Attributes

`debugFID`

CHAPTER TWO

MARIGOLD.EXTRACTS_AND_LOADS

Functions

`dump_data_from_tabs`([dump_file, skip_dir])	
`extractIskandraniData`([dump_file])	
`extractLocalDataFromDir`(path[, dump_file, ...])	Function for getting all local data from spreadsheets in a directory, path
`extractPitotData`([dump_file, in_dir, ...])	
`extractProbeData`([dump_file, in_dir, ...])	
`extractYangData`([dump_file])	
`loadData`(data_file)	
`loadIskandraniData`([data_file])	
`loadPitotData`([data_file])	
`loadProbeData`([data_file])	
`loadYangData`([data_file])	

CHAPTER THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX