

Ecosystem

Version 0.0 beta

Generated by Doxygen 1.8.3.1

Mon Sep 21 2015 16:56:05

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	File Index	5
3.1	File List	5
4	Class Documentation	7
4.1	ARNOLDI_DATA Struct Reference	7
4.1.1	Detailed Description	8
4.1.2	Member Data Documentation	8
4.1.2.1	beta	8
4.1.2.2	e1	8
4.1.2.3	Hkp1	8
4.1.2.4	hp1	8
4.1.2.5	iter	8
4.1.2.6	k	8
4.1.2.7	Output	8
4.1.2.8	sum	8
4.1.2.9	v	8
4.1.2.10	Vk	8
4.1.2.11	w	9
4.1.2.12	yk	9
4.2	Atom Class Reference	9
4.2.1	Constructor & Destructor Documentation	10
4.2.1.1	Atom	10
4.2.1.2	~Atom	10
4.2.1.3	Atom	10
4.2.1.4	Atom	10
4.2.2	Member Function Documentation	10

4.2.2.1	AtomCategory	10
4.2.2.2	AtomicNumber	10
4.2.2.3	AtomicWeight	10
4.2.2.4	AtomName	10
4.2.2.5	AtomState	10
4.2.2.6	AtomSymbol	10
4.2.2.7	BondingElectrons	10
4.2.2.8	DisplayInfo	10
4.2.2.9	editAtomicWeight	10
4.2.2.10	editElectrons	10
4.2.2.11	editNeutrons	10
4.2.2.12	editOxidationState	10
4.2.2.13	editProtons	10
4.2.2.14	editValence	11
4.2.2.15	Electrons	11
4.2.2.16	Neutrons	11
4.2.2.17	OxidationState	11
4.2.2.18	Protons	11
4.2.2.19	Register	11
4.2.2.20	Register	11
4.2.2.21	removeElectron	11
4.2.2.22	removeNeutron	11
4.2.2.23	removeProton	11
4.2.3	Member Data Documentation	11
4.2.3.1	atomic_number	11
4.2.3.2	atomic_weight	11
4.2.3.3	Category	11
4.2.3.4	electrons	11
4.2.3.5	Name	11
4.2.3.6	NaturalState	11
4.2.3.7	neutrons	11
4.2.3.8	oxidation_state	11
4.2.3.9	protons	11
4.2.3.10	Symbol	11
4.2.3.11	valence_e	11
4.3	BACKTRACK_DATA Struct Reference	11
4.3.1	Detailed Description	12
4.3.2	Member Data Documentation	12
4.3.2.1	alpha	12
4.3.2.2	constRho	12

4.3.2.3	Fk	12
4.3.2.4	lambdaMin	12
4.3.2.5	normFkp1	12
4.3.2.6	rho	12
4.3.2.7	xk	13
4.4	BiCGSTAB_DATA Struct Reference	13
4.4.1	Detailed Description	14
4.4.2	Member Data Documentation	14
4.4.2.1	alpha	14
4.4.2.2	bestres	14
4.4.2.3	bestx	14
4.4.2.4	beta	14
4.4.2.5	breakdown	14
4.4.2.6	iter	15
4.4.2.7	maxit	15
4.4.2.8	omega	15
4.4.2.9	omega_old	15
4.4.2.10	Output	15
4.4.2.11	p	15
4.4.2.12	r	15
4.4.2.13	r0	15
4.4.2.14	relres	15
4.4.2.15	relres_base	15
4.4.2.16	res	15
4.4.2.17	rho	15
4.4.2.18	rho_old	16
4.4.2.19	s	16
4.4.2.20	t	16
4.4.2.21	tol_abs	16
4.4.2.22	tol_rel	16
4.4.2.23	v	16
4.4.2.24	x	16
4.4.2.25	y	16
4.4.2.26	z	16
4.5	CGS_DATA Struct Reference	16
4.5.1	Detailed Description	18
4.5.2	Member Data Documentation	18
4.5.2.1	alpha	18
4.5.2.2	bestres	18
4.5.2.3	bestx	18

4.5.2.4	beta	18
4.5.2.5	breakdown	18
4.5.2.6	c	18
4.5.2.7	iter	18
4.5.2.8	maxit	18
4.5.2.9	Output	18
4.5.2.10	p	18
4.5.2.11	r	19
4.5.2.12	r0	19
4.5.2.13	relres	19
4.5.2.14	relres_base	19
4.5.2.15	res	19
4.5.2.16	rho	19
4.5.2.17	sigma	19
4.5.2.18	tol_abs	19
4.5.2.19	tol_rel	19
4.5.2.20	u	19
4.5.2.21	v	19
4.5.2.22	w	19
4.5.2.23	x	20
4.5.2.24	z	20
4.6	Document Class Reference	20
4.6.1	Constructor & Destructor Documentation	21
4.6.1.1	Document	21
4.6.1.2	~Document	21
4.6.1.3	Document	21
4.6.1.4	Document	21
4.6.1.5	Document	21
4.6.1.6	Document	21
4.6.1.7	Document	21
4.6.2	Member Function Documentation	21
4.6.2.1	addHeadKey	21
4.6.2.2	addPair	21
4.6.2.3	addPair	21
4.6.2.4	begin	21
4.6.2.5	begin	21
4.6.2.6	changeKey	21
4.6.2.7	clear	22
4.6.2.8	copyAnchor2Alias	22
4.6.2.9	DisplayContents	22

4.6.2.10	end	22
4.6.2.11	end	22
4.6.2.12	getAlias	22
4.6.2.13	getAnchoredHeader	22
4.6.2.14	getDataMap	22
4.6.2.15	getHeader	22
4.6.2.16	getHeadFromSubAlias	22
4.6.2.17	getHeadMap	22
4.6.2.18	getName	22
4.6.2.19	getState	22
4.6.2.20	isAlias	22
4.6.2.21	isAnchor	22
4.6.2.22	operator()	22
4.6.2.23	operator()	22
4.6.2.24	operator=	22
4.6.2.25	operator[]	22
4.6.2.26	operator[]	22
4.6.2.27	resetKeys	22
4.6.2.28	revalidateAllKeys	22
4.6.2.29	setAlias	22
4.6.2.30	setName	22
4.6.2.31	setNameAliasPair	22
4.6.2.32	setState	22
4.6.2.33	size	22
4.6.3	Member Data Documentation	22
4.6.3.1	Head_Map	23
4.7	DOGFISH_DATA Struct Reference	23
4.7.1	Member Data Documentation	23
4.7.1.1	DirichletBC	23
4.7.1.2	end_time	23
4.7.1.3	eval_DI	23
4.7.1.4	eval_kf	23
4.7.1.5	eval_qs	23
4.7.1.6	eval_R	23
4.7.1.7	fiber_diameter	23
4.7.1.8	fiber_length	24
4.7.1.9	finch_dat	24
4.7.1.10	NonLinear	24
4.7.1.11	NumComp	24
4.7.1.12	OutputFile	24

4.7.1.13	param_dat	24
4.7.1.14	Print2Console	24
4.7.1.15	Print2File	24
4.7.1.16	t_counter	24
4.7.1.17	t_print	24
4.7.1.18	time	24
4.7.1.19	time_old	24
4.7.1.20	total_sorption	24
4.7.1.21	total_sorption_old	24
4.7.1.22	total_steps	24
4.7.1.23	user_data	24
4.8	DOGFISH_PARAM Struct Reference	24
4.8.1	Member Data Documentation	24
4.8.1.1	film_transfer_coeff	24
4.8.1.2	initial_sorption	25
4.8.1.3	intraparticle_diffusion	25
4.8.1.4	sorbed_molefraction	25
4.8.1.5	species	25
4.8.1.6	surface_concentration	25
4.9	FINCH_DATA Struct Reference	25
4.9.1	Member Data Documentation	27
4.9.1.1	beta	27
4.9.1.2	callroutine	27
4.9.1.3	CC_E	27
4.9.1.4	CC_I	27
4.9.1.5	CheckMass	27
4.9.1.6	CL_E	27
4.9.1.7	CL_I	27
4.9.1.8	CN	27
4.9.1.9	CR_E	27
4.9.1.10	CR_I	27
4.9.1.11	d	27
4.9.1.12	DIC	27
4.9.1.13	Dirichlet	27
4.9.1.14	discretize	27
4.9.1.15	Dn	27
4.9.1.16	Dnp1	27
4.9.1.17	Do	27
4.9.1.18	dt	27
4.9.1.19	dt_old	27

4.9.1.20	dz	28
4.9.1.21	evalprecon	28
4.9.1.22	evalres	28
4.9.1.23	ExplicitFlux	28
4.9.1.24	fC_E	28
4.9.1.25	fC_I	28
4.9.1.26	fL_E	28
4.9.1.27	fL_I	28
4.9.1.28	Fn	28
4.9.1.29	Fnp1	28
4.9.1.30	fR_E	28
4.9.1.31	fR_I	28
4.9.1.32	gE	28
4.9.1.33	gl	28
4.9.1.34	Iterative	28
4.9.1.35	kfn	28
4.9.1.36	kfnp1	28
4.9.1.37	kIC	28
4.9.1.38	kn	28
4.9.1.39	knp1	28
4.9.1.40	ko	28
4.9.1.41	L	28
4.9.1.42	lambda_E	28
4.9.1.43	lambda_I	28
4.9.1.44	LN	28
4.9.1.45	max_iter	28
4.9.1.46	ME	28
4.9.1.47	MI	28
4.9.1.48	NE	29
4.9.1.49	NI	29
4.9.1.50	nl_method	29
4.9.1.51	NormTrack	29
4.9.1.52	OE	29
4.9.1.53	OI	29
4.9.1.54	param_data	29
4.9.1.55	picard_dat	29
4.9.1.56	pjfnk_dat	29
4.9.1.57	pres	29
4.9.1.58	res	29
4.9.1.59	resettime	29

4.9.1.60	RIC	29
4.9.1.61	Rn	29
4.9.1.62	Rnp1	29
4.9.1.63	Ro	29
4.9.1.64	s	29
4.9.1.65	setbcs	29
4.9.1.66	setic	29
4.9.1.67	setparams	29
4.9.1.68	setpostprocess	29
4.9.1.69	setpreprocess	29
4.9.1.70	settime	29
4.9.1.71	Sn	29
4.9.1.72	Snp1	29
4.9.1.73	solve	29
4.9.1.74	SteadyState	29
4.9.1.75	T	29
4.9.1.76	t	30
4.9.1.77	t_old	30
4.9.1.78	tol_abs	30
4.9.1.79	tol_rel	30
4.9.1.80	total_iter	30
4.9.1.81	u_star	30
4.9.1.82	uAvg	30
4.9.1.83	uAvg_old	30
4.9.1.84	ubest	30
4.9.1.85	uIC	30
4.9.1.86	un	30
4.9.1.87	unm1	30
4.9.1.88	unp1	30
4.9.1.89	uo	30
4.9.1.90	Update	30
4.9.1.91	uT	30
4.9.1.92	uT_old	30
4.9.1.93	uz_I_E	30
4.9.1.94	uz_I_I	30
4.9.1.95	uz_lm1_E	30
4.9.1.96	uz_lm1_I	30
4.9.1.97	uz_lp1_E	30
4.9.1.98	uz_lp1_I	30
4.9.1.99	vIC	30

4.9.1.100 vn	30
4.9.1.101 vnp1	30
4.9.1.102 vo	30
4.10 GCR_DATA Struct Reference	31
4.10.1 Detailed Description	32
4.10.2 Member Data Documentation	32
4.10.2.1 alpha	32
4.10.2.2 bestres	32
4.10.2.3 bestx	32
4.10.2.4 beta	32
4.10.2.5 breakdown	32
4.10.2.6 c	32
4.10.2.7 c_temp	32
4.10.2.8 iter_inner	32
4.10.2.9 iter_outer	32
4.10.2.10 maxit	32
4.10.2.11 Output	33
4.10.2.12 r	33
4.10.2.13 relres	33
4.10.2.14 relres_base	33
4.10.2.15 res	33
4.10.2.16 restart	33
4.10.2.17 tol_abs	33
4.10.2.18 tol_rel	33
4.10.2.19 total_iter	33
4.10.2.20 transpose_dat	33
4.10.2.21 u	33
4.10.2.22 u_temp	33
4.10.2.23 x	34
4.11 GMRESLP_DATA Struct Reference	34
4.11.1 Detailed Description	35
4.11.2 Member Data Documentation	35
4.11.2.1 arnoldi_dat	35
4.11.2.2 bestres	35
4.11.2.3 bestx	35
4.11.2.4 iter	35
4.11.2.5 maxit	35
4.11.2.6 Output	35
4.11.2.7 r	35
4.11.2.8 relres	35

4.11.2.9	relres_base	35
4.11.2.10	res	35
4.11.2.11	restart	36
4.11.2.12	steps	36
4.11.2.13	tol_abs	36
4.11.2.14	tol_rel	36
4.11.2.15	x	36
4.12	GMRESR_DATA Struct Reference	36
4.12.1	Detailed Description	37
4.12.2	Member Data Documentation	37
4.12.2.1	arg	37
4.12.2.2	gcr_abs_tol	37
4.12.2.3	gcr_dat	37
4.12.2.4	gcr_maxit	37
4.12.2.5	GCR_Output	38
4.12.2.6	gcr_rel_tol	38
4.12.2.7	gcr_restart	38
4.12.2.8	gmres_dat	38
4.12.2.9	gmres_maxit	38
4.12.2.10	GMRES_Output	38
4.12.2.11	gmres_restart	38
4.12.2.12	gmres_tol	38
4.12.2.13	iter_inner	38
4.12.2.14	iter_outer	38
4.12.2.15	matvec	38
4.12.2.16	matvec_data	38
4.12.2.17	N	39
4.12.2.18	term_precon	39
4.12.2.19	terminal_precon	39
4.12.2.20	total_iter	39
4.13	GMRESRP_DATA Struct Reference	39
4.13.1	Detailed Description	40
4.13.2	Member Data Documentation	40
4.13.2.1	bestres	40
4.13.2.2	bestx	40
4.13.2.3	e0	40
4.13.2.4	e0_bar	41
4.13.2.5	H	41
4.13.2.6	H_bar	41
4.13.2.7	iter_inner	41

4.13.2.8	iter_outer	41
4.13.2.9	iter_total	41
4.13.2.10	maxit	41
4.13.2.11	Output	41
4.13.2.12	r	41
4.13.2.13	relres	41
4.13.2.14	relres_base	41
4.13.2.15	res	41
4.13.2.16	restart	42
4.13.2.17	sum	42
4.13.2.18	tol_abs	42
4.13.2.19	tol_rel	42
4.13.2.20	v	42
4.13.2.21	Vk	42
4.13.2.22	w	42
4.13.2.23	x	42
4.13.2.24	y	42
4.14	GPAST_DATA Struct Reference	42
4.14.1	Member Data Documentation	43
4.14.1.1	gama_inf	43
4.14.1.2	He	43
4.14.1.3	Plo	43
4.14.1.4	po	43
4.14.1.5	poi	43
4.14.1.6	present	43
4.14.1.7	q	43
4.14.1.8	qo	43
4.14.1.9	x	43
4.14.1.10	y	43
4.15	GSTA_DATA Struct Reference	43
4.15.1	Member Data Documentation	43
4.15.1.1	dHo	43
4.15.1.2	dSo	43
4.15.1.3	m	43
4.15.1.4	qmax	44
4.16	GSTA_OPT_DATA Struct Reference	44
4.16.1	Member Data Documentation	44
4.16.1.1	all_pars	44
4.16.1.2	best_par	44
4.16.1.3	Fobj	44

4.16.1.4	iso	44
4.16.1.5	Kno	44
4.16.1.6	n_par	44
4.16.1.7	norms	44
4.16.1.8	opt_qmax	44
4.16.1.9	P	44
4.16.1.10	q	44
4.16.1.11	qmax	45
4.16.1.12	total_eval	45
4.17	Header Class Reference	45
4.17.1	Constructor & Destructor Documentation	46
4.17.1.1	Header	46
4.17.1.2	~Header	46
4.17.1.3	Header	46
4.17.1.4	Header	46
4.17.1.5	Header	46
4.17.1.6	Header	46
4.17.1.7	Header	46
4.17.2	Member Function Documentation	46
4.17.2.1	addPair	46
4.17.2.2	addPair	46
4.17.2.3	addSubKey	46
4.17.2.4	begin	46
4.17.2.5	begin	46
4.17.2.6	changeKey	46
4.17.2.7	clear	46
4.17.2.8	copyAnchor2Alias	46
4.17.2.9	DisplayContents	46
4.17.2.10	end	47
4.17.2.11	end	47
4.17.2.12	getAlias	47
4.17.2.13	getAnchoredSub	47
4.17.2.14	getDataMap	47
4.17.2.15	getName	47
4.17.2.16	getState	47
4.17.2.17	getSubHeader	47
4.17.2.18	getSubMap	47
4.17.2.19	isAlias	47
4.17.2.20	isAnchor	47
4.17.2.21	operator()	47

4.17.2.22	operator()	47
4.17.2.23	operator=	47
4.17.2.24	operator[]	47
4.17.2.25	operator[]	47
4.17.2.26	resetKeys	47
4.17.2.27	setAlias	47
4.17.2.28	setName	47
4.17.2.29	setNameAliasPair	47
4.17.2.30	setState	47
4.17.2.31	size	47
4.17.3	Member Data Documentation	47
4.17.3.1	Sub_Map	47
4.18	KeyValueMap Class Reference	48
4.18.1	Constructor & Destructor Documentation	49
4.18.1.1	KeyValueMap	49
4.18.1.2	~KeyValueMap	49
4.18.1.3	KeyValueMap	49
4.18.1.4	KeyValueMap	49
4.18.1.5	KeyValueMap	49
4.18.2	Member Function Documentation	49
4.18.2.1	addKey	49
4.18.2.2	addPair	49
4.18.2.3	addPair	49
4.18.2.4	addPair	49
4.18.2.5	assertType	49
4.18.2.6	begin	49
4.18.2.7	begin	49
4.18.2.8	clear	49
4.18.2.9	DisplayMap	49
4.18.2.10	editValue4Key	49
4.18.2.11	editValue4Key	49
4.18.2.12	end	49
4.18.2.13	end	49
4.18.2.14	findAllTypes	49
4.18.2.15	findType	49
4.18.2.16	getBool	49
4.18.2.17	getDouble	49
4.18.2.18	getInt	49
4.18.2.19	getMap	49
4.18.2.20	getPair	49

4.18.2.21	getString	49
4.18.2.22	getType	50
4.18.2.23	getValue	50
4.18.2.24	operator=	50
4.18.2.25	operator[]	50
4.18.2.26	operator[]	50
4.18.2.27	size	50
4.18.3	Member Data Documentation	50
4.18.3.1	Key_Value	50
4.19	MAGPIE_DATA Struct Reference	50
4.19.1	Member Data Documentation	50
4.19.1.1	gpast_dat	50
4.19.1.2	gsta_dat	50
4.19.1.3	mspd_dat	50
4.19.1.4	sys_dat	50
4.20	MassBalance Class Reference	50
4.20.1	Constructor & Destructor Documentation	51
4.20.1.1	MassBalance	51
4.20.1.2	~MassBalance	51
4.20.2	Member Function Documentation	51
4.20.2.1	Display_Info	51
4.20.2.2	Eval_Residual	51
4.20.2.3	Get_Delta	51
4.20.2.4	Get_Name	51
4.20.2.5	Get_TotalConcentration	51
4.20.2.6	Initialize_List	51
4.20.2.7	Set_Delta	51
4.20.2.8	Set_Name	51
4.20.2.9	Set_TotalConcentration	51
4.20.2.10	Sum_Delta	51
4.20.3	Member Data Documentation	52
4.20.3.1	Delta	52
4.20.3.2	List	52
4.20.3.3	Name	52
4.20.3.4	TotalConcentration	52
4.21	MasterSpeciesList Class Reference	52
4.21.1	Constructor & Destructor Documentation	53
4.21.1.1	MasterSpeciesList	53
4.21.1.2	~MasterSpeciesList	53
4.21.1.3	MasterSpeciesList	53

4.21.2	Member Function Documentation	53
4.21.2.1	alkalinity	53
4.21.2.2	charge	53
4.21.2.3	DisplayAll	53
4.21.2.4	DisplayConcentrations	53
4.21.2.5	DisplayInfo	53
4.21.2.6	Eval_ChargeResidual	53
4.21.2.7	get_index	53
4.21.2.8	get_species	53
4.21.2.9	list_size	53
4.21.2.10	operator=	53
4.21.2.11	set_alkalinity	53
4.21.2.12	set_list_size	53
4.21.2.13	set_species	53
4.21.2.14	set_species	53
4.21.2.15	speciesName	53
4.21.3	Member Data Documentation	53
4.21.3.1	residual_alkalinity	53
4.21.3.2	size	53
4.21.3.3	species	53
4.22	Matrix< T > Class Template Reference	54
4.22.1	Constructor & Destructor Documentation	55
4.22.1.1	Matrix	55
4.22.1.2	Matrix	55
4.22.1.3	Matrix	55
4.22.1.4	~Matrix	55
4.22.2	Member Function Documentation	55
4.22.2.1	adjoint	55
4.22.2.2	cofactor	55
4.22.2.3	columnExtend	55
4.22.2.4	columnExtract	55
4.22.2.5	columnProjection	55
4.22.2.6	columnReplace	55
4.22.2.7	columns	55
4.22.2.8	columnShrink	55
4.22.2.9	columnVectorFill	55
4.22.2.10	ConstantICFill	55
4.22.2.11	determinate	55
4.22.2.12	diagonalSolve	55
4.22.2.13	dirichletBCFill	56

4.22.2.14	Display	56
4.22.2.15	edit	56
4.22.2.16	inner_product	56
4.22.2.17	IntegralAvg	56
4.22.2.18	IntegralTotal	56
4.22.2.19	inverse	56
4.22.2.20	ladshawSolve	56
4.22.2.21	lowerHessenberg2Triangular	56
4.22.2.22	lowerHessenbergSolve	56
4.22.2.23	lowerTriangularSolve	56
4.22.2.24	naturalLaplacian3D	56
4.22.2.25	norm	56
4.22.2.26	operator()	56
4.22.2.27	operator()	56
4.22.2.28	operator*	56
4.22.2.29	operator*	56
4.22.2.30	operator+	56
4.22.2.31	operator-	56
4.22.2.32	operator/	56
4.22.2.33	operator=	56
4.22.2.34	rowExtend	56
4.22.2.35	rowExtract	56
4.22.2.36	rowReplace	56
4.22.2.37	rows	56
4.22.2.38	rowShrink	56
4.22.2.39	set_size	56
4.22.2.40	SolnTransform	57
4.22.2.41	sphericalAvg	57
4.22.2.42	sphericalBCFill	57
4.22.2.43	sum	57
4.22.2.44	transpose	57
4.22.2.45	transpose_multiply	57
4.22.2.46	tridiagonalFill	57
4.22.2.47	tridiagonalSolve	57
4.22.2.48	tridiagonalVectorFill	57
4.22.2.49	upperHessenberg2Triangular	57
4.22.2.50	upperHessenbergSolve	57
4.22.2.51	upperTriangularSolve	57
4.22.2.52	zeros	57
4.22.3	Member Data Documentation	57

4.22.3.1	Data	57
4.22.3.2	num_cols	57
4.22.3.3	num_rows	57
4.23	Mechanism Class Reference	57
4.23.1	Member Data Documentation	58
4.23.1.1	List	58
4.23.1.2	reactions	58
4.23.1.3	species_index	58
4.23.1.4	weight	58
4.24	MIXED_GAS Struct Reference	58
4.24.1	Member Data Documentation	58
4.24.1.1	binary_diffusion	58
4.24.1.2	char_length	58
4.24.1.3	CheckMolefractions	58
4.24.1.4	gas_temperature	58
4.24.1.5	kinematic_viscosity	58
4.24.1.6	molefraction	58
4.24.1.7	N	58
4.24.1.8	Reynolds	59
4.24.1.9	species_dat	59
4.24.1.10	total_density	59
4.24.1.11	total_dyn_vis	59
4.24.1.12	total_molecular_weight	59
4.24.1.13	total_pressure	59
4.24.1.14	total_specific_heat	59
4.24.1.15	velocity	59
4.25	Molecule Class Reference	59
4.25.1	Constructor & Destructor Documentation	60
4.25.1.1	Molecule	60
4.25.1.2	~Molecule	60
4.25.1.3	Molecule	60
4.25.2	Member Function Documentation	60
4.25.2.1	calculateAvgOxiState	60
4.25.2.2	Charge	60
4.25.2.3	DisplayInfo	60
4.25.2.4	editAllOxidationStates	61
4.25.2.5	editCharge	61
4.25.2.6	editEnergy	61
4.25.2.7	editEnthalpy	61
4.25.2.8	editEntropy	61

4.25.2.9	editHS	61
4.25.2.10	editOneOxidationState	61
4.25.2.11	Energy	61
4.25.2.12	Enthalpy	61
4.25.2.13	Entropy	61
4.25.2.14	HaveEnergy	61
4.25.2.15	HaveHS	61
4.25.2.16	isRegistered	61
4.25.2.17	MolarWeight	61
4.25.2.18	MolecularFormula	61
4.25.2.19	MoleculeName	61
4.25.2.20	MoleculePhase	61
4.25.2.21	recalculateMolarWeight	61
4.25.2.22	Register	61
4.25.2.23	Register	61
4.25.2.24	removeAllAtoms	61
4.25.2.25	removeOneAtom	61
4.25.2.26	setFormula	61
4.25.2.27	setMolarWeigth	61
4.25.3	Member Data Documentation	61
4.25.3.1	atoms	61
4.25.3.2	charge	61
4.25.3.3	formation_energy	61
4.25.3.4	formation_enthalpy	62
4.25.3.5	formation_entropy	62
4.25.3.6	Formula	62
4.25.3.7	haveG	62
4.25.3.8	haveHS	62
4.25.3.9	molar_weight	62
4.25.3.10	Name	62
4.25.3.11	Phase	62
4.25.3.12	registered	62
4.26	MONKFISH_DATA Struct Reference	62
4.26.1	Member Data Documentation	63
4.26.1.1	avg_fiber_density	63
4.26.1.2	DirichletBC	63
4.26.1.3	dog_dat	63
4.26.1.4	domain_diameter	63
4.26.1.5	end_time	63
4.26.1.6	eval_ads	63

4.26.1.7 eval_Cex	63
4.26.1.8 eval_Dex	63
4.26.1.9 eval_eps	63
4.26.1.10 eval_kf	63
4.26.1.11 eval_Ret	63
4.26.1.12 eval_rho	63
4.26.1.13 finch_dat	63
4.26.1.14 haveMinMax	63
4.26.1.15 level	63
4.26.1.16 max_fiber_density	63
4.26.1.17 max_porosity	63
4.26.1.18 min_fiber_density	63
4.26.1.19 min_porosity	63
4.26.1.20 MultiScale	63
4.26.1.21 NonLinear	63
4.26.1.22 NumComp	64
4.26.1.23 Output	64
4.26.1.24 param_dat	64
4.26.1.25 Print2Console	64
4.26.1.26 Print2File	64
4.26.1.27 single_fiber_density	64
4.26.1.28 t_counter	64
4.26.1.29 t_print	64
4.26.1.30 time	64
4.26.1.31 time_old	64
4.26.1.32 total_sorption	64
4.26.1.33 total_sorption_old	64
4.26.1.34 total_steps	64
4.26.1.35 user_data	64
4.27 MONKFISH_PARAM Struct Reference	64
4.27.1 Member Data Documentation	65
4.27.1.1 avg_sorption	65
4.27.1.2 avg_sorption_old	65
4.27.1.3 exterior_concentration	65
4.27.1.4 exterior_transfer_coeff	65
4.27.1.5 film_transfer_coeff	65
4.27.1.6 initial_sorption	65
4.27.1.7 interparticle_diffusion	65
4.27.1.8 intraparticle_diffusion	65
4.27.1.9 sorbed_molefraction	65

4.27.1.10 sorption_bc	65
4.27.1.11 species	65
4.28 mSPD_DATA Struct Reference	65
4.28.1 Member Data Documentation	65
4.28.1.1 eMax	65
4.28.1.2 eta	65
4.28.1.3 gama	65
4.28.1.4 s	65
4.28.1.5 v	65
4.29 NUM_JAC_DATA Struct Reference	66
4.29.1 Detailed Description	66
4.29.2 Member Data Documentation	66
4.29.2.1 dxj	66
4.29.2.2 eps	66
4.29.2.3 Fx	66
4.29.2.4 Fxp	66
4.30 OPTRANS_DATA Struct Reference	67
4.30.1 Detailed Description	67
4.30.2 Member Data Documentation	67
4.30.2.1 Ai	67
4.30.2.2 li	67
4.31 PCG_DATA Struct Reference	67
4.31.1 Detailed Description	68
4.31.2 Member Data Documentation	68
4.31.2.1 alpha	68
4.31.2.2 Ap	68
4.31.2.3 bestres	68
4.31.2.4 bestx	69
4.31.2.5 beta	69
4.31.2.6 iter	69
4.31.2.7 maxit	69
4.31.2.8 Output	69
4.31.2.9 p	69
4.31.2.10 r	69
4.31.2.11 r_old	69
4.31.2.12 relres	69
4.31.2.13 relres_base	69
4.31.2.14 res	69
4.31.2.15 tol_abs	69
4.31.2.16 tol_rel	70

4.31.2.17 x	70
4.31.2.18 z	70
4.31.2.19 z_old	70
4.32 PeriodicTable Class Reference	70
4.32.1 Constructor & Destructor Documentation	71
4.32.1.1 PeriodicTable	71
4.32.1.2 ~PeriodicTable	71
4.32.1.3 PeriodicTable	71
4.32.1.4 PeriodicTable	71
4.32.1.5 PeriodicTable	71
4.32.2 Member Function Documentation	71
4.32.2.1 DisplayTable	71
4.32.3 Member Data Documentation	71
4.32.3.1 number_elements	71
4.32.3.2 Table	71
4.33 PICARD_DATA Struct Reference	71
4.33.1 Detailed Description	72
4.33.2 Member Data Documentation	72
4.33.2.1 bestres	72
4.33.2.2 bestx	72
4.33.2.3 iter	72
4.33.2.4 maxit	72
4.33.2.5 Output	72
4.33.2.6 r	72
4.33.2.7 relres	72
4.33.2.8 relres_base	72
4.33.2.9 res	73
4.33.2.10 tol_abs	73
4.33.2.11 tol_rel	73
4.33.2.12 x0	73
4.34 PJFNK_DATA Struct Reference	73
4.34.1 Detailed Description	75
4.34.2 Member Data Documentation	75
4.34.2.1 backtrack_dat	75
4.34.2.2 bestx	75
4.34.2.3 bicgstab_dat	75
4.34.2.4 Bounce	75
4.34.2.5 cgs_dat	75
4.34.2.6 eps	75
4.34.2.7 F	75

4.34.2.8	funeval	75
4.34.2.9	Fv	75
4.34.2.10	gcr_dat	75
4.34.2.11	gmreslp_dat	76
4.34.2.12	gmresr_dat	76
4.34.2.13	gmresrp_dat	76
4.34.2.14	l_iter	76
4.34.2.15	L_Output	76
4.34.2.16	lin_tol_abs	76
4.34.2.17	lin_tol_rel	76
4.34.2.18	linear_solver	76
4.34.2.19	LineSearch	76
4.34.2.20	nl_bestres	76
4.34.2.21	nl_iter	76
4.34.2.22	nl_maxit	76
4.34.2.23	NL_Output	77
4.34.2.24	nl_relres	77
4.34.2.25	nl_res	77
4.34.2.26	nl_res_base	77
4.34.2.27	nl_tol_abs	77
4.34.2.28	nl_tol_rel	77
4.34.2.29	pcg_dat	77
4.34.2.30	precon	77
4.34.2.31	precon_data	77
4.34.2.32	res_data	77
4.34.2.33	v	77
4.34.2.34	x	77
4.35	Precipitation Class Reference	78
4.36	PURE_GAS Struct Reference	78
4.36.1	Member Data Documentation	78
4.36.1.1	density	78
4.36.1.2	dynamic_viscosity	78
4.36.1.3	molecular_diffusion	78
4.36.1.4	molecular_weight	78
4.36.1.5	Schmidt	78
4.36.1.6	specific_heat	78
4.36.1.7	Sutherland_Const	78
4.36.1.8	Sutherland_Temp	79
4.36.1.9	Sutherland_Viscosity	79
4.37	Reaction Class Reference	79

4.37.1	Constructor & Destructor Documentation	80
4.37.1.1	Reaction	80
4.37.1.2	~Reaction	80
4.37.2	Member Function Documentation	80
4.37.2.1	calculateEnergies	80
4.37.2.2	calculateEquilibrium	80
4.37.2.3	checkSpeciesEnergies	80
4.37.2.4	Display_Info	80
4.37.2.5	Eval_Residual	80
4.37.2.6	Get_Energy	80
4.37.2.7	Get_Enthalpy	80
4.37.2.8	Get_Entropy	80
4.37.2.9	Get_Equilibrium	80
4.37.2.10	Get_Stoichiometric	80
4.37.2.11	haveEquilibrium	80
4.37.2.12	Initialize_List	80
4.37.2.13	Set_Energy	80
4.37.2.14	Set_Enthalpy	80
4.37.2.15	Set_EnthalpyANDEntropy	80
4.37.2.16	Set_Entropy	80
4.37.2.17	Set_Equilibrium	80
4.37.2.18	Set_Stoichiometric	80
4.37.3	Member Data Documentation	80
4.37.3.1	CanCalcG	80
4.37.3.2	CanCalcHS	81
4.37.3.3	energy	81
4.37.3.4	enthalpy	81
4.37.3.5	entropy	81
4.37.3.6	Equilibrium	81
4.37.3.7	HaveEquil	81
4.37.3.8	HaveG	81
4.37.3.9	HaveHS	81
4.37.3.10	List	81
4.37.3.11	Stoichiometric	81
4.38	SCOPSOWL_DATA Struct Reference	81
4.38.1	Member Data Documentation	82
4.38.1.1	binder_fraction	82
4.38.1.2	binder_poresize	82
4.38.1.3	binder_porosity	82
4.38.1.4	char_macro	82

4.38.1.5	char_micro	82
4.38.1.6	coord_macro	82
4.38.1.7	coord_micro	82
4.38.1.8	crystal_radius	82
4.38.1.9	DirichletBC	82
4.38.1.10	eval_ads	82
4.38.1.11	eval_diff	82
4.38.1.12	eval_kf	82
4.38.1.13	eval_retard	82
4.38.1.14	eval_surfDiff	82
4.38.1.15	finch_dat	82
4.38.1.16	gas_dat	82
4.38.1.17	gas_temperature	82
4.38.1.18	gas_velocity	83
4.38.1.19	Heterogeneous	83
4.38.1.20	level	83
4.38.1.21	magpie_dat	83
4.38.1.22	NonLinear	83
4.38.1.23	OutputFile	83
4.38.1.24	param_dat	83
4.38.1.25	pellet_density	83
4.38.1.26	pellet_radius	83
4.38.1.27	Print2Console	83
4.38.1.28	Print2File	83
4.38.1.29	sim_time	83
4.38.1.30	skua_dat	83
4.38.1.31	SurfDiff	83
4.38.1.32	t	83
4.38.1.33	t_counter	83
4.38.1.34	t_old	83
4.38.1.35	t_print	83
4.38.1.36	tempy	83
4.38.1.37	total_pressure	83
4.38.1.38	total_steps	83
4.38.1.39	user_data	83
4.38.1.40	y	83
4.39	SCOPSOWL_OPT_DATA Struct Reference	83
4.39.1	Member Data Documentation	84
4.39.1.1	abs_tol_bias	84
4.39.1.2	adsorb_index	84

4.39.1.3	CompareFile	84
4.39.1.4	current_equil	84
4.39.1.5	current_points	84
4.39.1.6	current_press	84
4.39.1.7	current_temp	84
4.39.1.8	diffusion_type	84
4.39.1.9	e_norm	84
4.39.1.10	e_norm_old	85
4.39.1.11	evaluation	85
4.39.1.12	f_bias	85
4.39.1.13	f_bias_old	85
4.39.1.14	max_bias	85
4.39.1.15	max_guess_iter	85
4.39.1.16	min_bias	85
4.39.1.17	num_curves	85
4.39.1.18	num_params	85
4.39.1.19	Optimize	85
4.39.1.20	owl_dat	85
4.39.1.21	param_guess	85
4.39.1.22	param_guess_old	85
4.39.1.23	ParamFile	85
4.39.1.24	q_data	85
4.39.1.25	q_sim	85
4.39.1.26	rel_tol_norm	85
4.39.1.27	Rough	85
4.39.1.28	simulation_equil	85
4.39.1.29	t	85
4.39.1.30	total_eval	85
4.39.1.31	y_base	85
4.40	SCOPSOWL_PARAM_DATA Struct Reference	85
4.40.1	Member Data Documentation	86
4.40.1.1	activation_energy	86
4.40.1.2	Adsorbable	86
4.40.1.3	affinity	86
4.40.1.4	dq_dc	86
4.40.1.5	dq_dco	86
4.40.1.6	film_transfer	86
4.40.1.7	pore_diffusion	86
4.40.1.8	qAvg	86
4.40.1.9	qAvg_old	86

4.40.1.10 qIntegralAvg	86
4.40.1.11 qIntegralAvg_old	86
4.40.1.12 qo	86
4.40.1.13 Qst	86
4.40.1.14 Qst_old	86
4.40.1.15 QstAvg	86
4.40.1.16 QstAvg_old	87
4.40.1.17 Qsto	87
4.40.1.18 ref_diffusion	87
4.40.1.19 ref_pressure	87
4.40.1.20 ref_temperature	87
4.40.1.21 speciesName	87
4.40.1.22 xIC	87
4.41 SHARK_DATA Struct Reference	87
4.41.1 Member Data Documentation	88
4.41.1.1 act_fun	88
4.41.1.2 activity_data	88
4.41.1.3 activity_new	88
4.41.1.4 activity_old	88
4.41.1.5 Conc_new	88
4.41.1.6 Conc_old	88
4.41.1.7 Console_Output	88
4.41.1.8 const_pH	88
4.41.1.9 Contains_pH	88
4.41.1.10 Contains_pOH	88
4.41.1.11 Converged	88
4.41.1.12 dielectric_const	88
4.41.1.13 dt	88
4.41.1.14 dt_min	89
4.41.1.15 EvalActivity	89
4.41.1.16 File_Output	89
4.41.1.17 lin_precon	89
4.41.1.18 MassBalanceList	89
4.41.1.19 MasterList	89
4.41.1.20 Newton_data	89
4.41.1.21 Norm	89
4.41.1.22 num_mbe	89
4.41.1.23 num_other	89
4.41.1.24 num_ssr	89
4.41.1.25 num_usr	89

4.41.1.26 numvar	89
4.41.1.27 other_data	89
4.41.1.28 OtherList	89
4.41.1.29 OutputFile	89
4.41.1.30 pH	89
4.41.1.31 pH_index	89
4.41.1.32 pOH_index	89
4.41.1.33 precon_data	89
4.41.1.34 ReactionList	89
4.41.1.35 Residual	89
4.41.1.36 residual_data	89
4.41.1.37 simulationtime	89
4.41.1.38 SpeciationCurve	89
4.41.1.39 steadystate	89
4.41.1.40 t_count	89
4.41.1.41 t_out	89
4.41.1.42 temperature	90
4.41.1.43 time	90
4.41.1.44 time_old	90
4.41.1.45 TimeAdaptivity	90
4.41.1.46 timesteps	90
4.41.1.47 totalsteps	90
4.41.1.48 UnsteadyList	90
4.41.1.49 X_new	90
4.41.1.50 X_old	90
4.41.1.51 yaml_object	90
4.42 SKUA_DATA Struct Reference	90
4.42.1 Member Data Documentation	91
4.42.1.1 char_measure	91
4.42.1.2 coord	91
4.42.1.3 DirichletBC	91
4.42.1.4 eval_diff	91
4.42.1.5 eval_kf	91
4.42.1.6 finch_dat	91
4.42.1.7 gas_dat	91
4.42.1.8 gas_velocity	91
4.42.1.9 magpie_dat	91
4.42.1.10 NonLinear	91
4.42.1.11 OutputFile	91
4.42.1.12 param_dat	91

4.42.1.13 pellet_radius	91
4.42.1.14 Print2Console	91
4.42.1.15 Print2File	91
4.42.1.16 qTn	91
4.42.1.17 qTnp1	91
4.42.1.18 sim_time	91
4.42.1.19 t	91
4.42.1.20 t_counter	91
4.42.1.21 t_old	91
4.42.1.22 t_print	91
4.42.1.23 total_steps	91
4.42.1.24 user_data	91
4.42.1.25 y	91
4.43 SKUA_OPT_DATA Struct Reference	92
4.43.1 Member Data Documentation	92
4.43.1.1 abs_tol_bias	92
4.43.1.2 adsorb_index	92
4.43.1.3 CompareFile	92
4.43.1.4 current_equil	92
4.43.1.5 current_points	92
4.43.1.6 current_press	92
4.43.1.7 current_temp	92
4.43.1.8 diffusion_type	93
4.43.1.9 e_norm	93
4.43.1.10 e_norm_old	93
4.43.1.11 evaluation	93
4.43.1.12 f_bias	93
4.43.1.13 f_bias_old	93
4.43.1.14 max_bias	93
4.43.1.15 max_guess_iter	93
4.43.1.16 min_bias	93
4.43.1.17 num_curves	93
4.43.1.18 num_params	93
4.43.1.19 Optimize	93
4.43.1.20 param_guess	93
4.43.1.21 param_guess_old	93
4.43.1.22 ParamFile	93
4.43.1.23 q_data	93
4.43.1.24 q_sim	93
4.43.1.25 rel_tol_norm	93

4.43.1.26 Rough	93
4.43.1.27 simulation_equil	93
4.43.1.28 skua_dat	93
4.43.1.29 t	93
4.43.1.30 total_eval	93
4.43.1.31 y_base	93
4.44 SKUA_PARAM Struct Reference	93
4.44.1 Member Data Documentation	94
4.44.1.1 activation_energy	94
4.44.1.2 Adsorbable	94
4.44.1.3 affinity	94
4.44.1.4 film_transfer	94
4.44.1.5 Qstn	94
4.44.1.6 Qstnp1	94
4.44.1.7 ref_diffusion	94
4.44.1.8 ref_pressure	94
4.44.1.9 ref_temperature	94
4.44.1.10 speciesName	94
4.44.1.11 xIC	94
4.44.1.12 xn	94
4.44.1.13 xnp1	94
4.44.1.14 y_eff	94
4.45 Speciation_Test01_Data Struct Reference	94
4.45.1 Member Data Documentation	95
4.45.1.1 C	95
4.45.1.2 CT	95
4.45.1.3 Jacobian	95
4.45.1.4 logC	95
4.45.1.5 logKa1	95
4.45.1.6 logKa2	95
4.45.1.7 logKw	95
4.45.1.8 N	95
4.45.1.9 NaT	95
4.45.1.10 NumJac	95
4.45.1.11 x	95
4.46 SubHeader Class Reference	95
4.46.1 Constructor & Destructor Documentation	96
4.46.1.1 SubHeader	96
4.46.1.2 ~SubHeader	96
4.46.1.3 SubHeader	96

4.46.1.4	SubHeader	96
4.46.1.5	SubHeader	96
4.46.1.6	SubHeader	96
4.46.2	Member Function Documentation	96
4.46.2.1	addPair	96
4.46.2.2	addPair	96
4.46.2.3	clear	97
4.46.2.4	DisplayContents	97
4.46.2.5	getAlias	97
4.46.2.6	getMap	97
4.46.2.7	getName	97
4.46.2.8	getState	97
4.46.2.9	isAlias	97
4.46.2.10	isAnchor	97
4.46.2.11	operator=	97
4.46.2.12	operator[]	97
4.46.2.13	operator[]	97
4.46.2.14	setAlias	97
4.46.2.15	setAlias	97
4.46.2.16	setName	97
4.46.2.17	setNameAliasPair	97
4.46.2.18	setState	97
4.46.3	Member Data Documentation	97
4.46.3.1	alias	97
4.46.3.2	Data_Map	97
4.46.3.3	name	97
4.46.3.4	state	97
4.47	SYSTEM_DATA Struct Reference	97
4.47.1	Member Data Documentation	98
4.47.1.1	As	98
4.47.1.2	avg_norm	98
4.47.1.3	Carrier	98
4.47.1.4	I	98
4.47.1.5	Ideal	98
4.47.1.6	J	98
4.47.1.7	K	98
4.47.1.8	max_norm	98
4.47.1.9	N	98
4.47.1.10	Output	98
4.47.1.11	Par	98

4.47.1.12 PI	98
4.47.1.13 pi	98
4.47.1.14 PT	98
4.47.1.15 qT	98
4.47.1.16 Recover	98
4.47.1.17 Sys	98
4.47.1.18 T	99
4.47.1.19 total_eval	99
4.48 TRAJECTORY_DATA Struct Reference	99
4.48.1 Member Data Documentation	100
4.48.1.1 a	100
4.48.1.2 A_separator	100
4.48.1.3 A_wire	100
4.48.1.4 b	100
4.48.1.5 B0	100
4.48.1.6 beta	100
4.48.1.7 Cap	100
4.48.1.8 chi_p	100
4.48.1.9 dt	100
4.48.1.10 dX	100
4.48.1.11 dY	100
4.48.1.12 eta	100
4.48.1.13 H	100
4.48.1.14 H0	100
4.48.1.15 Hamaker	100
4.48.1.16 k	100
4.48.1.17 L	100
4.48.1.18 L_wire	100
4.48.1.19 M	100
4.48.1.20 m_rand	100
4.48.1.21 mp	100
4.48.1.22 Ms	100
4.48.1.23 mu_0	100
4.48.1.24 n_rand	101
4.48.1.25 POL	101
4.48.1.26 porosity	101
4.48.1.27 q_bar	101
4.48.1.28 Q_in	101
4.48.1.29 rho_f	101
4.48.1.30 rho_p	101

4.48.1.31	Rs	101
4.48.1.32	s_rand	101
4.48.1.33	sigma_m	101
4.48.1.34	sigma_n	101
4.48.1.35	sigma_v	101
4.48.1.36	sigma_vz	101
4.48.1.37	sigma_z	101
4.48.1.38	t_rand	101
4.48.1.39	Temp	101
4.48.1.40	V0	101
4.48.1.41	V_separator	101
4.48.1.42	V_wire	101
4.48.1.43	X	101
4.48.1.44	Y	101
4.48.1.45	Y_initial	101
4.49	UI_DATA Struct Reference	101
4.49.1	Detailed Description	102
4.49.2	Member Data Documentation	102
4.49.2.1	argc	102
4.49.2.2	argv	102
4.49.2.3	BasicUI	102
4.49.2.4	count	103
4.49.2.5	Files	103
4.49.2.6	input_files	103
4.49.2.7	max	103
4.49.2.8	MissingArg	103
4.49.2.9	option	103
4.49.2.10	path	103
4.49.2.11	Path	103
4.49.2.12	user_input	103
4.49.2.13	value_type	103
4.50	UnsteadyPrecipitation Class Reference	103
4.51	UnsteadyReaction Class Reference	104
4.51.1	Constructor & Destructor Documentation	105
4.51.1.1	UnsteadyReaction	105
4.51.1.2	~UnsteadyReaction	105
4.51.2	Member Function Documentation	105
4.51.2.1	calculateEnergies	105
4.51.2.2	calculateEquilibrium	105
4.51.2.3	calculateRate	105

4.51.2.4	checkSpeciesEnergies	106
4.51.2.5	Display_Info	106
4.51.2.6	Eval_IC_Residual	106
4.51.2.7	Eval_ReactionRate	106
4.51.2.8	Eval_Residual	106
4.51.2.9	Eval_Residual	106
4.51.2.10	Explicit_Eval	106
4.51.2.11	Get_ActivationEnergy	106
4.51.2.12	Get_Affinity	106
4.51.2.13	Get_Energy	106
4.51.2.14	Get_Enthalpy	106
4.51.2.15	Get_Entropy	106
4.51.2.16	Get_Equilibrium	106
4.51.2.17	Get_Forward	106
4.51.2.18	Get_ForwardRef	106
4.51.2.19	Get_InitialValue	106
4.51.2.20	Get_MaximumValue	106
4.51.2.21	Get_Reverse	106
4.51.2.22	Get_ReverseRef	106
4.51.2.23	Get_Species_Index	106
4.51.2.24	Get_Stoichiometric	106
4.51.2.25	Get_TimeStep	106
4.51.2.26	haveEquilibrium	106
4.51.2.27	haveRate	106
4.51.2.28	Initialize_List	106
4.51.2.29	Set_ActivationEnergy	106
4.51.2.30	Set_Affinity	106
4.51.2.31	Set_Energy	106
4.51.2.32	Set_Enthalpy	107
4.51.2.33	Set_EnthalpyANDEntropy	107
4.51.2.34	Set_Entropy	107
4.51.2.35	Set_Equilibrium	107
4.51.2.36	Set_Forward	107
4.51.2.37	Set_ForwardRef	107
4.51.2.38	Set_InitialValue	107
4.51.2.39	Set_MaximumValue	107
4.51.2.40	Set_Reverse	107
4.51.2.41	Set_ReverseRef	107
4.51.2.42	Set_Species_Index	107
4.51.2.43	Set_Species_Index	107

4.51.2.44	Set_Stoichiometric	107
4.51.2.45	Set_TimeStep	107
4.51.3	Member Data Documentation	107
4.51.3.1	activation_energy	107
4.51.3.2	forward_rate	107
4.51.3.3	forward_ref_rate	107
4.51.3.4	HaveForRef	107
4.51.3.5	HaveForward	107
4.51.3.6	HaveReverse	107
4.51.3.7	HaveRevRef	107
4.51.3.8	initial_value	107
4.51.3.9	max_value	107
4.51.3.10	reverse_rate	107
4.51.3.11	reverse_ref_rate	107
4.51.3.12	species_index	107
4.51.3.13	temperature_affinity	107
4.51.3.14	time_step	108
4.52	ValueTypePair Class Reference	108
4.52.1	Constructor & Destructor Documentation	108
4.52.1.1	ValueTypePair	108
4.52.1.2	~ValueTypePair	108
4.52.1.3	ValueTypePair	108
4.52.1.4	ValueTypePair	108
4.52.1.5	ValueTypePair	108
4.52.2	Member Function Documentation	108
4.52.2.1	assertType	109
4.52.2.2	DisplayPair	109
4.52.2.3	editPair	109
4.52.2.4	editValue	109
4.52.2.5	findType	109
4.52.2.6	getBool	109
4.52.2.7	getDouble	109
4.52.2.8	getInt	109
4.52.2.9	getPair	109
4.52.2.10	getString	109
4.52.2.11	getType	109
4.52.2.12	getValue	109
4.52.2.13	operator=	109
4.52.3	Member Data Documentation	109
4.52.3.1	type	109

4.52.3.2	Value_Type	109
4.53	yaml_cpp_class Class Reference	109
4.53.1	Constructor & Destructor Documentation	110
4.53.1.1	yaml_cpp_class	110
4.53.1.2	~yaml_cpp_class	110
4.53.2	Member Function Documentation	110
4.53.2.1	cleanup	110
4.53.2.2	DisplayContents	110
4.53.2.3	executeYamlRead	110
4.53.2.4	getYamlWrapper	110
4.53.2.5	readInputFile	110
4.53.2.6	setInputFile	110
4.53.3	Member Data Documentation	110
4.53.3.1	current_token	110
4.53.3.2	file_name	110
4.53.3.3	input_file	110
4.53.3.4	previous_token	110
4.53.3.5	token_parser	110
4.53.3.6	yaml_wrapper	110
4.54	YamlWrapper Class Reference	110
4.54.1	Constructor & Destructor Documentation	111
4.54.1.1	YamlWrapper	111
4.54.1.2	~YamlWrapper	111
4.54.1.3	YamlWrapper	111
4.54.1.4	YamlWrapper	111
4.54.2	Member Function Documentation	111
4.54.2.1	addDocKey	111
4.54.2.2	begin	111
4.54.2.3	begin	111
4.54.2.4	changeKey	111
4.54.2.5	clear	112
4.54.2.6	copyAnchor2Alias	112
4.54.2.7	DisplayContents	112
4.54.2.8	end	112
4.54.2.9	end	112
4.54.2.10	getAnchoredDoc	112
4.54.2.11	getDocFromHeadAlias	112
4.54.2.12	getDocFromSubAlias	112
4.54.2.13	getDocMap	112
4.54.2.14	getDocument	112

4.54.2.15 operator()	112
4.54.2.16 operator()	112
4.54.2.17 operator=	112
4.54.2.18 resetKeys	112
4.54.2.19 revalidateAllKeys	112
4.54.2.20 size	112
4.54.3 Member Data Documentation	112
4.54.3.1 Doc_Map	112
5 File Documentation	113
5.1 dogfish.cpp File Reference	113
5.1.1 Function Documentation	113
5.1.1.1 default_FilmMTCoeff	113
5.1.1.2 default_IntraDiffusion	113
5.1.1.3 default_Retardation	113
5.1.1.4 default_SurfaceConcentration	113
5.1.1.5 DOGFISH	114
5.1.1.6 DOGFISH_Executioner	114
5.1.1.7 DOGFISH_postprocesses	114
5.1.1.8 DOGFISH_preprocesses	114
5.1.1.9 DOGFISH_reset	114
5.1.1.10 DOGFISH_TESTS	114
5.1.1.11 print2file_DOGFISH_header	114
5.1.1.12 print2file_DOGFISH_result_new	114
5.1.1.13 print2file_DOGFISH_result_old	114
5.1.1.14 print2file_species_header	114
5.1.1.15 set_DOGFISH_ICs	114
5.1.1.16 set_DOGFISH_params	114
5.1.1.17 set_DOGFISH_timestep	114
5.1.1.18 setup_DOGFISH_DATA	114
5.2 dogfish.h File Reference	114
5.2.1 Function Documentation	115
5.2.1.1 default_FilmMTCoeff	115
5.2.1.2 default_IntraDiffusion	115
5.2.1.3 default_Retardation	115
5.2.1.4 default_SurfaceConcentration	115
5.2.1.5 DOGFISH	115
5.2.1.6 DOGFISH_Executioner	115
5.2.1.7 DOGFISH_postprocesses	115
5.2.1.8 DOGFISH_preprocesses	115

5.2.1.9	DOGFISH_reset	115
5.2.1.10	DOGFISH_TESTS	115
5.2.1.11	print2file_DOGFISH_header	115
5.2.1.12	print2file_DOGFISH_result_new	115
5.2.1.13	print2file_DOGFISH_result_old	115
5.2.1.14	print2file_species_header	115
5.2.1.15	set_DOGFISH_ICs	115
5.2.1.16	set_DOGFISH_params	115
5.2.1.17	set_DOGFISH_timestep	115
5.2.1.18	setup_DOGFISH_DATA	115
5.3	eel.cpp File Reference	115
5.3.1	Function Documentation	116
5.3.1.1	EEL_TESTS	116
5.4	eel.h File Reference	116
5.4.1	Function Documentation	116
5.4.1.1	EEL_TESTS	116
5.5	egret.cpp File Reference	116
5.5.1	Function Documentation	116
5.5.1.1	calculate_properties	116
5.5.1.2	EGRET_TESTS	117
5.5.1.3	initialize_data	117
5.5.1.4	set_variables	117
5.6	egret.h File Reference	117
5.6.1	Macro Definition Documentation	117
5.6.1.1	CE3	117
5.6.1.2	Cstd	117
5.6.1.3	D_ii	117
5.6.1.4	D_ij	118
5.6.1.5	Dp_ij	118
5.6.1.6	FilmMTCoeff	118
5.6.1.7	Mu	118
5.6.1.8	Nu	118
5.6.1.9	PE3	118
5.6.1.10	Po	118
5.6.1.11	PSI	118
5.6.1.12	Pstd	118
5.6.1.13	RE3	118
5.6.1.14	ReNum	118
5.6.1.15	Rstd	118
5.6.1.16	ScNum	118

5.6.2	Function Documentation	118
5.6.2.1	calculate_properties	118
5.6.2.2	EGRET_TESTS	118
5.6.2.3	initialize_data	118
5.6.2.4	set_variables	118
5.7	error.cpp File Reference	118
5.7.1	Function Documentation	118
5.7.1.1	error	118
5.8	error.h File Reference	118
5.8.1	Macro Definition Documentation	119
5.8.1.1	mError	119
5.8.2	Enumeration Type Documentation	119
5.8.2.1	error_type	119
5.8.3	Function Documentation	120
5.8.3.1	error	120
5.9	finch.cpp File Reference	120
5.9.1	Function Documentation	121
5.9.1.1	buckley_leverett_ic	121
5.9.1.2	buckley_leverett_params	121
5.9.1.3	burgers_bcs	121
5.9.1.4	burgers_ic	121
5.9.1.5	burgers_params	122
5.9.1.6	check_Mass	122
5.9.1.7	default_bcs	122
5.9.1.8	default_execution	122
5.9.1.9	default_ic	122
5.9.1.10	default_params	122
5.9.1.11	default_postprocess	122
5.9.1.12	default_precon	122
5.9.1.13	default_preprocess	122
5.9.1.14	default_res	122
5.9.1.15	default_reset	122
5.9.1.16	default_solve	122
5.9.1.17	default_timestep	122
5.9.1.18	FINCH_TESTS	122
5.9.1.19	I_direct	122
5.9.1.20	lark_picard_step	122
5.9.1.21	max	122
5.9.1.22	min	122
5.9.1.23	minmod	122

5.9.1.24	minmod_discretization	122
5.9.1.25	nl_picard	122
5.9.1.26	ospre_discretization	122
5.9.1.27	print2file_dim_header	122
5.9.1.28	print2file_newline	122
5.9.1.29	print2file_result_new	122
5.9.1.30	print2file_result_old	122
5.9.1.31	print2file_tab	122
5.9.1.32	print2file_time_header	122
5.9.1.33	setup_FINCH_DATA	123
5.9.1.34	uAverage	123
5.9.1.35	uTotal	123
5.9.1.36	vanAlbada_discretization	123
5.10	finch.h File Reference	123
5.10.1	Macro Definition Documentation	124
5.10.1.1	Cartesian	124
5.10.1.2	Cylindrical	124
5.10.1.3	FINCH_Picard	124
5.10.1.4	LARK_Picard	124
5.10.1.5	LARK_PJFNK	124
5.10.1.6	Spherical	124
5.10.2	Function Documentation	124
5.10.2.1	buckley_leverett_ic	124
5.10.2.2	buckley_leverett_params	124
5.10.2.3	burgers_bcs	124
5.10.2.4	burgers_ic	124
5.10.2.5	burgers_params	124
5.10.2.6	check_Mass	124
5.10.2.7	default_bcs	124
5.10.2.8	default_execution	125
5.10.2.9	default_ic	125
5.10.2.10	default_params	125
5.10.2.11	default_postprocess	125
5.10.2.12	default_precon	125
5.10.2.13	default_preprocess	125
5.10.2.14	default_res	125
5.10.2.15	default_reset	125
5.10.2.16	default_solve	125
5.10.2.17	default_timestep	125
5.10.2.18	FINCH_TESTS	125

5.10.2.19	l_direct	125
5.10.2.20	lark_picard_step	125
5.10.2.21	max	125
5.10.2.22	min	125
5.10.2.23	minmod	125
5.10.2.24	minmod_discretization	125
5.10.2.25	nl_picard	125
5.10.2.26	ospre_discretization	125
5.10.2.27	print2file_dim_header	125
5.10.2.28	print2file_newline	125
5.10.2.29	print2file_result_new	125
5.10.2.30	print2file_result_old	125
5.10.2.31	print2file_tab	125
5.10.2.32	print2file_time_header	125
5.10.2.33	setup_FINCH_DATA	126
5.10.2.34	uAverage	126
5.10.2.35	uTotal	126
5.10.2.36	vanAlbada_discretization	126
5.11	flock.h File Reference	126
5.12	gsta_opt.cpp File Reference	126
5.12.1	Function Documentation	127
5.12.1.1	avgPar	127
5.12.1.2	avgValue	127
5.12.1.3	eduGuess	127
5.12.1.4	eval_GSTA	127
5.12.1.5	gsta_optimize	127
5.12.1.6	gstaFunc	127
5.12.1.7	gstaObjFunc	127
5.12.1.8	isSmooth	127
5.12.1.9	minIndex	127
5.12.1.10	minValue	127
5.12.1.11	orderMag	127
5.12.1.12	orthoLinReg	127
5.12.1.13	roundIt	127
5.12.1.14	rSq	127
5.12.1.15	twoFifths	127
5.12.1.16	weightedAvg	127
5.13	gsta_opt.h File Reference	127
5.13.1	Macro Definition Documentation	128
5.13.1.1	Na	128

5.13.1.2	Po	128
5.13.1.3	R	128
5.13.2	Function Documentation	128
5.13.2.1	avgPar	128
5.13.2.2	avgValue	128
5.13.2.3	eduGuess	128
5.13.2.4	error	128
5.13.2.5	eval_GSTA	128
5.13.2.6	gsta_optimize	128
5.13.2.7	gstaFunc	128
5.13.2.8	gstaObjFunc	128
5.13.2.9	isSmooth	128
5.13.2.10	minIndex	129
5.13.2.11	minValue	129
5.13.2.12	orderMag	129
5.13.2.13	orthoLinReg	129
5.13.2.14	roundIt	129
5.13.2.15	rSq	129
5.13.2.16	twoFifths	129
5.13.2.17	weightedAvg	129
5.14	lark.cpp File Reference	129
5.14.1	Detailed Description	130
5.14.2	Function Documentation	130
5.14.2.1	arnoldi	130
5.14.2.2	backtrackLineSearch	130
5.14.2.3	bicgstab	130
5.14.2.4	cgs	130
5.14.2.5	fom	131
5.14.2.6	gcr	131
5.14.2.7	gmresLeftPreconditioned	131
5.14.2.8	gmresPreconditioner	131
5.14.2.9	gmresr	131
5.14.2.10	gmresRightPreconditioned	131
5.14.2.11	jacvec	131
5.14.2.12	LARK_TESTS	131
5.14.2.13	NumericalJacobian	131
5.14.2.14	operatorTranspose	131
5.14.2.15	pcg	131
5.14.2.16	picard	131
5.14.2.17	pjfnk	131

5.14.2.18	update_arnoldi_solution	131
5.15	lark.h File Reference	131
5.15.1	Detailed Description	133
5.15.2	Enumeration Type Documentation	134
5.15.2.1	krylov_method	134
5.15.3	Function Documentation	135
5.15.3.1	arnoldi	135
5.15.3.2	backtrackLineSearch	135
5.15.3.3	bicgstab	135
5.15.3.4	cgs	135
5.15.3.5	fom	135
5.15.3.6	gcr	135
5.15.3.7	gmresLeftPreconditioned	135
5.15.3.8	gmresPreconditioner	135
5.15.3.9	gmresr	135
5.15.3.10	gmresRightPreconditioned	135
5.15.3.11	jacvec	135
5.15.3.12	LARK_TESTS	135
5.15.3.13	NumericalJacobian	135
5.15.3.14	operatorTranspose	136
5.15.3.15	pcg	136
5.15.3.16	picard	136
5.15.3.17	pjfnk	136
5.15.3.18	update_arnoldi_solution	136
5.16	macaw.cpp File Reference	136
5.16.1	Function Documentation	136
5.16.1.1	MACAW_TESTS	136
5.17	macaw.h File Reference	136
5.17.1	Macro Definition Documentation	137
5.17.1.1	M_PI	137
5.17.2	Function Documentation	137
5.17.2.1	MACAW_TESTS	137
5.18	magpie.cpp File Reference	137
5.18.1	Function Documentation	137
5.18.1.1	dq_dp	137
5.18.1.2	eMax	137
5.18.1.3	eval_eta	137
5.18.1.4	eval_GPAST	137
5.18.1.5	eval_po	137
5.18.1.6	eval_po_PI	137

5.18.1.7	eval_po_qo	137
5.18.1.8	grad_mSPD	138
5.18.1.9	initialGuess_mSPD	138
5.18.1.10	lnact_mSPD	138
5.18.1.11	MAGPIE	138
5.18.1.12	MAGPIE_SCENARIOS	138
5.18.1.13	PI	138
5.18.1.14	q_p	138
5.18.1.15	qo	138
5.18.1.16	Qst	138
5.18.1.17	qT	138
5.19	magpie.h File Reference	138
5.19.1	Macro Definition Documentation	139
5.19.1.1	A	139
5.19.1.2	DBL_EPSILON	139
5.19.1.3	He	139
5.19.1.4	kB	139
5.19.1.5	lnKo	139
5.19.1.6	Na	139
5.19.1.7	Po	139
5.19.1.8	R	139
5.19.1.9	shapeFactor	139
5.19.1.10	V	139
5.19.1.11	Z	139
5.19.2	Function Documentation	139
5.19.2.1	dq_dp	139
5.19.2.2	eMax	139
5.19.2.3	eval_eta	139
5.19.2.4	eval_GPAST	140
5.19.2.5	eval_po	140
5.19.2.6	eval_po_PI	140
5.19.2.7	eval_po_qo	140
5.19.2.8	grad_mSPD	140
5.19.2.9	initialGuess_mSPD	140
5.19.2.10	lnact_mSPD	140
5.19.2.11	MAGPIE	140
5.19.2.12	MAGPIE_SCENARIOS	140
5.19.2.13	PI	140
5.19.2.14	q_p	140
5.19.2.15	qo	140

5.19.2.16 Qst	140
5.19.2.17 qT	140
5.20 main.cpp File Reference	140
5.20.1 Detailed Description	140
5.20.2 Function Documentation	141
5.20.2.1 main	141
5.21 mola.cpp File Reference	141
5.21.1 Function Documentation	141
5.21.1.1 MOLA_TESTS	141
5.22 mola.h File Reference	141
5.22.1 Function Documentation	141
5.22.1.1 MOLA_TESTS	141
5.23 monkfish.cpp File Reference	141
5.23.1 Function Documentation	142
5.23.1.1 default_density	142
5.23.1.2 default_exterior_concentration	142
5.23.1.3 default_film_transfer	142
5.23.1.4 default_interparticle_diffusion	142
5.23.1.5 default_monk_adsorption	142
5.23.1.6 default_monk_equilibrium	142
5.23.1.7 default_monkfish_retardation	142
5.23.1.8 default_porosity	142
5.23.1.9 MONKFISH_TESTS	142
5.24 monkfish.h File Reference	142
5.24.1 Function Documentation	143
5.24.1.1 default_density	143
5.24.1.2 default_exterior_concentration	143
5.24.1.3 default_film_transfer	143
5.24.1.4 default_interparticle_diffusion	143
5.24.1.5 default_monk_adsorption	143
5.24.1.6 default_monk_equilibrium	143
5.24.1.7 default_monkfish_retardation	143
5.24.1.8 default_porosity	143
5.24.1.9 MONKFISH_TESTS	143
5.24.1.10 setup_MONKFISH_DATA	143
5.25 sandbox.cpp File Reference	143
5.25.1 Function Documentation	143
5.25.1.1 RUN_SANDBOX	143
5.25.1.2 Speciation_Test01_Function	144
5.25.1.3 Speciation_Test01_Guess	144

5.25.1.4	Speciation_Test01_Jacobian	144
5.25.1.5	Speciation_Test01_MatVec	144
5.26	sandbox.h File Reference	144
5.26.1	Function Documentation	144
5.26.1.1	RUN_SANDBOX	144
5.26.1.2	Speciation_Test01_Function	144
5.26.1.3	Speciation_Test01_Guess	144
5.26.1.4	Speciation_Test01_Jacobian	144
5.26.1.5	Speciation_Test01_MatVec	144
5.27	school.h File Reference	144
5.28	scopsowl.cpp File Reference	145
5.28.1	Function Documentation	145
5.28.1.1	const_filmMassTransfer	145
5.28.1.2	const_pore_diffusion	145
5.28.1.3	CURVE_TEST03	145
5.28.1.4	CURVE_TEST04	145
5.28.1.5	CURVE_TEST05	145
5.28.1.6	default_adsorption	146
5.28.1.7	default_effective_diffusion	146
5.28.1.8	default_filmMassTransfer	146
5.28.1.9	default_pore_diffusion	146
5.28.1.10	default_retardation	146
5.28.1.11	default_surf_diffusion	146
5.28.1.12	LARGE_CYCLE_TEST01	146
5.28.1.13	print2file_SCOPSOWL_header	146
5.28.1.14	print2file_SCOPSOWL_result_new	146
5.28.1.15	print2file_SCOPSOWL_result_old	146
5.28.1.16	print2file_SCOPSOWL_time_header	146
5.28.1.17	print2file_species_header	146
5.28.1.18	SCOPSOWL	146
5.28.1.19	SCOPSOWL_Executioner	146
5.28.1.20	SCOPSOWL_postprocesses	146
5.28.1.21	SCOPSOWL_preprocesses	146
5.28.1.22	SCOPSOWL_reset	146
5.28.1.23	SCOPSOWL_SCENARIOS	146
5.28.1.24	SCOPSOWL_TESTS	146
5.28.1.25	set_SCOPSOWL_ICs	146
5.28.1.26	set_SCOPSOWL_params	146
5.28.1.27	set_SCOPSOWL_timestep	146
5.28.1.28	setup_SCOPSOWL_DATA	146

5.28.1.29 SMALL_CYCLE_TEST02	146
5.29 scopsowl.h File Reference	146
5.29.1 Macro Definition Documentation	148
5.29.1.1 avgDp	148
5.29.1.2 Dk	148
5.29.1.3 Dp	148
5.29.1.4 SCOPSOWL_HPP_	148
5.29.2 Function Documentation	148
5.29.2.1 const_filmMassTransfer	148
5.29.2.2 const_pore_diffusion	148
5.29.2.3 CURVE_TEST03	148
5.29.2.4 CURVE_TEST04	148
5.29.2.5 CURVE_TEST05	148
5.29.2.6 default_adsorption	148
5.29.2.7 default_effective_diffusion	148
5.29.2.8 default_filmMassTransfer	148
5.29.2.9 default_pore_diffusion	148
5.29.2.10 default_retardation	148
5.29.2.11 default_surf_diffusion	148
5.29.2.12 LARGE_CYCLE_TEST01	148
5.29.2.13 print2file_SCOPSOWL_header	148
5.29.2.14 print2file_SCOPSOWL_result_new	148
5.29.2.15 print2file_SCOPSOWL_result_old	148
5.29.2.16 print2file_SCOPSOWL_time_header	148
5.29.2.17 print2file_species_header	148
5.29.2.18 SCOPSOWL	148
5.29.2.19 SCOPSOWL_Executioner	148
5.29.2.20 SCOPSOWL_postprocesses	148
5.29.2.21 SCOPSOWL_preprocesses	148
5.29.2.22 SCOPSOWL_reset	148
5.29.2.23 SCOPSOWL_SCENARIOS	149
5.29.2.24 SCOPSOWL_TESTS	149
5.29.2.25 set_SCOPSOWL_ICs	149
5.29.2.26 set_SCOPSOWL_params	149
5.29.2.27 set_SCOPSOWL_timestep	149
5.29.2.28 setup_SCOPSOWL_DATA	149
5.29.2.29 SMALL_CYCLE_TEST02	149
5.30 scopsowl_opt.cpp File Reference	149
5.30.1 Function Documentation	149
5.30.1.1 eval_SCOPSOWL_Uptake	149

5.30.1.2	initial_guess_SCOPSOWL	149
5.30.1.3	SCOPSOWL_OPT_set_y	149
5.30.1.4	SCOPSOWL_OPTIMIZE	149
5.31	scopsowl_opt.h File Reference	149
5.31.1	Function Documentation	150
5.31.1.1	eval_SCOPSOWL_Uptake	150
5.31.1.2	initial_guess_SCOPSOWL	150
5.31.1.3	SCOPSOWL_OPT_set_y	150
5.31.1.4	SCOPSOWL_OPTIMIZE	150
5.32	shark.cpp File Reference	150
5.32.1	Function Documentation	151
5.32.1.1	act_choice	151
5.32.1.2	Convert2Concentration	151
5.32.1.3	Convert2LogConcentration	151
5.32.1.4	Davies_equation	151
5.32.1.5	DaviesLadshaw_equation	151
5.32.1.6	DebyeHuckel_equation	151
5.32.1.7	ideal_solution	151
5.32.1.8	linearsolve_choice	151
5.32.1.9	linsearch_choice	151
5.32.1.10	print2file_shark_header	151
5.32.1.11	print2file_shark_info	151
5.32.1.12	print2file_shark_results_new	151
5.32.1.13	print2file_shark_results_old	151
5.32.1.14	read_equilrxn	151
5.32.1.15	read_massbalance	151
5.32.1.16	read_options	151
5.32.1.17	read_scenario	151
5.32.1.18	read_species	151
5.32.1.19	read_unsteadyrxn	151
5.32.1.20	setup_SHARK_DATA	152
5.32.1.21	SHARK	152
5.32.1.22	shark_add_customResidual	152
5.32.1.23	shark_energy_calculations	152
5.32.1.24	shark_executioner	152
5.32.1.25	shark_guess	152
5.32.1.26	shark_initial_conditions	152
5.32.1.27	shark_parameter_check	152
5.32.1.28	shark_pH_finder	152
5.32.1.29	shark_postprocesses	152

5.32.1.30	shark_preprocesses	152
5.32.1.31	shark_reset	152
5.32.1.32	shark_residual	152
5.32.1.33	SHARK_SCENARIO	152
5.32.1.34	shark_solver	152
5.32.1.35	shark_temperature_calculations	152
5.32.1.36	SHARK_TESTS	152
5.32.1.37	shark_timestep_adapt	152
5.32.1.38	shark_timestep_const	152
5.33	shark.h File Reference	152
5.33.1	Macro Definition Documentation	154
5.33.1.1	Rstd	154
5.33.2	Typedef Documentation	154
5.33.2.1	SHARK_DATA	154
5.33.3	Enumeration Type Documentation	154
5.33.3.1	valid_act	154
5.33.4	Function Documentation	154
5.33.4.1	act_choice	154
5.33.4.2	Convert2Concentration	154
5.33.4.3	Convert2LogConcentration	154
5.33.4.4	Davies_equation	154
5.33.4.5	DaviesLadshaw_equation	154
5.33.4.6	DebyeHuckel_equation	154
5.33.4.7	ideal_solution	154
5.33.4.8	linearsolve_choice	154
5.33.4.9	linsearch_choice	154
5.33.4.10	print2file_shark_header	155
5.33.4.11	print2file_shark_info	155
5.33.4.12	print2file_shark_results_new	155
5.33.4.13	print2file_shark_results_old	155
5.33.4.14	read_equilrxn	155
5.33.4.15	read_massbalance	155
5.33.4.16	read_options	155
5.33.4.17	read_scenario	155
5.33.4.18	read_species	155
5.33.4.19	read_unsteadyrxn	155
5.33.4.20	setup_SHARK_DATA	155
5.33.4.21	SHARK	155
5.33.4.22	shark_add_customResidual	155
5.33.4.23	shark_energy_calculations	155

5.33.4.24	shark_executioner	155
5.33.4.25	shark_guess	155
5.33.4.26	shark_initial_conditions	155
5.33.4.27	shark_parameter_check	155
5.33.4.28	shark_pH_finder	155
5.33.4.29	shark_postprocesses	155
5.33.4.30	shark_preprocesses	155
5.33.4.31	shark_reset	155
5.33.4.32	shark_residual	155
5.33.4.33	SHARK_SCENARIO	155
5.33.4.34	shark_solver	155
5.33.4.35	shark_temperature_calculations	155
5.33.4.36	SHARK_TESTS	156
5.33.4.37	shark_timestep_adapt	156
5.33.4.38	shark_timestep_const	156
5.34	skua.cpp File Reference	156
5.34.1	Function Documentation	156
5.34.1.1	const_Dc	156
5.34.1.2	const_kf	156
5.34.1.3	default_Dc	156
5.34.1.4	default_kf	156
5.34.1.5	empirical_kf	157
5.34.1.6	molefractionCheck	157
5.34.1.7	print2file_SKUA_header	157
5.34.1.8	print2file_SKUA_results_new	157
5.34.1.9	print2file_SKUA_results_old	157
5.34.1.10	print2file_SKUA_time_header	157
5.34.1.11	print2file_species_header	157
5.34.1.12	set_SKUA_ICs	157
5.34.1.13	set_SKUA_params	157
5.34.1.14	set_SKUA_timestep	157
5.34.1.15	setup_SKUA_DATA	157
5.34.1.16	simple_darken_Dc	157
5.34.1.17	SKUA	157
5.34.1.18	SKUA_CYCLE_TEST01	157
5.34.1.19	SKUA_CYCLE_TEST02	157
5.34.1.20	SKUA_Executioner	157
5.34.1.21	SKUA_LOW_TEST03	157
5.34.1.22	SKUA_MID_TEST04	157
5.34.1.23	SKUA_postprocesses	157

5.34.1.24	SKUA_preprocesses	157
5.34.1.25	SKUA_reset	157
5.34.1.26	SKUA_SCENARIOS	157
5.34.1.27	SKUA_TESTS	157
5.34.1.28	theoretical_darken_Dc	157
5.35	skua.h File Reference	157
5.35.1	Macro Definition Documentation	158
5.35.1.1	D_c	158
5.35.1.2	D_inf	158
5.35.1.3	D_o	158
5.35.1.4	SKUA_HPP_	158
5.35.2	Function Documentation	159
5.35.2.1	const_Dc	159
5.35.2.2	const_kf	159
5.35.2.3	default_Dc	159
5.35.2.4	default_kf	159
5.35.2.5	empirical_kf	159
5.35.2.6	molefractionCheck	159
5.35.2.7	print2file_SKUA_header	159
5.35.2.8	print2file_SKUA_results_new	159
5.35.2.9	print2file_SKUA_results_old	159
5.35.2.10	print2file_SKUA_time_header	159
5.35.2.11	print2file_species_header	159
5.35.2.12	set_SKUA_ICs	159
5.35.2.13	set_SKUA_params	159
5.35.2.14	set_SKUA_timestep	159
5.35.2.15	setup_SKUA_DATA	159
5.35.2.16	simple_darken_Dc	159
5.35.2.17	SKUA	159
5.35.2.18	SKUA_CYCLE_TEST01	159
5.35.2.19	SKUA_CYCLE_TEST02	159
5.35.2.20	SKUA_Executioner	159
5.35.2.21	SKUA_LOW_TEST03	159
5.35.2.22	SKUA_MID_TEST04	159
5.35.2.23	SKUA_postprocesses	159
5.35.2.24	SKUA_preprocesses	159
5.35.2.25	SKUA_reset	159
5.35.2.26	SKUA_SCENARIOS	159
5.35.2.27	SKUA_TESTS	159
5.35.2.28	theoretical_darken_Dc	160

5.36 skua_opt.cpp File Reference	160
5.36.1 Function Documentation	160
5.36.1.1 eval_SKUA_Uptake	160
5.36.1.2 initial_guess_SKUA	160
5.36.1.3 SKUA_OPT_set_y	160
5.36.1.4 SKUA_OPTIMIZE	160
5.37 skua_opt.h File Reference	160
5.37.1 Function Documentation	160
5.37.1.1 eval_SKUA_Uptake	160
5.37.1.2 initial_guess_SKUA	160
5.37.1.3 SKUA_OPT_set_y	160
5.37.1.4 SKUA_OPTIMIZE	161
5.38 Trajectory.cpp File Reference	161
5.38.1 Function Documentation	161
5.38.1.1 Brown_RAD	161
5.38.1.2 Brown_THETA	161
5.38.1.3 CARTESIAN	161
5.38.1.4 DISPLACEMENT	161
5.38.1.5 Grav_R	161
5.38.1.6 Grav_T	162
5.38.1.7 LOCATION	162
5.38.1.8 Magnetic_R	162
5.38.1.9 Magnetic_T	162
5.38.1.10 Number_Generator	162
5.38.1.11 POLAR	162
5.38.1.12 RADIAL_FORCE	162
5.38.1.13 Removal_Efficiency	162
5.38.1.14 Run_Trajectory	162
5.38.1.15 TANGENTIAL_FORCE	162
5.38.1.16 Trajectory_SetupConstants	162
5.38.1.17 V_RAD	162
5.38.1.18 V_THETA	162
5.38.1.19 Van_R	162
5.39 Trajectory.h File Reference	162
5.39.1 Function Documentation	163
5.39.1.1 Brown_RAD	163
5.39.1.2 Brown_THETA	163
5.39.1.3 CARTESIAN	163
5.39.1.4 DISPLACEMENT	163
5.39.1.5 Grav_R	163

5.39.1.6	Grav_T	163
5.39.1.7	LOCATION	163
5.39.1.8	Magnetic_R	163
5.39.1.9	Magnetic_T	163
5.39.1.10	Number_Generator	163
5.39.1.11	POLAR	163
5.39.1.12	RADIAL_FORCE	163
5.39.1.13	Removal_Efficiency	164
5.39.1.14	Run_Trajectory	164
5.39.1.15	TANGENTIAL_FORCE	164
5.39.1.16	Trajectory_SetupConstants	164
5.39.1.17	V_RAD	164
5.39.1.18	V_THETA	164
5.39.1.19	Van_R	164
5.40	ui.cpp File Reference	164
5.40.1	Detailed Description	165
5.40.2	Function Documentation	165
5.40.2.1	allLower	165
5.40.2.2	ai_help	166
5.40.2.3	bui_help	166
5.40.2.4	display_help	166
5.40.2.5	display_version	166
5.40.2.6	exec	166
5.40.2.7	exec_loop	167
5.40.2.8	exit	167
5.40.2.9	help	167
5.40.2.10	input	167
5.40.2.11	invalid_input	167
5.40.2.12	number_files	168
5.40.2.13	path	168
5.40.2.14	run_exec	168
5.40.2.15	run_executable	168
5.40.2.16	run_test	168
5.40.2.17	test	169
5.40.2.18	test_loop	169
5.40.2.19	valid_addon_options	169
5.40.2.20	valid_exec_string	169
5.40.2.21	valid_input_execute	169
5.40.2.22	valid_input_main	170
5.40.2.23	valid_input_tests	170

5.40.2.24	valid_test_string	170
5.40.2.25	version	170
5.41	ui.h File Reference	170
5.41.1	Detailed Description	172
5.41.2	Macro Definition Documentation	173
5.41.2.1	ECO_EXECUTABLE	173
5.41.2.2	ECO_VERSION	173
5.41.2.3	UI_HPP_	173
5.41.3	Enumeration Type Documentation	173
5.41.3.1	valid_options	173
5.41.4	Function Documentation	174
5.41.4.1	allLower	174
5.41.4.2	aui_help	174
5.41.4.3	bui_help	174
5.41.4.4	display_help	174
5.41.4.5	display_version	174
5.41.4.6	exec	174
5.41.4.7	exec_loop	175
5.41.4.8	exit	175
5.41.4.9	help	175
5.41.4.10	input	175
5.41.4.11	invalid_input	175
5.41.4.12	number_files	176
5.41.4.13	path	176
5.41.4.14	run_exec	176
5.41.4.15	run_executable	176
5.41.4.16	run_test	177
5.41.4.17	test	177
5.41.4.18	test_loop	177
5.41.4.19	valid_addon_options	177
5.41.4.20	valid_exec_string	177
5.41.4.21	valid_input_execute	178
5.41.4.22	valid_input_main	178
5.41.4.23	valid_input_tests	178
5.41.4.24	valid_test_string	178
5.41.4.25	version	178
5.42	yaml_wrapper.cpp File Reference	179
5.42.1	Function Documentation	179
5.42.1.1	YAML_CPP_TEST	179
5.42.1.2	YAML_WRAPPER_TESTS	179

5.43	yaml_wrapper.h File Reference	179
5.43.1	Typedef Documentation	180
5.43.1.1	data_type	180
5.43.1.2	header_state	180
5.43.2	Enumeration Type Documentation	180
5.43.2.1	data_type	180
5.43.2.2	header_state	180
5.43.3	Function Documentation	180
5.43.3.1	YAML_CPP_TEST	180
5.43.3.2	YAML_WRAPPER_TESTS	180

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ARNOLDI_DATA	7
Atom	9
Molecule	59
MasterSpeciesList	52
PeriodicTable	70
BACKTRACK_DATA	11
BiCGSTAB_DATA	13
CGS_DATA	16
DOGFISH_DATA	23
DOGFISH_PARAM	24
FINCH_DATA	25
GCR_DATA	31
GMRESLP_DATA	34
GMRESR_DATA	36
GMRESRP_DATA	39
GPAST_DATA	42
GSTA_DATA	43
GSTA_OPT_DATA	44
KeyValueMap	48
MAGPIE_DATA	50
MassBalance	50
Matrix< T >	54
Matrix< double >	54
Matrix< int >	54
Mechanism	57
MIXED_GAS	58
MONKFISH_DATA	62
MONKFISH_PARAM	64
mSPD_DATA	65
NUM_JAC_DATA	66
OPTRANS_DATA	67
PCG_DATA	67
PICARD_DATA	71
PJFNK_DATA	73
PURE_GAS	78
Reaction	79
Precipitation	78

UnsteadyPrecipitation	103
UnsteadyReaction	104
SCOPSOWL_DATA	81
SCOPSOWL_OPT_DATA	83
SCOPSOWL_PARAM_DATA	85
SHARK_DATA	87
SKUA_DATA	90
SKUA_OPT_DATA	92
SKUA_PARAM	93
Speciation_Test01_Data	94
SubHeader	95
Document	20
Header	45
SYSTEM_DATA	97
TRAJECTORY_DATA	99
UI_DATA	101
ValueTypePair	108
yaml_cpp_class	109
YamlWrapper	110

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ARNOLDI_DATA	
Data structure for the construction of the Krylov subspaces for a linear system	7
Atom	9
BACKTRACK_DATA	
Data structure for the implementation of Backtracking Linesearch	11
BiCGSTAB_DATA	
Data structure for the implementation of the BiCGSTAB algorithm for non-symmetric linear systems	13
CGS_DATA	
Data structure for the implementation of the CGS algorithm for non-symmetric linear systems .	16
Document	20
DOGFISH_DATA	23
DOGFISH_PARAM	24
FINCH_DATA	25
GCR_DATA	
Data structure for the implementation of the GCR algorithm for non-symmetric linear systems .	31
GMRESLP_DATA	
Data structure for implementation of the Restarted GMRES algorithm with Left Preconditioning	34
GMRESR_DATA	
Data structure for the implementation of GCR with Nested GMRES preconditioning (i.e., GMRESR)	36
GMRESRP_DATA	
Data structure for the Restarted GMRES algorithm with Right Preconditioning	39
GPAST_DATA	42
GSTA_DATA	43
GSTA_OPT_DATA	44
Header	45
KeyValueMap	48
MAGPIE_DATA	50
MassBalance	50
MasterSpeciesList	52
Matrix< T >	54
Mechanism	57
MIXED_GAS	58
Molecule	59
MONKFISH_DATA	62
MONKFISH_PARAM	64
mSPD_DATA	65

NUM_JAC_DATA	
Data structure to form a numerical jacobian matrix with finite differences	66
OPTRANS_DATA	
Data structure for implementation of linear operator transposition	67
PCG_DATA	
Data structure for implementation of the PCG algorithms for symmetric linear systems	67
PeriodicTable	70
PICARD_DATA	
Data structure for the implementation of a Picard or Fixed-Point iteration for non-linear systems	71
PJFNK_DATA	
Data structure for the implementation of the PJFNK algorithm for non-linear systems	73
Precipitation	78
PURE_GAS	78
Reaction	79
SCOPSOWL_DATA	81
SCOPSOWL_OPT_DATA	83
SCOPSOWL_PARAM_DATA	85
SHARK_DATA	87
SKUA_DATA	90
SKUA_OPT_DATA	92
SKUA_PARAM	93
Speciation_Test01_Data	94
SubHeader	95
SYSTEM_DATA	97
TRAJECTORY_DATA	99
UI_DATA	
Data structure holding the UI arguments	101
UnsteadyPrecipitation	103
UnsteadyReaction	104
ValueTypePair	108
yaml_cpp_class	109
YamlWrapper	110

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

dogfish.cpp	113
dogfish.h	114
eel.cpp	115
eel.h	116
egret.cpp	116
egret.h	117
error.cpp	118
error.h	118
finch.cpp	120
finch.h	123
flock.h	126
gsta_opt.cpp	126
gsta_opt.h	127
lark.cpp	
Linear Algebra Residual Kernels	129
lark.h	
Linear Algebra Residual Kernels	131
macaw.cpp	136
macaw.h	136
magpie.cpp	137
magpie.h	138
main.cpp	
Main Function	140
mola.cpp	141
mola.h	141
monkfish.cpp	141
monkfish.h	142
sandbox.cpp	143
sandbox.h	144
school.h	144
scopsowl.cpp	145
scopsowl.h	146
scopsowl_opt.cpp	149
scopsowl_opt.h	149
shark.cpp	150
shark.h	152
skua.cpp	156
skua.h	157

skua_opt.cpp	160
skua_opt.h	160
Trajectory.cpp	161
Trajectory.h	162
ui.cpp	
User Interface for Ecosystem	164
ui.h	
User Interface for Ecosystem	170
yaml_wrapper.cpp	179
yaml_wrapper.h	179

Chapter 4

Class Documentation

4.1 ARNOLDI_DATA Struct Reference

Data structure for the construction of the Krylov subspaces for a linear system.

```
#include <lark.h>
```

Public Attributes

- int **k**
Desired size of the Krylov subspace.
- int **iter**
Actual size of the Krylov subspace.
- double **beta**
Normalization parameter.
- double **hp1**
Additional row element of H (separate storage for holding)
- bool **Output** = true
True = print messages to console.
- std::vector< **Matrix**< double > > **Vk**
(N) x (k) orthonormal vector basis stored as a vector of column matrices
- **Matrix**< double > **Hkp1**
(k+1) x (k) upper Hessenberg matrix
- **Matrix**< double > **yk**
(k) x (1) vector search direction
- **Matrix**< double > **e1**
(k) x (1) orthonormal vector with 1 in first position
- **Matrix**< double > **w**
(N) x (1) interim result of the matrix_vector multiplication
- **Matrix**< double > **v**
(N) x (1) holding cell for the column entries of Vk and other interims
- **Matrix**< double > **sum**
(N) x (1) running sum of subspace vectors for use in altering w

4.1.1 Detailed Description

Data structure for the construction of the Krylov subspaces for a linear system.

C-style object used in conjunction with the Arnoldi algorithm to construct an orthonormal basis and upper Hessenberg representation of a given linear operator. This is used to solve a linear system both iteratively (i.e., in conjunction with GMRESLP) and directly (i.e., in conjunction with FOM). Alternatively, you can just store the factorized components for later use in another routine.

4.1.2 Member Data Documentation

4.1.2.1 `double ARNOLDI_DATA::beta`

Normalization parameter.

4.1.2.2 `Matrix<double> ARNOLDI_DATA::e1`

(k) x (1) orthonormal vector with 1 in first position

4.1.2.3 `Matrix<double> ARNOLDI_DATA::Hkp1`

(k+1) x (k) upper Hessenberg matrix

4.1.2.4 `double ARNOLDI_DATA::hp1`

Additional row element of H (separate storage for holding)

4.1.2.5 `int ARNOLDI_DATA::iter`

Actual size of the Krylov subspace.

4.1.2.6 `int ARNOLDI_DATA::k`

Desired size of the Krylov subspace.

4.1.2.7 `bool ARNOLDI_DATA::Output = true`

True = print messages to console.

4.1.2.8 `Matrix<double> ARNOLDI_DATA::sum`

(N) x (1) running sum of subspace vectors for use in altering w

4.1.2.9 `Matrix<double> ARNOLDI_DATA::v`

(N) x (1) holding cell for the column entries of V_k and other interims

4.1.2.10 `std::vector< Matrix<double> > ARNOLDI_DATA::Vk`

(N) x (k) orthonormal vector basis stored as a vector of column matrices

4.1.2.11 `Matrix<double> ARNOLDI_DATA::w`

(N) x (1) interim result of the matrix_vector multiplication

4.1.2.12 `Matrix<double> ARNOLDI_DATA::yk`

(k) x (1) vector search direction

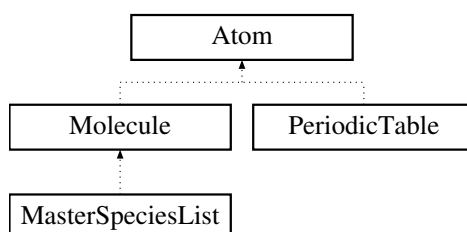
The documentation for this struct was generated from the following file:

- [lark.h](#)

4.2 Atom Class Reference

```
#include <eel.h>
```

Inheritance diagram for Atom:



Public Member Functions

- [Atom \(\)](#)
- [~Atom \(\)](#)
- [Atom \(std::string Name\)](#)
- [Atom \(int number\)](#)
- void [Register \(std::string Symbol\)](#)
- void [Register \(int number\)](#)
- void [editAtomicWeight \(double AW\)](#)
- void [editOxidationState \(int state\)](#)
- void [editProtons \(int proton\)](#)
- void [editNeutrons \(int neutron\)](#)
- void [editElectrons \(int electron\)](#)
- void [editValence \(int val\)](#)
- void [removeProton \(\)](#)
- void [removeNeutron \(\)](#)
- void [removeElectron \(\)](#)
- double [AtomicWeight \(\)](#)
- int [OxidationState \(\)](#)
- int [Protons \(\)](#)
- int [Neutrons \(\)](#)
- int [Electrons \(\)](#)
- int [BondingElectrons \(\)](#)
- std::string [AtomName \(\)](#)
- std::string [AtomSymbol \(\)](#)
- std::string [AtomCategory \(\)](#)
- std::string [AtomState \(\)](#)
- int [AtomicNumber \(\)](#)
- void [DisplayInfo \(\)](#)

Protected Attributes

- double [atomic_weight](#)
- int [oxidation_state](#)
- int [protons](#)
- int [neutrons](#)
- int [electrons](#)
- int [valence_e](#)

Private Attributes

- std::string [Name](#)
- std::string [Symbol](#)
- std::string [Category](#)
- std::string [NaturalState](#)
- int [atomic_number](#)

4.2.1 Constructor & Destructor Documentation

4.2.1.1 `Atom::Atom ()`

4.2.1.2 `Atom::~~Atom ()`

4.2.1.3 `Atom::Atom (std::string Name)`

4.2.1.4 `Atom::Atom (int number)`

4.2.2 Member Function Documentation

4.2.2.1 `std::string Atom::AtomCategory ()`

4.2.2.2 `int Atom::AtomicNumber ()`

4.2.2.3 `double Atom::AtomicWeight ()`

4.2.2.4 `std::string Atom::AtomName ()`

4.2.2.5 `std::string Atom::AtomState ()`

4.2.2.6 `std::string Atom::AtomSymbol ()`

4.2.2.7 `int Atom::BondingElectrons ()`

4.2.2.8 `void Atom::DisplayInfo ()`

4.2.2.9 `void Atom::editAtomicWeight (double AW)`

4.2.2.10 `void Atom::editElectrons (int electron)`

4.2.2.11 `void Atom::editNeutrons (int neutron)`

4.2.2.12 `void Atom::editOxidationState (int state)`

4.2.2.13 `void Atom::editProtons (int proton)`

- 4.2.2.14 void Atom::editValence (int *val*)
- 4.2.2.15 int Atom::Electrons ()
- 4.2.2.16 int Atom::Neutrons ()
- 4.2.2.17 int Atom::OxidationState ()
- 4.2.2.18 int Atom::Protons ()
- 4.2.2.19 void Atom::Register (std::string *Symbol*)
- 4.2.2.20 void Atom::Register (int *number*)
- 4.2.2.21 void Atom::removeElectron ()
- 4.2.2.22 void Atom::removeNeutron ()
- 4.2.2.23 void Atom::removeProton ()

4.2.3 Member Data Documentation

- 4.2.3.1 int Atom::atomic_number [private]
- 4.2.3.2 double Atom::atomic_weight [protected]
- 4.2.3.3 std::string Atom::Category [private]
- 4.2.3.4 int Atom::electrons [protected]
- 4.2.3.5 std::string Atom::Name [private]
- 4.2.3.6 std::string Atom::NaturalState [private]
- 4.2.3.7 int Atom::neutrons [protected]
- 4.2.3.8 int Atom::oxidation_state [protected]
- 4.2.3.9 int Atom::protons [protected]
- 4.2.3.10 std::string Atom::Symbol [private]
- 4.2.3.11 int Atom::valence_e [protected]

The documentation for this class was generated from the following files:

- [eel.h](#)
- [eel.cpp](#)

4.3 BACKTRACK_DATA Struct Reference

Data structure for the implementation of Backtracking Linesearch.

```
#include <lark.h>
```

Public Attributes

- double `alpha` = 1e-4
Scaling parameter for determination of search step size.
- double `rho` = 0.1
Scaling parameter for to change step size by.
- double `lambdaMin` = DBL_EPSILON
Smallest allowable step length.
- double `normFkp1`
New residual norm of the Newton step.
- bool `constRho` = false
True = use a constant value for rho.
- `Matrix< double > Fk`
Old residual vector of the Newton step.
- `Matrix< double > xk`
Old solution vector of the Newton step.

4.3.1 Detailed Description

Data structure for the implementation of Backtracking Linesearch.

C-style object used in conjunction with the Backtracking Linesearch algorithm to smooth out convergence of Newton based iterative methods for non-linear systems of equations. The actual algorithm has been separated from the interior of the Newton method so that it can be included in any future Newton based iterative methods being developed.

4.3.2 Member Data Documentation

4.3.2.1 double BACKTRACK_DATA::alpha = 1e-4

Scaling parameter for determination of search step size.

4.3.2.2 bool BACKTRACK_DATA::constRho = false

True = use a constant value for rho.

4.3.2.3 Matrix<double> BACKTRACK_DATA::Fk

Old residual vector of the Newton step.

4.3.2.4 double BACKTRACK_DATA::lambdaMin = DBL_EPSILON

Smallest allowable step length.

4.3.2.5 double BACKTRACK_DATA::normFkp1

New residual norm of the Newton step.

4.3.2.6 double BACKTRACK_DATA::rho = 0.1

Scaling parameter for to change step size by.

4.3.2.7 Matrix<double> BACKTRACK_DATA::xk

Old solution vector of the Newton step.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.4 BiCGSTAB_DATA Struct Reference

Data structure for the implementation of the BiCGSTAB algorithm for non-symmetric linear systems.

```
#include <lark.h>
```

Public Attributes

- int [maxit](#) = 0
*Maximum allowable iterations - default = min(2*vector_size,1000)*
- int [iter](#) = 0
Actual number of iterations.
- bool [breakdown](#)
Boolean to determine if the method broke down.
- double [alpha](#)
Step size parameter for next solution.
- double [beta](#)
Step size parameter for search direction.
- double [rho](#)
Scaling parameter for alpha and beta.
- double [rho_old](#)
Previous scaling parameter for alpha and beta.
- double [omega](#)
Scaling parameter and additional step length.
- double [omega_old](#)
Previous scaling parameter and step length.
- double [tol_rel](#) = 1e-6
Relative tolerance for convergence - default = 1e-6.
- double [tol_abs](#) = 1e-6
Absolution tolerance for convergence - default = 1e-6.
- double [res](#)
Absolute residual norm.
- double [relres](#)
Relative residual norm.
- double [relres_base](#)
Initial residual norm.
- double [bestres](#)
Best found residual norm.
- bool [Output](#) = true
True = print messages to console.
- Matrix< double > [x](#)
Current solution to the linear system.
- Matrix< double > [bestx](#)

- *Best found solution to the linear system.*
- `Matrix< double > r`
Residual vector for the linear system.
- `Matrix< double > r0`
Initial residual vector.
- `Matrix< double > v`
Search direction for p.
- `Matrix< double > p`
Search direction for updating.
- `Matrix< double > y`
Preconditioned search direction.
- `Matrix< double > s`
Residual updating vector.
- `Matrix< double > z`
Preconditioned residual updating vector.
- `Matrix< double > t`
Search direction for residual updates.

4.4.1 Detailed Description

Data structure for the implementation of the BiCGSTAB algorithm for non-symmetric linear systems.

C-style object used in conjunction with the Bi-Conjugate Gradient STABalized (BiCGSTAB) algorithm to solve a linear system of equations. This algorithm is generally more efficient than any GMRES or GCR variant, but may not always reduce the residual at each step. However, if used with preconditioning, then this algorithm is very efficient, especially when used for solving grid-based linear systems.

4.4.2 Member Data Documentation

4.4.2.1 `double BiCGSTAB_DATA::alpha`

Step size parameter for next solution.

4.4.2.2 `double BiCGSTAB_DATA::bestres`

Best found residual norm.

4.4.2.3 `Matrix<double> BiCGSTAB_DATA::bestx`

Best found solution to the linear system.

4.4.2.4 `double BiCGSTAB_DATA::beta`

Step size parameter for search direction.

4.4.2.5 `bool BiCGSTAB_DATA::breakdown`

Boolean to determine if the method broke down.

4.4.2.6 int BiCGSTAB_DATA::iter = 0

Actual number of iterations.

4.4.2.7 int BiCGSTAB_DATA::maxit = 0

Maximum allowable iterations - default = $\min(2 \times \text{vector_size}, 1000)$

4.4.2.8 double BiCGSTAB_DATA::omega

Scaling parameter and additional step length.

4.4.2.9 double BiCGSTAB_DATA::omega_old

Previous scaling parameter and step length.

4.4.2.10 bool BiCGSTAB_DATA::Output = true

True = print messages to console.

4.4.2.11 Matrix<double> BiCGSTAB_DATA::p

Search direction for updating.

4.4.2.12 Matrix<double> BiCGSTAB_DATA::r

Residual vector for the linear system.

4.4.2.13 Matrix<double> BiCGSTAB_DATA::r0

Initial residual vector.

4.4.2.14 double BiCGSTAB_DATA::relres

Relative residual norm.

4.4.2.15 double BiCGSTAB_DATA::relres_base

Initial residual norm.

4.4.2.16 double BiCGSTAB_DATA::res

Absolute residual norm.

4.4.2.17 double BiCGSTAB_DATA::rho

Scaling parameter for alpha and beta.

4.4.2.18 `double BiCGSTAB_DATA::rho_old`

Previous scaling parameter for alpha and beta.

4.4.2.19 `Matrix<double> BiCGSTAB_DATA::s`

Residual updating vector.

4.4.2.20 `Matrix<double> BiCGSTAB_DATA::t`

Search direction for residual updates.

4.4.2.21 `double BiCGSTAB_DATA::tol_abs = 1e-6`

Absolution tolerance for convergence - default = 1e-6.

4.4.2.22 `double BiCGSTAB_DATA::tol_rel = 1e-6`

Relative tolerance for convergence - default = 1e-6.

4.4.2.23 `Matrix<double> BiCGSTAB_DATA::v`

Search direction for p.

4.4.2.24 `Matrix<double> BiCGSTAB_DATA::x`

Current solution to the linear system.

4.4.2.25 `Matrix<double> BiCGSTAB_DATA::y`

Preconditioned search direction.

4.4.2.26 `Matrix<double> BiCGSTAB_DATA::z`

Preconditioned residual updating vector.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.5 CGS_DATA Struct Reference

Data structure for the implementation of the CGS algorithm for non-symmetric linear systems.

```
#include <lark.h>
```

Public Attributes

- `int maxit = 0`
*Maximum allowable iterations - default = min(2*vector_size,1000)*

- int `iter` = 0
Actual number of iterations.
- bool `breakdown`
Boolean to determine if the method broke down.
- double `alpha`
Step size parameter for next solution.
- double `beta`
Step size parameter for search direction.
- double `rho`
Scaling parameter for alpha and beta.
- double `sigma`
Scaling parameter and additional step length.
- double `tol_rel` = 1e-6
Relative tolerance for convergence - default = 1e-6.
- double `tol_abs` = 1e-6
Absolution tolerance for convergence - default = 1e-6.
- double `res`
Absolute residual norm.
- double `relres`
Relative residual norm.
- double `relres_base`
Initial residual norm.
- double `bestres`
Best found residual norm.
- bool `Output` = true
True = print messages to console.
- `Matrix`< double > `x`
Current solution to the linear system.
- `Matrix`< double > `bestx`
Best found solution to the linear system.
- `Matrix`< double > `r`
Residual vector for the linear system.
- `Matrix`< double > `r0`
Initial residual vector.
- `Matrix`< double > `u`
Search direction for v.
- `Matrix`< double > `w`
Updates sigma and u.
- `Matrix`< double > `v`
Search direction for x.
- `Matrix`< double > `p`
Preconditioning result for w, z, and matvec for Ax.
- `Matrix`< double > `c`
Holds the matvec result between A and p.
- `Matrix`< double > `z`
Full search direction for x.

4.5.1 Detailed Description

Data structure for the implementation of the CGS algorithm for non-symmetric linear systems.

C-style object to be used in conjunction with the Conjugate Gradient Squared (CGS) algorithm to solve linear systems of equations. This algorithm is slightly less computational work than BiCGSTAB, but is much less stable. As a result, I do not recommend using this algorithm unless you also use some form of preconditioning.

4.5.2 Member Data Documentation

4.5.2.1 `double CGS_DATA::alpha`

Step size parameter for next solution.

4.5.2.2 `double CGS_DATA::bestres`

Best found residual norm.

4.5.2.3 `Matrix<double> CGS_DATA::bestx`

Best found solution to the linear system.

4.5.2.4 `double CGS_DATA::beta`

Step size parameter for search direction.

4.5.2.5 `bool CGS_DATA::breakdown`

Boolean to determine if the method broke down.

4.5.2.6 `Matrix<double> CGS_DATA::c`

Holds the matvec result between A and p.

4.5.2.7 `int CGS_DATA::iter = 0`

Actual number of iterations.

4.5.2.8 `int CGS_DATA::maxit = 0`

Maximum allowable iterations - default = $\min(2 \cdot \text{vector_size}, 1000)$

4.5.2.9 `bool CGS_DATA::Output = true`

True = print messages to console.

4.5.2.10 `Matrix<double> CGS_DATA::p`

Preconditioning result for w, z, and matvec for Ax.

4.5.2.11 Matrix<double> CGS_DATA::r

Residual vector for the linear system.

4.5.2.12 Matrix<double> CGS_DATA::r0

Initial residual vector.

4.5.2.13 double CGS_DATA::relres

Relative residual norm.

4.5.2.14 double CGS_DATA::relres_base

Initial residual norm.

4.5.2.15 double CGS_DATA::res

Absolute residual norm.

4.5.2.16 double CGS_DATA::rho

Scaling parameter for alpha and beta.

4.5.2.17 double CGS_DATA::sigma

Scaling parameter and additional step length.

4.5.2.18 double CGS_DATA::tol_abs = 1e-6

Absolution tolerance for convergence - default = 1e-6.

4.5.2.19 double CGS_DATA::tol_rel = 1e-6

Relative tolerance for convergence - default = 1e-6.

4.5.2.20 Matrix<double> CGS_DATA::u

Search direction for v.

4.5.2.21 Matrix<double> CGS_DATA::v

Search direction for x.

4.5.2.22 Matrix<double> CGS_DATA::w

Updates sigma and u.

4.5.2.23 `Matrix<double> CGS_DATA::x`

Current solution to the linear system.

4.5.2.24 `Matrix<double> CGS_DATA::z`

Full search direction for x.

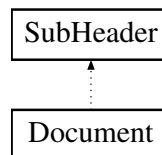
The documentation for this struct was generated from the following file:

- [lark.h](#)

4.6 Document Class Reference

```
#include <yaml_wrapper.h>
```

Inheritance diagram for Document:



Public Member Functions

- `Document ()`
- `~Document ()`
- `Document (const Document &doc)`
- `Document (std::string name)`
- `Document (const KeyValueType &map)`
- `Document (std::string name, const KeyValueType &map)`
- `Document (std::string key, const Header &head)`
- `Document & operator= (const Document &doc)`
- `ValueTypePair & operator[] (const std::string key)`
- `ValueTypePair operator[] (const std::string key) const`
- `Header & operator() (const std::string key)`
- `Header operator() (const std::string key) const`
- `std::map< std::string, Header > & getHeadMap ()`
- `KeyValueType & getDataMap ()`
- `Header & getHeader (std::string key)`
- `std::map< std::string, Header > ::const_iterator end () const`
- `std::map< std::string, Header > ::iterator end ()`
- `std::map< std::string, Header > ::const_iterator begin () const`
- `std::map< std::string, Header > ::iterator begin ()`
- `void clear ()`
- `void resetKeys ()`
- `void changeKey (std::string oldKey, std::string newKey)`
- `void revalidateAllKeys ()`

- void [addPair](#) (std::string key, std::string val)
- void [addPair](#) (std::string key, std::string val, int t)
- void [setName](#) (std::string [name](#))
- void [setAlias](#) (std::string [alias](#))
- void [setNameAliasPair](#) (std::string n, std::string a, int s)
- void [setState](#) (int [state](#))
- void [DisplayContents](#) ()
- void [addHeadKey](#) (std::string key)
- void [copyAnchor2Alias](#) (std::string [alias](#), [Header](#) &ref)
- int [size](#) ()
- std::string [getName](#) ()
- std::string [getAlias](#) ()
- int [getState](#) ()
- bool [isAlias](#) ()
- bool [isAnchor](#) ()
- [Header](#) & [getAnchoredHeader](#) (std::string [alias](#))
- [Header](#) & [getHeadFromSubAlias](#) (std::string [alias](#))

Private Attributes

- std::map< std::string, [Header](#) > [Head_Map](#)

Additional Inherited Members

4.6.1 Constructor & Destructor Documentation

4.6.1.1 [Document::Document](#) ()

4.6.1.2 [Document::~~Document](#) ()

4.6.1.3 [Document::Document](#) (const [Document](#) & *doc*)

4.6.1.4 [Document::Document](#) (std::string *name*)

4.6.1.5 [Document::Document](#) (const [KeyValueMap](#) & *map*)

4.6.1.6 [Document::Document](#) (std::string *name*, const [KeyValueMap](#) & *map*)

4.6.1.7 [Document::Document](#) (std::string *key*, const [Header](#) & *head*)

4.6.2 Member Function Documentation

4.6.2.1 void [Document::addHeadKey](#) (std::string *key*)

4.6.2.2 void [Document::addPair](#) (std::string *key*, std::string *val*)

4.6.2.3 void [Document::addPair](#) (std::string *key*, std::string *val*, int *t*)

4.6.2.4 std::map< std::string, [Header](#) >::const_iterator [Document::begin](#) () const

4.6.2.5 std::map< std::string, [Header](#) >::iterator [Document::begin](#) ()

4.6.2.6 void [Document::changeKey](#) (std::string *oldKey*, std::string *newKey*)

- 4.6.2.7 void Document::clear ()
- 4.6.2.8 void Document::copyAnchor2Alias (std::string *alias*, Header & *ref*)
- 4.6.2.9 void Document::DisplayContents ()
- 4.6.2.10 std::map< std::string, Header >::const_iterator Document::end () const
- 4.6.2.11 std::map< std::string, Header >::iterator Document::end ()
- 4.6.2.12 std::string Document::getAlias ()
- 4.6.2.13 Header & Document::getAnchoredHeader (std::string *alias*)
- 4.6.2.14 KeyValueMap & Document::getDataMap ()
- 4.6.2.15 Header & Document::getHeader (std::string *key*)
- 4.6.2.16 Header & Document::getHeadFromSubAlias (std::string *alias*)
- 4.6.2.17 std::map< std::string, Header > & Document::getHeadMap ()
- 4.6.2.18 std::string Document::getName ()
- 4.6.2.19 int Document::getState ()
- 4.6.2.20 bool Document::isAlias ()
- 4.6.2.21 bool Document::isAnchor ()
- 4.6.2.22 Header & Document::operator() (const std::string *key*)
- 4.6.2.23 Header Document::operator() (const std::string *key*) const
- 4.6.2.24 Document & Document::operator= (const Document & *doc*)
- 4.6.2.25 ValueTypePair & Document::operator[] (const std::string *key*)
- 4.6.2.26 ValueTypePair Document::operator[] (const std::string *key*) const
- 4.6.2.27 void Document::resetKeys ()
- 4.6.2.28 void Document::revalidateAllKeys ()
- 4.6.2.29 void Document::setAlias (std::string *alias*)
- 4.6.2.30 void Document::setName (std::string *name*)
- 4.6.2.31 void Document::setNameAliasPair (std::string *n*, std::string *a*, int *s*)
- 4.6.2.32 void Document::setState (int *state*)
- 4.6.2.33 int Document::size ()

4.6.3 Member Data Documentation

4.6.3.1 `std::map<std::string, Header> Document::Head_Map` [private]

The documentation for this class was generated from the following files:

- [yaml_wrapper.h](#)
- [yaml_wrapper.cpp](#)

4.7 DOGFISH_DATA Struct Reference

```
#include <dogfish.h>
```

Public Attributes

- unsigned long int `total_steps` = 0
- double `time_old` = 0.0
- double `time` = 0.0
- bool `Print2File` = true
- bool `Print2Console` = true
- bool `DirichletBC` = false
- bool `NonLinear` = false
- double `t_counter` = 0.0
- double `t_print`
- int `NumComp`
- double `end_time`
- double `total_sorption_old`
- double `total_sorption`
- double `fiber_length`
- double `fiber_diameter`
- FILE * `OutputFile`
- double(* `eval_R`)(int i, int l, const void *data)
- double(* `eval_DI`)(int i, int l, const void *data)
- double(* `eval_kf`)(int i, const void *data)
- double(* `eval_qs`)(int i, const void *data)
- const void * `user_data`
- std::vector< `FINCH_DATA` > `finch_dat`
- std::vector< `DOGFISH_PARAM` > `param_dat`

4.7.1 Member Data Documentation

4.7.1.1 `bool DOGFISH_DATA::DirichletBC` = false

4.7.1.2 `double DOGFISH_DATA::end_time`

4.7.1.3 `double(* DOGFISH_DATA::eval_DI)(int i, int l, const void *data)`

4.7.1.4 `double(* DOGFISH_DATA::eval_kf)(int i, const void *data)`

4.7.1.5 `double(* DOGFISH_DATA::eval_qs)(int i, const void *data)`

4.7.1.6 `double(* DOGFISH_DATA::eval_R)(int i, int l, const void *data)`

4.7.1.7 `double DOGFISH_DATA::fiber_diameter`

4.7.1.8 double DOGFISH_DATA::fiber_length

4.7.1.9 std::vector<FINCH_DATA> DOGFISH_DATA::finch_dat

4.7.1.10 bool DOGFISH_DATA::NonLinear = false

4.7.1.11 int DOGFISH_DATA::NumComp

4.7.1.12 FILE* DOGFISH_DATA::OutputFile

4.7.1.13 std::vector<DOGFISH_PARAM> DOGFISH_DATA::param_dat

4.7.1.14 bool DOGFISH_DATA::Print2Console = true

4.7.1.15 bool DOGFISH_DATA::Print2File = true

4.7.1.16 double DOGFISH_DATA::t_counter = 0.0

4.7.1.17 double DOGFISH_DATA::t_print

4.7.1.18 double DOGFISH_DATA::time = 0.0

4.7.1.19 double DOGFISH_DATA::time_old = 0.0

4.7.1.20 double DOGFISH_DATA::total_sorption

4.7.1.21 double DOGFISH_DATA::total_sorption_old

4.7.1.22 unsigned long int DOGFISH_DATA::total_steps = 0

4.7.1.23 const void* DOGFISH_DATA::user_data

The documentation for this struct was generated from the following file:

- [dogfish.h](#)

4.8 DOGFISH_PARAM Struct Reference

```
#include <dogfish.h>
```

Public Attributes

- double [intraparticle_diffusion](#)
- double [film_transfer_coeff](#)
- double [surface_concentration](#)
- double [initial_sorption](#)
- double [sorbed_molefraction](#)
- [Molecule species](#)

4.8.1 Member Data Documentation

4.8.1.1 double DOGFISH_PARAM::film_transfer_coeff

4.8.1.2 double DOGFISH_PARAM::initial_sorption

4.8.1.3 double DOGFISH_PARAM::intraparticle_diffusion

4.8.1.4 double DOGFISH_PARAM::sorbed_molefraction

4.8.1.5 Molecule DOGFISH_PARAM::species

4.8.1.6 double DOGFISH_PARAM::surface_concentration

The documentation for this struct was generated from the following file:

- [dogfish.h](#)

4.9 FINCH_DATA Struct Reference

```
#include <finch.h>
```

Public Attributes

- int [d](#) = 0
- double [dt](#) = 0.0125
- double [dt_old](#) = 0.0125
- double [T](#) = 1.0
- double [dz](#) = 0.1
- double [L](#) = 1.0
- double [s](#) = 1.0
- double [t](#) = 0.0
- double [t_old](#) = 0.0
- double [uT](#) = 0.0
- double [uT_old](#) = 0.0
- double [uAvg](#) = 0.0
- double [uAvg_old](#) = 0.0
- double [uIC](#) = 0.0
- double [vIC](#) = 1.0
- double [DIC](#) = 1.0
- double [kIC](#) = 1.0
- double [RIC](#) = 1.0
- double [uo](#) = 1.0
- double [vo](#) = 1.0
- double [Do](#) = 1.0
- double [ko](#) = 1.0
- double [Ro](#) = 1.0
- double [kfn](#) = 1.0
- double [kfnp1](#) = 1.0
- double [lambda_I](#)
- double [lambda_E](#)
- int [LN](#) = 10
- bool [CN](#) = true
- bool [Update](#) = false
- bool [Dirichlet](#) = false
- bool [CheckMass](#) = false
- bool [ExplicitFlux](#) = false

- bool `Iterative` = true
- bool `SteadyState` = false
- bool `NormTrack` = true
- double `beta` = 0.5
- double `tol_rel` = 1e-6
- double `tol_abs` = 1e-6
- int `max_iter` = 20
- int `total_iter` = 0
- int `nl_method` = `FINCH_Picard`
- std::vector< double > `CL_I`
- std::vector< double > `CL_E`
- std::vector< double > `CC_I`
- std::vector< double > `CC_E`
- std::vector< double > `CR_I`
- std::vector< double > `CR_E`
- std::vector< double > `fL_I`
- std::vector< double > `fL_E`
- std::vector< double > `fC_I`
- std::vector< double > `fC_E`
- std::vector< double > `fR_I`
- std::vector< double > `fR_E`
- std::vector< double > `OI`
- std::vector< double > `OE`
- std::vector< double > `NI`
- std::vector< double > `NE`
- std::vector< double > `MI`
- std::vector< double > `ME`
- std::vector< double > `uz_I_I`
- std::vector< double > `uz_lm1_I`
- std::vector< double > `uz_lp1_I`
- std::vector< double > `uz_I_E`
- std::vector< double > `uz_lm1_E`
- std::vector< double > `uz_lp1_E`
- Matrix< double > `unm1`
- Matrix< double > `un`
- Matrix< double > `unp1`
- Matrix< double > `u_star`
- Matrix< double > `ubest`
- Matrix< double > `vn`
- Matrix< double > `vnp1`
- Matrix< double > `Dn`
- Matrix< double > `Dnp1`
- Matrix< double > `kn`
- Matrix< double > `knp1`
- Matrix< double > `Sn`
- Matrix< double > `Snp1`
- Matrix< double > `Rn`
- Matrix< double > `Rnp1`
- Matrix< double > `Fn`
- Matrix< double > `Fnp1`
- Matrix< double > `gl`
- Matrix< double > `gE`
- Matrix< double > `res`
- Matrix< double > `pres`
- int(* `callroutine`)(const void *user_data)

- `int(* setic)(const void *user_data)`
- `int(* settime)(const void *user_data)`
- `int(* setpreprocess)(const void *user_data)`
- `int(* solve)(const void *user_data)`
- `int(* setparams)(const void *user_data)`
- `int(* discretize)(const void *user_data)`
- `int(* setbcs)(const void *user_data)`
- `int(* evalres)(const Matrix< double > &x, Matrix< double > &res, const void *user_data)`
- `int(* evalprecon)(const Matrix< double > &b, Matrix< double > &p, const void *user_data)`
- `int(* setpostprocess)(const void *user_data)`
- `int(* resettime)(const void *user_data)`
- `PICARD_DATA picard_dat`
- `PJFNK_DATA pjfnk_dat`
- `const void * param_data`

4.9.1 Member Data Documentation

4.9.1.1 `double FINCH_DATA::beta = 0.5`

4.9.1.2 `int(* FINCH_DATA::callroutine)(const void *user_data)`

4.9.1.3 `std::vector<double> FINCH_DATA::CC_E`

4.9.1.4 `std::vector<double> FINCH_DATA::CC_I`

4.9.1.5 `bool FINCH_DATA::CheckMass = false`

4.9.1.6 `std::vector<double> FINCH_DATA::CL_E`

4.9.1.7 `std::vector<double> FINCH_DATA::CL_I`

4.9.1.8 `bool FINCH_DATA::CN = true`

4.9.1.9 `std::vector<double> FINCH_DATA::CR_E`

4.9.1.10 `std::vector<double> FINCH_DATA::CR_I`

4.9.1.11 `int FINCH_DATA::d = 0`

4.9.1.12 `double FINCH_DATA::DIC = 1.0`

4.9.1.13 `bool FINCH_DATA::Dirichlet = false`

4.9.1.14 `int(* FINCH_DATA::discretize)(const void *user_data)`

4.9.1.15 `Matrix<double> FINCH_DATA::Dn`

4.9.1.16 `Matrix<double> FINCH_DATA::Dnp1`

4.9.1.17 `double FINCH_DATA::Do = 1.0`

4.9.1.18 `double FINCH_DATA::dt = 0.0125`

4.9.1.19 `double FINCH_DATA::dt_old = 0.0125`

- 4.9.1.20 `double FINCH_DATA::dz = 0.1`
- 4.9.1.21 `int(* FINCH_DATA::evalprecon)(const Matrix< double > &b, Matrix< double > &p, const void *user_data)`
- 4.9.1.22 `int(* FINCH_DATA::evalres)(const Matrix< double > &x, Matrix< double > &res, const void *user_data)`
- 4.9.1.23 `bool FINCH_DATA::ExplicitFlux = false`
- 4.9.1.24 `std::vector<double> FINCH_DATA::fC_E`
- 4.9.1.25 `std::vector<double> FINCH_DATA::fC_I`
- 4.9.1.26 `std::vector<double> FINCH_DATA::fL_E`
- 4.9.1.27 `std::vector<double> FINCH_DATA::fL_I`
- 4.9.1.28 `Matrix<double> FINCH_DATA::Fn`
- 4.9.1.29 `Matrix<double> FINCH_DATA::Fnp1`
- 4.9.1.30 `std::vector<double> FINCH_DATA::fR_E`
- 4.9.1.31 `std::vector<double> FINCH_DATA::fR_I`
- 4.9.1.32 `Matrix<double> FINCH_DATA::gE`
- 4.9.1.33 `Matrix<double> FINCH_DATA::gl`
- 4.9.1.34 `bool FINCH_DATA::Iterative = true`
- 4.9.1.35 `double FINCH_DATA::kfn = 1.0`
- 4.9.1.36 `double FINCH_DATA::kfnp1 = 1.0`
- 4.9.1.37 `double FINCH_DATA::kIC = 1.0`
- 4.9.1.38 `Matrix<double> FINCH_DATA::kn`
- 4.9.1.39 `Matrix<double> FINCH_DATA::knp1`
- 4.9.1.40 `double FINCH_DATA::ko = 1.0`
- 4.9.1.41 `double FINCH_DATA::L = 1.0`
- 4.9.1.42 `double FINCH_DATA::lambda_E`
- 4.9.1.43 `double FINCH_DATA::lambda_I`
- 4.9.1.44 `int FINCH_DATA::LN = 10`
- 4.9.1.45 `int FINCH_DATA::max_iter = 20`
- 4.9.1.46 `std::vector<double> FINCH_DATA::ME`
- 4.9.1.47 `std::vector<double> FINCH_DATA::MI`

- 4.9.1.48 `std::vector<double> FINCH_DATA::NE`
- 4.9.1.49 `std::vector<double> FINCH_DATA::NI`
- 4.9.1.50 `int FINCH_DATA::nl_method = FINCH_Picard`
- 4.9.1.51 `bool FINCH_DATA::NormTrack = true`
- 4.9.1.52 `std::vector<double> FINCH_DATA::OE`
- 4.9.1.53 `std::vector<double> FINCH_DATA::OI`
- 4.9.1.54 `const void* FINCH_DATA::param_data`
- 4.9.1.55 `PICARD_DATA FINCH_DATA::picard_dat`
- 4.9.1.56 `PJFNK_DATA FINCH_DATA::pjfnk_dat`
- 4.9.1.57 `Matrix<double> FINCH_DATA::pres`
- 4.9.1.58 `Matrix<double> FINCH_DATA::res`
- 4.9.1.59 `int(* FINCH_DATA::resettime)(const void *user_data)`
- 4.9.1.60 `double FINCH_DATA::RIC = 1.0`
- 4.9.1.61 `Matrix<double> FINCH_DATA::Rn`
- 4.9.1.62 `Matrix<double> FINCH_DATA::Rnp1`
- 4.9.1.63 `double FINCH_DATA::Ro = 1.0`
- 4.9.1.64 `double FINCH_DATA::s = 1.0`
- 4.9.1.65 `int(* FINCH_DATA::setbcs)(const void *user_data)`
- 4.9.1.66 `int(* FINCH_DATA::setic)(const void *user_data)`
- 4.9.1.67 `int(* FINCH_DATA::setparams)(const void *user_data)`
- 4.9.1.68 `int(* FINCH_DATA::setpostprocess)(const void *user_data)`
- 4.9.1.69 `int(* FINCH_DATA::setpreprocess)(const void *user_data)`
- 4.9.1.70 `int(* FINCH_DATA::settime)(const void *user_data)`
- 4.9.1.71 `Matrix<double> FINCH_DATA::Sn`
- 4.9.1.72 `Matrix<double> FINCH_DATA::Snp1`
- 4.9.1.73 `int(* FINCH_DATA::solve)(const void *user_data)`
- 4.9.1.74 `bool FINCH_DATA::SteadyState = false`
- 4.9.1.75 `double FINCH_DATA::T = 1.0`

4.9.1.76 `double FINCH_DATA::t = 0.0`

4.9.1.77 `double FINCH_DATA::t_old = 0.0`

4.9.1.78 `double FINCH_DATA::tol_abs = 1e-6`

4.9.1.79 `double FINCH_DATA::tol_rel = 1e-6`

4.9.1.80 `int FINCH_DATA::total_iter = 0`

4.9.1.81 `Matrix<double> FINCH_DATA::u_star`

4.9.1.82 `double FINCH_DATA::uAvg = 0.0`

4.9.1.83 `double FINCH_DATA::uAvg_old = 0.0`

4.9.1.84 `Matrix<double> FINCH_DATA::ubest`

4.9.1.85 `double FINCH_DATA::uIC = 0.0`

4.9.1.86 `Matrix<double> FINCH_DATA::un`

4.9.1.87 `Matrix<double> FINCH_DATA::unm1`

4.9.1.88 `Matrix<double> FINCH_DATA::unp1`

4.9.1.89 `double FINCH_DATA::uo = 1.0`

4.9.1.90 `bool FINCH_DATA::Update = false`

4.9.1.91 `double FINCH_DATA::uT = 0.0`

4.9.1.92 `double FINCH_DATA::uT_old = 0.0`

4.9.1.93 `std::vector<double> FINCH_DATA::uz_I_E`

4.9.1.94 `std::vector<double> FINCH_DATA::uz_I_I`

4.9.1.95 `std::vector<double> FINCH_DATA::uz_lm1_E`

4.9.1.96 `std::vector<double> FINCH_DATA::uz_lm1_I`

4.9.1.97 `std::vector<double> FINCH_DATA::uz_lp1_E`

4.9.1.98 `std::vector<double> FINCH_DATA::uz_lp1_I`

4.9.1.99 `double FINCH_DATA::vIC = 1.0`

4.9.1.100 `Matrix<double> FINCH_DATA::vn`

4.9.1.101 `Matrix<double> FINCH_DATA::vnp1`

4.9.1.102 `double FINCH_DATA::vo = 1.0`

The documentation for this struct was generated from the following file:

- [finch.h](#)

4.10 GCR_DATA Struct Reference

Data structure for the implementation of the GCR algorithm for non-symmetric linear systems.

```
#include <lark.h>
```

Public Attributes

- int **restart** = -1
Restart parameter for outer iterations - default = 50.
- int **maxit** = 0
Maximum allowable outer iterations.
- int **iter_outer** = 0
Number of outer iterations taken.
- int **iter_inner** = 0
Number of inner iterations taken.
- int **total_iter** = 0
Total number of iterations taken.
- bool **breakdown** = false
Boolean to determine if a step has failed.
- double **alpha**
Inner iteration step size.
- double **beta**
Outer iteration step size.
- double **tol_rel** = 1e-6
Relative tolerance for convergence - default = 1e-6.
- double **tol_abs** = 1e-6
Absolute tolerance for convergence - default = 1e-6.
- double **res**
Absolute residual norm for linear system.
- double **relres**
Relative residual norm for linear system.
- double **relres_base**
Initial residual norm of the linear system.
- double **bestres**
Best found residual norm of the linear system.
- bool **Output** = true
True = print messages to the console.
- **Matrix**< double > **x**
Current solution to the linear system.
- **Matrix**< double > **bestx**
Best found solution to the linear system.
- **Matrix**< double > **r**
Residual Vector.
- **Matrix**< double > **c_temp**
Temporary c vector to be updated.
- **Matrix**< double > **u_temp**
Temporary u vector to be updated.
- std::vector< **Matrix**< double > > **u**
Vector span for updating x.
- std::vector< **Matrix**< double > > **c**
Vector span for updating r.
- **OPTRANS_DATA** **transpose_dat**
Data structure for Operator Transposition.

4.10.1 Detailed Description

Data structure for the implementation of the GCR algorithm for non-symmetric linear systems.

C-style object used in conjunction with the Generalized Conjugate Residual (GCR) algorithm for solving a non-symmetric linear system of equations. When the linear system in question has a positive-definite-symmetric component to it, then this algorithm is equivalent to GMRESRP. However, it is generally less efficient than GMRESRP and can suffer breakdowns.

4.10.2 Member Data Documentation

4.10.2.1 `double GCR_DATA::alpha`

Inner iteration step size.

4.10.2.2 `double GCR_DATA::bestres`

Best found residual norm of the linear system.

4.10.2.3 `Matrix<double> GCR_DATA::bestx`

Best found solution to the linear system.

4.10.2.4 `double GCR_DATA::beta`

Outer iteration step size.

4.10.2.5 `bool GCR_DATA::breakdown = false`

Boolean to determine if a step has failed.

4.10.2.6 `std::vector<Matrix<double> > GCR_DATA::c`

Vector span for updating r.

4.10.2.7 `Matrix<double> GCR_DATA::c_temp`

Temporary c vector to be updated.

4.10.2.8 `int GCR_DATA::iter_inner = 0`

Number of inner iterations taken.

4.10.2.9 `int GCR_DATA::iter_outer = 0`

Number of outer iterations taken.

4.10.2.10 `int GCR_DATA::maxit = 0`

Maximum allowable outer iterations.

4.10.2.11 `bool GCR_DATA::Output = true`

True = print messages to the console.

4.10.2.12 `Matrix<double> GCR_DATA::r`

Residual Vector.

4.10.2.13 `double GCR_DATA::relres`

Relative residual norm for linear system.

4.10.2.14 `double GCR_DATA::relres_base`

Initial residual norm of the linear system.

4.10.2.15 `double GCR_DATA::res`

Absolute residual norm for linear system.

4.10.2.16 `int GCR_DATA::restart = -1`

Restart parameter for outer iterations - default = 50.

4.10.2.17 `double GCR_DATA::tol_abs = 1e-6`

Absolute tolerance for convergence - default = 1e-6.

4.10.2.18 `double GCR_DATA::tol_rel = 1e-6`

Relative tolerance for convergence - default = 1e-6.

4.10.2.19 `int GCR_DATA::total_iter = 0`

Total number of iterations taken.

4.10.2.20 `OPTRANS_DATA GCR_DATA::transpose_dat`

Data structure for Operator Transposition.

4.10.2.21 `std::vector<Matrix<double> > GCR_DATA::u`

Vector span for updating x.

4.10.2.22 `Matrix<double> GCR_DATA::u_temp`

Temporary u vector to be updated.

4.10.2.23 `Matrix<double> GCR_DATA::x`

Current solution to the linear system.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.11 GMRESLP_DATA Struct Reference

Data structure for implementation of the Restarted GMRES algorithm with Left Preconditioning.

```
#include <lark.h>
```

Public Attributes

- int `restart` = -1
Restart parameter - default = min(vector_size,50)
- int `maxit` = 0
Maximum allowable iterations - default = min(vector_size,1000)
- int `iter` = 0
Number of iterations needed for convergence.
- int `steps` = 0
Total number of gmres iterations and krylov iterations.
- double `tol_rel` = 1e-6
Relative tolerance for convergence - default = 1e-6.
- double `tol_abs` = 1e-6
Absolution tolerance for convergence - default = 1e-6.
- double `res`
Absolution redisual norm of the linear system.
- double `relres`
Relative residual norm of the linear system.
- double `relres_base`
Initial residual norm of the linear system.
- double `bestres`
Best found residual norm of the linear system.
- bool `Output` = true
True = print messages to console.
- `Matrix< double > x`
Current solution to the linear system.
- `Matrix< double > bestx`
Best found solution to the linear system.
- `Matrix< double > r`
Residual vector for the linear system.
- `ARNOLDI_DATA arnoldi_dat`
Data structure for the kyrlov subspace.

4.11.1 Detailed Description

Data structure for implementation of the Restarted GMRES algorithm with Left Preconditioning.

C-style object used in conjunction with Generalized Minimum RESidual Left-Preconditioned (GMRESLP) and Full Orthogonalization Method (FOM) algorithms to iteratively or directly solve a linear system of equations. When using with GMRESLP, you can only check/observe the linear residuals before a restart or after the Arnoldi space is constructed. This is because this object uses Left-side Preconditioning. A faster routine may be GMRESRP, which is able to construct residuals after each Arnoldi iteration.

4.11.2 Member Data Documentation

4.11.2.1 ARNOLDI_DATA GMRESLP_DATA::arnoldi_dat

Data structure for the krylov subspace.

4.11.2.2 double GMRESLP_DATA::bestres

Best found residual norm of the linear system.

4.11.2.3 Matrix<double> GMRESLP_DATA::bestx

Best found solution to the linear system.

4.11.2.4 int GMRESLP_DATA::iter = 0

Number of iterations needed for convergence.

4.11.2.5 int GMRESLP_DATA::maxit = 0

Maximum allowable iterations - default = min(vector_size,1000)

4.11.2.6 bool GMRESLP_DATA::Output = true

True = print messages to console.

4.11.2.7 Matrix<double> GMRESLP_DATA::r

Residual vector for the linear system.

4.11.2.8 double GMRESLP_DATA::relres

Relative residual norm of the linear system.

4.11.2.9 double GMRESLP_DATA::relres_base

Initial residual norm of the linear system.

4.11.2.10 double GMRESLP_DATA::res

Absolute residual norm of the linear system.

4.11.2.11 int GMRESLP_DATA::restart = -1

Restart parameter - default = min(vector_size,50)

4.11.2.12 int GMRESLP_DATA::steps = 0

Total number of gmres iterations and krylov iterations.

4.11.2.13 double GMRESLP_DATA::tol_abs = 1e-6

Absolution tolerance for convergence - default = 1e-6.

4.11.2.14 double GMRESLP_DATA::tol_rel = 1e-6

Relative tolerance for convergence - default = 1e-6.

4.11.2.15 Matrix<double> GMRESLP_DATA::x

Current solution to the linear system.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.12 GMRESR_DATA Struct Reference

Data structure for the implementation of GCR with Nested GMRES preconditioning (i.e., GMRESR)

```
#include <lark.h>
```

Public Attributes

- int [gcr_restart](#) = -1
Number of GCR restarts (default = 50, max = N)
- int [gcr_maxit](#) = 0
Number of GCR iterations.
- int [gmres_restart](#) = -1
Number of GMRES restarts (max = 20)
- int [gmres_maxit](#) = 1
Number of GMRES iterations (max = 5, default = 1)
- int [N](#)
Dimension of the linear system.
- int [total_iter](#)
Total GMRES and GCR iterations.
- int [iter_outer](#)
Total GCR iterations.
- int [iter_inner](#)
Total GMRES iterations.
- bool [GCR_Output](#) = true
True = print GCR messages.
- bool [GMRES_Output](#) = false

- True = print GMRES messages.*
- double `gmres_tol` = 0.1
Tolerance relative to GCR iterations.
- double `gcr_rel_tol` = 1e-6
Relative outer residual tolerance.
- double `gcr_abs_tol` = 1e-6
Absolute outer residual tolerance.
- `Matrix< double > arg`
Argument matrix passed between preconditioner and iterator.
- `GCR_DATA gcr_dat`
Data structure for the outer GCR steps.
- `GMRESRP_DATA gmres_dat`
Data structure for the inner GMRES steps.
- `int(* matvec)(const Matrix< double > &x, Matrix< double > &Ax, const void *matvec_data)`
User supplied matrix-vector product function.
- `int(* terminal_precon)(const Matrix< double > &r, Matrix< double > &p, const void *precon_data)`
Optional user supplied terminal preconditioner.
- `const void * matvec_data`
Data structure for the user's matvec function.
- `const void * term_precon`
Data structure for the user's terminal preconditioner.

4.12.1 Detailed Description

Data structure for the implementation of GCR with Nested GMRES preconditioning (i.e., GMRESR)

C-style object to be used in conjunction with the Generalized Minimum RESidual Recursive (GMRESR) algorithm. Although the name suggests that this method used GMRES recursively, what it is actually doing is nesting GMRESRP iterations inside the GCR method to form a preconditioner for GCR. The name GMRESR came from literature (Vorst and Vuik, "GMRESR: A family of nested GMRES methods", 1991).

4.12.2 Member Data Documentation

4.12.2.1 `Matrix<double> GMRESR_DATA::arg`

Argument matrix passed between preconditioner and iterator.

4.12.2.2 `double GMRESR_DATA::gcr_abs_tol = 1e-6`

Absolute outer residual tolerance.

4.12.2.3 `GCR_DATA GMRESR_DATA::gcr_dat`

Data structure for the outer GCR steps.

4.12.2.4 `int GMRESR_DATA::gcr_maxit = 0`

Number of GCR iterations.

4.12.2.5 `bool GMRESR_DATA::GCR_Output = true`

True = print GCR messages.

4.12.2.6 `double GMRESR_DATA::gcr_rel_tol = 1e-6`

Relative outer residual tolerance.

4.12.2.7 `int GMRESR_DATA::gcr_restart = -1`

Number of GCR restarts (default = 50, max = N)

4.12.2.8 `GMRESRP_DATA GMRESR_DATA::gmres_dat`

Data structure for the inner GMRES steps.

4.12.2.9 `int GMRESR_DATA::gmres_maxit = 1`

Number of GMRES iterations (max = 5, default = 1)

4.12.2.10 `bool GMRESR_DATA::GMRES_Output = false`

True = print GMRES messages.

4.12.2.11 `int GMRESR_DATA::gmres_restart = -1`

Number of GMRES restarts (max = 20)

4.12.2.12 `double GMRESR_DATA::gmres_tol = 0.1`

Tolerance relative to GCR iterations.

4.12.2.13 `int GMRESR_DATA::iter_inner`

Total GMRES iterations.

4.12.2.14 `int GMRESR_DATA::iter_outer`

Total GCR iterations.

4.12.2.15 `int(* GMRESR_DATA::matvec)(const Matrix< double > &x, Matrix< double > &Ax, const void *matvec_data)`

User supplied matrix-vector product function.

4.12.2.16 `const void* GMRESR_DATA::matvec_data`

Data structure for the user's matvec function.

4.12.2.17 int GMRESR_DATA::N

Dimension of the linear system.

4.12.2.18 const void* GMRESR_DATA::term_precon

Data structure for the user's terminal preconditioner.

4.12.2.19 int(* GMRESR_DATA::terminal_precon)(const Matrix< double > &r, Matrix< double > &p, const void *precon_data)

Optional user supplied terminal preconditioner.

4.12.2.20 int GMRESR_DATA::total_iter

Total GMRES and GCR iterations.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.13 GMRESR_DATA Struct Reference

Data structure for the Restarted GMRES algorithm with Right Preconditioning.

```
#include <lark.h>
```

Public Attributes

- int [restart](#) = -1
Restart parameter - default = min(50,vector_size)
- int [maxit](#) = 0
Maximum allowable outer iterations.
- int [iter_outer](#) = 0
Total number of outer iterations.
- int [iter_inner](#) = 0
Total number of inner iterations.
- int [iter_total](#) = 0
Total number of overall iterations.
- double [tol_rel](#) = 1e-6
Relative tolerance for convergence - default = 1e-6.
- double [tol_abs](#) = 1e-6
Absolute tolerance for convergence - default = 1e-6.
- double [res](#)
Absolute residual norm for linear system.
- double [relres](#)
Relative residual norm for linear system.
- double [relres_base](#)
Initial residual norm of the linear system.
- double [bestres](#)
Best found residual norm of the linear system.

- `bool Output = true`
True = print messages to console.
- `Matrix< double > x`
Current solution to the linear system.
- `Matrix< double > bestx`
Best found solution to the linear system.
- `Matrix< double > r`
Residual vector for the linear system.
- `std::vector< Matrix< double > > Vk`
(N x k) orthonormal vector basis
- `std::vector< std::vector< double > > H`
(k+1 x k) upper Hessenberg storage matrix
- `std::vector< std::vector< double > > H_bar`
(k+1 x k) Factorized matrix
- `std::vector< double > y`
(k x 1) Vector search direction
- `std::vector< double > e0`
(k+1 x 1) Normalized vector with residual info
- `std::vector< double > e0_bar`
(k+1 x 1) Factorized normal vector
- `Matrix< double > w`
(N) x (1) interim result of the matrix_vector multiplication
- `Matrix< double > v`
(N) x (1) holding cell for the column entries of Vk and other interims
- `Matrix< double > sum`
(N) x (1) running sum of subspace vectors for use in altering w

4.13.1 Detailed Description

Data structure for the Restarted GMRES algorithm with Right Preconditioning.

C-style object used in conjunction with Generalized Minimum RESidual Right Preconditioned (GMRESRP) algorithm to iteratively solve a linear system of equations. Unlike GMRESLP, the GMRESRP method is capable of checking linear residuals at both the inner and outer steps. As a result, this algorithm may terminate earlier than GMRESLP if it has found a suitable solution during one of the inner steps.

4.13.2 Member Data Documentation

4.13.2.1 `double GMRESRP_DATA::bestres`

Best found residual norm of the linear system.

4.13.2.2 `Matrix<double> GMRESRP_DATA::bestx`

Best found solution to the linear system.

4.13.2.3 `std::vector< double > GMRESRP_DATA::e0`

(k+1 x 1) Normalized vector with residual info

4.13.2.4 `std::vector< double > GMRESRP_DATA::e0_bar`

(k+1 x 1) Factorized normal vector

4.13.2.5 `std::vector< std::vector< double > > GMRESRP_DATA::H`

(k+1 x k) upper Hessenberg storage matrix

4.13.2.6 `std::vector< std::vector< double > > GMRESRP_DATA::H_bar`

(k+1 x k) Factorized matrix

4.13.2.7 `int GMRESRP_DATA::iter_inner = 0`

Total number of inner iterations.

4.13.2.8 `int GMRESRP_DATA::iter_outer = 0`

Total number of outer iterations.

4.13.2.9 `int GMRESRP_DATA::iter_total = 0`

Total number of overall iterations.

4.13.2.10 `int GMRESRP_DATA::maxit = 0`

Maximum allowable outer iterations.

4.13.2.11 `bool GMRESRP_DATA::Output = true`

True = print messages to console.

4.13.2.12 `Matrix<double> GMRESRP_DATA::r`

Residual vector for the linear system.

4.13.2.13 `double GMRESRP_DATA::relres`

Relative residual norm for linear system.

4.13.2.14 `double GMRESRP_DATA::relres_base`

Initial residual norm of the linear system.

4.13.2.15 `double GMRESRP_DATA::res`

Absolute residual norm for linear system.

4.13.2.16 `int GMRESRP_DATA::restart = -1`

Restart parameter - default = min(50,vector_size)

4.13.2.17 `Matrix<double> GMRESRP_DATA::sum`

(N) x (1) running sum of subspace vectors for use in altering w

4.13.2.18 `double GMRESRP_DATA::tol_abs = 1e-6`

Absolute tolerance for convergence - default = 1e-6.

4.13.2.19 `double GMRESRP_DATA::tol_rel = 1e-6`

Relative tolerance for convergence - default = 1e-6.

4.13.2.20 `Matrix<double> GMRESRP_DATA::v`

(N) x (1) holding cell for the column entries of V_k and other interims

4.13.2.21 `std::vector< Matrix<double> > GMRESRP_DATA::Vk`

(N x k) orthonormal vector basis

4.13.2.22 `Matrix<double> GMRESRP_DATA::w`

(N) x (1) interim result of the matrix_vector multiplication

4.13.2.23 `Matrix<double> GMRESRP_DATA::x`

Current solution to the linear system.

4.13.2.24 `std::vector< double > GMRESRP_DATA::y`

(k x 1) Vector search direction

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.14 GPAST_DATA Struct Reference

```
#include <magpie.h>
```

Public Attributes

- double [x](#)
- double [y](#)
- double [He](#)
- double [q](#)

- `std::vector< double > gama_inf`
- `double qo`
- `double Plo`
- `std::vector< double > po`
- `double poi`
- `bool present`

4.14.1 Member Data Documentation

4.14.1.1 `std::vector<double> GPAST_DATA::gama_inf`

4.14.1.2 `double GPAST_DATA::He`

4.14.1.3 `double GPAST_DATA::Plo`

4.14.1.4 `std::vector<double> GPAST_DATA::po`

4.14.1.5 `double GPAST_DATA::poi`

4.14.1.6 `bool GPAST_DATA::present`

4.14.1.7 `double GPAST_DATA::q`

4.14.1.8 `double GPAST_DATA::qo`

4.14.1.9 `double GPAST_DATA::x`

4.14.1.10 `double GPAST_DATA::y`

The documentation for this struct was generated from the following file:

- [magpie.h](#)

4.15 GSTA_DATA Struct Reference

```
#include <magpie.h>
```

Public Attributes

- `double qmax`
- `int m`
- `std::vector< double > dHo`
- `std::vector< double > dSo`

4.15.1 Member Data Documentation

4.15.1.1 `std::vector<double> GSTA_DATA::dHo`

4.15.1.2 `std::vector<double> GSTA_DATA::dSo`

4.15.1.3 `int GSTA_DATA::m`

4.15.1.4 double GSTA_DATA::qmax

The documentation for this struct was generated from the following file:

- [magpie.h](#)

4.16 GSTA_OPT_DATA Struct Reference

```
#include <gsta_opt.h>
```

Public Attributes

- int [total_eval](#)
- int [n_par](#)
- double [qmax](#)
- int [iso](#)
- std::vector< std::vector< double > > [Fobj](#)
- std::vector< std::vector< double > > [q](#)
- std::vector< std::vector< double > > [P](#)
- std::vector< std::vector< double > > [best_par](#)
- std::vector< std::vector< double > > [Kno](#)
- std::vector< std::vector< std::vector< double > > > [all_pars](#)
- std::vector< std::vector< double > > [norms](#)
- std::vector< double > [opt_qmax](#)

4.16.1 Member Data Documentation

4.16.1.1 std::vector<std::vector<std::vector<double> > > GSTA_OPT_DATA::all_pars

4.16.1.2 std::vector<std::vector<double> > GSTA_OPT_DATA::best_par

4.16.1.3 std::vector<std::vector<double> > GSTA_OPT_DATA::Fobj

4.16.1.4 int GSTA_OPT_DATA::iso

4.16.1.5 std::vector<std::vector<double> > GSTA_OPT_DATA::Kno

4.16.1.6 int GSTA_OPT_DATA::n_par

4.16.1.7 std::vector<std::vector<double> > GSTA_OPT_DATA::norms

4.16.1.8 std::vector<double> GSTA_OPT_DATA::opt_qmax

4.16.1.9 std::vector<std::vector<double> > GSTA_OPT_DATA::P

4.16.1.10 std::vector<std::vector<double> > GSTA_OPT_DATA::q

4.16.1.11 double GSTA_OPT_DATA::qmax

4.16.1.12 int GSTA_OPT_DATA::total_eval

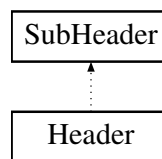
The documentation for this struct was generated from the following file:

- [gsta_opt.h](#)

4.17 Header Class Reference

```
#include <yaml_wrapper.h>
```

Inheritance diagram for Header:



Public Member Functions

- [Header](#) ()
- [~Header](#) ()
- [Header](#) (const [Header](#) &head)
- [Header](#) (std::string [name](#))
- [Header](#) (const [KeyValueMap](#) &map)
- [Header](#) (std::string [name](#), const [KeyValueMap](#) &map)
- [Header](#) (std::string key, const [SubHeader](#) &sub)
- [Header](#) & [operator=](#) (const [Header](#) &head)
- [ValueTypePair](#) & [operator\[\]](#) (const std::string key)
- [ValueTypePair](#) [operator\[\]](#) (const std::string key) const
- [SubHeader](#) & [operator\(\)](#) (const std::string key)
- [SubHeader](#) [operator\(\)](#) (const std::string key) const
- std::map< std::string, [SubHeader](#) > & [getSubMap](#) ()
- [KeyValueMap](#) & [getDataMap](#) ()
- [SubHeader](#) & [getSubHeader](#) (std::string key)
- std::map< std::string, [SubHeader](#) >::const_iterator [end](#) () const
- std::map< std::string, [SubHeader](#) >::iterator [end](#) ()
- std::map< std::string, [SubHeader](#) >::const_iterator [begin](#) () const
- std::map< std::string, [SubHeader](#) >::iterator [begin](#) ()
- void [clear](#) ()
- void [resetKeys](#) ()
- void [changeKey](#) (std::string oldKey, std::string newKey)
- void [addPair](#) (std::string key, std::string val)
- void [addPair](#) (std::string key, std::string val, int t)
- void [setName](#) (std::string [name](#))
- void [setAlias](#) (std::string [alias](#))

- void [setNameAliasPair](#) (std::string n, std::string a, int s)
- void [setState](#) (int state)
- void [DisplayContents](#) ()
- void [addSubKey](#) (std::string key)
- void [copyAnchor2Alias](#) (std::string alias, [SubHeader](#) &ref)
- int [size](#) ()
- std::string [getName](#) ()
- std::string [getAlias](#) ()
- int [getState](#) ()
- bool [isAlias](#) ()
- bool [isAnchor](#) ()
- [SubHeader](#) & [getAnchoredSub](#) (std::string alias)

Private Attributes

- std::map< std::string, [SubHeader](#) > [Sub_Map](#)

Additional Inherited Members

4.17.1 Constructor & Destructor Documentation

4.17.1.1 [Header::Header](#) ()

4.17.1.2 [Header::~~Header](#) ()

4.17.1.3 [Header::Header](#) (const [Header](#) & head)

4.17.1.4 [Header::Header](#) (std::string name)

4.17.1.5 [Header::Header](#) (const [KeyValueMap](#) & map)

4.17.1.6 [Header::Header](#) (std::string name, const [KeyValueMap](#) & map)

4.17.1.7 [Header::Header](#) (std::string key, const [SubHeader](#) & sub)

4.17.2 Member Function Documentation

4.17.2.1 void [Header::addPair](#) (std::string key, std::string val)

4.17.2.2 void [Header::addPair](#) (std::string key, std::string val, int t)

4.17.2.3 void [Header::addSubKey](#) (std::string key)

4.17.2.4 std::map< std::string, [SubHeader](#) >::const_iterator [Header::begin](#) () const

4.17.2.5 std::map< std::string, [SubHeader](#) >::iterator [Header::begin](#) ()

4.17.2.6 void [Header::changeKey](#) (std::string oldKey, std::string newKey)

4.17.2.7 void [Header::clear](#) ()

4.17.2.8 void [Header::copyAnchor2Alias](#) (std::string alias, [SubHeader](#) & ref)

4.17.2.9 void [Header::DisplayContents](#) ()

4.17.2.10 `std::map< std::string, SubHeader >::const_iterator Header::end () const`

4.17.2.11 `std::map< std::string, SubHeader >::iterator Header::end ()`

4.17.2.12 `std::string Header::getAlias ()`

4.17.2.13 `SubHeader & Header::getAnchoredSub (std::string alias)`

4.17.2.14 `KeyValueMap & Header::getDataMap ()`

4.17.2.15 `std::string Header::getName ()`

4.17.2.16 `int Header::getState ()`

4.17.2.17 `SubHeader & Header::getSubHeader (std::string key)`

4.17.2.18 `std::map< std::string, SubHeader > & Header::getSubMap ()`

4.17.2.19 `bool Header::isAlias ()`

4.17.2.20 `bool Header::isAnchor ()`

4.17.2.21 `SubHeader & Header::operator() (const std::string key)`

4.17.2.22 `SubHeader Header::operator() (const std::string key) const`

4.17.2.23 `Header & Header::operator= (const Header & head)`

4.17.2.24 `ValueTypePair & Header::operator[] (const std::string key)`

4.17.2.25 `ValueTypePair Header::operator[] (const std::string key) const`

4.17.2.26 `void Header::resetKeys ()`

4.17.2.27 `void Header::setAlias (std::string alias)`

4.17.2.28 `void Header::setName (std::string name)`

4.17.2.29 `void Header::setNameAliasPair (std::string n, std::string a, int s)`

4.17.2.30 `void Header::setState (int state)`

4.17.2.31 `int Header::size ()`

4.17.3 Member Data Documentation

4.17.3.1 `std::map<std::string, SubHeader> Header::Sub_Map [private]`

The documentation for this class was generated from the following files:

- [yaml_wrapper.h](#)
- [yaml_wrapper.cpp](#)

4.18 KeyValueType Class Reference

```
#include <yaml_wrapper.h>
```

Public Member Functions

- [KeyValueType](#) ()
- [~KeyValueType](#) ()
- [KeyValueType](#) (const std::map< std::string, std::string > &map)
- [KeyValueType](#) (std::string key, std::string value)
- [KeyValueType](#) (const [KeyValueType](#) &map)
- [KeyValueType](#) & [operator=](#) (const [KeyValueType](#) &map)
- [ValueTypePair](#) & [operator\[\]](#) (const std::string key)
- [ValueTypePair](#) [operator\[\]](#) (const std::string key) const
- std::map< std::string, [ValueTypePair](#) > & [getMap](#) ()
- std::map< std::string, [ValueTypePair](#) > ::const_iterator [end](#) () const
- std::map< std::string, [ValueTypePair](#) > ::iterator [end](#) ()
- std::map< std::string, [ValueTypePair](#) > ::const_iterator [begin](#) () const
- std::map< std::string, [ValueTypePair](#) > ::iterator [begin](#) ()
- void [clear](#) ()
- void [addKey](#) (std::string key)
- void [editValue4Key](#) (std::string val, std::string key)
- void [editValue4Key](#) (std::string val, int type, std::string key)
- void [addPair](#) (std::string key, [ValueTypePair](#) val)
- void [addPair](#) (std::string key, std::string val)
- void [addPair](#) (std::string key, std::string val, int type)
- void [findType](#) (std::string key)
- void [assertType](#) (std::string key, int type)
- void [findAllTypes](#) ()
- void [DisplayMap](#) ()
- int [size](#) ()
- std::string [getString](#) (std::string key)
- bool [getBool](#) (std::string key)
- double [getDouble](#) (std::string key)
- int [getInt](#) (std::string key)
- std::string [getValue](#) (std::string key)
- int [getType](#) (std::string key)
- [ValueTypePair](#) & [getPair](#) (std::string key)

Private Attributes

- std::map< std::string, [ValueTypePair](#) > [Key_Value](#)

4.18.1 Constructor & Destructor Documentation

4.18.1.1 KeyValueType::KeyValueType ()

4.18.1.2 KeyValueType::~~KeyValueType ()

4.18.1.3 KeyValueType::KeyValueType (const std::map< std::string, std::string > & map)

4.18.1.4 KeyValueType::KeyValueType (std::string key, std::string value)

4.18.1.5 KeyValueType::KeyValueType (const KeyValueType & map)

4.18.2 Member Function Documentation

4.18.2.1 void KeyValueType::addKey (std::string key)

4.18.2.2 void KeyValueType::addPair (std::string key, ValueTypePair val)

4.18.2.3 void KeyValueType::addPair (std::string key, std::string val)

4.18.2.4 void KeyValueType::addPair (std::string key, std::string val, int type)

4.18.2.5 void KeyValueType::assertType (std::string key, int type)

4.18.2.6 std::map< std::string, ValueTypePair >::const_iterator KeyValueType::begin () const

4.18.2.7 std::map< std::string, ValueTypePair >::iterator KeyValueType::begin ()

4.18.2.8 void KeyValueType::clear ()

4.18.2.9 void KeyValueType::DisplayMap ()

4.18.2.10 void KeyValueType::editValue4Key (std::string val, std::string key)

4.18.2.11 void KeyValueType::editValue4Key (std::string val, int type, std::string key)

4.18.2.12 std::map< std::string, ValueTypePair >::const_iterator KeyValueType::end () const

4.18.2.13 std::map< std::string, ValueTypePair >::iterator KeyValueType::end ()

4.18.2.14 void KeyValueType::findAllTypes ()

4.18.2.15 void KeyValueType::findType (std::string key)

4.18.2.16 bool KeyValueType::getBool (std::string key)

4.18.2.17 double KeyValueType::getDouble (std::string key)

4.18.2.18 int KeyValueType::getInt (std::string key)

4.18.2.19 std::map< std::string, ValueTypePair > & KeyValueType::getMap ()

4.18.2.20 ValueTypePair & KeyValueType::getPair (std::string key)

4.18.2.21 std::string KeyValueType::getString (std::string key)

4.18.2.22 `int KeyValueType::getType (std::string key)`

4.18.2.23 `std::string KeyValueType::getValue (std::string key)`

4.18.2.24 `KeyValueType & KeyValueType::operator= (const KeyValueType & map)`

4.18.2.25 `ValueTypePair & KeyValueType::operator[] (const std::string key)`

4.18.2.26 `ValueTypePair KeyValueType::operator[] (const std::string key) const`

4.18.2.27 `int KeyValueType::size ()`

4.18.3 Member Data Documentation

4.18.3.1 `std::map<std::string, ValueTypePair > KeyValueType::Key_Value [private]`

The documentation for this class was generated from the following files:

- [yaml_wrapper.h](#)
- [yaml_wrapper.cpp](#)

4.19 MAGPIE_DATA Struct Reference

```
#include <magpie.h>
```

Public Attributes

- `std::vector< GSTA_DATA > gsta_dat`
- `std::vector< mSPD_DATA > mspd_dat`
- `std::vector< GPAST_DATA > gpast_dat`
- `SYSTEM_DATA sys_dat`

4.19.1 Member Data Documentation

4.19.1.1 `std::vector<GPAST_DATA> MAGPIE_DATA::gpast_dat`

4.19.1.2 `std::vector<GSTA_DATA> MAGPIE_DATA::gsta_dat`

4.19.1.3 `std::vector<mSPD_DATA> MAGPIE_DATA::mspd_dat`

4.19.1.4 `SYSTEM_DATA MAGPIE_DATA::sys_dat`

The documentation for this struct was generated from the following file:

- [magpie.h](#)

4.20 MassBalance Class Reference

```
#include <shark.h>
```

Public Member Functions

- [MassBalance](#) ()
- [~MassBalance](#) ()
- void [Initialize_List](#) ([MasterSpeciesList](#) &[List](#))
- void [Display_Info](#) ()
- void [Set_Delta](#) (int *i*, double *v*)
- void [Set_TotalConcentration](#) (double *v*)
- void [Set_Name](#) (std::string *name*)
- double [Get_Delta](#) (int *i*)
- double [Sum_Delta](#) ()
- double [Get_TotalConcentration](#) ()
- std::string [Get_Name](#) ()
- double [Eval_Residual](#) (const [Matrix](#)< double > &*x*)

Protected Attributes

- [MasterSpeciesList](#) * [List](#)
- std::vector< double > [Delta](#)
- double [TotalConcentration](#)

Private Attributes

- std::string [Name](#)

4.20.1 Constructor & Destructor Documentation

4.20.1.1 [MassBalance::MassBalance](#) ()

4.20.1.2 [MassBalance::~~MassBalance](#) ()

4.20.2 Member Function Documentation

4.20.2.1 void [MassBalance::Display_Info](#) ()

4.20.2.2 double [MassBalance::Eval_Residual](#) (const [Matrix](#)< double > & *x*)

4.20.2.3 double [MassBalance::Get_Delta](#) (int *i*)

4.20.2.4 std::string [MassBalance::Get_Name](#) ()

4.20.2.5 double [MassBalance::Get_TotalConcentration](#) ()

4.20.2.6 void [MassBalance::Initialize_List](#) ([MasterSpeciesList](#) & *List*)

4.20.2.7 void [MassBalance::Set_Delta](#) (int *i*, double *v*)

4.20.2.8 void [MassBalance::Set_Name](#) (std::string *name*)

4.20.2.9 void [MassBalance::Set_TotalConcentration](#) (double *v*)

4.20.2.10 double [MassBalance::Sum_Delta](#) ()

4.20.3 Member Data Documentation

4.20.3.1 `std::vector<double> MassBalance::Delta` [protected]

4.20.3.2 `MasterSpeciesList* MassBalance::List` [protected]

4.20.3.3 `std::string MassBalance::Name` [private]

4.20.3.4 `double MassBalance::TotalConcentration` [protected]

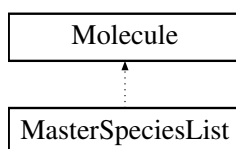
The documentation for this class was generated from the following files:

- [shark.h](#)
- [shark.cpp](#)

4.21 MasterSpeciesList Class Reference

```
#include <shark.h>
```

Inheritance diagram for MasterSpeciesList:



Public Member Functions

- [MasterSpeciesList](#) ()
- [~MasterSpeciesList](#) ()
- [MasterSpeciesList](#) (const [MasterSpeciesList](#) &msl)
- [MasterSpeciesList](#) & [operator=](#) (const [MasterSpeciesList](#) &msl)
- void [set_list_size](#) (int i)
- void [set_species](#) (int i, std::string formula)
- void [set_species](#) (int i, int [charge](#), double enthalpy, double entropy, double energy, bool HS, bool G, std::string [Phase](#), std::string [Name](#), std::string [Formula](#), std::string lin_formula)
- void [DisplayInfo](#) (int i)
- void [DisplayAll](#) ()
- void [DisplayConcentrations](#) ([Matrix](#)< double > &C)
- void [set_alkalinity](#) (double alk)
- int [list_size](#) ()
- [Molecule](#) & [get_species](#) (int i)
- int [get_index](#) (std::string name)
- double [charge](#) (int i)
- double [alkalinity](#) ()
- std::string [speciesName](#) (int i)
- double [Eval_ChargeResidual](#) (const [Matrix](#)< double > &x)

Protected Attributes

- int [size](#)
- std::vector< [Molecule](#) > [species](#)
- double [residual_alkalinity](#)

Additional Inherited Members

4.21.1 Constructor & Destructor Documentation

4.21.1.1 MasterSpeciesList::MasterSpeciesList ()

4.21.1.2 MasterSpeciesList::~~MasterSpeciesList ()

4.21.1.3 MasterSpeciesList::MasterSpeciesList (const MasterSpeciesList & *msl*)

4.21.2 Member Function Documentation

4.21.2.1 double MasterSpeciesList::alkalinity ()

4.21.2.2 double MasterSpeciesList::charge (int *i*)

4.21.2.3 void MasterSpeciesList::DisplayAll ()

4.21.2.4 void MasterSpeciesList::DisplayConcentrations (Matrix< double > & *C*)

4.21.2.5 void MasterSpeciesList::DisplayInfo (int *i*)

4.21.2.6 double MasterSpeciesList::Eval_ChargeResidual (const Matrix< double > & *x*)

4.21.2.7 int MasterSpeciesList::get_index (std::string *name*)

4.21.2.8 Molecule & MasterSpeciesList::get_species (int *i*)

4.21.2.9 int MasterSpeciesList::list_size ()

4.21.2.10 MasterSpeciesList & MasterSpeciesList::operator= (const MasterSpeciesList & *msl*)

4.21.2.11 void MasterSpeciesList::set_alkalinity (double *alk*)

4.21.2.12 void MasterSpeciesList::set_list_size (int *i*)

4.21.2.13 void MasterSpeciesList::set_species (int *i*, std::string *formula*)

4.21.2.14 void MasterSpeciesList::set_species (int *i*, int *charge*, double *enthalpy*, double *entropy*, double *energy*, bool *HS*, bool *G*, std::string *Phase*, std::string *Name*, std::string *Formula*, std::string *lin_formula*)

4.21.2.15 std::string MasterSpeciesList::speciesName (int *i*)

4.21.3 Member Data Documentation

4.21.3.1 double MasterSpeciesList::residual_alkalinity [protected]

4.21.3.2 int MasterSpeciesList::size [protected]

4.21.3.3 std::vector<Molecule> MasterSpeciesList::species [protected]

The documentation for this class was generated from the following files:

- [shark.h](#)
- [shark.cpp](#)

4.22 Matrix< T > Class Template Reference

```
#include <macaw.h>
```

Public Member Functions

- [Matrix](#) (int [rows](#), int [columns](#))
- [T & operator\(\)](#) (int i, int j)
- [T operator\(\)](#) (int i, int j) const
- [Matrix](#) (const [Matrix](#) &M)
- [Matrix & operator=](#) (const [Matrix](#) &M)
- [Matrix](#) ()
- [~Matrix](#) ()
- void [set_size](#) (int i, int j)
- void [zeros](#) ()
- void [edit](#) (int i, int j, T value)
- int [rows](#) ()
- int [columns](#) ()
- [T determinate](#) ()
- [T norm](#) ()
- [T sum](#) ()
- [T inner_product](#) (const [Matrix](#) &x)
- [Matrix & cofactor](#) (const [Matrix](#) &M)
- [Matrix operator+](#) (const [Matrix](#) &M)
- [Matrix operator-](#) (const [Matrix](#) &M)
- [Matrix operator*](#) (const T)
- [Matrix operator/](#) (const T)
- [Matrix operator*](#) (const [Matrix](#) &M)
- [Matrix & transpose](#) (const [Matrix](#) &M)
- [Matrix & transpose_multiply](#) (const [Matrix](#) &MT, const [Matrix](#) &v)
- [Matrix & adjoint](#) (const [Matrix](#) &M)
- [Matrix & inverse](#) (const [Matrix](#) &M)
- void [Display](#) (const std::string Name)
- [Matrix & tridiagonalSolve](#) (const [Matrix](#) &A, const [Matrix](#) &b)
- [Matrix & ladshawSolve](#) (const [Matrix](#) &A, const [Matrix](#) &d)
- [Matrix & tridiagonalFill](#) (const T A, const T B, const T C, bool [Spherical](#))
- [Matrix & naturalLaplacian3D](#) (int m)
- [Matrix & sphericalBCFill](#) (int node, const T coeff, T variable)
- [Matrix & ConstantICFill](#) (const T IC)
- [Matrix & SolnTransform](#) (const [Matrix](#) &A, bool Forward)
- [T sphericalAvg](#) (double radius, double dr, double bound, bool Dirichlet)
- [T IntegralAvg](#) (double radius, double dr, double bound, bool Dirichlet)
- [T IntegralTotal](#) (double dr, double bound, bool Dirichlet)
- [Matrix & tridiagonalVectorFill](#) (const std::vector< T > &A, const std::vector< T > &B, const std::vector< T > &C)
- [Matrix & columnVectorFill](#) (const std::vector< T > &A)
- [Matrix & columnProjection](#) (const [Matrix](#) &b, const [Matrix](#) &b_old, const double dt, const double dt_old)
- [Matrix & dirichletBCFill](#) (int node, const T coeff, T variable)
- [Matrix & diagonalSolve](#) (const [Matrix](#) &D, const [Matrix](#) &v)
- [Matrix & upperTriangularSolve](#) (const [Matrix](#) &U, const [Matrix](#) &v)
- [Matrix & lowerTriangularSolve](#) (const [Matrix](#) &L, const [Matrix](#) &v)
- [Matrix & upperHessenberg2Triangular](#) ([Matrix](#) &b)
- [Matrix & lowerHessenberg2Triangular](#) ([Matrix](#) &b)
- [Matrix & upperHessenbergSolve](#) (const [Matrix](#) &H, const [Matrix](#) &v)

- [Matrix](#) & [lowerHessenbergSolve](#) (const [Matrix](#) &H, const [Matrix](#) &v)
- [Matrix](#) & [columnExtract](#) (int j, const [Matrix](#) &M)
- [Matrix](#) & [rowExtract](#) (int i, const [Matrix](#) &M)
- [Matrix](#) & [columnReplace](#) (int j, const [Matrix](#) &v)
- [Matrix](#) & [rowReplace](#) (int i, const [Matrix](#) &v)
- void [rowShrink](#) ()
- void [columnShrink](#) ()
- void [rowExtend](#) (const [Matrix](#) &v)
- void [columnExtend](#) (const [Matrix](#) &v)

Protected Attributes

- int [num_rows](#)
- int [num_cols](#)
- std::vector< T > [Data](#)

4.22.1 Constructor & Destructor Documentation

- 4.22.1.1 `template<class T> Matrix< T >::Matrix (int rows, int columns)`
- 4.22.1.2 `template<class T> Matrix< T >::Matrix (const Matrix< T > & M)`
- 4.22.1.3 `template<class T> Matrix< T >::Matrix ()`
- 4.22.1.4 `template<class T> Matrix< T >::~~Matrix ()`

4.22.2 Member Function Documentation

- 4.22.2.1 `template<class T> Matrix< T > & Matrix< T >::adjoint (const Matrix< T > & M)`
- 4.22.2.2 `template<class T> Matrix< T > & Matrix< T >::cofactor (const Matrix< T > & M)`
- 4.22.2.3 `template<class T> void Matrix< T >::columnExtend (const Matrix< T > & v)`
- 4.22.2.4 `template<class T> Matrix< T > & Matrix< T >::columnExtract (int j, const Matrix< T > & M)`
- 4.22.2.5 `template<class T> Matrix< T > & Matrix< T >::columnProjection (const Matrix< T > & b, const Matrix< T > & b_old, const double dt, const double dt_old)`
- 4.22.2.6 `template<class T> Matrix< T > & Matrix< T >::columnReplace (int j, const Matrix< T > & v)`
- 4.22.2.7 `template<class T> int Matrix< T >::columns ()`
- 4.22.2.8 `template<class T> void Matrix< T >::columnShrink ()`
- 4.22.2.9 `template<class T> Matrix< T > & Matrix< T >::columnVectorFill (const std::vector< T > & A)`
- 4.22.2.10 `template<class T> Matrix< T > & Matrix< T >::ConstantICFill (const T IC)`
- 4.22.2.11 `template<class T> T Matrix< T >::determinate ()`
- 4.22.2.12 `template<class T> Matrix< T > & Matrix< T >::diagonalSolve (const Matrix< T > & D, const Matrix< T > & v)`

- 4.22.2.13 `template<class T> Matrix< T > & Matrix< T >::dirichletBCFill (int node, const T coeff, T variable)`
- 4.22.2.14 `template<class T> void Matrix< T >::Display (const std::string Name)`
- 4.22.2.15 `template<class T> void Matrix< T >::edit (int i, int j, T value)`
- 4.22.2.16 `template<class T> T Matrix< T >::inner_product (const Matrix< T > & x)`
- 4.22.2.17 `template<class T> T Matrix< T >::IntegralAvg (double radius, double dr, double bound, bool Dirichlet)`
- 4.22.2.18 `template<class T> T Matrix< T >::IntegralTotal (double dr, double bound, bool Dirichlet)`
- 4.22.2.19 `template<class T> Matrix< T > & Matrix< T >::inverse (const Matrix< T > & M)`
- 4.22.2.20 `template<class T> Matrix< T > & Matrix< T >::ladshawSolve (const Matrix< T > & A, const Matrix< T > & d)`
- 4.22.2.21 `template<class T> Matrix< T > & Matrix< T >::lowerHessenberg2Triangular (Matrix< T > & b)`
- 4.22.2.22 `template<class T> Matrix< T > & Matrix< T >::lowerHessenbergSolve (const Matrix< T > & H, const Matrix< T > & v)`
- 4.22.2.23 `template<class T> Matrix< T > & Matrix< T >::lowerTriangularSolve (const Matrix< T > & L, const Matrix< T > & v)`
- 4.22.2.24 `template<class T> Matrix< T > & Matrix< T >::naturalLaplacian3D (int m)`
- 4.22.2.25 `template<class T> T Matrix< T >::norm ()`
- 4.22.2.26 `template<class T> T & Matrix< T >::operator() (int i, int j)`
- 4.22.2.27 `template<class T> T Matrix< T >::operator() (int i, int j) const`
- 4.22.2.28 `template<class T> Matrix< T > Matrix< T >::operator* (const T a)`
- 4.22.2.29 `template<class T> Matrix< T > Matrix< T >::operator* (const Matrix< T > & M)`
- 4.22.2.30 `template<class T> Matrix< T > Matrix< T >::operator+ (const Matrix< T > & M)`
- 4.22.2.31 `template<class T> Matrix< T > Matrix< T >::operator- (const Matrix< T > & M)`
- 4.22.2.32 `template<class T> Matrix< T > Matrix< T >::operator/ (const T a)`
- 4.22.2.33 `template<class T> Matrix< T > & Matrix< T >::operator= (const Matrix< T > & M)`
- 4.22.2.34 `template<class T> void Matrix< T >::rowExtend (const Matrix< T > & v)`
- 4.22.2.35 `template<class T> Matrix< T > & Matrix< T >::rowExtract (int i, const Matrix< T > & M)`
- 4.22.2.36 `template<class T> Matrix< T > & Matrix< T >::rowReplace (int i, const Matrix< T > & v)`
- 4.22.2.37 `template<class T> int Matrix< T >::rows ()`
- 4.22.2.38 `template<class T> void Matrix< T >::rowShrink ()`
- 4.22.2.39 `template<class T> void Matrix< T >::set_size (int i, int j)`

- 4.22.2.40 `template<class T> Matrix< T> & Matrix< T>::SolnTransform (const Matrix< T> & A, bool Forward)`
- 4.22.2.41 `template<class T> T Matrix< T>::sphericalAvg (double radius, double dr, double bound, bool Dirichlet)`
- 4.22.2.42 `template<class T> Matrix< T> & Matrix< T>::sphericalBCFill (int node, const T coeff, T variable)`
- 4.22.2.43 `template<class T> T Matrix< T>::sum ()`
- 4.22.2.44 `template<class T> Matrix< T> & Matrix< T>::transpose (const Matrix< T> & M)`
- 4.22.2.45 `template<class T> Matrix< T> & Matrix< T>::transpose_multiply (const Matrix< T> & MT, const Matrix< T> & v)`
- 4.22.2.46 `template<class T> Matrix< T> & Matrix< T>::tridiagonalFill (const T A, const T B, const T C, bool Spherical)`
- 4.22.2.47 `template<class T> Matrix< T> & Matrix< T>::tridiagonalSolve (const Matrix< T> & A, const Matrix< T> & b)`
- 4.22.2.48 `template<class T> Matrix< T> & Matrix< T>::tridiagonalVectorFill (const std::vector< T> & A, const std::vector< T> & B, const std::vector< T> & C)`
- 4.22.2.49 `template<class T> Matrix< T> & Matrix< T>::upperHessenberg2Triangular (Matrix< T> & b)`
- 4.22.2.50 `template<class T> Matrix< T> & Matrix< T>::upperHessenbergSolve (const Matrix< T> & H, const Matrix< T> & v)`
- 4.22.2.51 `template<class T> Matrix< T> & Matrix< T>::upperTriangularSolve (const Matrix< T> & U, const Matrix< T> & v)`
- 4.22.2.52 `template<class T> void Matrix< T>::zeros ()`

4.22.3 Member Data Documentation

- 4.22.3.1 `template<class T> std::vector<T> Matrix< T>::Data [protected]`
- 4.22.3.2 `template<class T> int Matrix< T>::num_cols [protected]`
- 4.22.3.3 `template<class T> int Matrix< T>::num_rows [protected]`

The documentation for this class was generated from the following file:

- [macaw.h](#)

4.23 Mechanism Class Reference

```
#include <shark.h>
```

Protected Attributes

- [MasterSpeciesList](#) * [List](#)
- std::vector< [UnsteadyReaction](#) > [reactions](#)
- std::vector< double > [weight](#)
- int [species_index](#)

4.23.1 Member Data Documentation

4.23.1.1 **MasterSpeciesList*** Mechanism::List [protected]

4.23.1.2 **std::vector<UnsteadyReaction>** Mechanism::reactions [protected]

4.23.1.3 **int** Mechanism::species_index [protected]

4.23.1.4 **std::vector<double>** Mechanism::weight [protected]

The documentation for this class was generated from the following file:

- [shark.h](#)

4.24 MIXED_GAS Struct Reference

```
#include <egret.h>
```

Public Attributes

- int [N](#)
- bool [CheckMolefractions](#) = true
- double [total_pressure](#)
- double [gas_temperature](#)
- double [velocity](#)
- double [char_length](#)
- std::vector< double > [molefraction](#)
- double [total_density](#)
- double [total_dyn_vis](#)
- double [kinematic_viscosity](#)
- double [total_molecular_weight](#)
- double [total_specific_heat](#)
- double [Reynolds](#)
- [Matrix](#)< double > [binary_diffusion](#)
- std::vector< [PURE_GAS](#) > [species_dat](#)

4.24.1 Member Data Documentation

4.24.1.1 **Matrix<double>** MIXED_GAS::binary_diffusion

4.24.1.2 **double** MIXED_GAS::char_length

4.24.1.3 **bool** MIXED_GAS::CheckMolefractions = true

4.24.1.4 **double** MIXED_GAS::gas_temperature

4.24.1.5 **double** MIXED_GAS::kinematic_viscosity

4.24.1.6 **std::vector<double>** MIXED_GAS::molefraction

4.24.1.7 **int** MIXED_GAS::N

4.24.1.8 double MIXED_GAS::Reynolds

4.24.1.9 std::vector<PURE_GAS> MIXED_GAS::species_dat

4.24.1.10 double MIXED_GAS::total_density

4.24.1.11 double MIXED_GAS::total_dyn_vis

4.24.1.12 double MIXED_GAS::total_molecular_weight

4.24.1.13 double MIXED_GAS::total_pressure

4.24.1.14 double MIXED_GAS::total_specific_heat

4.24.1.15 double MIXED_GAS::velocity

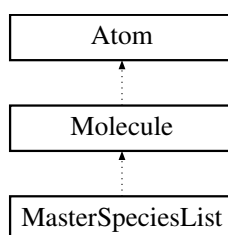
The documentation for this struct was generated from the following file:

- [egret.h](#)

4.25 Molecule Class Reference

```
#include <mola.h>
```

Inheritance diagram for Molecule:



Public Member Functions

- [Molecule](#) ()
- [~Molecule](#) ()
- [Molecule](#) (int [charge](#), double enthalpy, double entropy, double energy, bool HS, bool G, std::string [Phase](#), std::string [Name](#), std::string [Formula](#), std::string lin_formula)
- void [Register](#) (int [charge](#), double enthalpy, double entropy, double energy, bool HS, bool G, std::string [Phase](#), std::string [Name](#), std::string [Formula](#), std::string lin_formula)
- void [Register](#) (std::string formula)
- void [setFormula](#) (std::string form)
- void [recalculateMolarWeight](#) ()
- void [setMolarWeight](#) (double mw)
- void [editCharge](#) (int c)
- void [editOneOxidationState](#) (int state, std::string [Symbol](#))
- void [editAllOxidationStates](#) (int state, std::string [Symbol](#))
- void [calculateAvgOxiState](#) (std::string [Symbol](#))
- void [editEnthalpy](#) (double enthalpy)
- void [editEntropy](#) (double entropy)
- void [editHS](#) (double H, double S)

- void [editEnergy](#) (double *energy*)
- void [removeOneAtom](#) (std::string *Symbol*)
- void [removeAllAtoms](#) (std::string *Symbol*)
- int [Charge](#) ()
- double [MolarWeight](#) ()
- bool [HaveHS](#) ()
- bool [HaveEnergy](#) ()
- bool [isRegistered](#) ()
- double [Enthalpy](#) ()
- double [Entropy](#) ()
- double [Energy](#) ()
- std::string [MoleculeName](#) ()
- std::string [MolecularFormula](#) ()
- std::string [MoleculePhase](#) ()
- void [DisplayInfo](#) ()

Protected Attributes

- int [charge](#)
- double [molar_weight](#)
- double [formation_enthalpy](#)
- double [formation_entropy](#)
- double [formation_energy](#)
- std::string [Phase](#)
- std::vector< [Atom](#) > [atoms](#)

Private Attributes

- std::string [Name](#)
- std::string [Formula](#)
- bool [haveG](#)
- bool [haveHS](#)
- bool [registered](#)

Additional Inherited Members

4.25.1 Constructor & Destructor Documentation

4.25.1.1 [Molecule::Molecule](#) ()

4.25.1.2 [Molecule::~~Molecule](#) ()

4.25.1.3 [Molecule::Molecule](#) (int *charge*, double *enthalpy*, double *entropy*, double *energy*, bool *HS*, bool *G*, std::string *Phase*, std::string *Name*, std::string *Formula*, std::string *lin_formula*)

4.25.2 Member Function Documentation

4.25.2.1 [void Molecule::calculateAvgOxiState](#) (std::string *Symbol*)

4.25.2.2 [int Molecule::Charge](#) ()

4.25.2.3 [void Molecule::DisplayInfo](#) ()

- 4.25.2.4 void Molecule::editAllOxidationStates (int *state*, std::string *Symbol*)
- 4.25.2.5 void Molecule::editCharge (int *c*)
- 4.25.2.6 void Molecule::editEnergy (double *energy*)
- 4.25.2.7 void Molecule::editEnthalpy (double *enthalpy*)
- 4.25.2.8 void Molecule::editEntropy (double *entropy*)
- 4.25.2.9 void Molecule::editHS (double *H*, double *S*)
- 4.25.2.10 void Molecule::editOneOxidationState (int *state*, std::string *Symbol*)
- 4.25.2.11 double Molecule::Energy ()
- 4.25.2.12 double Molecule::Enthalpy ()
- 4.25.2.13 double Molecule::Entropy ()
- 4.25.2.14 bool Molecule::HaveEnergy ()
- 4.25.2.15 bool Molecule::HaveHS ()
- 4.25.2.16 bool Molecule::isRegistered ()
- 4.25.2.17 double Molecule::MolarWeight ()
- 4.25.2.18 std::string Molecule::MolecularFormula ()
- 4.25.2.19 std::string Molecule::MoleculeName ()
- 4.25.2.20 std::string Molecule::MoleculePhase ()
- 4.25.2.21 void Molecule::recalculateMolarWeight ()
- 4.25.2.22 void Molecule::Register (int *charge*, double *enthalpy*, double *entropy*, double *energy*, bool *HS*, bool *G*, std::string *Phase*, std::string *Name*, std::string *Formula*, std::string *lin_formula*)
- 4.25.2.23 void Molecule::Register (std::string *formula*)
- 4.25.2.24 void Molecule::removeAllAtoms (std::string *Symbol*)
- 4.25.2.25 void Molecule::removeOneAtom (std::string *Symbol*)
- 4.25.2.26 void Molecule::setFormula (std::string *form*)
- 4.25.2.27 void Molecule::setMolarWeigth (double *mw*)

4.25.3 Member Data Documentation

- 4.25.3.1 std::vector<Atom> Molecule::atoms [protected]
- 4.25.3.2 int Molecule::charge [protected]
- 4.25.3.3 double Molecule::formation_energy [protected]

4.25.3.4 `double Molecule::formation_enthalpy` [protected]

4.25.3.5 `double Molecule::formation_entropy` [protected]

4.25.3.6 `std::string Molecule::Formula` [private]

4.25.3.7 `bool Molecule::haveG` [private]

4.25.3.8 `bool Molecule::haveHS` [private]

4.25.3.9 `double Molecule::molar_weight` [protected]

4.25.3.10 `std::string Molecule::Name` [private]

4.25.3.11 `std::string Molecule::Phase` [protected]

4.25.3.12 `bool Molecule::registered` [private]

The documentation for this class was generated from the following files:

- [mola.h](#)
- [mola.cpp](#)

4.26 MONKFISH_DATA Struct Reference

```
#include <monkfish.h>
```

Public Attributes

- unsigned long int `total_steps` = 0
- double `time_old` = 0.0
- double `time` = 0.0
- bool `Print2File` = true
- bool `Print2Console` = true
- bool `DirichletBC` = true
- bool `NonLinear` = false
- bool `haveMinMax` = false
- bool `MultiScale` = true
- int `level` = 2
- double `t_counter` = 0.0
- double `t_print`
- int `NumComp`
- double `end_time`
- double `total_sorption_old`
- double `total_sorption`
- double `single_fiber_density`
- double `avg_fiber_density`
- double `max_fiber_density`
- double `min_fiber_density`
- double `max_porosity`
- double `min_porosity`
- double `domain_diameter`
- FILE * `Output`

- `double(* eval_eps)(int i, int l, const void *user_data)`
- `double(* eval_rho)(int i, int l, const void *user_data)`
- `double(* eval_Dex)(int i, int l, const void *user_data)`
- `double(* eval_ads)(int i, int l, const void *user_data)`
- `double(* eval_Ret)(int i, int l, const void *user_data)`
- `double(* eval_Cex)(int i, const void *user_data)`
- `double(* eval_kf)(int i, const void *user_data)`
- `const void * user_data`
- `std::vector< FINCH_DATA > finch_dat`
- `std::vector< MONKFISH_PARAM > param_dat`
- `std::vector< DOGFISH_DATA > dog_dat`

4.26.1 Member Data Documentation

- 4.26.1.1 `double MONKFISH_DATA::avg_fiber_density`
- 4.26.1.2 `bool MONKFISH_DATA::DirichletBC = true`
- 4.26.1.3 `std::vector< DOGFISH_DATA > MONKFISH_DATA::dog_dat`
- 4.26.1.4 `double MONKFISH_DATA::domain_diameter`
- 4.26.1.5 `double MONKFISH_DATA::end_time`
- 4.26.1.6 `double(* MONKFISH_DATA::eval_ads)(int i, int l, const void *user_data)`
- 4.26.1.7 `double(* MONKFISH_DATA::eval_Cex)(int i, const void *user_data)`
- 4.26.1.8 `double(* MONKFISH_DATA::eval_Dex)(int i, int l, const void *user_data)`
- 4.26.1.9 `double(* MONKFISH_DATA::eval_eps)(int i, int l, const void *user_data)`
- 4.26.1.10 `double(* MONKFISH_DATA::eval_kf)(int i, const void *user_data)`
- 4.26.1.11 `double(* MONKFISH_DATA::eval_Ret)(int i, int l, const void *user_data)`
- 4.26.1.12 `double(* MONKFISH_DATA::eval_rho)(int i, int l, const void *user_data)`
- 4.26.1.13 `std::vector< FINCH_DATA > MONKFISH_DATA::finch_dat`
- 4.26.1.14 `bool MONKFISH_DATA::haveMinMax = false`
- 4.26.1.15 `int MONKFISH_DATA::level = 2`
- 4.26.1.16 `double MONKFISH_DATA::max_fiber_density`
- 4.26.1.17 `double MONKFISH_DATA::max_porosity`
- 4.26.1.18 `double MONKFISH_DATA::min_fiber_density`
- 4.26.1.19 `double MONKFISH_DATA::min_porosity`
- 4.26.1.20 `bool MONKFISH_DATA::MultiScale = true`
- 4.26.1.21 `bool MONKFISH_DATA::NonLinear = false`

- 4.26.1.22 int MONKFISH_DATA::NumComp
- 4.26.1.23 FILE* MONKFISH_DATA::Output
- 4.26.1.24 std::vector<MONKFISH_PARAM> MONKFISH_DATA::param_dat
- 4.26.1.25 bool MONKFISH_DATA::Print2Console = true
- 4.26.1.26 bool MONKFISH_DATA::Print2File = true
- 4.26.1.27 double MONKFISH_DATA::single_fiber_density
- 4.26.1.28 double MONKFISH_DATA::t_counter = 0.0
- 4.26.1.29 double MONKFISH_DATA::t_print
- 4.26.1.30 double MONKFISH_DATA::time = 0.0
- 4.26.1.31 double MONKFISH_DATA::time_old = 0.0
- 4.26.1.32 double MONKFISH_DATA::total_sorption
- 4.26.1.33 double MONKFISH_DATA::total_sorption_old
- 4.26.1.34 unsigned long int MONKFISH_DATA::total_steps = 0
- 4.26.1.35 const void* MONKFISH_DATA::user_data

The documentation for this struct was generated from the following file:

- [monkfish.h](#)

4.27 MONKFISH_PARAM Struct Reference

```
#include <monkfish.h>
```

Public Attributes

- double [interparticle_diffusion](#)
- double [exterior_concentration](#)
- double [exterior_transfer_coeff](#)
- double [sorbed_molefraction](#)
- double [initial_sorption](#)
- double [sorption_bc](#)
- double [intraparticle_diffusion](#)
- double [film_transfer_coeff](#)
- [Matrix](#)< double > [avg_sorption](#)
- [Matrix](#)< double > [avg_sorption_old](#)
- [Molecule species](#)

4.27.1 Member Data Documentation

- 4.27.1.1 **Matrix**<double> MONKFISH_PARAM::avg_sorption
- 4.27.1.2 **Matrix**<double> MONKFISH_PARAM::avg_sorption_old
- 4.27.1.3 **double** MONKFISH_PARAM::exterior_concentration
- 4.27.1.4 **double** MONKFISH_PARAM::exterior_transfer_coeff
- 4.27.1.5 **double** MONKFISH_PARAM::film_transfer_coeff
- 4.27.1.6 **double** MONKFISH_PARAM::initial_sorption
- 4.27.1.7 **double** MONKFISH_PARAM::interparticle_diffusion
- 4.27.1.8 **double** MONKFISH_PARAM::intraparticle_diffusion
- 4.27.1.9 **double** MONKFISH_PARAM::sorbed_molefraction
- 4.27.1.10 **double** MONKFISH_PARAM::sorption_bc
- 4.27.1.11 **Molecule** MONKFISH_PARAM::species

The documentation for this struct was generated from the following file:

- [monkfish.h](#)

4.28 mSPD_DATA Struct Reference

```
#include <magpie.h>
```

Public Attributes

- **double** [s](#)
- **double** [v](#)
- **double** [eMax](#)
- **std::vector**< **double** > [eta](#)
- **double** [gama](#)

4.28.1 Member Data Documentation

- 4.28.1.1 **double** mSPD_DATA::eMax
- 4.28.1.2 **std::vector**<**double**> mSPD_DATA::eta
- 4.28.1.3 **double** mSPD_DATA::gama
- 4.28.1.4 **double** mSPD_DATA::s
- 4.28.1.5 **double** mSPD_DATA::v

The documentation for this struct was generated from the following file:

- [magpie.h](#)

4.29 NUM_JAC_DATA Struct Reference

Data structure to form a numerical jacobian matrix with finite differences.

```
#include <lark.h>
```

Public Attributes

- double [eps](#) = sqrt(DBL_EPSILON)
Perturbation value.
- [Matrix](#)< double > [Fx](#)
Vector of function evaluations at x.
- [Matrix](#)< double > [Fxp](#)
Vector of function evaluations at x+eps.
- [Matrix](#)< double > [dxj](#)
Vector of perturbed x values.

4.29.1 Detailed Description

Data structure to form a numerical jacobian matrix with finite differences.

C-style object to be used in conjunction with the Numerical Jacobian algorithm. This algorithm will used double-precision finite-differences to formulate an approximate Jacobian matrix at the given variable state for the given residual/non-linear function.

4.29.2 Member Data Documentation

4.29.2.1 [Matrix](#)<double> NUM_JAC_DATA::dxj

Vector of perturbed x values.

4.29.2.2 double NUM_JAC_DATA::eps = sqrt(DBL_EPSILON)

Perturbation value.

4.29.2.3 [Matrix](#)<double> NUM_JAC_DATA::Fx

Vector of function evaluations at x.

4.29.2.4 [Matrix](#)<double> NUM_JAC_DATA::Fxp

Vector of function evaluations at x+eps.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.30 OPTRANS_DATA Struct Reference

Data structure for implementation of linear operator transposition.

```
#include <lark.h>
```

Public Attributes

- [Matrix](#)< double > [li](#)
The ith column vector of the identity operator.
- [Matrix](#)< double > [Ai](#)
The ith column vector of the user's linear operator.

4.30.1 Detailed Description

Data structure for implementation of linear operator transposition.

C-style object used in conjunction with the Operator Transpose algorithm to form an action of $A^T r$ when A is only available as a linear operator and not a matrix. This is a sub-routine required by GCR and GMRESR to stabilize the outer iterations.

4.30.2 Member Data Documentation

4.30.2.1 [Matrix](#)<double> OPTRANS_DATA::Ai

The ith column vector of the user's linear operator.

4.30.2.2 [Matrix](#)<double> OPTRANS_DATA::li

The ith column vector of the identity operator.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.31 PCG_DATA Struct Reference

Data structure for implementation of the PCG algorithms for symmetric linear systems.

```
#include <lark.h>
```

Public Attributes

- int [maxit](#) = 0
Maximum allowable iterations - default = min(vector_size,1000)
- int [iter](#) = 0
Actual number of iterations taken.
- double [alpha](#)
Step size for new solution.
- double [beta](#)
Step size for new search direction.
- double [tol_rel](#) = 1e-6

- *Relative tolerance for convergence - default = 1e-6.*
- double `tol_abs` = 1e-6
- *Absolution tolerance for convergence - default = 1e-6.*
- double `res`
- *Absolute residual norm.*
- double `relres`
- *Relative residual norm.*
- double `relres_base`
- *Initial residual norm.*
- double `bestres`
- *Best found residual norm.*
- bool `Output` = true
- *True = print messages to console.*
- `Matrix< double > x`
- *Current solution to the linear system.*
- `Matrix< double > bestx`
- *Best found solution to the linear system.*
- `Matrix< double > r`
- *Residual vector for the linear system.*
- `Matrix< double > r_old`
- *Previous residual vector.*
- `Matrix< double > z`
- *Preconditioned residual vector (result of precon function)*
- `Matrix< double > z_old`
- *Previous preconditioned residual vector.*
- `Matrix< double > p`
- *Search direction.*
- `Matrix< double > Ap`
- *Result of matrix-vector multiplication.*

4.31.1 Detailed Description

Data structure for implementation of the PCG algorithms for symmetric linear systems.

C-style object used in conjunction with the Preconditioned Conjugate Gradient (PCG) algorithm to iteratively solve a symmetric linear system of equations. This algorithm is optimal if your linear system is symmetric, but will not work at all if your system is asymmetric. For asymmetric systems, use one of the other linear methods.

4.31.2 Member Data Documentation

4.31.2.1 double PCG_DATA::alpha

Step size for new solution.

4.31.2.2 Matrix<double> PCG_DATA::Ap

Result of matrix-vector multiplication.

4.31.2.3 double PCG_DATA::bestres

Best found residual norm.

4.31.2.4 Matrix<double> PCG_DATA::bestx

Best found solution to the linear system.

4.31.2.5 double PCG_DATA::beta

Step size for new search direction.

4.31.2.6 int PCG_DATA::iter = 0

Actual number of iterations taken.

4.31.2.7 int PCG_DATA::maxit = 0

Maximum allowable iterations - default = min(vector_size,1000)

4.31.2.8 bool PCG_DATA::Output = true

True = print messages to console.

4.31.2.9 Matrix<double> PCG_DATA::p

Search direction.

4.31.2.10 Matrix<double> PCG_DATA::r

Residual vector for the linear system.

4.31.2.11 Matrix<double> PCG_DATA::r_old

Previous residual vector.

4.31.2.12 double PCG_DATA::relres

Relative residual norm.

4.31.2.13 double PCG_DATA::relres_base

Initial residual norm.

4.31.2.14 double PCG_DATA::res

Absolute residual norm.

4.31.2.15 double PCG_DATA::tol_abs = 1e-6

Absolution tolerance for convergence - default = 1e-6.

4.31.2.16 `double PCG_DATA::tol_rel = 1e-6`

Relative tolerance for convergence - default = 1e-6.

4.31.2.17 `Matrix<double> PCG_DATA::x`

Current solution to the linear system.

4.31.2.18 `Matrix<double> PCG_DATA::z`

Preconditioned residual vector (result of precon function)

4.31.2.19 `Matrix<double> PCG_DATA::z_old`

Previous preconditioned residual vector.

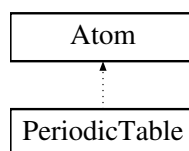
The documentation for this struct was generated from the following file:

- [lark.h](#)

4.32 PeriodicTable Class Reference

```
#include <eel.h>
```

Inheritance diagram for PeriodicTable:



Public Member Functions

- [PeriodicTable](#) ()
- [~PeriodicTable](#) ()
- [PeriodicTable](#) (int *n, int N)
- [PeriodicTable](#) (std::vector< std::string > &[Symbol](#))
- [PeriodicTable](#) (std::vector< int > &n)
- void [DisplayTable](#) ()

Protected Attributes

- std::vector< [Atom](#) > [Table](#)

Private Attributes

- int [number_elements](#)

Additional Inherited Members

4.32.1 Constructor & Destructor Documentation

4.32.1.1 `PeriodicTable::PeriodicTable ()`

4.32.1.2 `PeriodicTable::~~PeriodicTable ()`

4.32.1.3 `PeriodicTable::PeriodicTable (int * n, int N)`

4.32.1.4 `PeriodicTable::PeriodicTable (std::vector< std::string > & Symbol)`

4.32.1.5 `PeriodicTable::PeriodicTable (std::vector< int > & n)`

4.32.2 Member Function Documentation

4.32.2.1 `void PeriodicTable::DisplayTable ()`

4.32.3 Member Data Documentation

4.32.3.1 `int PeriodicTable::number_elements` `[private]`

4.32.3.2 `std::vector<Atom> PeriodicTable::Table` `[protected]`

The documentation for this class was generated from the following files:

- [eel.h](#)
- [eel.cpp](#)

4.33 PICARD_DATA Struct Reference

Data structure for the implementation of a Picard or Fixed-Point iteration for non-linear systems.

```
#include <lark.h>
```

Public Attributes

- `int` [maxit](#) = 0
*Maximum allowable iterations - default = min(3*vec_size,1000)*
- `int` [iter](#) = 0
Actual number of iterations.
- `double` [tol_rel](#) = 1e-6
Relative tolerance for convergence - default = 1e-6.
- `double` [tol_abs](#) = 1e-6
Absolution tolerance for convergence - default = 1e-6.
- `double` [res](#)
Residual norm of the iterate.
- `double` [relres](#)
Relative residual norm of the iterate.
- `double` [relres_base](#)
Initial residual norm.
- `double` [bestres](#)
Best found residual norm.

- `bool Output = true`
True = print messages to console.
- `Matrix< double > x0`
Previous iterate solution vector.
- `Matrix< double > bestx`
Best found solution vector.
- `Matrix< double > r`
Residual of the non-linear system.

4.33.1 Detailed Description

Data structure for the implementation of a Picard or Fixed-Point iteration for non-linear systems.

C-style object used in conjunction with the Picard algorithm for solving a non-linear system of equations. This is an extraordinarily simple iterative method by which a weak or loose form of the non-linear system is solved based on an initial guess. User must supplied a residual function for the non-linear system and a function representing the weak solution. Generally, this method is less efficient than Newton methods, but is significantly cheaper.

4.33.2 Member Data Documentation

4.33.2.1 `double PICARD_DATA::bestres`

Best found residual norm.

4.33.2.2 `Matrix<double> PICARD_DATA::bestx`

Best found solution vector.

4.33.2.3 `int PICARD_DATA::iter = 0`

Actual number of iterations.

4.33.2.4 `int PICARD_DATA::maxit = 0`

Maximum allowable iterations - default = $\min(3 \cdot \text{vec_size}, 1000)$

4.33.2.5 `bool PICARD_DATA::Output = true`

True = print messages to console.

4.33.2.6 `Matrix<double> PICARD_DATA::r`

Residual of the non-linear system.

4.33.2.7 `double PICARD_DATA::relres`

Relative residual norm of the iterate.

4.33.2.8 `double PICARD_DATA::relres_base`

Initial residual norm.

4.33.2.9 double PICARD_DATA::res

Residual norm of the iterate.

4.33.2.10 double PICARD_DATA::tol_abs = 1e-6

Absolution tolerance for convergence - default = 1e-6.

4.33.2.11 double PICARD_DATA::tol_rel = 1e-6

Relative tolerance for convergence - default = 1e-6.

4.33.2.12 Matrix<double> PICARD_DATA::x0

Previous iterate solution vector.

The documentation for this struct was generated from the following file:

- [lark.h](#)

4.34 PJFNK_DATA Struct Reference

Data structure for the implementation of the PJFNK algorithm for non-linear systems.

```
#include <lark.h>
```

Public Attributes

- int [nl_iter](#) = 0
Number of non-linear iterations.
- int [l_iter](#) = 0
Number of linear iterations.
- int [nl_maxit](#) = 0
Maximum allowable non-linear steps.
- int [linear_solver](#) = -1
Flag to denote which linear solver to use - default = PJFNK Chooses.
- double [nl_tol_abs](#) = 1e-6
Absolute Convergence tolerance for non-linear system - default = 1e-6.
- double [nl_tol_rel](#) = 1e-6
Relative Convergence tol for the non-linear system - default = 1e-6.
- double [lin_tol_rel](#) = 1e-6
Relative tolerance of the linear solver - default = 1e-6.
- double [lin_tol_abs](#) = 1e-6
Absolute tolerance of the linear solver - default = 1e-6.
- double [nl_res](#)
Absolute residual norm for the non-linear system.
- double [nl_relres](#)
Relative residual for the non-linear system.
- double [nl_res_base](#)
Initial residual norm for the non-linear system.
- double [nl_bestres](#)

- Best found residual norm.*

 - double `eps` = `sqrt(DBL_EPSILON)`

Value of epsilon used jacvec - default = `sqrt(DBL_EPSILON)`
 - bool `NL_Output` = true

True = print PJFNK messages to console.
 - bool `L_Output` = false

True = print Linear messages to console.
 - bool `LineSearch` = false

True = use Backtracking Linesearch for global convergence.
 - bool `Bounce` = false

True = allow Linesearch to go outside local well, False = Strict local convergence.
 - `Matrix< double > F`

Stored fuction evaluation at x (also the residual)
 - `Matrix< double > Fv`

*Stored function evaluation at $x + \text{eps} * v$.*
 - `Matrix< double > v`

*Stored vector of $x + \text{eps} * v$.*
 - `Matrix< double > x`

Current solution vector for the non-linear system.
 - `Matrix< double > bestx`

Best found solution vector to the non-linear system.
 - `GMRESLP_DATA gmreslp_dat`

Data structure for the GMRESLP method.
 - `PCG_DATA pcg_dat`

Data structure for the PCG method.
 - `BiCGSTAB_DATA bicgstab_dat`

Data structure for the BiCGSTAB method.
 - `CGS_DATA cgs_dat`

Data structure for the CGS method.
 - `GMRESRP_DATA gmresrp_dat`

Data structure for the GMRESRP method.
 - `GCR_DATA gcr_dat`

Data structure for the GCR method.
 - `GMRESR_DATA gmresr_dat`

Data structure for the GMRESR method.
 - `BACKTRACK_DATA backtrack_dat`

Data structure for the Backtracking Linesearch algorithm.
 - const void * `res_data`

Data structure pointer for user's residual data.
 - const void * `precon_data`

Data structure pointer for user's preconditioning data.
 - int(* `funeval`)(const `Matrix< double > &x`, `Matrix< double > &F`, const void *`res_data`)

Function pointer for the user's function $F(x)$ using there data.
 - int(* `precon`)(const `Matrix< double > &r`, `Matrix< double > &p`, const void *`precon_data`)

Function pointer for the user's preconditioning function for the linear system.

4.34.1 Detailed Description

Data structure for the implementation of the PJFNK algorithm for non-linear systems.

C-style object to be used in conjunction with the Preconditioned Jacobian-Free Newton-Krylov (PJFNK) method for solving a non-linear system of equations. You can use any of the Krylov methods listed in the `krylov_method` enum to solve the linear sub-problem. When FOM is specified as the Krylov method, this algorithm becomes equivalent to an exact Newton method. If no Krylov method is specified, then the algorithm will try to pick a method based on the problem size and availability of preconditioning.

4.34.2 Member Data Documentation

4.34.2.1 BACKTRACK_DATA PJFNK_DATA::backtrack_dat

Data structure for the Backtracking Linesearch algorithm.

4.34.2.2 Matrix<double> PJFNK_DATA::bestx

Best found solution vector to the non-linear system.

4.34.2.3 BiCGSTAB_DATA PJFNK_DATA::bicgstab_dat

Data structure for the BiCGSTAB method.

4.34.2.4 bool PJFNK_DATA::Bounce = false

True = allow Linesearch to go outside local well, False = Strict local convergence.

4.34.2.5 CGS_DATA PJFNK_DATA::cgs_dat

Data structure for the CGS method.

4.34.2.6 double PJFNK_DATA::eps =sqrt(DBL_EPSILON)

Value of epsilon used jacvec - default = sqrt(DBL_EPSILON)

4.34.2.7 Matrix<double> PJFNK_DATA::F

Stored function evaluation at x (also the residual)

4.34.2.8 int(* PJFNK_DATA::funeval)(const Matrix< double > &x, Matrix< double > &F, const void *res_data)

Function pointer for the user's function F(x) using there data.

4.34.2.9 Matrix<double> PJFNK_DATA::Fv

Stored function evaluation at $x + \text{eps} * v$.

4.34.2.10 GCR_DATA PJFNK_DATA::gcr_dat

Data structure for the GCR method.

4.34.2.11 `GMRESLP_DATA PJFNK_DATA::gmreslp_dat`

Data structure for the GMRESLP method.

4.34.2.12 `GMRESR_DATA PJFNK_DATA::gmresr_dat`

Data structure for the GMRESR method.

4.34.2.13 `GMRESRP_DATA PJFNK_DATA::gmresrp_dat`

Data structure for the GMRESRP method.

4.34.2.14 `int PJFNK_DATA::l_iter = 0`

Number of linear iterations.

4.34.2.15 `bool PJFNK_DATA::L_Output = false`

True = print Linear messages to console.

4.34.2.16 `double PJFNK_DATA::lin_tol_abs = 1e-6`

Absolute tolerance of the linear solver - default = 1e-6.

4.34.2.17 `double PJFNK_DATA::lin_tol_rel = 1e-6`

Relative tolerance of the linear solver - default = 1e-6.

4.34.2.18 `int PJFNK_DATA::linear_solver = -1`

Flag to denote which linear solver to use - default = PJFNK Chooses.

4.34.2.19 `bool PJFNK_DATA::LineSearch = false`

True = use Backtracking Linesearch for global convergence.

4.34.2.20 `double PJFNK_DATA::nl_bestres`

Best found residual norm.

4.34.2.21 `int PJFNK_DATA::nl_iter = 0`

Number of non-linear iterations.

4.34.2.22 `int PJFNK_DATA::nl_maxit = 0`

Maximum allowable non-linear steps.

4.34.2.23 `bool PJFNK_DATA::NL_Output = true`

True = print PJFNK messages to console.

4.34.2.24 `double PJFNK_DATA::nl_relres`

Relative residual for the non-linear system.

4.34.2.25 `double PJFNK_DATA::nl_res`

Absolute residual norm for the non-linear system.

4.34.2.26 `double PJFNK_DATA::nl_res_base`

Initial residual norm for the non-linear system.

4.34.2.27 `double PJFNK_DATA::nl_tol_abs = 1e-6`

Absolute Convergence tolerance for non-linear system - default = 1e-6.

4.34.2.28 `double PJFNK_DATA::nl_tol_rel = 1e-6`

Relative Convergence tol for the non-linear system - default = 1e-6.

4.34.2.29 `PCG_DATA PJFNK_DATA::pcg_dat`

Data structure for the PCG method.

4.34.2.30 `int(* PJFNK_DATA::precon)(const Matrix< double > &r, Matrix< double > &p, const void *precon_data)`

Function pointer for the user's preconditioning function for the linear system.

4.34.2.31 `const void* PJFNK_DATA::precon_data`

Data structure pointer for user's preconditioning data.

4.34.2.32 `const void* PJFNK_DATA::res_data`

Data structure pointer for user's residual data.

4.34.2.33 `Matrix<double> PJFNK_DATA::v`

Stored vector of $x + \epsilon v$.

4.34.2.34 `Matrix<double> PJFNK_DATA::x`

Current solution vector for the non-linear system.

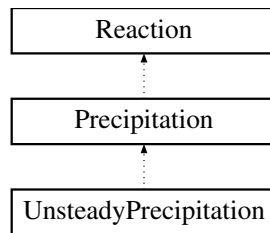
The documentation for this struct was generated from the following file:

- [lark.h](#)

4.35 Precipitation Class Reference

```
#include <shark.h>
```

Inheritance diagram for Precipitation:



Additional Inherited Members

The documentation for this class was generated from the following file:

- [shark.h](#)

4.36 PURE_GAS Struct Reference

```
#include <egret.h>
```

Public Attributes

- double [molecular_weight](#)
- double [Sutherland_Temp](#)
- double [Sutherland_Const](#)
- double [Sutherland_Viscosity](#)
- double [specific_heat](#)
- double [molecular_diffusion](#)
- double [dynamic_viscosity](#)
- double [density](#)
- double [Schmidt](#)

4.36.1 Member Data Documentation

4.36.1.1 double PURE_GAS::density

4.36.1.2 double PURE_GAS::dynamic_viscosity

4.36.1.3 double PURE_GAS::molecular_diffusion

4.36.1.4 double PURE_GAS::molecular_weight

4.36.1.5 double PURE_GAS::Schmidt

4.36.1.6 double PURE_GAS::specific_heat

4.36.1.7 double PURE_GAS::Sutherland_Const

4.36.1.8 double PURE_GAS::Sutherland_Temp

4.36.1.9 double PURE_GAS::Sutherland_Viscosity

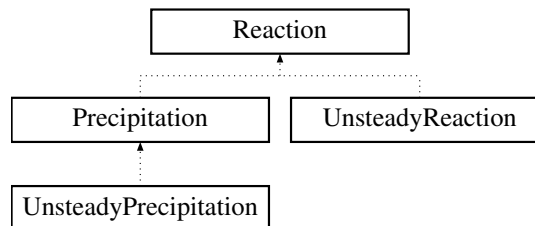
The documentation for this struct was generated from the following file:

- [egret.h](#)

4.37 Reaction Class Reference

```
#include <shark.h>
```

Inheritance diagram for Reaction:



Public Member Functions

- [Reaction](#) ()
- [~Reaction](#) ()
- void [Initialize_List](#) (MasterSpeciesList &List)
- void [Display_Info](#) ()
- void [Set_Stoichiometric](#) (int i, double v)
- void [Set_Equilibrium](#) (double v)
- void [Set_Enthalpy](#) (double H)
- void [Set_Entropy](#) (double S)
- void [Set_EnthalpyANDEntropy](#) (double H, double S)
- void [Set_Energy](#) (double G)
- void [checkSpeciesEnergies](#) ()
- void [calculateEnergies](#) ()
- void [calculateEquilibrium](#) (double T)
- bool [haveEquilibrium](#) ()
- double [Get_Stoichiometric](#) (int i)
- double [Get_Equilibrium](#) ()
- double [Get_Enthalpy](#) ()
- double [Get_Entropy](#) ()
- double [Get_Energy](#) ()
- double [Eval_Residual](#) (const [Matrix](#)< double > &x, const [Matrix](#)< double > &gama)

Protected Attributes

- [MasterSpeciesList](#) * List
- std::vector< double > [Stoichiometric](#)
- double [Equilibrium](#)
- double [enthalpy](#)
- double [entropy](#)

- double [energy](#)
- bool [CanCalcHS](#)
- bool [CanCalcG](#)
- bool [HaveHS](#)
- bool [HaveG](#)
- bool [HaveEquil](#)

4.37.1 Constructor & Destructor Documentation

4.37.1.1 `Reaction::Reaction ()`

4.37.1.2 `Reaction::~~Reaction ()`

4.37.2 Member Function Documentation

4.37.2.1 `void Reaction::calculateEnergies ()`

4.37.2.2 `void Reaction::calculateEquilibrium (double T)`

4.37.2.3 `void Reaction::checkSpeciesEnergies ()`

4.37.2.4 `void Reaction::Display_Info ()`

4.37.2.5 `double Reaction::Eval_Residual (const Matrix< double > & x, const Matrix< double > & gama)`

4.37.2.6 `double Reaction::Get_Energy ()`

4.37.2.7 `double Reaction::Get_Enthalpy ()`

4.37.2.8 `double Reaction::Get_Entropy ()`

4.37.2.9 `double Reaction::Get_Equilibrium ()`

4.37.2.10 `double Reaction::Get_Stoichiometric (int i)`

4.37.2.11 `bool Reaction::haveEquilibrium ()`

4.37.2.12 `void Reaction::Initialize_List (MasterSpeciesList & List)`

4.37.2.13 `void Reaction::Set_Energy (double G)`

4.37.2.14 `void Reaction::Set_Enthalpy (double H)`

4.37.2.15 `void Reaction::Set_EnthalpyANDEntropy (double H, double S)`

4.37.2.16 `void Reaction::Set_Entropy (double S)`

4.37.2.17 `void Reaction::Set_Equilibrium (double v)`

4.37.2.18 `void Reaction::Set_Stoichiometric (int i, double v)`

4.37.3 Member Data Documentation

4.37.3.1 `bool Reaction::CanCalcG` [protected]

- 4.37.3.2 `bool Reaction::CanCalcHS` [protected]
- 4.37.3.3 `double Reaction::energy` [protected]
- 4.37.3.4 `double Reaction::enthalpy` [protected]
- 4.37.3.5 `double Reaction::entropy` [protected]
- 4.37.3.6 `double Reaction::Equilibrium` [protected]
- 4.37.3.7 `bool Reaction::HaveEquil` [protected]
- 4.37.3.8 `bool Reaction::HaveG` [protected]
- 4.37.3.9 `bool Reaction::HaveHS` [protected]
- 4.37.3.10 `MasterSpeciesList* Reaction::List` [protected]
- 4.37.3.11 `std::vector<double> Reaction::Stoichiometric` [protected]

The documentation for this class was generated from the following files:

- [shark.h](#)
- [shark.cpp](#)

4.38 SCOPSOWL_DATA Struct Reference

```
#include <scopsowl.h>
```

Public Attributes

- unsigned long int [total_steps](#)
- int [coord_macro](#)
- int [coord_micro](#)
- int [level](#) = 2
- double [sim_time](#)
- double [t_old](#)
- double [t](#)
- double [t_counter](#) = 0.0
- double [t_print](#)
- bool [Print2File](#) = true
- bool [Print2Console](#) = true
- bool [SurfDiff](#) = true
- bool [Heterogeneous](#) = true
- double [gas_velocity](#)
- double [total_pressure](#)
- double [gas_temperature](#)
- double [pellet_radius](#)
- double [crystal_radius](#)
- double [char_macro](#)
- double [char_micro](#)
- double [binder_fraction](#)
- double [binder_porosity](#)

- double [binder_poresize](#)
- double [pellet_density](#)
- bool [DirichletBC](#) = false
- bool [NonLinear](#) = true
- std::vector< double > [y](#)
- std::vector< double > [tempy](#)
- FILE * [OutputFile](#)
- double(* [eval_ads](#))(int i, int l, const void *[user_data](#))
- double(* [eval_retard](#))(int i, int l, const void *[user_data](#))
- double(* [eval_diff](#))(int i, int l, const void *[user_data](#))
- double(* [eval_surfDiff](#))(int i, int l, const void *[user_data](#))
- double(* [eval_kf](#))(int i, const void *[user_data](#))
- const void * [user_data](#)
- [MIXED_GAS](#) * [gas_dat](#)
- [MAGPIE_DATA](#) [magpie_dat](#)
- std::vector< [FINCH_DATA](#) > [finch_dat](#)
- std::vector< [SCOPSOWL_PARAM_DATA](#) > [param_dat](#)
- std::vector< [SKUA_DATA](#) > [skua_dat](#)

4.38.1 Member Data Documentation

- 4.38.1.1 double [SCOPSOWL_DATA::binder_fraction](#)
- 4.38.1.2 double [SCOPSOWL_DATA::binder_poresize](#)
- 4.38.1.3 double [SCOPSOWL_DATA::binder_porosity](#)
- 4.38.1.4 double [SCOPSOWL_DATA::char_macro](#)
- 4.38.1.5 double [SCOPSOWL_DATA::char_micro](#)
- 4.38.1.6 int [SCOPSOWL_DATA::coord_macro](#)
- 4.38.1.7 int [SCOPSOWL_DATA::coord_micro](#)
- 4.38.1.8 double [SCOPSOWL_DATA::crystal_radius](#)
- 4.38.1.9 bool [SCOPSOWL_DATA::DirichletBC](#) = false
- 4.38.1.10 double(* [SCOPSOWL_DATA::eval_ads](#))(int i, int l, const void *[user_data](#))
- 4.38.1.11 double(* [SCOPSOWL_DATA::eval_diff](#))(int i, int l, const void *[user_data](#))
- 4.38.1.12 double(* [SCOPSOWL_DATA::eval_kf](#))(int i, const void *[user_data](#))
- 4.38.1.13 double(* [SCOPSOWL_DATA::eval_retard](#))(int i, int l, const void *[user_data](#))
- 4.38.1.14 double(* [SCOPSOWL_DATA::eval_surfDiff](#))(int i, int l, const void *[user_data](#))
- 4.38.1.15 std::vector< [FINCH_DATA](#) > [SCOPSOWL_DATA::finch_dat](#)
- 4.38.1.16 [MIXED_GAS](#)* [SCOPSOWL_DATA::gas_dat](#)
- 4.38.1.17 double [SCOPSOWL_DATA::gas_temperature](#)

- 4.38.1.18 double SCOPSOWL_DATA::gas_velocity
- 4.38.1.19 bool SCOPSOWL_DATA::Heterogeneous = true
- 4.38.1.20 int SCOPSOWL_DATA::level = 2
- 4.38.1.21 **MAGPIE_DATA** SCOPSOWL_DATA::magpie_dat
- 4.38.1.22 bool SCOPSOWL_DATA::NonLinear = true
- 4.38.1.23 FILE* SCOPSOWL_DATA::OutputFile
- 4.38.1.24 std::vector<SCOPSOWL_PARAM_DATA> SCOPSOWL_DATA::param_dat
- 4.38.1.25 double SCOPSOWL_DATA::pellet_density
- 4.38.1.26 double SCOPSOWL_DATA::pellet_radius
- 4.38.1.27 bool SCOPSOWL_DATA::Print2Console = true
- 4.38.1.28 bool SCOPSOWL_DATA::Print2File = true
- 4.38.1.29 double SCOPSOWL_DATA::sim_time
- 4.38.1.30 std::vector<SKUA_DATA> SCOPSOWL_DATA::skua_dat
- 4.38.1.31 bool SCOPSOWL_DATA::SurfDiff = true
- 4.38.1.32 double SCOPSOWL_DATA::t
- 4.38.1.33 double SCOPSOWL_DATA::t_counter = 0.0
- 4.38.1.34 double SCOPSOWL_DATA::t_old
- 4.38.1.35 double SCOPSOWL_DATA::t_print
- 4.38.1.36 std::vector<double> SCOPSOWL_DATA::tempy
- 4.38.1.37 double SCOPSOWL_DATA::total_pressure
- 4.38.1.38 unsigned long int SCOPSOWL_DATA::total_steps
- 4.38.1.39 const void* SCOPSOWL_DATA::user_data
- 4.38.1.40 std::vector<double> SCOPSOWL_DATA::y

The documentation for this struct was generated from the following file:

- [scopsowl.h](#)

4.39 SCOPSOWL_OPT_DATA Struct Reference

```
#include <scopsowl_opt.h>
```

Public Attributes

- int [num_curves](#)
- int [evaluation](#)
- unsigned long int [total_eval](#)
- int [current_points](#)
- int [num_params](#) = 1
- int [diffusion_type](#)
- int [adsorb_index](#)
- int [max_guess_iter](#) = 20
- bool [Optimize](#)
- bool [Rough](#)
- double [current_temp](#)
- double [current_press](#)
- double [current_equil](#)
- double [simulation_equil](#)
- double [max_bias](#)
- double [min_bias](#)
- double [e_norm](#)
- double [f_bias](#)
- double [e_norm_old](#)
- double [f_bias_old](#)
- double [param_guess](#)
- double [param_guess_old](#)
- double [rel_tol_norm](#) = 0.01
- double [abs_tol_bias](#) = 1.0
- std::vector< double > [y_base](#)
- std::vector< double > [q_data](#)
- std::vector< double > [q_sim](#)
- std::vector< double > [t](#)
- FILE * [ParamFile](#)
- FILE * [CompareFile](#)
- SCOPSOWL_DATA owl_dat

4.39.1 Member Data Documentation

4.39.1.1 double SCOPSOWL_OPT_DATA::abs_tol_bias = 1.0

4.39.1.2 int SCOPSOWL_OPT_DATA::adsorb_index

4.39.1.3 FILE* SCOPSOWL_OPT_DATA::CompareFile

4.39.1.4 double SCOPSOWL_OPT_DATA::current_equil

4.39.1.5 int SCOPSOWL_OPT_DATA::current_points

4.39.1.6 double SCOPSOWL_OPT_DATA::current_press

4.39.1.7 double SCOPSOWL_OPT_DATA::current_temp

4.39.1.8 int SCOPSOWL_OPT_DATA::diffusion_type

4.39.1.9 double SCOPSOWL_OPT_DATA::e_norm

- 4.39.1.10 double SCOPSOWL_OPT_DATA::e_norm_old
- 4.39.1.11 int SCOPSOWL_OPT_DATA::evaluation
- 4.39.1.12 double SCOPSOWL_OPT_DATA::f_bias
- 4.39.1.13 double SCOPSOWL_OPT_DATA::f_bias_old
- 4.39.1.14 double SCOPSOWL_OPT_DATA::max_bias
- 4.39.1.15 int SCOPSOWL_OPT_DATA::max_guess_iter = 20
- 4.39.1.16 double SCOPSOWL_OPT_DATA::min_bias
- 4.39.1.17 int SCOPSOWL_OPT_DATA::num_curves
- 4.39.1.18 int SCOPSOWL_OPT_DATA::num_params = 1
- 4.39.1.19 bool SCOPSOWL_OPT_DATA::Optimize
- 4.39.1.20 SCOPSOWL_DATA SCOPSOWL_OPT_DATA::owl_dat
- 4.39.1.21 double SCOPSOWL_OPT_DATA::param_guess
- 4.39.1.22 double SCOPSOWL_OPT_DATA::param_guess_old
- 4.39.1.23 FILE* SCOPSOWL_OPT_DATA::ParamFile
- 4.39.1.24 std::vector<double> SCOPSOWL_OPT_DATA::q_data
- 4.39.1.25 std::vector<double> SCOPSOWL_OPT_DATA::q_sim
- 4.39.1.26 double SCOPSOWL_OPT_DATA::rel_tol_norm = 0.01
- 4.39.1.27 bool SCOPSOWL_OPT_DATA::Rough
- 4.39.1.28 double SCOPSOWL_OPT_DATA::simulation_equil
- 4.39.1.29 std::vector<double> SCOPSOWL_OPT_DATA::t
- 4.39.1.30 unsigned long int SCOPSOWL_OPT_DATA::total_eval
- 4.39.1.31 std::vector<double> SCOPSOWL_OPT_DATA::y_base

The documentation for this struct was generated from the following file:

- [scopsowl_opt.h](#)

4.40 SCOPSOWL_PARAM_DATA Struct Reference

```
#include <scopsowl.h>
```

Public Attributes

- [Matrix](#)< double > [qAvg](#)

- [Matrix< double > qAvg_old](#)
- [Matrix< double > Qst](#)
- [Matrix< double > Qst_old](#)
- [Matrix< double > dq_dc](#)
- [double xIC](#)
- [double qIntegralAvg](#)
- [double qIntegralAvg_old](#)
- [double QstAvg](#)
- [double QstAvg_old](#)
- [double qo](#)
- [double Qsto](#)
- [double dq_dco](#)
- [double pore_diffusion](#)
- [double film_transfer](#)
- [double activation_energy](#)
- [double ref_diffusion](#)
- [double ref_temperature](#)
- [double affinity](#)
- [double ref_pressure](#)
- [bool Adsorbable](#)
- [std::string speciesName](#)

4.40.1 Member Data Documentation

- 4.40.1.1 [double SCOPSOWL_PARAM_DATA::activation_energy](#)
- 4.40.1.2 [bool SCOPSOWL_PARAM_DATA::Adsorbable](#)
- 4.40.1.3 [double SCOPSOWL_PARAM_DATA::affinity](#)
- 4.40.1.4 [Matrix<double> SCOPSOWL_PARAM_DATA::dq_dc](#)
- 4.40.1.5 [double SCOPSOWL_PARAM_DATA::dq_dco](#)
- 4.40.1.6 [double SCOPSOWL_PARAM_DATA::film_transfer](#)
- 4.40.1.7 [double SCOPSOWL_PARAM_DATA::pore_diffusion](#)
- 4.40.1.8 [Matrix<double> SCOPSOWL_PARAM_DATA::qAvg](#)
- 4.40.1.9 [Matrix<double> SCOPSOWL_PARAM_DATA::qAvg_old](#)
- 4.40.1.10 [double SCOPSOWL_PARAM_DATA::qIntegralAvg](#)
- 4.40.1.11 [double SCOPSOWL_PARAM_DATA::qIntegralAvg_old](#)
- 4.40.1.12 [double SCOPSOWL_PARAM_DATA::qo](#)
- 4.40.1.13 [Matrix<double> SCOPSOWL_PARAM_DATA::Qst](#)
- 4.40.1.14 [Matrix<double> SCOPSOWL_PARAM_DATA::Qst_old](#)
- 4.40.1.15 [double SCOPSOWL_PARAM_DATA::QstAvg](#)

- 4.40.1.16 double SCOPSOWL_PARAM_DATA::QstAvg_old
- 4.40.1.17 double SCOPSOWL_PARAM_DATA::Qsto
- 4.40.1.18 double SCOPSOWL_PARAM_DATA::ref_diffusion
- 4.40.1.19 double SCOPSOWL_PARAM_DATA::ref_pressure
- 4.40.1.20 double SCOPSOWL_PARAM_DATA::ref_temperature
- 4.40.1.21 std::string SCOPSOWL_PARAM_DATA::speciesName
- 4.40.1.22 double SCOPSOWL_PARAM_DATA::xIC

The documentation for this struct was generated from the following file:

- [scopsowl.h](#)

4.41 SHARK_DATA Struct Reference

```
#include <shark.h>
```

Public Attributes

- [MasterSpeciesList](#) MasterList
- std::vector< [Reaction](#) > ReactionList
- std::vector< [MassBalance](#) > MassBalanceList
- std::vector< [UnsteadyReaction](#) > UnsteadyList
- std::vector< double(*) (const [Matrix](#)< double > &x, [SHARK_DATA](#) *shark_dat, const void *data) > OtherList
- int numvar
- int num_ssr
- int num_mbe
- int num_usr
- int num_other = 0
- int act_fun = IDEAL
- int totalsteps = 0
- int timesteps = 0
- int pH_index = -1
- int pOH_index = -1
- double simulationtime = 0.0
- double dt = 0.1
- double dt_min = sqrt(DBL_EPSILON)
- double t_out = 0.0
- double t_count = 0.0
- double time = 0.0
- double time_old = 0.0
- double pH = 7.0
- double Norm = 0.0
- double dielectric_const = 78.325
- double temperature = 298.15

- bool [steadystate](#) = true
- bool [TimeAdaptivity](#) = false
- bool [const_pH](#) = false
- bool [SpeciationCurve](#) = false
- bool [Console_Output](#) = true
- bool [File_Output](#) = false
- bool [Contains_pH](#) = false
- bool [Contains_pOH](#) = false
- bool [Converged](#) = false
- [Matrix](#)< double > [X_old](#)
- [Matrix](#)< double > [X_new](#)
- [Matrix](#)< double > [Conc_old](#)
- [Matrix](#)< double > [Conc_new](#)
- [Matrix](#)< double > [activity_new](#)
- [Matrix](#)< double > [activity_old](#)
- int(* [EvalActivity](#))(const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *data)
- int(* [Residual](#))(const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *data)
- int(* [lin_precon](#))(const [Matrix](#)< double > &r, [Matrix](#)< double > &p, const void *data)
- [PJFNK_DATA](#) [Newton_data](#)
- const void * [activity_data](#)
- const void * [residual_data](#)
- const void * [precon_data](#)
- const void * [other_data](#)
- FILE * [OutputFile](#)
- [yaml_cpp_class](#) [yaml_object](#)

4.41.1 Member Data Documentation

4.41.1.1 int SHARK_DATA::act_fun = IDEAL

4.41.1.2 const void* SHARK_DATA::activity_data

4.41.1.3 [Matrix](#)<double> SHARK_DATA::activity_new

4.41.1.4 [Matrix](#)<double> SHARK_DATA::activity_old

4.41.1.5 [Matrix](#)<double> SHARK_DATA::Conc_new

4.41.1.6 [Matrix](#)<double> SHARK_DATA::Conc_old

4.41.1.7 bool SHARK_DATA::Console_Output = true

4.41.1.8 bool SHARK_DATA::const_pH = false

4.41.1.9 bool SHARK_DATA::Contains_pH = false

4.41.1.10 bool SHARK_DATA::Contains_pOH = false

4.41.1.11 bool SHARK_DATA::Converged = false

4.41.1.12 double SHARK_DATA::dielectric_const = 78.325

4.41.1.13 double SHARK_DATA::dt = 0.1

- 4.41.1.14 double SHARK_DATA::dt_min = sqrt(DBL_EPSILON)
- 4.41.1.15 int(* SHARK_DATA::EvalActivity)(const Matrix< double > &x, Matrix< double > &F, const void *data)
- 4.41.1.16 bool SHARK_DATA::File_Output = false
- 4.41.1.17 int(* SHARK_DATA::lin_precon)(const Matrix< double > &r, Matrix< double > &p, const void *data)
- 4.41.1.18 std::vector<MassBalance> SHARK_DATA::MassBalanceList
- 4.41.1.19 MasterSpeciesList SHARK_DATA::MasterList
- 4.41.1.20 PJFNK_DATA SHARK_DATA::Newton_data
- 4.41.1.21 double SHARK_DATA::Norm = 0.0
- 4.41.1.22 int SHARK_DATA::num_mbe
- 4.41.1.23 int SHARK_DATA::num_other = 0
- 4.41.1.24 int SHARK_DATA::num_ssr
- 4.41.1.25 int SHARK_DATA::num_usr
- 4.41.1.26 int SHARK_DATA::numvar
- 4.41.1.27 const void* SHARK_DATA::other_data
- 4.41.1.28 std::vector< double (*) (const Matrix<double> &x, SHARK_DATA *shark_dat, const void *data) > SHARK_DATA::OtherList
- 4.41.1.29 FILE* SHARK_DATA::OutputFile
- 4.41.1.30 double SHARK_DATA::pH = 7.0
- 4.41.1.31 int SHARK_DATA::pH_index = -1
- 4.41.1.32 int SHARK_DATA::pOH_index = -1
- 4.41.1.33 const void* SHARK_DATA::precon_data
- 4.41.1.34 std::vector<Reaction> SHARK_DATA::ReactionList
- 4.41.1.35 int(* SHARK_DATA::Residual)(const Matrix< double > &x, Matrix< double > &F, const void *data)
- 4.41.1.36 const void* SHARK_DATA::residual_data
- 4.41.1.37 double SHARK_DATA::simulationtime = 0.0
- 4.41.1.38 bool SHARK_DATA::SpeciationCurve = false
- 4.41.1.39 bool SHARK_DATA::steadystate = true
- 4.41.1.40 double SHARK_DATA::t_count = 0.0
- 4.41.1.41 double SHARK_DATA::t_out = 0.0

- 4.41.1.42 double SHARK_DATA::temperature = 298.15
- 4.41.1.43 double SHARK_DATA::time = 0.0
- 4.41.1.44 double SHARK_DATA::time_old = 0.0
- 4.41.1.45 bool SHARK_DATA::TimeAdaptivity = false
- 4.41.1.46 int SHARK_DATA::timesteps = 0
- 4.41.1.47 int SHARK_DATA::totalsteps = 0
- 4.41.1.48 std::vector<UnsteadyReaction> SHARK_DATA::UnsteadyList
- 4.41.1.49 Matrix<double> SHARK_DATA::X_new
- 4.41.1.50 Matrix<double> SHARK_DATA::X_old
- 4.41.1.51 yaml_cpp_class SHARK_DATA::yaml_object

The documentation for this struct was generated from the following file:

- [shark.h](#)

4.42 SKUA_DATA Struct Reference

```
#include <skua.h>
```

Public Attributes

- unsigned long int [total_steps](#)
- int [coord](#)
- double [sim_time](#)
- double [t_old](#)
- double [t](#)
- double [t_counter](#) = 0.0
- double [t_print](#)
- double [qTn](#)
- double [qTnp1](#)
- bool [Print2File](#) = true
- bool [Print2Console](#) = true
- double [gas_velocity](#)
- double [pellet_radius](#)
- double [char_measure](#)
- bool [DirichletBC](#) = true
- bool [NonLinear](#) = true
- std::vector< double > [y](#)
- FILE * [OutputFile](#)
- double(* [eval_diff](#))(int i, int l, const void *[user_data](#))
- double(* [eval_kf](#))(int i, const void *[user_data](#))
- const void * [user_data](#)
- [MAGPIE_DATA](#) [magpie_dat](#)
- [MIXED_GAS](#) * [gas_dat](#)
- std::vector< [FINCH_DATA](#) > [finch_dat](#)
- std::vector< [SKUA_PARAM](#) > [param_dat](#)

4.42.1 Member Data Documentation

- 4.42.1.1 double SKUA_DATA::char_measure
- 4.42.1.2 int SKUA_DATA::coord
- 4.42.1.3 bool SKUA_DATA::DirichletBC = true
- 4.42.1.4 double(* SKUA_DATA::eval_diff)(int i, int l, const void *user_data)
- 4.42.1.5 double(* SKUA_DATA::eval_kf)(int i, const void *user_data)
- 4.42.1.6 std::vector<FINCH_DATA> SKUA_DATA::finch_dat
- 4.42.1.7 MIXED_GAS* SKUA_DATA::gas_dat
- 4.42.1.8 double SKUA_DATA::gas_velocity
- 4.42.1.9 MAGPIE_DATA SKUA_DATA::magpie_dat
- 4.42.1.10 bool SKUA_DATA::NonLinear = true
- 4.42.1.11 FILE* SKUA_DATA::OutputFile
- 4.42.1.12 std::vector<SKUA_PARAM> SKUA_DATA::param_dat
- 4.42.1.13 double SKUA_DATA::pellet_radius
- 4.42.1.14 bool SKUA_DATA::Print2Console = true
- 4.42.1.15 bool SKUA_DATA::Print2File = true
- 4.42.1.16 double SKUA_DATA::qTn
- 4.42.1.17 double SKUA_DATA::qTnp1
- 4.42.1.18 double SKUA_DATA::sim_time
- 4.42.1.19 double SKUA_DATA::t
- 4.42.1.20 double SKUA_DATA::t_counter = 0.0
- 4.42.1.21 double SKUA_DATA::t_old
- 4.42.1.22 double SKUA_DATA::t_print
- 4.42.1.23 unsigned long int SKUA_DATA::total_steps
- 4.42.1.24 const void* SKUA_DATA::user_data
- 4.42.1.25 std::vector<double> SKUA_DATA::y

The documentation for this struct was generated from the following file:

- [skua.h](#)

4.43 SKUA_OPT_DATA Struct Reference

```
#include <skua_opt.h>
```

Public Attributes

- int [num_curves](#)
- int [evaluation](#)
- unsigned long int [total_eval](#)
- int [current_points](#)
- int [num_params](#) = 1
- int [diffusion_type](#)
- int [adsorb_index](#)
- int [max_guess_iter](#) = 20
- bool [Optimize](#)
- bool [Rough](#)
- double [current_temp](#)
- double [current_press](#)
- double [current_equil](#)
- double [simulation_equil](#)
- double [max_bias](#)
- double [min_bias](#)
- double [e_norm](#)
- double [f_bias](#)
- double [e_norm_old](#)
- double [f_bias_old](#)
- double [param_guess](#)
- double [param_guess_old](#)
- double [rel_tol_norm](#) = 0.1
- double [abs_tol_bias](#) = 0.1
- std::vector< double > [y_base](#)
- std::vector< double > [q_data](#)
- std::vector< double > [q_sim](#)
- std::vector< double > [t](#)
- FILE * [ParamFile](#)
- FILE * [CompareFile](#)
- [SKUA_DATA](#) [skua_dat](#)

4.43.1 Member Data Documentation

4.43.1.1 double SKUA_OPT_DATA::abs_tol_bias = 0.1

4.43.1.2 int SKUA_OPT_DATA::adsorb_index

4.43.1.3 FILE* SKUA_OPT_DATA::CompareFile

4.43.1.4 double SKUA_OPT_DATA::current_equil

4.43.1.5 int SKUA_OPT_DATA::current_points

4.43.1.6 double SKUA_OPT_DATA::current_press

4.43.1.7 double SKUA_OPT_DATA::current_temp

- 4.43.1.8 int SKUA_OPT_DATA::diffusion_type
- 4.43.1.9 double SKUA_OPT_DATA::e_norm
- 4.43.1.10 double SKUA_OPT_DATA::e_norm_old
- 4.43.1.11 int SKUA_OPT_DATA::evaluation
- 4.43.1.12 double SKUA_OPT_DATA::f_bias
- 4.43.1.13 double SKUA_OPT_DATA::f_bias_old
- 4.43.1.14 double SKUA_OPT_DATA::max_bias
- 4.43.1.15 int SKUA_OPT_DATA::max_guess_iter = 20
- 4.43.1.16 double SKUA_OPT_DATA::min_bias
- 4.43.1.17 int SKUA_OPT_DATA::num_curves
- 4.43.1.18 int SKUA_OPT_DATA::num_params = 1
- 4.43.1.19 bool SKUA_OPT_DATA::Optimize
- 4.43.1.20 double SKUA_OPT_DATA::param_guess
- 4.43.1.21 double SKUA_OPT_DATA::param_guess_old
- 4.43.1.22 FILE* SKUA_OPT_DATA::ParamFile
- 4.43.1.23 std::vector<double> SKUA_OPT_DATA::q_data
- 4.43.1.24 std::vector<double> SKUA_OPT_DATA::q_sim
- 4.43.1.25 double SKUA_OPT_DATA::rel_tol_norm = 0.1
- 4.43.1.26 bool SKUA_OPT_DATA::Rough
- 4.43.1.27 double SKUA_OPT_DATA::simulation_equil
- 4.43.1.28 SKUA_DATA SKUA_OPT_DATA::skua_dat
- 4.43.1.29 std::vector<double> SKUA_OPT_DATA::t
- 4.43.1.30 unsigned long int SKUA_OPT_DATA::total_eval
- 4.43.1.31 std::vector<double> SKUA_OPT_DATA::y_base

The documentation for this struct was generated from the following file:

- [skua_opt.h](#)

4.44 SKUA_PARAM Struct Reference

```
#include <skua.h>
```

Public Attributes

- double [activation_energy](#)
- double [ref_diffusion](#)
- double [ref_temperature](#)
- double [affinity](#)
- double [ref_pressure](#)
- double [film_transfer](#)
- double [xIC](#)
- double [y_eff](#)
- double [Qstn](#)
- double [Qstnp1](#)
- double [xn](#)
- double [xnp1](#)
- bool [Adsorbable](#)
- std::string [speciesName](#)

4.44.1 Member Data Documentation

4.44.1.1 double SKUA_PARAM::activation_energy

4.44.1.2 bool SKUA_PARAM::Adsorbable

4.44.1.3 double SKUA_PARAM::affinity

4.44.1.4 double SKUA_PARAM::film_transfer

4.44.1.5 double SKUA_PARAM::Qstn

4.44.1.6 double SKUA_PARAM::Qstnp1

4.44.1.7 double SKUA_PARAM::ref_diffusion

4.44.1.8 double SKUA_PARAM::ref_pressure

4.44.1.9 double SKUA_PARAM::ref_temperature

4.44.1.10 std::string SKUA_PARAM::speciesName

4.44.1.11 double SKUA_PARAM::xIC

4.44.1.12 double SKUA_PARAM::xn

4.44.1.13 double SKUA_PARAM::xnp1

4.44.1.14 double SKUA_PARAM::y_eff

The documentation for this struct was generated from the following file:

- [skua.h](#)

4.45 Speciation_Test01_Data Struct Reference

```
#include <sandbox.h>
```

Public Attributes

- int `N` = 4
- const double `logKw` = -14.0
- const double `logKa1` = -6.35
- const double `logKa2` = -10.33
- double `CT` = 0.1786
- double `NaT` = 0.1786
- std::vector< `Molecule` > `x`
- `Matrix`< double > `Jacobian`
- `Matrix`< double > `NumJac`
- `Matrix`< double > `logC`
- `Matrix`< double > `C`

4.45.1 Member Data Documentation

4.45.1.1 `Matrix`<double> `Speciation_Test01_Data::C`

4.45.1.2 double `Speciation_Test01_Data::CT` = 0.1786

4.45.1.3 `Matrix`<double> `Speciation_Test01_Data::Jacobian`

4.45.1.4 `Matrix`<double> `Speciation_Test01_Data::logC`

4.45.1.5 const double `Speciation_Test01_Data::logKa1` = -6.35

4.45.1.6 const double `Speciation_Test01_Data::logKa2` = -10.33

4.45.1.7 const double `Speciation_Test01_Data::logKw` = -14.0

4.45.1.8 int `Speciation_Test01_Data::N` = 4

4.45.1.9 double `Speciation_Test01_Data::NaT` = 0.1786

4.45.1.10 `Matrix`<double> `Speciation_Test01_Data::NumJac`

4.45.1.11 std::vector<`Molecule`> `Speciation_Test01_Data::x`

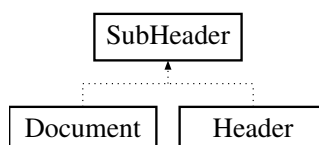
The documentation for this struct was generated from the following file:

- [sandbox.h](#)

4.46 SubHeader Class Reference

```
#include <yaml_wrapper.h>
```

Inheritance diagram for SubHeader:



Public Member Functions

- [SubHeader](#) ()
- [~SubHeader](#) ()
- [SubHeader](#) (const [SubHeader](#) &subheader)
- [SubHeader](#) (const [KeyValueType](#) &map)
- [SubHeader](#) (std::string [name](#))
- [SubHeader](#) (std::string [name](#), const [KeyValueType](#) &map)
- [SubHeader](#) & [operator=](#) (const [SubHeader](#) &sub)
- [ValueTypePair](#) & [operator\[\]](#) (const std::string key)
- [ValueTypePair](#) [operator\[\]](#) (const std::string key) const
- [KeyValueType](#) & [getMap](#) ()
- void [clear](#) ()
- void [addPair](#) (std::string key, std::string val)
- void [addPair](#) (std::string key, std::string val, int type)
- void [setName](#) (std::string [name](#))
- void [setAlias](#) (std::string [alias](#))
- void [setAlias](#) (std::string [alias](#), int [state](#))
- void [setNameAliasPair](#) (std::string [name](#), std::string [alias](#), int [state](#))
- void [setState](#) (int [state](#))
- void [DisplayContents](#) ()
- std::string [getName](#) ()
- std::string [getAlias](#) ()
- bool [isAlias](#) ()
- bool [isAnchor](#) ()
- int [getState](#) ()

Protected Attributes

- [KeyValueType](#) [Data_Map](#)
- std::string [name](#)
- std::string [alias](#)
- int [state](#)

4.46.1 Constructor & Destructor Documentation

4.46.1.1 [SubHeader::SubHeader](#) ()

4.46.1.2 [SubHeader::~SubHeader](#) ()

4.46.1.3 [SubHeader::SubHeader](#) (const [SubHeader](#) & *subheader*)

4.46.1.4 [SubHeader::SubHeader](#) (const [KeyValueType](#) & *map*)

4.46.1.5 [SubHeader::SubHeader](#) (std::string *name*)

4.46.1.6 [SubHeader::SubHeader](#) (std::string *name*, const [KeyValueType](#) & *map*)

4.46.2 Member Function Documentation

4.46.2.1 void [SubHeader::addPair](#) (std::string *key*, std::string *val*)

4.46.2.2 void [SubHeader::addPair](#) (std::string *key*, std::string *val*, int *type*)

- 4.46.2.3 void SubHeader::clear ()
- 4.46.2.4 void SubHeader::DisplayContents ()
- 4.46.2.5 std::string SubHeader::getAlias ()
- 4.46.2.6 KeyValueTypeMap & SubHeader::getMap ()
- 4.46.2.7 std::string SubHeader::getName ()
- 4.46.2.8 int SubHeader::getState ()
- 4.46.2.9 bool SubHeader::isAlias ()
- 4.46.2.10 bool SubHeader::isAnchor ()
- 4.46.2.11 SubHeader & SubHeader::operator= (const SubHeader & *sub*)
- 4.46.2.12 ValuePair & SubHeader::operator[] (const std::string *key*)
- 4.46.2.13 ValuePair SubHeader::operator[] (const std::string *key*) const
- 4.46.2.14 void SubHeader::setAlias (std::string *alias*)
- 4.46.2.15 void SubHeader::setAlias (std::string *alias*, int *state*)
- 4.46.2.16 void SubHeader::setName (std::string *name*)
- 4.46.2.17 void SubHeader::setNameAliasPair (std::string *name*, std::string *alias*, int *state*)
- 4.46.2.18 void SubHeader::setState (int *state*)

4.46.3 Member Data Documentation

- 4.46.3.1 std::string SubHeader::alias [protected]
- 4.46.3.2 KeyValueTypeMap SubHeader::Data_Map [protected]
- 4.46.3.3 std::string SubHeader::name [protected]
- 4.46.3.4 int SubHeader::state [protected]

The documentation for this class was generated from the following files:

- [yaml_wrapper.h](#)
- [yaml_wrapper.cpp](#)

4.47 SYSTEM_DATA Struct Reference

```
#include <magpie.h>
```

Public Attributes

- double [T](#)

- double [PT](#)
- double [qT](#)
- double [PI](#)
- double [pi](#)
- double [As](#)
- int [N](#)
- int [I](#)
- int [J](#)
- int [K](#)
- unsigned long int [total_eval](#)
- double [avg_norm](#)
- double [max_norm](#)
- int [Sys](#)
- int [Par](#)
- bool [Recover](#)
- bool [Carrier](#)
- bool [Ideal](#)
- bool [Output](#)

4.47.1 Member Data Documentation

4.47.1.1 double `SYSTEM_DATA::As`

4.47.1.2 double `SYSTEM_DATA::avg_norm`

4.47.1.3 bool `SYSTEM_DATA::Carrier`

4.47.1.4 int `SYSTEM_DATA::I`

4.47.1.5 bool `SYSTEM_DATA::Ideal`

4.47.1.6 int `SYSTEM_DATA::J`

4.47.1.7 int `SYSTEM_DATA::K`

4.47.1.8 double `SYSTEM_DATA::max_norm`

4.47.1.9 int `SYSTEM_DATA::N`

4.47.1.10 bool `SYSTEM_DATA::Output`

4.47.1.11 int `SYSTEM_DATA::Par`

4.47.1.12 double `SYSTEM_DATA::PI`

4.47.1.13 double `SYSTEM_DATA::pi`

4.47.1.14 double `SYSTEM_DATA::PT`

4.47.1.15 double `SYSTEM_DATA::qT`

4.47.1.16 bool `SYSTEM_DATA::Recover`

4.47.1.17 int `SYSTEM_DATA::Sys`

4.47.1.18 double SYSTEM_DATA::T

4.47.1.19 unsigned long int SYSTEM_DATA::total_eval

The documentation for this struct was generated from the following file:

- [magpie.h](#)

4.48 TRAJECTORY_DATA Struct Reference

```
#include <Trajectory.h>
```

Public Attributes

- double [mu_0](#) = 12.57e-7
- double [rho_f](#) = 1000.0
- double [eta](#) = 0.001
- double [Hamaker](#) = 1.3e-21
- double [Temp](#) = 298
- double [k](#) = 1.38e-23
- double [Rs](#) = 0.0026925
- double [L](#) = 0.0611
- double [porosity](#) = 0.8979
- double [V_separator](#)
- double [a](#) = 33.0e-6
- double [V_wire](#)
- double [L_wire](#)
- double [A_separator](#)
- double [A_wire](#)
- double [B0](#) = 1.0
- double [H0](#)
- double [Ms](#) = 0.6
- double [b](#) = 0.25e-6
- double [chi_p](#) = 3.87e-6
- double [rho_p](#) = 8700.0
- double [Q_in](#)
- double [V0](#)
- double [Y_initial](#) = 20.0
- double [dt](#)
- double [M](#)
- double [mp](#)
- double [beta](#)
- double [q_bar](#)
- double [sigma_v](#)
- double [sigma_vz](#)
- double [sigma_z](#)
- double [sigma_n](#)
- double [sigma_m](#)
- double [n_rand](#)
- double [m_rand](#)
- double [s_rand](#)
- double [t_rand](#)
- [Matrix](#)< double > [POL](#)

- [Matrix< double > H](#)
- [Matrix< double > dX](#)
- [Matrix< double > dY](#)
- [Matrix< double > X](#)
- [Matrix< double > Y](#)
- [Matrix< int > Cap](#)

4.48.1 Member Data Documentation

4.48.1.1 double TRAJECTORY_DATA::a = 33.0e-6

4.48.1.2 double TRAJECTORY_DATA::A_separator

4.48.1.3 double TRAJECTORY_DATA::A_wire

4.48.1.4 double TRAJECTORY_DATA::b = 0.25e-6

4.48.1.5 double TRAJECTORY_DATA::B0 = 1.0

4.48.1.6 double TRAJECTORY_DATA::beta

4.48.1.7 [Matrix<int>](#) TRAJECTORY_DATA::Cap

4.48.1.8 double TRAJECTORY_DATA::chi_p = 3.87e-6

4.48.1.9 double TRAJECTORY_DATA::dt

4.48.1.10 [Matrix<double>](#) TRAJECTORY_DATA::dX

4.48.1.11 [Matrix<double>](#) TRAJECTORY_DATA::dY

4.48.1.12 double TRAJECTORY_DATA::eta = 0.001

4.48.1.13 [Matrix<double>](#) TRAJECTORY_DATA::H

4.48.1.14 double TRAJECTORY_DATA::H0

4.48.1.15 double TRAJECTORY_DATA::Hamaker = 1.3e-21

4.48.1.16 double TRAJECTORY_DATA::k = 1.38e-23

4.48.1.17 double TRAJECTORY_DATA::L = 0.0611

4.48.1.18 double TRAJECTORY_DATA::L_wire

4.48.1.19 double TRAJECTORY_DATA::M

4.48.1.20 double TRAJECTORY_DATA::m_rand

4.48.1.21 double TRAJECTORY_DATA::mp

4.48.1.22 double TRAJECTORY_DATA::Ms = 0.6

4.48.1.23 double TRAJECTORY_DATA::mu_0 = 12.57e-7

- 4.48.1.24 double TRAJECTORY_DATA::n_rand
- 4.48.1.25 Matrix<double> TRAJECTORY_DATA::POL
- 4.48.1.26 double TRAJECTORY_DATA::porosity = 0.8979
- 4.48.1.27 double TRAJECTORY_DATA::q_bar
- 4.48.1.28 double TRAJECTORY_DATA::Q_in
- 4.48.1.29 double TRAJECTORY_DATA::rho_f = 1000.0
- 4.48.1.30 double TRAJECTORY_DATA::rho_p = 8700.0
- 4.48.1.31 double TRAJECTORY_DATA::Rs = 0.0026925
- 4.48.1.32 double TRAJECTORY_DATA::s_rand
- 4.48.1.33 double TRAJECTORY_DATA::sigma_m
- 4.48.1.34 double TRAJECTORY_DATA::sigma_n
- 4.48.1.35 double TRAJECTORY_DATA::sigma_v
- 4.48.1.36 double TRAJECTORY_DATA::sigma_vz
- 4.48.1.37 double TRAJECTORY_DATA::sigma_z
- 4.48.1.38 double TRAJECTORY_DATA::t_rand
- 4.48.1.39 double TRAJECTORY_DATA::Temp = 298
- 4.48.1.40 double TRAJECTORY_DATA::V0
- 4.48.1.41 double TRAJECTORY_DATA::V_separator
- 4.48.1.42 double TRAJECTORY_DATA::V_wire
- 4.48.1.43 Matrix<double> TRAJECTORY_DATA::X
- 4.48.1.44 Matrix<double> TRAJECTORY_DATA::Y
- 4.48.1.45 double TRAJECTORY_DATA::Y_initial = 20.0

The documentation for this struct was generated from the following file:

- [Trajectory.h](#)

4.49 UI_DATA Struct Reference

Data structure holding the UI arguments.

```
#include <ui.h>
```

Public Attributes

- [ValueTypePair value_type](#)
Data pair for input, tells what the input is and it's type.
- `std::vector< std::string >` [user_input](#)
What is read in from the console at any point.
- `std::vector< std::string >` [input_files](#)
A vector of input file names and directories given by user.
- `std::string` [path](#)
Path to where input files are located.
- `int` [count](#) = 0
Number of times a questing has been asked.
- `int` [max](#) = 3
Maximum allowable recursions of a question.
- `int` [option](#)
Current option choosen by the user.
- `bool` [Path](#) = false
True if user gives path as an option.
- `bool` [Files](#) = false
True if user gives input files as an option.
- `bool` [MissingArg](#) = true
True if an input argument is missing; False if everything is ok.
- `bool` [BasicUI](#) = true
True if using Basic UI; False if using Advanced UI.
- `int` [argc](#)
Number of console arguments given on input.
- `const char *` [argv](#) []
Actual console arguments given at execution.

4.49.1 Detailed Description

Data structure holding the UI arguments.

C-Style object for interfacing with users request upon execution of the program. User input is stored in objects below and a series of booleans is used to determine how and what to execute.

4.49.2 Member Data Documentation

4.49.2.1 `int UI_DATA::argc`

Number of console arguments given on input.

4.49.2.2 `const char* UI_DATA::argv[]`

Actual console arguments given at execution.

4.49.2.3 `bool UI_DATA::BasicUI = true`

True if using Basic UI; False if using Advanced UI.

4.49.2.4 `int UI_DATA::count = 0`

Number of times a questing has been asked.

4.49.2.5 `bool UI_DATA::Files = false`

True if user gives input files as an option.

4.49.2.6 `std::vector<std::string> UI_DATA::input_files`

A vector of input file names and directories given by user.

4.49.2.7 `int UI_DATA::max = 3`

Maximum allowable recursions of a question.

4.49.2.8 `bool UI_DATA::MissingArg = true`

True if an input argument is missing; False if everything is ok.

4.49.2.9 `int UI_DATA::option`

Current option choosen by the user.

4.49.2.10 `std::string UI_DATA::path`

Path to where input files are located.

4.49.2.11 `bool UI_DATA::Path = false`

True if user gives path as an option.

4.49.2.12 `std::vector<std::string> UI_DATA::user_input`

What is read in from the console at any point.

4.49.2.13 `ValueTypePair UI_DATA::value_type`

Data pair for input, tells what the input is and it's type.

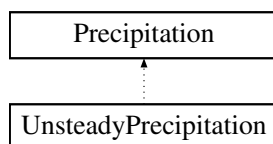
The documentation for this struct was generated from the following file:

- [ui.h](#)

4.50 UnsteadyPrecipitation Class Reference

```
#include <shark.h>
```

Inheritance diagram for UnsteadyPrecipitation:



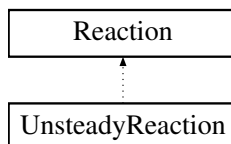
The documentation for this class was generated from the following file:

- [shark.h](#)

4.51 UnsteadyReaction Class Reference

```
#include <shark.h>
```

Inheritance diagram for UnsteadyReaction:



Public Member Functions

- [UnsteadyReaction](#) ()
- [~UnsteadyReaction](#) ()
- void [Initialize_List](#) (MasterSpeciesList &List)
- void [Display_Info](#) ()
- void [Set_Species_Index](#) (int i)
- void [Set_Species_Index](#) (std::string formula)
- void [Set_Stoichiometric](#) (int i, double v)
- void [Set_Equilibrium](#) (double v)
- void [Set_Enthalpy](#) (double H)
- void [Set_Entropy](#) (double S)
- void [Set_EnthalpyANDEntropy](#) (double H, double S)
- void [Set_Energy](#) (double G)
- void [Set_InitialValue](#) (double ic)
- void [Set_MaximumValue](#) (double max)
- void [Set_Forward](#) (double forward)
- void [Set_Reverse](#) (double reverse)
- void [Set_ForwardRef](#) (double Fref)
- void [Set_ReverseRef](#) (double Rref)
- void [Set_ActivationEnergy](#) (double E)
- void [Set_Affinity](#) (double b)
- void [Set_TimeStep](#) (double dt)
- void [checkSpeciesEnergies](#) ()
- void [calculateEnergies](#) ()
- void [calculateEquilibrium](#) (double T)
- void [calculateRate](#) (double T)
- bool [haveEquilibrium](#) ()
- bool [haveRate](#) ()
- int [Get_Species_Index](#) ()

- double [Get_Stoichiometric](#) (int i)
- double [Get_Equilibrium](#) ()
- double [Get_Enthalpy](#) ()
- double [Get_Entropy](#) ()
- double [Get_Energy](#) ()
- double [Get_InitialValue](#) ()
- double [Get_MaximumValue](#) ()
- double [Get_Forward](#) ()
- double [Get_Reverse](#) ()
- double [Get_ForwardRef](#) ()
- double [Get_ReverseRef](#) ()
- double [Get_ActivationEnergy](#) ()
- double [Get_Affinity](#) ()
- double [Get_TimeStep](#) ()
- double [Eval_ReactionRate](#) (const [Matrix](#)< double > &x, const [Matrix](#)< double > &gama)
- double [Eval_Residual](#) (const [Matrix](#)< double > &x_new, const [Matrix](#)< double > &x_old, const [Matrix](#)< double > &gama_new, const [Matrix](#)< double > &gama_old)
- double [Eval_Residual](#) (const [Matrix](#)< double > &x, const [Matrix](#)< double > &gama)
- double [Eval_IC_Residual](#) (const [Matrix](#)< double > &x)
- double [Explicit_Eval](#) (const [Matrix](#)< double > &x, const [Matrix](#)< double > &gama)

Protected Attributes

- double [initial_value](#)
- double [max_value](#)
- double [forward_rate](#)
- double [reverse_rate](#)
- double [forward_ref_rate](#)
- double [reverse_ref_rate](#)
- double [activation_energy](#)
- double [temperature_affinity](#)
- double [time_step](#)
- bool [HaveForward](#)
- bool [HaveReverse](#)
- bool [HaveForRef](#)
- bool [HaveRevRef](#)
- int [species_index](#)

Additional Inherited Members

4.51.1 Constructor & Destructor Documentation

4.51.1.1 [UnsteadyReaction::UnsteadyReaction](#) ()

4.51.1.2 [UnsteadyReaction::~~UnsteadyReaction](#) ()

4.51.2 Member Function Documentation

4.51.2.1 [void UnsteadyReaction::calculateEnergies](#) ()

4.51.2.2 [void UnsteadyReaction::calculateEquilibrium](#) (double T)

4.51.2.3 [void UnsteadyReaction::calculateRate](#) (double T)

- 4.51.2.4 void UnsteadyReaction::checkSpeciesEnergies ()
- 4.51.2.5 void UnsteadyReaction::Display_Info ()
- 4.51.2.6 double UnsteadyReaction::Eval_IC_Residual (const Matrix< double > & x)
- 4.51.2.7 double UnsteadyReaction::Eval_ReactionRate (const Matrix< double > & x, const Matrix< double > & gama)
- 4.51.2.8 double UnsteadyReaction::Eval_Residual (const Matrix< double > & x_new, const Matrix< double > & x_old, const Matrix< double > & gama_new, const Matrix< double > & gama_old)
- 4.51.2.9 double UnsteadyReaction::Eval_Residual (const Matrix< double > & x, const Matrix< double > & gama)
- 4.51.2.10 double UnsteadyReaction::Explicit_Eval (const Matrix< double > & x, const Matrix< double > & gama)
- 4.51.2.11 double UnsteadyReaction::Get_ActivationEnergy ()
- 4.51.2.12 double UnsteadyReaction::Get_Affinity ()
- 4.51.2.13 double UnsteadyReaction::Get_Energy ()
- 4.51.2.14 double UnsteadyReaction::Get_Enthalpy ()
- 4.51.2.15 double UnsteadyReaction::Get_Entropy ()
- 4.51.2.16 double UnsteadyReaction::Get_Equilibrium ()
- 4.51.2.17 double UnsteadyReaction::Get_Forward ()
- 4.51.2.18 double UnsteadyReaction::Get_ForwardRef ()
- 4.51.2.19 double UnsteadyReaction::Get_InitialValue ()
- 4.51.2.20 double UnsteadyReaction::Get_MaximumValue ()
- 4.51.2.21 double UnsteadyReaction::Get_Reverse ()
- 4.51.2.22 double UnsteadyReaction::Get_ReverseRef ()
- 4.51.2.23 int UnsteadyReaction::Get_Species_Index ()
- 4.51.2.24 double UnsteadyReaction::Get_Stoichiometric (int i)
- 4.51.2.25 double UnsteadyReaction::Get_TimeStep ()
- 4.51.2.26 bool UnsteadyReaction::haveEquilibrium ()
- 4.51.2.27 bool UnsteadyReaction::haveRate ()
- 4.51.2.28 void UnsteadyReaction::Initialize_List (MasterSpeciesList & List)
- 4.51.2.29 void UnsteadyReaction::Set_ActivationEnergy (double E)
- 4.51.2.30 void UnsteadyReaction::Set_Affinity (double b)
- 4.51.2.31 void UnsteadyReaction::Set_Energy (double G)

- 4.51.2.32 void UnsteadyReaction::Set_Enthalpy (double *H*)
- 4.51.2.33 void UnsteadyReaction::Set_EnthalpyANDEntropy (double *H*, double *S*)
- 4.51.2.34 void UnsteadyReaction::Set_Entropy (double *S*)
- 4.51.2.35 void UnsteadyReaction::Set_Equilibrium (double *v*)
- 4.51.2.36 void UnsteadyReaction::Set_Forward (double *forward*)
- 4.51.2.37 void UnsteadyReaction::Set_ForwardRef (double *Fref*)
- 4.51.2.38 void UnsteadyReaction::Set_InitialValue (double *ic*)
- 4.51.2.39 void UnsteadyReaction::Set_MaximumValue (double *max*)
- 4.51.2.40 void UnsteadyReaction::Set_Reverse (double *reverse*)
- 4.51.2.41 void UnsteadyReaction::Set_ReverseRef (double *Rref*)
- 4.51.2.42 void UnsteadyReaction::Set_Species_Index (int *i*)
- 4.51.2.43 void UnsteadyReaction::Set_Species_Index (std::string *formula*)
- 4.51.2.44 void UnsteadyReaction::Set_Stoichiometric (int *i*, double *v*)
- 4.51.2.45 void UnsteadyReaction::Set_TimeStep (double *dt*)

4.51.3 Member Data Documentation

- 4.51.3.1 double UnsteadyReaction::activation_energy [protected]
- 4.51.3.2 double UnsteadyReaction::forward_rate [protected]
- 4.51.3.3 double UnsteadyReaction::forward_ref_rate [protected]
- 4.51.3.4 bool UnsteadyReaction::HaveForRef [protected]
- 4.51.3.5 bool UnsteadyReaction::HaveForward [protected]
- 4.51.3.6 bool UnsteadyReaction::HaveReverse [protected]
- 4.51.3.7 bool UnsteadyReaction::HaveRevRef [protected]
- 4.51.3.8 double UnsteadyReaction::initial_value [protected]
- 4.51.3.9 double UnsteadyReaction::max_value [protected]
- 4.51.3.10 double UnsteadyReaction::reverse_rate [protected]
- 4.51.3.11 double UnsteadyReaction::reverse_ref_rate [protected]
- 4.51.3.12 int UnsteadyReaction::species_index [protected]
- 4.51.3.13 double UnsteadyReaction::temperature_affinity [protected]

4.51.3.14 double UnsteadyReaction::time_step [protected]

The documentation for this class was generated from the following files:

- [shark.h](#)
- [shark.cpp](#)

4.52 ValueTypePair Class Reference

```
#include <yaml_wrapper.h>
```

Public Member Functions

- [ValueTypePair](#) ()
- [~ValueTypePair](#) ()
- [ValueTypePair](#) (const std::pair< std::string, int > &vt)
- [ValueTypePair](#) (std::string value, int [type](#))
- [ValueTypePair](#) (const [ValueTypePair](#) &vt)
- [ValueTypePair](#) & [operator=](#) (const [ValueTypePair](#) &vt)
- void [editValue](#) (std::string value)
- void [editPair](#) (std::string value, int [type](#))
- void [findType](#) ()
- void [assertType](#) (int [type](#))
- void [DisplayPair](#) ()
- std::string [getString](#) ()
- bool [getBool](#) ()
- double [getDouble](#) ()
- int [getInt](#) ()
- std::string [getValue](#) ()
- int [getType](#) ()
- std::pair< std::string, int > & [getPair](#) ()

Private Attributes

- std::pair< std::string, int > [Value_Type](#)
- int [type](#)

4.52.1 Constructor & Destructor Documentation

4.52.1.1 [ValueTypePair::ValueTypePair](#) ()

4.52.1.2 [ValueTypePair::~~ValueTypePair](#) ()

4.52.1.3 [ValueTypePair::ValueTypePair](#) (const std::pair< std::string, int > & vt)

4.52.1.4 [ValueTypePair::ValueTypePair](#) (std::string value, int type)

4.52.1.5 [ValueTypePair::ValueTypePair](#) (const [ValueTypePair](#) & vt)

4.52.2 Member Function Documentation

- 4.52.2.1 void ValueTypePair::assertType (int *type*)
- 4.52.2.2 void ValueTypePair::DisplayPair ()
- 4.52.2.3 void ValueTypePair::editPair (std::string *value*, int *type*)
- 4.52.2.4 void ValueTypePair::editValue (std::string *value*)
- 4.52.2.5 void ValueTypePair::findType ()
- 4.52.2.6 bool ValueTypePair::getBool ()
- 4.52.2.7 double ValueTypePair::getDouble ()
- 4.52.2.8 int ValueTypePair::getInt ()
- 4.52.2.9 std::pair< std::string, int > & ValueTypePair::getPair ()
- 4.52.2.10 std::string ValueTypePair::getString ()
- 4.52.2.11 int ValueTypePair::getType ()
- 4.52.2.12 std::string ValueTypePair::getValue ()
- 4.52.2.13 ValueTypePair & ValueTypePair::operator= (const ValueTypePair & *vt*)

4.52.3 Member Data Documentation

- 4.52.3.1 int ValueTypePair::type [private]
- 4.52.3.2 std::pair<std::string,int> ValueTypePair::Value_Type [private]

The documentation for this class was generated from the following files:

- [yaml_wrapper.h](#)
- [yaml_wrapper.cpp](#)

4.53 yaml_cpp_class Class Reference

```
#include <yaml_wrapper.h>
```

Public Member Functions

- [yaml_cpp_class](#) ()
- [~yaml_cpp_class](#) ()
- int [setInputFile](#) (const char *file)
- int [readInputFile](#) ()
- int [cleanup](#) ()
- int [executeYamlRead](#) (const char *file)
- [YamlWrapper](#) & [getYamlWrapper](#) ()
- void [DisplayContents](#) ()

Private Attributes

- [YamlWrapper](#) `yaml_wrapper`
- FILE * `input_file`
- const char * `file_name`
- `yaml_parser_t` `token_parser`
- `yaml_token_t` `current_token`
- `yaml_token_t` `previous_token`

4.53.1 Constructor & Destructor Documentation

4.53.1.1 `yaml_cpp_class::yaml_cpp_class ()`

4.53.1.2 `yaml_cpp_class::~~yaml_cpp_class ()`

4.53.2 Member Function Documentation

4.53.2.1 `int yaml_cpp_class::cleanup ()`

4.53.2.2 `void yaml_cpp_class::DisplayContents ()`

4.53.2.3 `int yaml_cpp_class::executeYamlRead (const char * file)`

4.53.2.4 `YamlWrapper & yaml_cpp_class::getYamlWrapper ()`

4.53.2.5 `int yaml_cpp_class::readInputFile ()`

4.53.2.6 `int yaml_cpp_class::setInputFile (const char * file)`

4.53.3 Member Data Documentation

4.53.3.1 `yaml_token_t yaml_cpp_class::current_token` [private]

4.53.3.2 `const char* yaml_cpp_class::file_name` [private]

4.53.3.3 `FILE* yaml_cpp_class::input_file` [private]

4.53.3.4 `yaml_token_t yaml_cpp_class::previous_token` [private]

4.53.3.5 `yaml_parser_t yaml_cpp_class::token_parser` [private]

4.53.3.6 `YamlWrapper yaml_cpp_class::yaml_wrapper` [private]

The documentation for this class was generated from the following files:

- [yaml_wrapper.h](#)
- [yaml_wrapper.cpp](#)

4.54 YamlWrapper Class Reference

```
#include <yaml_wrapper.h>
```

Public Member Functions

- [YamlWrapper](#) ()
- [~YamlWrapper](#) ()
- [YamlWrapper](#) (const [YamlWrapper](#) &yaml)
- [YamlWrapper](#) (std::string key, const [Document](#) &doc)
- [YamlWrapper](#) & [operator=](#) (const [YamlWrapper](#) &yaml)
- [Document](#) & [operator\(\)](#) (const std::string key)
- [Document](#) [operator\(\)](#) (const std::string key) const
- std::map< std::string, [Document](#) > & [getDocMap](#) ()
- [Document](#) & [getDocument](#) (std::string key)
- std::map< std::string, [Document](#) >::const_iterator [end](#) () const
- std::map< std::string, [Document](#) >::iterator [end](#) ()
- std::map< std::string, [Document](#) >::const_iterator [begin](#) () const
- std::map< std::string, [Document](#) >::iterator [begin](#) ()
- void [clear](#) ()
- void [resetKeys](#) ()
- void [changeKey](#) (std::string oldKey, std::string newKey)
- void [revalidateAllKeys](#) ()
- void [DisplayContents](#) ()
- void [addDocKey](#) (std::string key)
- void [copyAnchor2Alias](#) (std::string alias, [Document](#) &ref)
- int [size](#) ()
- [Document](#) & [getAnchoredDoc](#) (std::string alias)
- [Document](#) & [getDocFromHeadAlias](#) (std::string alias)
- [Document](#) & [getDocFromSubAlias](#) (std::string alias)

Private Attributes

- std::map< std::string, [Document](#) > [Doc_Map](#)

4.54.1 Constructor & Destructor Documentation

4.54.1.1 [YamlWrapper::YamlWrapper](#) ()

4.54.1.2 [YamlWrapper::~~YamlWrapper](#) ()

4.54.1.3 [YamlWrapper::YamlWrapper](#) (const [YamlWrapper](#) & *yaml*)

4.54.1.4 [YamlWrapper::YamlWrapper](#) (std::string *key*, const [Document](#) & *doc*)

4.54.2 Member Function Documentation

4.54.2.1 void [YamlWrapper::addDocKey](#) (std::string *key*)

4.54.2.2 std::map< std::string, [Document](#) >::const_iterator [YamlWrapper::begin](#) () const

4.54.2.3 std::map< std::string, [Document](#) >::iterator [YamlWrapper::begin](#) ()

4.54.2.4 void [YamlWrapper::changeKey](#) (std::string *oldKey*, std::string *newKey*)

- 4.54.2.5 `void YamlWrapper::clear ()`
- 4.54.2.6 `void YamlWrapper::copyAnchor2Alias (std::string alias, Document & ref)`
- 4.54.2.7 `void YamlWrapper::DisplayContents ()`
- 4.54.2.8 `std::map< std::string, Document >::const_iterator YamlWrapper::end () const`
- 4.54.2.9 `std::map< std::string, Document >::iterator YamlWrapper::end ()`
- 4.54.2.10 `Document & YamlWrapper::getAnchoredDoc (std::string alias)`
- 4.54.2.11 `Document & YamlWrapper::getDocFromHeadAlias (std::string alias)`
- 4.54.2.12 `Document & YamlWrapper::getDocFromSubAlias (std::string alias)`
- 4.54.2.13 `std::map< std::string, Document > & YamlWrapper::getDocMap ()`
- 4.54.2.14 `Document & YamlWrapper::getDocument (std::string key)`
- 4.54.2.15 `Document & YamlWrapper::operator() (const std::string key)`
- 4.54.2.16 `Document YamlWrapper::operator() (const std::string key) const`
- 4.54.2.17 `YamlWrapper & YamlWrapper::operator= (const YamlWrapper & yaml)`
- 4.54.2.18 `void YamlWrapper::resetKeys ()`
- 4.54.2.19 `void YamlWrapper::revalidateAllKeys ()`
- 4.54.2.20 `int YamlWrapper::size ()`

4.54.3 Member Data Documentation

- 4.54.3.1 `std::map<std::string, Document> YamlWrapper::Doc_Map [private]`

The documentation for this class was generated from the following files:

- [yaml_wrapper.h](#)
- [yaml_wrapper.cpp](#)

Chapter 5

File Documentation

5.1 dogfish.cpp File Reference

```
#include "dogfish.h"
```

Functions

- void [print2file_species_header](#) (FILE *Output, [DOGFISH_DATA](#) *dog_dat, int i)
- void [print2file_DOGFISH_header](#) ([DOGFISH_DATA](#) *dog_dat)
- void [print2file_DOGFISH_result_old](#) ([DOGFISH_DATA](#) *dog_dat)
- void [print2file_DOGFISH_result_new](#) ([DOGFISH_DATA](#) *dog_dat)
- double [default_Retardation](#) (int i, int l, const void *data)
- double [default_IntraDiffusion](#) (int i, int l, const void *data)
- double [default_FilmMTCoeff](#) (int i, const void *data)
- double [default_SurfaceConcentration](#) (int i, const void *data)
- int [setup_DOGFISH_DATA](#) (FILE *file, double(*eval_R)(int i, int l, const void *user_data), double(*eval_DI)(int i, int l, const void *user_data), double(*eval_kf)(int i, const void *user_data), double(*eval_qs)(int i, const void *user_data), const void *user_data, [DOGFISH_DATA](#) *dog_dat)
- int [DOGFISH_Executioner](#) ([DOGFISH_DATA](#) *dog_dat)
- int [set_DOGFISH_ICs](#) ([DOGFISH_DATA](#) *dog_dat)
- int [set_DOGFISH_timestep](#) ([DOGFISH_DATA](#) *dog_dat)
- int [DOGFISH_preprocesses](#) ([DOGFISH_DATA](#) *dog_dat)
- int [set_DOGFISH_params](#) (const void *user_data)
- int [DOGFISH_postprocesses](#) ([DOGFISH_DATA](#) *dog_dat)
- int [DOGFISH_reset](#) ([DOGFISH_DATA](#) *dog_dat)
- int [DOGFISH](#) ([DOGFISH_DATA](#) *dog_dat)
- int [DOGFISH_TESTS](#) ()

5.1.1 Function Documentation

5.1.1.1 double [default_FilmMTCoeff](#) (int *i*, const void * *data*)

5.1.1.2 double [default_IntraDiffusion](#) (int *i*, int *l*, const void * *data*)

5.1.1.3 double [default_Retardation](#) (int *i*, int *l*, const void * *data*)

5.1.1.4 double [default_SurfaceConcentration](#) (int *i*, const void * *data*)

- 5.1.1.5 int DOGFISH (DOGFISH_DATA * *dog_dat*)
- 5.1.1.6 int DOGFISH_Executioner (DOGFISH_DATA * *dog_dat*)
- 5.1.1.7 int DOGFISH_postprocesses (DOGFISH_DATA * *dog_dat*)
- 5.1.1.8 int DOGFISH_preprocesses (DOGFISH_DATA * *dog_dat*)
- 5.1.1.9 int DOGFISH_reset (DOGFISH_DATA * *dog_dat*)
- 5.1.1.10 int DOGFISH_TESTS ()
- 5.1.1.11 void print2file_DOGFISH_header (DOGFISH_DATA * *dog_dat*)
- 5.1.1.12 void print2file_DOGFISH_result_new (DOGFISH_DATA * *dog_dat*)
- 5.1.1.13 void print2file_DOGFISH_result_old (DOGFISH_DATA * *dog_dat*)
- 5.1.1.14 void print2file_species_header (FILE * *Output*, DOGFISH_DATA * *dog_dat*, int *i*)
- 5.1.1.15 int set_DOGFISH_ICs (DOGFISH_DATA * *dog_dat*)
- 5.1.1.16 int set_DOGFISH_params (const void * *user_data*)
- 5.1.1.17 int set_DOGFISH_timestep (DOGFISH_DATA * *dog_dat*)
- 5.1.1.18 int setup_DOGFISH_DATA (FILE * *file*, double(*)(int i, int l, const void **user_data*) *eval_R*, double(*)(int i, int l, const void **user_data*) *eval_DI*, double(*)(int i, const void **user_data*) *eval_kf*, double(*)(int i, const void **user_data*) *eval_qs*, const void * *user_data*, DOGFISH_DATA * *dog_dat*)

5.2 dogfish.h File Reference

```
#include "finch.h"
#include "mola.h"
```

Classes

- struct [DOGFISH_PARAM](#)
- struct [DOGFISH_DATA](#)

Functions

- void [print2file_species_header](#) (FILE **Output*, [DOGFISH_DATA](#) **dog_dat*, int *i*)
- void [print2file_DOGFISH_header](#) ([DOGFISH_DATA](#) **dog_dat*)
- void [print2file_DOGFISH_result_old](#) ([DOGFISH_DATA](#) **dog_dat*)
- void [print2file_DOGFISH_result_new](#) ([DOGFISH_DATA](#) **dog_dat*)
- double [default_Retardation](#) (int *i*, int *l*, const void **data*)
- double [default_IntraDiffusion](#) (int *i*, int *l*, const void **data*)
- double [default_FilmMTCoeff](#) (int *i*, const void **data*)
- double [default_SurfaceConcentration](#) (int *i*, const void **data*)
- int [setup_DOGFISH_DATA](#) (FILE **file*, double(**eval_R*)(int *i*, int *l*, const void **user_data*), double(**eval_DI*)(int *i*, int *l*, const void **user_data*), double(**eval_kf*)(int *i*, const void **user_data*), double(**eval_qs*)(int *i*, const void **user_data*), const void **user_data*, [DOGFISH_DATA](#) **dog_dat*)

- int [DOGFISH_Executioner](#) (DOGFISH_DATA *dog_dat)
- int [set_DOGFISH_ICs](#) (DOGFISH_DATA *dog_dat)
- int [set_DOGFISH_timestep](#) (DOGFISH_DATA *dog_dat)
- int [DOGFISH_preprocesses](#) (DOGFISH_DATA *dog_dat)
- int [set_DOGFISH_params](#) (const void *user_data)
- int [DOGFISH_postprocesses](#) (DOGFISH_DATA *dog_dat)
- int [DOGFISH_reset](#) (DOGFISH_DATA *dog_dat)
- int [DOGFISH](#) (DOGFISH_DATA *dog_dat)
- int [DOGFISH_TESTS](#) ()

5.2.1 Function Documentation

- 5.2.1.1 double [default_FilmMTCoeff](#) (int *i*, const void * *data*)
- 5.2.1.2 double [default_IntraDiffusion](#) (int *i*, int *l*, const void * *data*)
- 5.2.1.3 double [default_Retardation](#) (int *i*, int *l*, const void * *data*)
- 5.2.1.4 double [default_SurfaceConcentration](#) (int *i*, const void * *data*)
- 5.2.1.5 int [DOGFISH](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.6 int [DOGFISH_Executioner](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.7 int [DOGFISH_postprocesses](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.8 int [DOGFISH_preprocesses](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.9 int [DOGFISH_reset](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.10 int [DOGFISH_TESTS](#) ()
- 5.2.1.11 void [print2file_DOGFISH_header](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.12 void [print2file_DOGFISH_result_new](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.13 void [print2file_DOGFISH_result_old](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.14 void [print2file_species_header](#) (FILE * *Output*, DOGFISH_DATA * *dog_dat*, int *i*)
- 5.2.1.15 int [set_DOGFISH_ICs](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.16 int [set_DOGFISH_params](#) (const void * *user_data*)
- 5.2.1.17 int [set_DOGFISH_timestep](#) (DOGFISH_DATA * *dog_dat*)
- 5.2.1.18 int [setup_DOGFISH_DATA](#) (FILE * *file*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_R*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_DI*, double(*) (int *i*, const void * *user_data*) *eval_kf*, double(*) (int *i*, const void * *user_data*) *eval_qs*, const void * *user_data*, DOGFISH_DATA * *dog_dat*)

5.3 eel.cpp File Reference

```
#include "eel.h"
```

Functions

- int [EEL_TESTS](#) ()

5.3.1 Function Documentation

5.3.1.1 int EEL_TESTS ()

5.4 eel.h File Reference

```
#include <stdio.h>
#include <math.h>
#include <iostream>
#include <fstream>
#include <stdlib.h>
#include <vector>
#include <time.h>
#include <float.h>
#include <string>
#include "error.h"
```

Classes

- class [Atom](#)
- class [PeriodicTable](#)

Functions

- int [EEL_TESTS](#) ()

5.4.1 Function Documentation

5.4.1.1 int EEL_TESTS ()

5.5 egret.cpp File Reference

```
#include "egret.h"
```

Functions

- int [initialize_data](#) (int N, [MIXED_GAS](#) *gas_dat)
- int [set_variables](#) (double PT, double T, double us, double L, std::vector< double > &y, [MIXED_GAS](#) *gas_dat)
- int [calculate_properties](#) ([MIXED_GAS](#) *gas_dat)
- int [EGRET_TESTS](#) ()

5.5.1 Function Documentation

5.5.1.1 int calculate_properties ([MIXED_GAS](#) * *gas_dat*)

5.5.1.2 int EGRET_TESTS ()

5.5.1.3 int initialize_data (int *N*, MIXED_GAS * *gas_dat*)

5.5.1.4 int set_variables (double *PT*, double *T*, double *us*, double *L*, std::vector< double > & *y*, MIXED_GAS * *gas_dat*)

5.6 egret.h File Reference

```
#include "macaw.h"
```

Classes

- struct [PURE_GAS](#)
- struct [MIXED_GAS](#)

Macros

- #define [Rstd](#) 8.3144621
- #define [RE3](#) 8.3144621E+3
- #define [Po](#) 100.0
- #define [Cstd](#)(*p*, *T*) ((*p*)/([Rstd](#)**T*))
- #define [CE3](#)(*p*, *T*) ((*p*)/([RE3](#)**T*))
- #define [Pstd](#)(*c*, *T*) ((*c*)*[Rstd](#)**T*)
- #define [PE3](#)(*c*, *T*) ((*c*)*[RE3](#)**T*)
- #define [Nu](#)(*mu*, *rho*) ((*mu*)/(*rho*))
- #define [PSI](#)(*T*) (0.873143 + (0.000072375**T*))
- #define [Dp_ij](#)(*Dij*, *PT*) ((*PT***Dij*)/[Po](#))
- #define [D_ij](#)(*MWi*, *MWj*, *rhoi*, *rhoj*, *mui*, *muj*) ((4.0 / sqrt(2.0)) * pow((((1/*MWi*)+(1/*MWj*)),0.5)) / pow((pow((pow((*rhoi*/(1.385**mui*))),2.0)/*MWi*),0.25)+ pow((pow((*rhoj*/(1.385**muj*))),2.0)/*MWj*),0.25)),2.0)
- #define [Mu](#)(*muo*, *To*, *C*, *T*) (*muo* * ((*To* + *C*)/(*T* + *C*)) * pow((*T*/*To*),1.5))
- #define [D_ii](#)(*rhoi*, *mui*) (1.385**mui*/*rhoi*)
- #define [ReNum](#)(*u*, *L*, *nu*) (*u***L*/*nu*)
- #define [ScNum](#)(*nu*, *D*) (*nu*/*D*)
- #define [FilmMTCoeff](#)(*D*, *L*, *Re*, *Sc*) ((*D*/*L*)*(2.0 + (1.1*pow(*Re*,0.6)*pow(*Sc*,0.3))))

Functions

- int [initialize_data](#) (int *N*, MIXED_GAS **gas_dat*)
- int [set_variables](#) (double *PT*, double *T*, double *us*, double *L*, std::vector< double > &*y*, MIXED_GAS **gas_dat*)
- int [calculate_properties](#) (MIXED_GAS **gas_dat*)
- int [EGRET_TESTS](#) ()

5.6.1 Macro Definition Documentation

5.6.1.1 #define [CE3](#)(*p*, *T*) ((*p*)/([RE3](#)**T*))

5.6.1.2 #define [Cstd](#)(*p*, *T*) ((*p*)/([Rstd](#)**T*))

5.6.1.3 #define [D_ii](#)(*rhoi*, *mui*) (1.385**mui*/*rhoi*)

```
5.6.1.4 #define D_ij( MWi, MWj, rhoi, rhoj, mui, muj ) ( (4.0 / sqrt(2.0)) * pow(((1/MWi)+(1/MWj)),0.5) ) / pow(
    (pow((pow((rhoi/(1.385*mui)),2.0)/MWi),0.25)+ pow((pow((rhoj/(1.385*muj)),2.0)/MWj),0.25)),2.0 )
```

```
5.6.1.5 #define Dp_ij( Dij, PT ) ((PT*Dij)/Po)
```

```
5.6.1.6 #define FilmMTCoeff( D, L, Re, Sc ) ((D/L)*(2.0 + (1.1*pow(Re,0.6)*pow(Sc,0.3))))
```

```
5.6.1.7 #define Mu( muo, To, C, T ) (muo * ((To + C)/(T + C)) * pow((T/To),1.5) )
```

```
5.6.1.8 #define Nu( mu, rho ) ((mu)/(rho))
```

```
5.6.1.9 #define PE3( c, T ) ((c)*RE3*T)
```

```
5.6.1.10 #define Po 100.0
```

```
5.6.1.11 #define PSI( T ) (0.873143 + (0.000072375*T))
```

```
5.6.1.12 #define Pstd( c, T ) ((c)*Rstd*T)
```

```
5.6.1.13 #define RE3 8.3144621E+3
```

```
5.6.1.14 #define ReNum( u, L, nu ) (u*L/nu)
```

```
5.6.1.15 #define Rstd 8.3144621
```

```
5.6.1.16 #define ScNum( nu, D ) (nu/D)
```

5.6.2 Function Documentation

```
5.6.2.1 int calculate_properties ( MIXED_GAS * gas_dat )
```

```
5.6.2.2 int EGRET_TESTS ( )
```

```
5.6.2.3 int initialize_data ( int N, MIXED_GAS * gas_dat )
```

```
5.6.2.4 int set_variables ( double PT, double T, double us, double L, std::vector< double > & y, MIXED_GAS * gas_dat )
```

5.7 error.cpp File Reference

```
#include "error.h"
```

Functions

- void [error](#) (int flag)

5.7.1 Function Documentation

```
5.7.1.1 void error ( int flag )
```

5.8 error.h File Reference

```
#include <iostream>
```

Macros

- #define `mError(i)`

Enumerations

- enum `error_type` {
`generic_error`, `file_dne`, `indexing_error`, `magpie_reverse_error`,
`simulation_fail`, `invalid_components`, `invalid_boolean`, `invalid_molefraction`,
`invalid_gas_sum`, `invalid_solid_sum`, `scenario_fail`, `out_of_bounds`,
`non_square_matrix`, `dim_mis_match`, `empty_matrix`, `opt_no_support`,
`invalid_fraction`, `ortho_check_fail`, `unstable_matrix`, `no_diffusion`,
`negative_mass`, `negative_time`, `matvec_mis_match`, `arg_matrix_same`,
`singular_matrix`, `matrix_too_small`, `invalid_size`, `nullptr_func`,
`invalid_norm`, `vector_out_of_bounds`, `zero_vector`, `tensor_out_of_bounds`,
`non_real_edge`, `nullptr_error`, `invalid_atom`, `invalid_proton`,
`invalid_neutron`, `invalid_electron`, `invalid_valence`, `string_parse_error`,
`unregistered_name`, `rxn_rate_error`, `invalid_species`, `duplicate_variable`,
`missing_information`, `invalid_type`, `key_not_found`, `anchor_alias_dne`,
`initial_error`, `not_a_token`, `read_error`, `invalid_console_input` }

Functions

- void `error` (int flag)

5.8.1 Macro Definition Documentation

5.8.1.1 #define `mError(i)`

Value:

```
{error(i); \
std::cout << "Source: " << __FILE__ << "\nLine: " << __LINE__ << std::endl;}
```

5.8.2 Enumeration Type Documentation

5.8.2.1 enum `error_type`

Enumerator

generic_error
file_dne
indexing_error
magpie_reverse_error
simulation_fail
invalid_components
invalid_boolean
invalid_molefraction
invalid_gas_sum
invalid_solid_sum
scenario_fail
out_of_bounds

non_square_matrix
dim_mis_match
empty_matrix
opt_no_support
invalid_fraction
ortho_check_fail
unstable_matrix
no_diffusion
negative_mass
negative_time
matvec_mis_match
arg_matrix_same
singular_matrix
matrix_too_small
invalid_size
nullptr_func
invalid_norm
vector_out_of_bounds
zero_vector
tensor_out_of_bounds
non_real_edge
nullptr_error
invalid_atom
invalid_proton
invalid_neutron
invalid_electron
invalid_valence
string_parse_error
unregistered_name
rxn_rate_error
invalid_species
duplicate_variable
missing_information
invalid_type
key_not_found
anchor_alias_dne
initial_error
not_a_token
read_error
invalid_console_input

5.8.3 Function Documentation

5.8.3.1 void error (int *flag*)

5.9 finch.cpp File Reference

```
#include "finch.h"
```


Functions

- double [max](#) (std::vector< double > &values)
- double [min](#) (std::vector< double > &values)
- double [minmod](#) (std::vector< double > &values)
- int [uTotal](#) (FINCH_DATA *dat)
- int [uAverage](#) (FINCH_DATA *dat)
- int [check_Mass](#) (FINCH_DATA *dat)
- int [l_direct](#) (FINCH_DATA *dat)
- int [lark_picard_step](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &G, const void *data)
- int [nl_picard](#) (FINCH_DATA *dat)
- int [setup_FINCH_DATA](#) (int(*user_callroutine)(const void *user_data), int(*user_setic)(const void *user_data), int(*user_timestep)(const void *user_data), int(*user_preprocess)(const void *user_data), int(*user_solve)(const void *user_data), int(*user_setparams)(const void *user_data), int(*user_discretize)(const void *user_data), int(*user_bcs)(const void *user_data), int(*user_res)(const [Matrix](#)< double > &x, [Matrix](#)< double > &res, const void *user_data), int(*user_precon)(const [Matrix](#)< double > &b, [Matrix](#)< double > &p, const void *user_data), int(*user_postprocess)(const void *user_data), int(*user_reset)(const void *user_data), FINCH_DATA *dat, const void *param_data)
- void [print2file_dim_header](#) (FILE *Output, FINCH_DATA *dat)
- void [print2file_time_header](#) (FILE *Output, FINCH_DATA *dat)
- void [print2file_result_old](#) (FILE *Output, FINCH_DATA *dat)
- void [print2file_result_new](#) (FILE *Output, FINCH_DATA *dat)
- void [print2file_newline](#) (FILE *Output, FINCH_DATA *dat)
- void [print2file_tab](#) (FILE *Output, FINCH_DATA *dat)
- int [default_execution](#) (const void *user_data)
- int [default_ic](#) (const void *user_data)
- int [default_timestep](#) (const void *user_data)
- int [default_preprocess](#) (const void *user_data)
- int [default_solve](#) (const void *user_data)
- int [default_params](#) (const void *user_data)
- int [minmod_discretization](#) (const void *user_data)
- int [vanAlbada_discretization](#) (const void *user_data)
- int [ospre_discretization](#) (const void *user_data)
- int [default_bcs](#) (const void *user_data)
- int [default_res](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &res, const void *user_data)
- int [default_precon](#) (const [Matrix](#)< double > &b, [Matrix](#)< double > &p, const void *user_data)
- int [default_postprocess](#) (const void *user_data)
- int [default_reset](#) (const void *user_data)
- int [buckley_leverett_ic](#) (const void *user_data)
- int [buckley_leverett_params](#) (const void *user_data)
- int [burgers_ic](#) (const void *user_data)
- int [burgers_params](#) (const void *user_data)
- int [burgers_bcs](#) (const void *user_data)
- int [FINCH_TESTS](#) ()

5.9.1 Function Documentation

5.9.1.1 [int buckley_leverett_ic \(const void * user_data \)](#)

5.9.1.2 [int buckley_leverett_params \(const void * user_data \)](#)

5.9.1.3 [int burgers_bcs \(const void * user_data \)](#)

5.9.1.4 [int burgers_ic \(const void * user_data \)](#)

- 5.9.1.5 `int burgers_params (const void * user_data)`
- 5.9.1.6 `int check_Mass (FINCH_DATA * dat)`
- 5.9.1.7 `int default_bcs (const void * user_data)`
- 5.9.1.8 `int default_execution (const void * user_data)`
- 5.9.1.9 `int default_ic (const void * user_data)`
- 5.9.1.10 `int default_params (const void * user_data)`
- 5.9.1.11 `int default_postprocess (const void * user_data)`
- 5.9.1.12 `int default_precon (const Matrix< double > & b, Matrix< double > & p, const void * user_data)`
- 5.9.1.13 `int default_preprocess (const void * user_data)`
- 5.9.1.14 `int default_res (const Matrix< double > & x, Matrix< double > & res, const void * user_data)`
- 5.9.1.15 `int default_reset (const void * user_data)`
- 5.9.1.16 `int default_solve (const void * user_data)`
- 5.9.1.17 `int default_timestep (const void * user_data)`
- 5.9.1.18 `int FINCH_TESTS ()`
- 5.9.1.19 `int l_direct (FINCH_DATA * dat)`
- 5.9.1.20 `int lark_picard_step (const Matrix< double > & x, Matrix< double > & G, const void * data)`
- 5.9.1.21 `double max (std::vector< double > & values)`
- 5.9.1.22 `double min (std::vector< double > & values)`
- 5.9.1.23 `double minmod (std::vector< double > & values)`
- 5.9.1.24 `int minmod_discretization (const void * user_data)`
- 5.9.1.25 `int nl_picard (FINCH_DATA * dat)`
- 5.9.1.26 `int ospre_discretization (const void * user_data)`
- 5.9.1.27 `void print2file_dim_header (FILE * Output, FINCH_DATA * dat)`
- 5.9.1.28 `void print2file_newline (FILE * Output, FINCH_DATA * dat)`
- 5.9.1.29 `void print2file_result_new (FILE * Output, FINCH_DATA * dat)`
- 5.9.1.30 `void print2file_result_old (FILE * Output, FINCH_DATA * dat)`
- 5.9.1.31 `void print2file_tab (FILE * Output, FINCH_DATA * dat)`
- 5.9.1.32 `void print2file_time_header (FILE * Output, FINCH_DATA * dat)`

5.9.1.33 `int setup_FINCH_DATA (int (*)(const void *user_data) user_callroutine, int (*)(const void *user_data) user_setic, int (*)(const void *user_data) user_timestep, int (*)(const void *user_data) user_preprocess, int (*)(const void *user_data) user_solve, int (*)(const void *user_data) user_setparams, int (*)(const void *user_data) user_discretize, int (*)(const void *user_data) user_bcs, int (*)(const Matrix< double > &x, Matrix< double > &res, const void *user_data) user_res, int (*)(const Matrix< double > &b, Matrix< double > &p, const void *user_data) user_precon, int (*)(const void *user_data) user_postprocess, int (*)(const void *user_data) user_reset, FINCH_DATA * dat, const void * param_data)`

5.9.1.34 `int uAverage (FINCH_DATA * dat)`

5.9.1.35 `int uTotal (FINCH_DATA * dat)`

5.9.1.36 `int vanAlbada_discretization (const void * user_data)`

5.10 finch.h File Reference

```
#include "macaw.h"
#include "lark.h"
```

Classes

- struct [FINCH_DATA](#)

Macros

- `#define` [FINCH_Picard](#) 0
- `#define` [LARK_Picard](#) 1
- `#define` [LARK_PJFNK](#) 2
- `#define` [Cartesian](#) 0
- `#define` [Cylindrical](#) 1
- `#define` [Spherical](#) 2

Functions

- double [max](#) (std::vector< double > &values)
- double [min](#) (std::vector< double > &values)
- double [minmod](#) (std::vector< double > &values)
- int [uTotal](#) (FINCH_DATA *dat)
- int [uAverage](#) (FINCH_DATA *dat)
- int [check_Mass](#) (FINCH_DATA *dat)
- int [l_direct](#) (FINCH_DATA *dat)
- int [lark_picard_step](#) (const Matrix< double > &x, Matrix< double > &G, const void *data)
- int [nl_picard](#) (FINCH_DATA *dat)
- int [setup_FINCH_DATA](#) (int(*user_callroutine)(const void *user_data), int(*user_setic)(const void *user_data), int(*user_timestep)(const void *user_data), int(*user_preprocess)(const void *user_data), int(*user_solve)(const void *user_data), int(*user_setparams)(const void *user_data), int(*user_discretize)(const void *user_data), int(*user_bcs)(const void *user_data), int(*user_res)(const Matrix< double > &x, Matrix< double > &res, const void *user_data), int(*user_precon)(const Matrix< double > &b, Matrix< double > &p, const void *user_data), int(*user_postprocess)(const void *user_data), int(*user_reset)(const void *user_data), FINCH_DATA *dat, const void *param_data)
- void [print2file_dim_header](#) (FILE *Output, FINCH_DATA *dat)
- void [print2file_time_header](#) (FILE *Output, FINCH_DATA *dat)
- void [print2file_result_old](#) (FILE *Output, FINCH_DATA *dat)

- void [print2file_result_new](#) (FILE *Output, [FINCH_DATA](#) *dat)
- void [print2file_newline](#) (FILE *Output, [FINCH_DATA](#) *dat)
- void [print2file_tab](#) (FILE *Output, [FINCH_DATA](#) *dat)
- int [default_execution](#) (const void *user_data)
- int [default_ic](#) (const void *user_data)
- int [default_timestep](#) (const void *user_data)
- int [default_preprocess](#) (const void *user_data)
- int [default_solve](#) (const void *user_data)
- int [default_params](#) (const void *user_data)
- int [minmod_discretization](#) (const void *user_data)
- int [vanAlbada_discretization](#) (const void *user_data)
- int [ospre_discretization](#) (const void *user_data)
- int [default_bcs](#) (const void *user_data)
- int [default_res](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &res, const void *user_data)
- int [default_precon](#) (const [Matrix](#)< double > &b, [Matrix](#)< double > &p, const void *user_data)
- int [default_postprocess](#) (const void *user_data)
- int [default_reset](#) (const void *user_data)
- int [buckley_leverett_ic](#) (const void *user_data)
- int [buckley_leverett_params](#) (const void *user_data)
- int [burgers_ic](#) (const void *user_data)
- int [burgers_params](#) (const void *user_data)
- int [burgers_bcs](#) (const void *user_data)
- int [FINCH_TESTS](#) ()

5.10.1 Macro Definition Documentation

5.10.1.1 [#define Cartesian](#) 0

5.10.1.2 [#define Cylindrical](#) 1

5.10.1.3 [#define FINCH_Picard](#) 0

5.10.1.4 [#define LARK_Picard](#) 1

5.10.1.5 [#define LARK_PJFNK](#) 2

5.10.1.6 [#define Spherical](#) 2

5.10.2 Function Documentation

5.10.2.1 int [buckley_leverett_ic](#) (const void * *user_data*)

5.10.2.2 int [buckley_leverett_params](#) (const void * *user_data*)

5.10.2.3 int [burgers_bcs](#) (const void * *user_data*)

5.10.2.4 int [burgers_ic](#) (const void * *user_data*)

5.10.2.5 int [burgers_params](#) (const void * *user_data*)

5.10.2.6 int [check_Mass](#) ([FINCH_DATA](#) * *dat*)

5.10.2.7 int [default_bcs](#) (const void * *user_data*)

- 5.10.2.8 int default_execution (const void * *user_data*)
- 5.10.2.9 int default_ic (const void * *user_data*)
- 5.10.2.10 int default_params (const void * *user_data*)
- 5.10.2.11 int default_postprocess (const void * *user_data*)
- 5.10.2.12 int default_precon (const Matrix< double > & *b*, Matrix< double > & *p*, const void * *user_data*)
- 5.10.2.13 int default_preprocess (const void * *user_data*)
- 5.10.2.14 int default_res (const Matrix< double > & *x*, Matrix< double > & *res*, const void * *user_data*)
- 5.10.2.15 int default_reset (const void * *user_data*)
- 5.10.2.16 int default_solve (const void * *user_data*)
- 5.10.2.17 int default_timestep (const void * *user_data*)
- 5.10.2.18 int FINCH_TESTS ()
- 5.10.2.19 int l_direct (FINCH_DATA * *dat*)
- 5.10.2.20 int lark_picard_step (const Matrix< double > & *x*, Matrix< double > & *G*, const void * *data*)
- 5.10.2.21 double max (std::vector< double > & *values*)
- 5.10.2.22 double min (std::vector< double > & *values*)
- 5.10.2.23 double minmod (std::vector< double > & *values*)
- 5.10.2.24 int minmod_discretization (const void * *user_data*)
- 5.10.2.25 int nl_picard (FINCH_DATA * *dat*)
- 5.10.2.26 int ospre_discretization (const void * *user_data*)
- 5.10.2.27 void print2file_dim_header (FILE * *Output*, FINCH_DATA * *dat*)
- 5.10.2.28 void print2file_newline (FILE * *Output*, FINCH_DATA * *dat*)
- 5.10.2.29 void print2file_result_new (FILE * *Output*, FINCH_DATA * *dat*)
- 5.10.2.30 void print2file_result_old (FILE * *Output*, FINCH_DATA * *dat*)
- 5.10.2.31 void print2file_tab (FILE * *Output*, FINCH_DATA * *dat*)
- 5.10.2.32 void print2file_time_header (FILE * *Output*, FINCH_DATA * *dat*)

```
5.10.2.33 int setup_FINCH_DATA ( int(*) (const void *user_data) user_callroutine, int(*) (const void *user_data) user_setic,
int(*) (const void *user_data) user_timestep, int(*) (const void *user_data) user_preprocess, int(*) (const
void *user_data) user_solve, int(*) (const void *user_data) user_setparams, int(*) (const void *user_data)
user_discretize, int(*) (const void *user_data) user_bcs, int(*) (const Matrix< double > &x, Matrix< double >
&res, const void *user_data) user_res, int(*) (const Matrix< double > &b, Matrix< double > &p, const void
*user_data) user_precon, int(*) (const void *user_data) user_postprocess, int(*) (const void *user_data) user_reset,
FINCH_DATA * dat, const void * param_data )
```

```
5.10.2.34 int uAverage ( FINCH_DATA * dat )
```

```
5.10.2.35 int uTotal ( FINCH_DATA * dat )
```

```
5.10.2.36 int vanAlbada_discretization ( const void * user_data )
```

5.11 flock.h File Reference

```
#include "macaw.h"
#include "egret.h"
#include "finch.h"
#include "lark.h"
#include "skua.h"
#include "scopsowl.h"
#include "gsta_opt.h"
#include "magpie.h"
#include "skua_opt.h"
#include "scopsowl_opt.h"
#include "yaml_wrapper.h"
```

5.12 gsta_opt.cpp File Reference

```
#include "gsta_opt.h"
```

Functions

- int [roundIt](#) (double d)
- int [twoFifths](#) (int m)
- int [orderMag](#) (double x)
- int [minValue](#) (std::vector< int > array)
- int [minIndex](#) (std::vector< double > array)
- int [avgPar](#) (std::vector< int > array)
- double [avgValue](#) (std::vector< double > array)
- double [weightedAvg](#) (double *enorm, double *x, int n)
- double [rSq](#) (double *x, double *y, double slope, double vint, int m_dat)
- bool [isSmooth](#) (double *par, void *data)
- void [orthoLinReg](#) (double *x, double *y, double *par, int m_dat, int n_par)
- void [eduGuess](#) (double *P, double *q, double *par, int k, int m_dat, void *data)
- double [gstaFunc](#) (double p, const double *K, double qmax, int n_par)
- double [gstaObjFunc](#) (double *t, double *y, double *par, int m_dat, void *data)
- void [eval_GSTA](#) (const double *par, int m_dat, const void *data, double *fvec, int *info)
- int [gsta_optimize](#) (const char *fileName)

5.12.1 Function Documentation

5.12.1.1 `int avgPar (std::vector< int > array)`

5.12.1.2 `double avgValue (std::vector< double > array)`

5.12.1.3 `void eduGuess (double * P, double * q, double * par, int k, int m_dat, void * data)`

5.12.1.4 `void eval_GSTA (const double * par, int m_dat, const void * data, double * fvec, int * info)`

5.12.1.5 `int gsta_optimize (const char * fileName)`

5.12.1.6 `double gstaFunc (double p, const double * K, double qmax, int n_par)`

5.12.1.7 `double gstaObjFunc (double * t, double * y, double * par, int m_dat, void * data)`

5.12.1.8 `bool isSmooth (double * par, void * data)`

5.12.1.9 `int minIndex (std::vector< double > array)`

5.12.1.10 `int minValue (std::vector< int > array)`

5.12.1.11 `int orderMag (double x)`

5.12.1.12 `void orthoLinReg (double * x, double * y, double * par, int m_dat, int n_par)`

5.12.1.13 `int roundIt (double d)`

5.12.1.14 `double rSq (double * x, double * y, double slope, double vint, int m_dat)`

5.12.1.15 `int twoFifths (int m)`

5.12.1.16 `double weightedAvg (double * enorm, double * x, int n)`

5.13 gsta_opt.h File Reference

```
#include "lmcurve.h"
#include <stdio.h>
#include <math.h>
#include <iostream>
#include <fstream>
#include <stdlib.h>
#include <vector>
#include <time.h>
#include <float.h>
#include <string>
#include "error.h"
```

Classes

- struct [GSTA_OPT_DATA](#)

Macros

- `#define Po 100.0`
- `#define R 8.3144621`
- `#define Na 6.0221413E+23`

Functions

- `void error ()`
- `int roundIt (double d)`
- `int twoFifths (int m)`
- `int orderMag (double x)`
- `int minValue (std::vector< int > array)`
- `int minIndex (std::vector< double > array)`
- `int avgPar (std::vector< int > array)`
- `double avgValue (std::vector< double > array)`
- `double weightedAvg (double *enorm, double *x, int n)`
- `double rSq (double *x, double *y, double slope, double vint, int m_dat)`
- `bool isSmooth (double *par, void *data)`
- `void orthoLinReg (double *x, double *y, double *par, int m_dat, int n_par)`
- `void eduGuess (double *P, double *q, double *par, int k, int m_dat, void *data)`
- `double gstaFunc (double p, const double *K, double qmax, int n_par)`
- `double gstaObjFunc (double *t, double *y, double *par, int m_dat, void *data)`
- `void eval_GSTA (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `int gsta_optimize (const char *fileName)`

5.13.1 Macro Definition Documentation

5.13.1.1 `#define Na 6.0221413E+23`

5.13.1.2 `#define Po 100.0`

5.13.1.3 `#define R 8.3144621`

5.13.2 Function Documentation

5.13.2.1 `int avgPar (std::vector< int > array)`

5.13.2.2 `double avgValue (std::vector< double > array)`

5.13.2.3 `void eduGuess (double * P, double * q, double * par, int k, int m_dat, void * data)`

5.13.2.4 `void error ()`

5.13.2.5 `void eval_GSTA (const double * par, int m_dat, const void * data, double * fvec, int * info)`

5.13.2.6 `int gsta_optimize (const char * fileName)`

5.13.2.7 `double gstaFunc (double p, const double * K, double qmax, int n_par)`

5.13.2.8 `double gstaObjFunc (double * t, double * y, double * par, int m_dat, void * data)`

5.13.2.9 `bool isSmooth (double * par, void * data)`

- 5.13.2.10 int minIndex (std::vector< double > array)
- 5.13.2.11 int minValue (std::vector< int > array)
- 5.13.2.12 int orderMag (double x)
- 5.13.2.13 void orthoLinReg (double * x, double * y, double * par, int m_dat, int n_par)
- 5.13.2.14 int roundIt (double d)
- 5.13.2.15 double rSq (double * x, double * y, double slope, double vint, int m_dat)
- 5.13.2.16 int twoFifths (int m)
- 5.13.2.17 double weightedAvg (double * enorm, double * x, int n)

5.14 lark.cpp File Reference

Linear Algebra Residual Kernels.

```
#include "lark.h"
```

Functions

- int [update_arnoldi_solution](#) (Matrix< double > &x, Matrix< double > &x0, ARNOLDI_DATA *arnoldi_dat)
- int [arnoldi](#) (int(*matvec)(const Matrix< double > &v, Matrix< double > &w, const void *data), int(*precon)(const Matrix< double > &b, Matrix< double > &p, const void *data), Matrix< double > &r0, ARNOLDI_DATA *arnoldi_dat, const void *matvec_data, const void *precon_data)
- int [gmresLeftPreconditioned](#) (int(*matvec)(const Matrix< double > &v, Matrix< double > &w, const void *data), int(*precon)(const Matrix< double > &b, Matrix< double > &P, const void *data), Matrix< double > &b, GMRESLP_DATA *gmreslp_dat, const void *matvec_data, const void *precon_data)
- int [fom](#) (int(*matvec)(const Matrix< double > &v, Matrix< double > &w, const void *data), int(*precon)(const Matrix< double > &b, Matrix< double > &P, const void *data), Matrix< double > &b, GMRESLP_DATA *gmreslp_dat, const void *matvec_data, const void *precon_data)
- int [gmresRightPreconditioned](#) (int(*matvec)(const Matrix< double > &v, Matrix< double > &w, const void *data), int(*precon)(const Matrix< double > &b, Matrix< double > &p, const void *data), Matrix< double > &b, GMRESRP_DATA *gmresrp_dat, const void *matvec_data, const void *precon_data)
- int [pcg](#) (int(*matvec)(const Matrix< double > &p, Matrix< double > &Ap, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &z, const void *data), Matrix< double > &b, PCG_DATA *pcg_dat, const void *matvec_data, const void *precon_data)
- int [bicgstab](#) (int(*matvec)(const Matrix< double > &p, Matrix< double > &Ap, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &z, const void *data), Matrix< double > &b, BiCGSTAB_DATA *bicg_dat, const void *matvec_data, const void *precon_data)
- int [cgs](#) (int(*matvec)(const Matrix< double > &p, Matrix< double > &Ap, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &z, const void *data), Matrix< double > &b, CGS_DATA *cgs_dat, const void *matvec_data, const void *precon_data)
- int [operatorTranspose](#) (int(*matvec)(const Matrix< double > &v, Matrix< double > &Av, const void *data), Matrix< double > &r, Matrix< double > &u, OPTTRANS_DATA *transpose_dat, const void *matvec_data)
- int [gcr](#) (int(*matvec)(const Matrix< double > &x, Matrix< double > &Ax, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &Mr, const void *data), Matrix< double > &b, GCR_DATA *gcr_dat, const void *matvec_data, const void *precon_data)
- int [gmresPreconditioner](#) (const Matrix< double > &r, Matrix< double > &Mr, const void *data)
- int [gmresr](#) (int(*matvec)(const Matrix< double > &x, Matrix< double > &Ax, const void *data), int(*terminal_precon)(const Matrix< double > &r, Matrix< double > &Mr, const void *data), Matrix< double > &b, GMRESR_DATA *gmresr_dat, const void *matvec_data, const void *term_precon_data)

- `int picard (int(*res)(const Matrix< double > &x, Matrix< double > &r, const void *data), int(*evalx)(const Matrix< double > &x0, Matrix< double > &x, const void *data), Matrix< double > &x, PICARD_DATA *picard_dat, const void *res_data, const void *evalx_data)`
- `int jacvec (const Matrix< double > &v, Matrix< double > &Jv, const void *data)`
- `int backtrackLineSearch (int(*feval)(const Matrix< double > &x, Matrix< double > &F, const void *data), Matrix< double > &Fkp1, Matrix< double > &xkp1, Matrix< double > &pk, double normFk, BACKTRACK_DATA *backtrack_dat, const void *feval_data)`
- `int pjfnk (int(*res)(const Matrix< double > &x, Matrix< double > &F, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &p, const void *data), Matrix< double > &x, PJFNK_DATA *pjfnk_dat, const void *res_data, const void *precon_data)`
- `int NumericalJacobian (int(*Func)(const Matrix< double > &x, Matrix< double > &F, const void *user_data), const Matrix< double > &x, Matrix< double > &J, int Nx, int Nf, NUM_JAC_DATA *jac_dat, const void *user_data)`
- `int LARK_TESTS ()`

5.14.1 Detailed Description

Linear Algebra Residual Kernels. [lark.h](#)

Author

Austin Ladshaw

Version

0.0 beta

Date

10/14/2014

Copyright

This software was designed and built at the Georgia Institute of Technology by Austin Ladshaw for PhD research in the area of adsorption and surface science. Copyright (c) 2015, all rights reserved.

5.14.2 Function Documentation

- 5.14.2.1 `int arnoldi (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &p, const void *data) precon, Matrix< double > &r0, ARNOLDI_DATA * arnoldi_dat, const void * matvec_data, const void * precon_data)`
- 5.14.2.2 `int backtrackLineSearch (int(*) (const Matrix< double > &x, Matrix< double > &F, const void *data) feval, Matrix< double > &Fkp1, Matrix< double > &xkp1, Matrix< double > &pk, double normFk, BACKTRACK_DATA * backtrack_dat, const void * feval_data)`
- 5.14.2.3 `int bicgstab (int(*) (const Matrix< double > &p, Matrix< double > &Ap, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &z, const void *data) precon, Matrix< double > &b, BiCGSTAB_DATA * bicg_dat, const void * matvec_data, const void * precon_data)`
- 5.14.2.4 `int cgs (int(*) (const Matrix< double > &p, Matrix< double > &Ap, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &z, const void *data) precon, Matrix< double > &b, CGS_DATA * cgs_dat, const void * matvec_data, const void * precon_data)`

- 5.14.2.5 `int fom (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &P, const void *data) precon, Matrix< double > & b, GMRESLP_DATA * gmreslp_dat, const void * matvec_data, const void * precon_data)`
- 5.14.2.6 `int gcr (int(*) (const Matrix< double > &x, Matrix< double > &Ax, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &Mr, const void *data) precon, Matrix< double > & b, GCR_DATA * gcr_dat, const void * matvec_data, const void * precon_data)`
- 5.14.2.7 `int gmresLeftPreconditioned (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &P, const void *data) precon, Matrix< double > & b, GMRESLP_DATA * gmreslp_dat, const void * matvec_data, const void * precon_data)`
- 5.14.2.8 `int gmresPreconditioner (const Matrix< double > & r, Matrix< double > & Mr, const void * data)`
- 5.14.2.9 `int gmresr (int(*) (const Matrix< double > &x, Matrix< double > &Ax, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &Mr, const void *data) terminal_precon, Matrix< double > & b, GMRESR_DATA * gmresr_dat, const void * matvec_data, const void * term_precon_data)`
- 5.14.2.10 `int gmresRightPreconditioned (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &p, const void *data) precon, Matrix< double > & b, GMRESRP_DATA * gmresrp_dat, const void * matvec_data, const void * precon_data)`
- 5.14.2.11 `int jacvec (const Matrix< double > & v, Matrix< double > & Jv, const void * data)`
- 5.14.2.12 `int LARK_TESTS ()`
- 5.14.2.13 `int NumericalJacobian (int(*) (const Matrix< double > &x, Matrix< double > &F, const void *user_data) Func, const Matrix< double > & x, Matrix< double > & J, int Nx, int Nf, NUM_JAC_DATA * jac_dat, const void * user_data)`
- 5.14.2.14 `int operatorTranspose (int(*) (const Matrix< double > &v, Matrix< double > &Av, const void *data) matvec, Matrix< double > & r, Matrix< double > & u, OPTRANS_DATA * transpose_dat, const void * matvec_data)`
- 5.14.2.15 `int pcg (int(*) (const Matrix< double > &p, Matrix< double > &Ap, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &z, const void *data) precon, Matrix< double > & b, PCG_DATA * pcg_dat, const void * matvec_data, const void * precon_data)`
- 5.14.2.16 `int picard (int(*) (const Matrix< double > &x, Matrix< double > &r, const void *data) res, int(*) (const Matrix< double > &x0, Matrix< double > &x, const void *data) evalx, Matrix< double > & x, PICARD_DATA * picard_dat, const void * res_data, const void * evalx_data)`
- 5.14.2.17 `int pjfnk (int(*) (const Matrix< double > &x, Matrix< double > &F, const void *data) res, int(*) (const Matrix< double > &r, Matrix< double > &p, const void *data) precon, Matrix< double > & x, PJFNK_DATA * pjfnk_dat, const void * res_data, const void * precon_data)`
- 5.14.2.18 `int update_arnoldi_solution (Matrix< double > & x, Matrix< double > & x0, ARNOLDI_DATA * arnoldi_dat)`

5.15 lark.h File Reference

Linear Algebra Residual Kernels.

```
#include "macaw.h"
#include <float.h>
```

Classes

- struct [ARNOLDI_DATA](#)
Data structure for the construction of the Krylov subspaces for a linear system.
- struct [GMRESLP_DATA](#)
Data structure for implementation of the Restarted GMRES algorithm with Left Preconditioning.
- struct [GMRESRP_DATA](#)
Data structure for the Restarted GMRES algorithm with Right Preconditioning.
- struct [PCG_DATA](#)
Data structure for implementation of the PCG algorithms for symmetric linear systems.
- struct [BiCGSTAB_DATA](#)
Data structure for the implementation of the BiCGSTAB algorithm for non-symmetric linear systems.
- struct [CGS_DATA](#)
Data structure for the implementation of the CGS algorithm for non-symmetric linear systems.
- struct [OPTRANS_DATA](#)
Data structure for implementation of linear operator transposition.
- struct [GCR_DATA](#)
Data structure for the implementation of the GCR algorithm for non-symmetric linear systems.
- struct [GMRESR_DATA](#)
Data structure for the implementation of GCR with Nested GMRES preconditioning (i.e., GMRESR)
- struct [PICARD_DATA](#)
Data structure for the implementation of a Picard or Fixed-Point iteration for non-linear systems.
- struct [BACKTRACK_DATA](#)
Data structure for the implementation of Backtracking Linesearch.
- struct [PJFNK_DATA](#)
Data structure for the implementation of the PJFNK algorithm for non-linear systems.
- struct [NUM_JAC_DATA](#)
Data structure to form a numerical jacobian matrix with finite differences.

Enumerations

- enum [krylov_method](#) {
 [GMRESLP](#), [PCG](#), [BiCGSTAB](#), [CGS](#),
 [FOM](#), [GMRESRP](#), [GCR](#), [GMRESR](#) }
Enum of definitions for linear solver types in PJFNK.

Functions

- int [update_arnoldi_solution](#) ([Matrix](#)< double > &x, [Matrix](#)< double > &x0, [ARNOLDI_DATA](#) *arnoldi_dat)
- int [arnoldi](#) (int(*matvec)(const [Matrix](#)< double > &v, [Matrix](#)< double > &w, const void *data), int(*precon)(const [Matrix](#)< double > &b, [Matrix](#)< double > &p, const void *data), [Matrix](#)< double > &r0, [ARNOLDI_DATA](#) *arnoldi_dat, const void *matvec_data, const void *precon_data)
- int [gmresLeftPreconditioned](#) (int(*matvec)(const [Matrix](#)< double > &v, [Matrix](#)< double > &w, const void *data), int(*precon)(const [Matrix](#)< double > &b, [Matrix](#)< double > &p, const void *data), [Matrix](#)< double > &b, [GMRESLP_DATA](#) *gmreslp_dat, const void *matvec_data, const void *precon_data)
- int [fom](#) (int(*matvec)(const [Matrix](#)< double > &v, [Matrix](#)< double > &w, const void *data), int(*precon)(const [Matrix](#)< double > &b, [Matrix](#)< double > &p, const void *data), [Matrix](#)< double > &b, [GMRESLP_DATA](#) *gmreslp_dat, const void *matvec_data, const void *precon_data)
- int [gmresRightPreconditioned](#) (int(*matvec)(const [Matrix](#)< double > &v, [Matrix](#)< double > &w, const void *data), int(*precon)(const [Matrix](#)< double > &b, [Matrix](#)< double > &p, const void *data), [Matrix](#)< double > &b, [GMRESRP_DATA](#) *gmresrp_dat, const void *matvec_data, const void *precon_data)

- `int pcg (int(*matvec)(const Matrix< double > &p, Matrix< double > &Ap, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &z, const void *data), Matrix< double > &b, PCG_DATA *pcg_dat, const void *matvec_data, const void *precon_data)`
- `int bicgstab (int(*matvec)(const Matrix< double > &p, Matrix< double > &Ap, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &z, const void *data), Matrix< double > &b, BiCGSTAB_DATA *bicg_dat, const void *matvec_data, const void *precon_data)`
- `int cgs (int(*matvec)(const Matrix< double > &p, Matrix< double > &Ap, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &z, const void *data), Matrix< double > &b, CGS_DATA *cgs_dat, const void *matvec_data, const void *precon_data)`
- `int operatorTranspose (int(*matvec)(const Matrix< double > &v, Matrix< double > &Av, const void *data), Matrix< double > &r, Matrix< double > &u, OPTRANS_DATA *transpose_dat, const void *matvec_data)`
- `int gcr (int(*matvec)(const Matrix< double > &x, Matrix< double > &Ax, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &Mr, const void *data), Matrix< double > &b, GCR_DATA *gcr_dat, const void *matvec_data, const void *precon_data)`
- `int gmresPreconditioner (const Matrix< double > &r, Matrix< double > &Mr, const void *data)`
- `int gmresr (int(*matvec)(const Matrix< double > &x, Matrix< double > &Ax, const void *data), int(*terminal_precon)(const Matrix< double > &r, Matrix< double > &Mr, const void *data), Matrix< double > &b, GMRESR_DATA *gmresr_dat, const void *matvec_data, const void *term_precon_data)`
- `int picard (int(*res)(const Matrix< double > &x, Matrix< double > &r, const void *data), int(*evalx)(const Matrix< double > &x0, Matrix< double > &x, const void *data), Matrix< double > &x, PICARD_DATA *picard_dat, const void *res_data, const void *evalx_data)`
- `int jacvec (const Matrix< double > &v, Matrix< double > &Jv, const void *data)`
- `int backtrackLineSearch (int(*feval)(const Matrix< double > &x, Matrix< double > &F, const void *data), Matrix< double > &Fkp1, Matrix< double > &xkp1, Matrix< double > &pk, double normFk, BACKTRACK_DATA *backtrack_dat, const void *feval_data)`
- `int pjfnk (int(*res)(const Matrix< double > &x, Matrix< double > &F, const void *data), int(*precon)(const Matrix< double > &r, Matrix< double > &p, const void *data), Matrix< double > &x, PJFNK_DATA *pjfnk_dat, const void *res_data, const void *precon_data)`
- `int NumericalJacobian (int(*Func)(const Matrix< double > &x, Matrix< double > &F, const void *user_data), const Matrix< double > &x, Matrix< double > &J, int Nx, int Nf, NUM_JAC_DATA *jac_dat, const void *user_data)`
- `int LARK_TESTS ()`

5.15.1 Detailed Description

Linear Algebra Residual Kernels. [lark.cpp](#)

The functions contained within are designed to solve generic linear and non-linear square systems of equations given a function argument and data from the user. Optionally, the user can also provide a function to return a preconditioning result that will be applied to the system.

Having the user define how the preconditioning is carried out provides two major advantages: (1) we do not need to store and large, sparse preconditioning matrices and instead only store the preconditioned vector result and (2) this allows the user to use any kind of preconditioner they see fit for their problem.

The Arnoldi function is typically not called by the user, but can be if desired. It accepts the function arguments and a residual vector to form an orthonormal basis of the Krylov subspace using the Modified Gram-Schmidt process (aka Arnoldi Iteration). This function is called by GMRES to iteratively solve a linear system of equations. Note that you can use this function to directly solve the linear system as long as that system is not too large. Construction of the basis is expensive, which is why this is used as a sub-function of an iterative method.

The Restarted GMRES function will accept function arguments for a linear system and attempt to solve said system iteratively by constructing an orthonormal basis from the Krylov function. Note that this GMRES function does support restarting and will use restarting by default if the linear system is too large.

Also included is a GMRES algorithm without restarting. This will directly solve the linear system within residual tolerance using a Full Orthogonal basis set of that system. It is equivalent to calling the Krylov method with the k parameter equal to N (i.e. the number of equations). This method is nick-named the Full Orthogonalization Method (FOM), although the true FOM algorithm in literature is slightly different.

The PJFNK function will accept function arguments for a square, non-linear system of equations and attempt to solve it iteratively using both the GMRES and Krylov functions with Newton's method to convert the non-linear system into a linear system.

Also built here is a PCG implementation for solving symmetric linear systems. Can also be called by PJFNK if we know that the linear system (i.e. the Jacobian) is symmetric. This algorithm is significantly more efficient than GMRES, but is only valid if the system of equations is symmetric.

Other linear solvers implemented in this work are the BiCGSTAB and CGS algorithms for non-symmetric, positive definite matrices. These algorithms are significantly more computationally efficient than GMRES or FOM. However, they can both break down if the linear system is poorly conditioned. In general, you only want to use these methods if you have preconditioning available and your linear system is very, very large. Otherwise, you will be better suited to using GMRES or FOM.

There is also an implementation of the Generalized Conjugate Residual (GCR) method with and without restarting. This is a GMRES-like method that should give the exact solution within N iterations, where N is the original size of the matrix. Built on top of the GCR method is a GMRESR (or GMRES Recursive) algorithm that uses GCR as the base method and performs GMRESRP iterations as a preconditioner at each iteration of GCR. This is the only linear solver that has built-in preconditioning. As a result, it may be slower than other algorithms for simple problems, but generally will have much better convergence behavior and will almost always give better residual reduction, even for hard to solve problems.

NOTE: There are three GMRES implementations: (i) gmresLP, (ii) fom, and (iii) gmresRP. GMRESLP is a restarted GMRES implementation that is left preconditioned and only checks the residual on the outer loops. This may be less efficient than GMRESRP, which can check both outer and inner loop residuals. However, GMRESRP has to use right preconditioning, which also slightly changes the convergence behavior of the linear system. GMRES with left preconditioning and without restarting will just build the full subspace by default, thus solving the system exactly, but may require too much memory. You can do a GMRESRP unrestarted by specifying that the restart parameter be equal to the size of the problem.

Author

Austin Ladshaw

Version

0.0 beta

Date

10/14/2014

Copyright

This software was designed and built at the Georgia Institute of Technology by Austin Ladshaw for PhD research in the area of adsorption and surface science. Copyright (c) 2015, all rights reserved.

5.15.2 Enumeration Type Documentation

5.15.2.1 enum krylov_method

Enum of definitions for linear solver types in PJFNK.

Enum delineates the available Krylov Subspace methods that can be used to solve the linear sub-problem at each non-linear iteration in a Newton method.

Enumerator

GMRESLP

PCG

BiCGSTAB***CGS******FOM******GMRESRP******GCR******GMRESR***

5.15.3 Function Documentation

- 5.15.3.1 `int arnoldi (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &p, const void *data) precon, Matrix< double > &r0, ARNOLDI_DATA * arnoldi_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.2 `int backtrackLineSearch (int(*) (const Matrix< double > &x, Matrix< double > &F, const void *data) feval, Matrix< double > &Fkp1, Matrix< double > &xkp1, Matrix< double > &pk, double normFk, BACKTRACK_DATA * backtrack_dat, const void * feval_data)`
- 5.15.3.3 `int bicgstab (int(*) (const Matrix< double > &p, Matrix< double > &Ap, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &z, const void *data) precon, Matrix< double > &b, BiCGSTAB_DATA * bicg_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.4 `int cgs (int(*) (const Matrix< double > &p, Matrix< double > &Ap, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &z, const void *data) precon, Matrix< double > &b, CGS_DATA * cgs_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.5 `int fom (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &p, const void *data) precon, Matrix< double > &b, GMRESLP_DATA * gmreslp_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.6 `int gcr (int(*) (const Matrix< double > &x, Matrix< double > &Ax, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &Mr, const void *data) precon, Matrix< double > &b, GCR_DATA * gcr_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.7 `int gmresLeftPreconditioned (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &p, const void *data) precon, Matrix< double > &b, GMRESLP_DATA * gmreslp_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.8 `int gmresPreconditioner (const Matrix< double > &r, Matrix< double > &Mr, const void * data)`
- 5.15.3.9 `int gmresr (int(*) (const Matrix< double > &x, Matrix< double > &Ax, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &Mr, const void *data) terminal_precon, Matrix< double > &b, GMRESR_DATA * gmresr_dat, const void * matvec_data, const void * term_precon_data)`
- 5.15.3.10 `int gmresRightPreconditioned (int(*) (const Matrix< double > &v, Matrix< double > &w, const void *data) matvec, int(*) (const Matrix< double > &b, Matrix< double > &p, const void *data) precon, Matrix< double > &b, GMRESRP_DATA * gmresrp_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.11 `int jacvec (const Matrix< double > &v, Matrix< double > &Jv, const void * data)`
- 5.15.3.12 `int LARK_TESTS ()`
- 5.15.3.13 `int NumericalJacobian (int(*) (const Matrix< double > &x, Matrix< double > &F, const void *user_data) Func, const Matrix< double > &x, Matrix< double > &J, int Nx, int Nf, NUM_JAC_DATA * jac_dat, const void * user_data)`

- 5.15.3.14 `int operatorTranspose (int(*) (const Matrix< double > &v, Matrix< double > &Av, const void *data) matvec, Matrix< double > &r, Matrix< double > &u, OPTRANS_DATA * transpose_dat, const void * matvec_data)`
- 5.15.3.15 `int pcg (int(*) (const Matrix< double > &p, Matrix< double > &Ap, const void *data) matvec, int(*) (const Matrix< double > &r, Matrix< double > &z, const void *data) precon, Matrix< double > &b, PCG_DATA * pcg_dat, const void * matvec_data, const void * precon_data)`
- 5.15.3.16 `int picard (int(*) (const Matrix< double > &x, Matrix< double > &r, const void *data) res, int(*) (const Matrix< double > &x0, Matrix< double > &x, const void *data) evalx, Matrix< double > &x, PICARD_DATA * picard_dat, const void * res_data, const void * evalx_data)`
- 5.15.3.17 `int pjfnk (int(*) (const Matrix< double > &x, Matrix< double > &F, const void *data) res, int(*) (const Matrix< double > &r, Matrix< double > &p, const void *data) precon, Matrix< double > &x, PJFNK_DATA * pjfnk_dat, const void * res_data, const void * precon_data)`
- 5.15.3.18 `int update_arnoldi_solution (Matrix< double > &x, Matrix< double > &x0, ARNOLDI_DATA * arnoldi_dat)`

5.16 macaw.cpp File Reference

```
#include "macaw.h"
```

Functions

- int [MACAW_TESTS](#) ()

5.16.1 Function Documentation

5.16.1.1 `int MACAW_TESTS ()`

5.17 macaw.h File Reference

```
#include <stdio.h>
#include <math.h>
#include <iostream>
#include <fstream>
#include <stdlib.h>
#include <vector>
#include <time.h>
#include <float.h>
#include <string>
#include <exception>
#include "error.h"
```

Classes

- class [Matrix< T >](#)

Macros

- `#define M_PI 3.14159265358979323846264338327950288 /* pi */`

Functions

- int [MACAW_TESTS](#) ()

5.17.1 Macro Definition Documentation

5.17.1.1 `#define M_PI 3.14159265358979323846264338327950288 /* pi */`

5.17.2 Function Documentation

5.17.2.1 int [MACAW_TESTS](#) ()

5.18 magpie.cpp File Reference

```
#include "magpie.h"
```

Functions

- double [qo](#) (double po, const void *data, int i)
- double [dq_dp](#) (double p, const void *data, int i)
- double [q_p](#) (double p, const void *data, int i)
- double [PI](#) (double po, const void *data, int i)
- double [eMax](#) (const void *data, int i)
- double [Qst](#) (double po, const void *data, int i)
- double [lnact_mSPD](#) (const double *par, const void *data, int i, volatile double [PI](#))
- double [grad_mSPD](#) (const double *par, const void *data, int i)
- double [qT](#) (const double *par, const void *data)
- void [initialGuess_mSPD](#) (double *par, const void *data)
- void [eval_po_PI](#) (const double *par, int m_dat, const void *data, double *fvec, int *info)
- void [eval_po_qo](#) (const double *par, int m_dat, const void *data, double *fvec, int *info)
- void [eval_po](#) (const double *par, int m_dat, const void *data, double *fvec, int *info)
- void [eval_eta](#) (const double *par, int m_dat, const void *data, double *fvec, int *info)
- void [eval_GPAST](#) (const double *par, int m_dat, const void *data, double *fvec, int *info)
- int [MAGPIE](#) (const void *data)
- int [MAGPIE_SCENARIOS](#) (const char *inputFileName, const char *sceneFileName)

5.18.1 Function Documentation

5.18.1.1 double [dq_dp](#) (double *p*, const void * *data*, int *i*)

5.18.1.2 double [eMax](#) (const void * *data*, int *i*)

5.18.1.3 void [eval_eta](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)

5.18.1.4 void [eval_GPAST](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)

5.18.1.5 void [eval_po](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)

5.18.1.6 void [eval_po_PI](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)

5.18.1.7 void [eval_po_qo](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)

5.18.1.8 `double grad_mSPD (const double * par, const void * data, int i)`

5.18.1.9 `void initialGuess_mSPD (double * par, const void * data)`

5.18.1.10 `double lnact_mSPD (const double * par, const void * data, int i, volatile double PI)`

5.18.1.11 `int MAGPIE (const void * data)`

5.18.1.12 `int MAGPIE_SCENARIOS (const char * inputFileName, const char * sceneFileName)`

5.18.1.13 `double PI (double po, const void * data, int i)`

5.18.1.14 `double q_p (double p, const void * data, int i)`

5.18.1.15 `double qo (double po, const void * data, int i)`

5.18.1.16 `double Qst (double po, const void * data, int i)`

5.18.1.17 `double qT (const double * par, const void * data)`

5.19 magpie.h File Reference

```
#include "lmcurve.h"
#include <stdio.h>
#include <math.h>
#include <iostream>
#include <fstream>
#include <stdlib.h>
#include <vector>
#include <time.h>
#include <float.h>
#include <string>
#include "error.h"
```

Classes

- struct [GSTA_DATA](#)
- struct [mSPD_DATA](#)
- struct [GPAST_DATA](#)
- struct [SYSTEM_DATA](#)
- struct [MAGPIE_DATA](#)

Macros

- `#define DBL_EPSILON 2.2204460492503131e-016`
- `#define Z 10.0`
- `#define A 3.13E+09`
- `#define V 18.92`
- `#define Po 100.0`
- `#define R 8.3144621`
- `#define Na 6.0221413E+23`
- `#define kB 1.3806488E-23`
- `#define shapeFactor(v_i) (((Z - 2) * v_i) / (Z * V)) + (2 / Z)`

- `#define InKo(H, S, T) -(H / (R * T)) + (S / R)`
- `#define He(qm, K1, m) (qm * K1) / (m * Po)`

Functions

- `double qo (double po, const void *data, int i)`
- `double dq_dp (double p, const void *data, int i)`
- `double q_p (double p, const void *data, int i)`
- `double PI (double po, const void *data, int i)`
- `double Qst (double po, const void *data, int i)`
- `double eMax (const void *data, int i)`
- `double lnact_mSPD (const double *par, const void *data, int i, volatile double PI)`
- `double grad_mSPD (const double *par, const void *data, int i)`
- `double qT (const double *par, const void *data)`
- `void initialGuess_mSPD (double *par, const void *data)`
- `void eval_po_PI (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `void eval_po_qo (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `void eval_po (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `void eval_eta (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `void eval_GPAST (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `int MAGPIE (const void *data)`
- `int MAGPIE_SCENARIOS (const char *inputFileName, const char *sceneFileName)`

5.19.1 Macro Definition Documentation

5.19.1.1 `#define A 3.13E+09`

5.19.1.2 `#define DBL_EPSILON 2.2204460492503131e-016`

5.19.1.3 `#define He(qm, K1, m) (qm * K1) / (m * Po)`

5.19.1.4 `#define kB 1.3806488E-23`

5.19.1.5 `#define InKo(H, S, T) -(H / (R * T)) + (S / R)`

5.19.1.6 `#define Na 6.0221413E+23`

5.19.1.7 `#define Po 100.0`

5.19.1.8 `#define R 8.3144621`

5.19.1.9 `#define shapeFactor(v_i) (((Z - 2) * v_i) / (Z * V)) + (2 / Z)`

5.19.1.10 `#define V 18.92`

5.19.1.11 `#define Z 10.0`

5.19.2 Function Documentation

5.19.2.1 `double dq_dp (double p, const void * data, int i)`

5.19.2.2 `double eMax (const void * data, int i)`

5.19.2.3 `void eval_eta (const double * par, int m_dat, const void * data, double * fvec, int * info)`

```

5.19.2.4 void eval_GPAST ( const double * par, int m_dat, const void * data, double * fvec, int * info )

5.19.2.5 void eval_po ( const double * par, int m_dat, const void * data, double * fvec, int * info )

5.19.2.6 void eval_po_PI ( const double * par, int m_dat, const void * data, double * fvec, int * info )

5.19.2.7 void eval_po_qo ( const double * par, int m_dat, const void * data, double * fvec, int * info )

5.19.2.8 double grad_mSPD ( const double * par, const void * data, int i )

5.19.2.9 void initialGuess_mSPD ( double * par, const void * data )

5.19.2.10 double lnact_mSPD ( const double * par, const void * data, int i, volatile double PI )

5.19.2.11 int MAGPIE ( const void * data )

5.19.2.12 int MAGPIE_SCENARIOS ( const char * inputFileName, const char * sceneFileName )

5.19.2.13 double PI ( double po, const void * data, int i )

5.19.2.14 double q_p ( double p, const void * data, int i )

5.19.2.15 double qo ( double po, const void * data, int i )

5.19.2.16 double Qst ( double po, const void * data, int i )

5.19.2.17 double qT ( const double * par, const void * data )

```

5.20 main.cpp File Reference

Main Function.

```
#include "ui.h"
```

Functions

- int [main](#) (int argc, const char *argv[])

5.20.1 Detailed Description

Main Function. User input provided at time of execution is used to call the ui functions

Author

Austin Ladshaw

Version

0.0 beta

Date

08/25/2015

Copyright

This software was designed and built at the Georgia Institute of Technology by Austin Ladshaw for PhD research in the area of adsorption and surface science. Copyright (c) 2015, all rights reserved.

5.20.2 Function Documentation

5.20.2.1 `int main (int argc, const char * argv[])`

Command Line Interface: To Override, comment out this line and replace with your own run function

5.21 mola.cpp File Reference

```
#include "mola.h"
```

Functions

- `int MOLA_TESTS ()`

5.21.1 Function Documentation

5.21.1.1 `int MOLA_TESTS ()`

5.22 mola.h File Reference

```
#include <ctype.h>
#include "eel.h"
```

Classes

- `class Molecule`

Functions

- `int MOLA_TESTS ()`

5.22.1 Function Documentation

5.22.1.1 `int MOLA_TESTS ()`

5.23 monkfish.cpp File Reference

```
#include "monkfish.h"
```

Functions

- double [default_porosity](#) (int *i*, int *l*, const void **user_data*)
- double [default_density](#) (int *i*, int *l*, const void **user_data*)
- double [default_interparticle_diffusion](#) (int *i*, int *l*, const void **user_data*)
- double [default_monk_adsorption](#) (int *i*, int *l*, const void **user_data*)
- double [default_monk_equilibrium](#) (int *i*, int *l*, const void **user_data*)
- double [default_monkfish_retardation](#) (int *i*, int *l*, const void **user_data*)
- double [default_exterior_concentration](#) (int *i*, const void **user_data*)
- double [default_film_transfer](#) (int *i*, const void **user_data*)
- int [MONKFISH_TESTS](#) ()

5.23.1 Function Documentation

5.23.1.1 double [default_density](#) (int *i*, int *l*, const void * *user_data*)

5.23.1.2 double [default_exterior_concentration](#) (int *i*, const void * *user_data*)

5.23.1.3 double [default_film_transfer](#) (int *i*, const void * *user_data*)

5.23.1.4 double [default_interparticle_diffusion](#) (int *i*, int *l*, const void * *user_data*)

5.23.1.5 double [default_monk_adsorption](#) (int *i*, int *l*, const void * *user_data*)

5.23.1.6 double [default_monk_equilibrium](#) (int *i*, int *l*, const void * *user_data*)

5.23.1.7 double [default_monkfish_retardation](#) (int *i*, int *l*, const void * *user_data*)

5.23.1.8 double [default_porosity](#) (int *i*, int *l*, const void * *user_data*)

5.23.1.9 int [MONKFISH_TESTS](#) ()

5.24 monkfish.h File Reference

```
#include "dogfish.h"
```

Classes

- struct [MONKFISH_PARAM](#)
- struct [MONKFISH_DATA](#)

Functions

- double [default_porosity](#) (int *i*, int *l*, const void **user_data*)
- double [default_density](#) (int *i*, int *l*, const void **user_data*)
- double [default_interparticle_diffusion](#) (int *i*, int *l*, const void **user_data*)
- double [default_monk_adsorption](#) (int *i*, int *l*, const void **user_data*)
- double [default_monk_equilibrium](#) (int *i*, int *l*, const void **user_data*)
- double [default_monkfish_retardation](#) (int *i*, int *l*, const void **user_data*)
- double [default_exterior_concentration](#) (int *i*, const void **user_data*)
- double [default_film_transfer](#) (int *i*, const void **user_data*)

- int [setup_MONKFISH_DATA](#) (FILE *file, double(*eval_porosity)(int i, int l, const void *user_data), double(*eval_density)(int i, int l, const void *user_data), double(*eval_ext_diff)(int i, int l, const void *user_data), double(*eval_adsorb)(int i, int l, const void *user_data), double(*eval_retard)(int i, int l, const void *user_data), double(*eval_ext_conc)(int i, const void *user_data), double(*eval_ext_film)(int i, const void *user_data), double(*dog_diffusion)(int i, int l, const void *user_data), double(*dog_ext_film)(int i, const void *user_data), double(*dog_surf_conc)(int i, const void *user_data), const void *user_data, [MONKFISH_DATA](#) *monk_dat)
- int [MONKFISH_TESTS](#) ()

5.24.1 Function Documentation

5.24.1.1 double default_density (int *i*, int *l*, const void * *user_data*)

5.24.1.2 double default_exterior_concentration (int *i*, const void * *user_data*)

5.24.1.3 double default_film_transfer (int *i*, const void * *user_data*)

5.24.1.4 double default_interparticle_diffusion (int *i*, int *l*, const void * *user_data*)

5.24.1.5 double default_monk_adsorption (int *i*, int *l*, const void * *user_data*)

5.24.1.6 double default_monk_equilibrium (int *i*, int *l*, const void * *user_data*)

5.24.1.7 double default_monkfish_retardation (int *i*, int *l*, const void * *user_data*)

5.24.1.8 double default_porosity (int *i*, int *l*, const void * *user_data*)

5.24.1.9 int [MONKFISH_TESTS](#) ()

5.24.1.10 int [setup_MONKFISH_DATA](#) (FILE * *file*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_porosity*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_density*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_ext_diff*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_adsorb*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_retard*, double(*) (int *i*, const void * *user_data*) *eval_ext_conc*, double(*) (int *i*, const void * *user_data*) *eval_ext_film*, double(*) (int *i*, int *l*, const void * *user_data*) *dog_diffusion*, double(*) (int *i*, const void * *user_data*) *dog_ext_film*, double(*) (int *i*, const void * *user_data*) *dog_surf_conc*, const void * *user_data*, [MONKFISH_DATA](#) * *monk_dat*)

5.25 sandbox.cpp File Reference

```
#include "sandbox.h"
```

Functions

- int [Speciation_Test01_Function](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *res_data)
- int [Speciation_Test01_Jacobian](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &J, const void *precon_data)
- int [Speciation_Test01_Guess](#) (const void *user_data)
- int [Speciation_Test01_MatVec](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &Ax, const void *matvec_data)
- int [RUN_SANDBOX](#) ()

5.25.1 Function Documentation

5.25.1.1 int [RUN_SANDBOX](#) ()

5.25.1.2 `int Speciation_Test01_Function (const Matrix< double > &x, Matrix< double > &F, const void * res_data)`

5.25.1.3 `int Speciation_Test01_Guess (const void * user_data)`

5.25.1.4 `int Speciation_Test01_Jacobian (const Matrix< double > &x, Matrix< double > &J, const void * precon_data)`

5.25.1.5 `int Speciation_Test01_MatVec (const Matrix< double > &x, Matrix< double > &Ax, const void * matvec_data)`

5.26 sandbox.h File Reference

```
#include "flock.h"
#include "school.h"
```

Classes

- struct [Speciation_Test01_Data](#)

Functions

- `int Speciation_Test01_Function (const Matrix< double > &x, Matrix< double > &F, const void *res_data)`
- `int Speciation_Test01_Jacobian (const Matrix< double > &x, Matrix< double > &J, const void *precon_data)`
- `int Speciation_Test01_Guess (const void *user_data)`
- `int Speciation_Test01_MatVec (const Matrix< double > &x, Matrix< double > &Ax, const void *matvec_data)`
- `int RUN_SANDBOX ()`

5.26.1 Function Documentation

5.26.1.1 `int RUN_SANDBOX ()`

5.26.1.2 `int Speciation_Test01_Function (const Matrix< double > &x, Matrix< double > &F, const void * res_data)`

5.26.1.3 `int Speciation_Test01_Guess (const void * user_data)`

5.26.1.4 `int Speciation_Test01_Jacobian (const Matrix< double > &x, Matrix< double > &J, const void * precon_data)`

5.26.1.5 `int Speciation_Test01_MatVec (const Matrix< double > &x, Matrix< double > &Ax, const void * matvec_data)`

5.27 school.h File Reference

```
#include "eel.h"
#include "mola.h"
#include "shark.h"
#include "dogfish.h"
#include "monkfish.h"
#include "yaml_wrapper.h"
```


5.28 scopsowl.cpp File Reference

```
#include "scopsowl.h"
```

Functions

- void [print2file_species_header](#) (FILE *Output, [SCOPSOWL_DATA](#) *owl_dat, int i)
- void [print2file_SCOPSOWL_time_header](#) (FILE *Output, [SCOPSOWL_DATA](#) *owl_dat, int i)
- void [print2file_SCOPSOWL_header](#) ([SCOPSOWL_DATA](#) *owl_dat)
- void [print2file_SCOPSOWL_result_old](#) ([SCOPSOWL_DATA](#) *owl_dat)
- void [print2file_SCOPSOWL_result_new](#) ([SCOPSOWL_DATA](#) *owl_dat)
- double [default_adsorption](#) (int i, int l, const void *user_data)
- double [default_retardation](#) (int i, int l, const void *user_data)
- double [default_pore_diffusion](#) (int i, int l, const void *user_data)
- double [default_surf_diffusion](#) (int i, int l, const void *user_data)
- double [default_effective_diffusion](#) (int i, int l, const void *user_data)
- double [const_pore_diffusion](#) (int i, int l, const void *user_data)
- double [default_filmMassTransfer](#) (int i, const void *user_data)
- double [const_filmMassTransfer](#) (int i, const void *user_data)
- int [setup_SCOPSOWL_DATA](#) (FILE *file, double(*eval_sorption)(int i, int l, const void *user_data), double(*eval_retardation)(int i, int l, const void *user_data), double(*eval_pore_diff)(int i, int l, const void *user_data), double(*eval_filmMT)(int i, const void *user_data), double(*eval_surface_diff)(int i, int l, const void *user_data), const void *user_data, [MIXED_GAS](#) *gas_data, [SCOPSOWL_DATA](#) *owl_data)
- int [SCOPSOWL_Executioner](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [set_SCOPSOWL_ICs](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [set_SCOPSOWL_timestep](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL_preprocesses](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [set_SCOPSOWL_params](#) (const void *user_data)
- int [SCOPSOWL_postprocesses](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL_reset](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [LARGE_CYCLE_TEST01](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SMALL_CYCLE_TEST02](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [CURVE_TEST03](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [CURVE_TEST04](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [CURVE_TEST05](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL_SCENARIOS](#) (const char *scene, const char *sorbent, const char *comp, const char *sorbate)
- int [SCOPSOWL_TESTS](#) ()

5.28.1 Function Documentation

5.28.1.1 [double const_filmMassTransfer](#) (int i, const void * user_data)

5.28.1.2 [double const_pore_diffusion](#) (int i, int l, const void * user_data)

5.28.1.3 [int CURVE_TEST03](#) ([SCOPSOWL_DATA](#) * owl_dat)

5.28.1.4 [int CURVE_TEST04](#) ([SCOPSOWL_DATA](#) * owl_dat)

5.28.1.5 [int CURVE_TEST05](#) ([SCOPSOWL_DATA](#) * owl_dat)

- 5.28.1.6 double default_adsorption (int *i*, int *l*, const void * *user_data*)
- 5.28.1.7 double default_effective_diffusion (int *i*, int *l*, const void * *user_data*)
- 5.28.1.8 double default_filmMassTransfer (int *i*, const void * *user_data*)
- 5.28.1.9 double default_pore_diffusion (int *i*, int *l*, const void * *user_data*)
- 5.28.1.10 double default_retardation (int *i*, int *l*, const void * *user_data*)
- 5.28.1.11 double default_surf_diffusion (int *i*, int *l*, const void * *user_data*)
- 5.28.1.12 int LARGE_CYCLE_TEST01 (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.13 void print2file_SCOPSOWL_header (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.14 void print2file_SCOPSOWL_result_new (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.15 void print2file_SCOPSOWL_result_old (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.16 void print2file_SCOPSOWL_time_header (FILE * *Output*, SCOPSOWL_DATA * *owl_dat*, int *i*)
- 5.28.1.17 void print2file_species_header (FILE * *Output*, SCOPSOWL_DATA * *owl_dat*, int *i*)
- 5.28.1.18 int SCOPSOWL (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.19 int SCOPSOWL_Executioner (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.20 int SCOPSOWL_postprocesses (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.21 int SCOPSOWL_preprocesses (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.22 int SCOPSOWL_reset (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.23 int SCOPSOWL_SCENARIOS (const char * *scene*, const char * *sorbent*, const char * *comp*, const char * *sorbate*)
- 5.28.1.24 int SCOPSOWL_TESTS ()
- 5.28.1.25 int set_SCOPSOWL_ICs (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.26 int set_SCOPSOWL_params (const void * *user_data*)
- 5.28.1.27 int set_SCOPSOWL_timestep (SCOPSOWL_DATA * *owl_dat*)
- 5.28.1.28 int setup_SCOPSOWL_DATA (FILE * *file*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_sorption*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_retardation*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_pore_diff*, double(*) (int *i*, const void * *user_data*) *eval_filmMT*, double(*) (int *i*, int *l*, const void * *user_data*) *eval_surface_diff*, const void * *user_data*, MIXED_GAS * *gas_data*, SCOPSOWL_DATA * *owl_data*)
- 5.28.1.29 int SMALL_CYCLE_TEST02 (SCOPSOWL_DATA * *owl_dat*)

5.29 scopsowl.h File Reference

```
#include "egret.h"
#include "skua.h"
```

Classes

- struct [SCOPSOWL_PARAM_DATA](#)
- struct [SCOPSOWL_DATA](#)

Macros

- #define [SCOPSOWL_HPP_](#)
- #define [Dp](#)(Dm, ep) (ep*ep*Dm)
- #define [Dk](#)(rp, T, MW) (9700.0*rp*pow((T/MW),0.5))
- #define [avgDp](#)(Dp, Dk) (pow((((1/Dp)+(1/Dk)),-1.0))

Functions

- void [print2file_species_header](#) (FILE *Output, [SCOPSOWL_DATA](#) *owl_dat, int i)
- void [print2file_SCOPSOWL_time_header](#) (FILE *Output, [SCOPSOWL_DATA](#) *owl_dat, int i)
- void [print2file_SCOPSOWL_header](#) ([SCOPSOWL_DATA](#) *owl_dat)
- void [print2file_SCOPSOWL_result_old](#) ([SCOPSOWL_DATA](#) *owl_dat)
- void [print2file_SCOPSOWL_result_new](#) ([SCOPSOWL_DATA](#) *owl_dat)
- double [default_adsorption](#) (int i, int l, const void *user_data)
- double [default_retardation](#) (int i, int l, const void *user_data)
- double [default_pore_diffusion](#) (int i, int l, const void *user_data)
- double [default_surf_diffusion](#) (int i, int l, const void *user_data)
- double [default_effective_diffusion](#) (int i, int l, const void *user_data)
- double [const_pore_diffusion](#) (int i, int l, const void *user_data)
- double [default_filmMassTransfer](#) (int i, const void *user_data)
- double [const_filmMassTransfer](#) (int i, const void *user_data)
- int [setup_SCOPSOWL_DATA](#) (FILE *file, double(*eval_sorption)(int i, int l, const void *user_data), double(*eval_retardation)(int i, int l, const void *user_data), double(*eval_pore_diff)(int i, int l, const void *user_data), double(*eval_filmMT)(int i, const void *user_data), double(*eval_surface_diff)(int i, int l, const void *user_data), const void *user_data, [MIXED_GAS](#) *gas_data, [SCOPSOWL_DATA](#) *owl_data)
- int [SCOPSOWL_Executioner](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [set_SCOPSOWL_ICs](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [set_SCOPSOWL_timestep](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL_preprocesses](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [set_SCOPSOWL_params](#) (const void *user_data)
- int [SCOPSOWL_postprocesses](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL_reset](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [LARGE_CYCLE_TEST01](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SMALL_CYCLE_TEST02](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [CURVE_TEST03](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [CURVE_TEST04](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [CURVE_TEST05](#) ([SCOPSOWL_DATA](#) *owl_dat)
- int [SCOPSOWL_SCENARIOS](#) (const char *scene, const char *sorbent, const char *comp, const char *sorbate)
- int [SCOPSOWL_TESTS](#) ()

5.29.1 Macro Definition Documentation

5.29.1.1 `#define avgDp(Dp, Dk) (pow(((1/Dp)+(1/Dk)),-1.0))`

5.29.1.2 `#define Dk(rp, T, MW) (9700.0*rp*pow((T/MW),0.5))`

5.29.1.3 `#define Dp(Dm, ep) (ep*ep*Dm)`

5.29.1.4 `#define SCOPSOWL_HPP_`

5.29.2 Function Documentation

5.29.2.1 `double const_filmMassTransfer (int i, const void * user_data)`

5.29.2.2 `double const_pore_diffusion (int i, int l, const void * user_data)`

5.29.2.3 `int CURVE_TEST03 (SCOPSOWL_DATA * owl_dat)`

5.29.2.4 `int CURVE_TEST04 (SCOPSOWL_DATA * owl_dat)`

5.29.2.5 `int CURVE_TEST05 (SCOPSOWL_DATA * owl_dat)`

5.29.2.6 `double default_adsorption (int i, int l, const void * user_data)`

5.29.2.7 `double default_effective_diffusion (int i, int l, const void * user_data)`

5.29.2.8 `double default_filmMassTransfer (int i, const void * user_data)`

5.29.2.9 `double default_pore_diffusion (int i, int l, const void * user_data)`

5.29.2.10 `double default_retardation (int i, int l, const void * user_data)`

5.29.2.11 `double default_surf_diffusion (int i, int l, const void * user_data)`

5.29.2.12 `int LARGE_CYCLE_TEST01 (SCOPSOWL_DATA * owl_dat)`

5.29.2.13 `void print2file_SCOPSOWL_header (SCOPSOWL_DATA * owl_dat)`

5.29.2.14 `void print2file_SCOPSOWL_result_new (SCOPSOWL_DATA * owl_dat)`

5.29.2.15 `void print2file_SCOPSOWL_result_old (SCOPSOWL_DATA * owl_dat)`

5.29.2.16 `void print2file_SCOPSOWL_time_header (FILE * Output, SCOPSOWL_DATA * owl_dat, int i)`

5.29.2.17 `void print2file_species_header (FILE * Output, SCOPSOWL_DATA * owl_dat, int i)`

5.29.2.18 `int SCOPSOWL (SCOPSOWL_DATA * owl_dat)`

5.29.2.19 `int SCOPSOWL_Executioner (SCOPSOWL_DATA * owl_dat)`

5.29.2.20 `int SCOPSOWL_postprocesses (SCOPSOWL_DATA * owl_dat)`

5.29.2.21 `int SCOPSOWL_preprocesses (SCOPSOWL_DATA * owl_dat)`

5.29.2.22 `int SCOPSOWL_reset (SCOPSOWL_DATA * owl_dat)`

- 5.29.2.23 int SCOPSOWL_SCENARIOS (const char * *scene*, const char * *sorbent*, const char * *comp*, const char * *sorbate*)
- 5.29.2.24 int SCOPSOWL_TESTS ()
- 5.29.2.25 int set_SCOPSOWL_ICs (SCOPSOWL_DATA * *owl_dat*)
- 5.29.2.26 int set_SCOPSOWL_params (const void * *user_data*)
- 5.29.2.27 int set_SCOPSOWL_timestep (SCOPSOWL_DATA * *owl_dat*)
- 5.29.2.28 int setup_SCOPSOWL_DATA (FILE * *file*, double(*) (int i, int l, const void * *user_data*) *eval_sorption*, double(*) (int i, int l, const void * *user_data*) *eval_retardation*, double(*) (int i, int l, const void * *user_data*) *eval_pore_diff*, double(*) (int i, const void * *user_data*) *eval_filmMT*, double(*) (int i, int l, const void * *user_data*) *eval_surface_diff*, const void * *user_data*, MIXED_GAS * *gas_data*, SCOPSOWL_DATA * *owl_data*)
- 5.29.2.29 int SMALL_CYCLE_TEST02 (SCOPSOWL_DATA * *owl_dat*)

5.30 scopsowl_opt.cpp File Reference

```
#include "scopsowl_opt.h"
```

Functions

- int [SCOPSOWL_OPT_set_y](#) (SCOPSOWL_OPT_DATA * *owl_opt*)
- int [initial_guess_SCOPSOWL](#) (SCOPSOWL_OPT_DATA * *owl_opt*)
- void [eval_SCOPSOWL_Uptake](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)
- int [SCOPSOWL_OPTIMIZE](#) (const char * *scene*, const char * *sorbent*, const char * *comp*, const char * *sorbate*, const char * *data*)

5.30.1 Function Documentation

- 5.30.1.1 void [eval_SCOPSOWL_Uptake](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)
- 5.30.1.2 int [initial_guess_SCOPSOWL](#) (SCOPSOWL_OPT_DATA * *owl_opt*)
- 5.30.1.3 int [SCOPSOWL_OPT_set_y](#) (SCOPSOWL_OPT_DATA * *owl_opt*)
- 5.30.1.4 int [SCOPSOWL_OPTIMIZE](#) (const char * *scene*, const char * *sorbent*, const char * *comp*, const char * *sorbate*, const char * *data*)

5.31 scopsowl_opt.h File Reference

```
#include "scopsowl.h"
```

Classes

- struct [SCOPSOWL_OPT_DATA](#)

Functions

- int [SCOPSOWL_OPT_set_y](#) ([SCOPSOWL_OPT_DATA](#) *owl_opt)
- int [initial_guess_SCOPSOWL](#) ([SCOPSOWL_OPT_DATA](#) *owl_opt)
- void [eval_SCOPSOWL_Uptake](#) (const double *par, int m_dat, const void *data, double *fvec, int *info)
- int [SCOPSOWL_OPTIMIZE](#) (const char *scene, const char *sorbent, const char *comp, const char *sorbate, const char *data)

5.31.1 Function Documentation

5.31.1.1 void [eval_SCOPSOWL_Uptake](#) (const double * *par*, int *m_dat*, const void * *data*, double * *fvec*, int * *info*)

5.31.1.2 int [initial_guess_SCOPSOWL](#) ([SCOPSOWL_OPT_DATA](#) * *owl_opt*)

5.31.1.3 int [SCOPSOWL_OPT_set_y](#) ([SCOPSOWL_OPT_DATA](#) * *owl_opt*)

5.31.1.4 int [SCOPSOWL_OPTIMIZE](#) (const char * *scene*, const char * *sorbent*, const char * *comp*, const char * *sorbate*, const char * *data*)

5.32 shark.cpp File Reference

```
#include "shark.h"
```

Functions

- void [print2file_shark_info](#) ([SHARK_DATA](#) *shark_dat)
- void [print2file_shark_header](#) ([SHARK_DATA](#) *shark_dat)
- void [print2file_shark_results_new](#) ([SHARK_DATA](#) *shark_dat)
- void [print2file_shark_results_old](#) ([SHARK_DATA](#) *shark_dat)
- int [ideal_solution](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *data)
- int [Davies_equation](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *data)
- int [DebyeHuckel_equation](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *data)
- int [DaviesLadshaw_equation](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *data)
- int [act_choice](#) (const std::string &input)
- bool [linsearch_choice](#) (const std::string &input)
- int [linearsolve_choice](#) (const std::string &input)
- int [Convert2LogConcentration](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &logx)
- int [Convert2Concentration](#) (const [Matrix](#)< double > &logx, [Matrix](#)< double > &x)
- int [read_scenario](#) ([SHARK_DATA](#) *shark_dat)
- int [read_options](#) ([SHARK_DATA](#) *shark_dat)
- int [read_species](#) ([SHARK_DATA](#) *shark_dat)
- int [read_massbalance](#) ([SHARK_DATA](#) *shark_dat)
- int [read_equilrxn](#) ([SHARK_DATA](#) *shark_dat)
- int [read_unsteadyrxn](#) ([SHARK_DATA](#) *shark_dat)
- int [setup_SHARK_DATA](#) (FILE *file, int(*residual)(const [Matrix](#)< double > &x, [Matrix](#)< double > &res, const void *data), int(*activity)(const [Matrix](#)< double > &x, [Matrix](#)< double > &gama, const void *data), int(*precond)(const [Matrix](#)< double > &r, [Matrix](#)< double > &p, const void *data), [SHARK_DATA](#) *dat, const void *activity_data, const void *residual_data, const void *precon_data, const void *other_data)
- int [shark_add_customResidual](#) (int i, double(*other_res)(const [Matrix](#)< double > &x, [SHARK_DATA](#) *shark_dat, const void *other_data), [SHARK_DATA](#) *shark_dat)
- int [shark_parameter_check](#) ([SHARK_DATA](#) *shark_dat)
- int [shark_energy_calculations](#) ([SHARK_DATA](#) *shark_dat)
- int [shark_temperature_calculations](#) ([SHARK_DATA](#) *shark_dat)

- int [shark_ph_finder](#) (SHARK_DATA *shark_dat)
- int [shark_guess](#) (SHARK_DATA *shark_dat)
- int [shark_initial_conditions](#) (SHARK_DATA *shark_dat)
- int [shark_executioner](#) (SHARK_DATA *shark_dat)
- int [shark_timestep_const](#) (SHARK_DATA *shark_dat)
- int [shark_timestep_adapt](#) (SHARK_DATA *shark_dat)
- int [shark_preprocesses](#) (SHARK_DATA *shark_dat)
- int [shark_solver](#) (SHARK_DATA *shark_dat)
- int [shark_postprocesses](#) (SHARK_DATA *shark_dat)
- int [shark_reset](#) (SHARK_DATA *shark_dat)
- int [shark_residual](#) (const Matrix< double > &x, Matrix< double > &F, const void *data)
- int [SHARK](#) (SHARK_DATA *shark_dat)
- int [SHARK_SCENARIO](#) (const char *yaml_input)
- int [SHARK_TESTS](#) ()

5.32.1 Function Documentation

- 5.32.1.1 int [act_choice](#) (const std::string & *input*)
- 5.32.1.2 int [Convert2Concentration](#) (const Matrix< double > & *logx*, Matrix< double > & *x*)
- 5.32.1.3 int [Convert2LogConcentration](#) (const Matrix< double > & *x*, Matrix< double > & *logx*)
- 5.32.1.4 int [Davies.equation](#) (const Matrix< double > & *x*, Matrix< double > & *F*, const void * *data*)
- 5.32.1.5 int [DaviesLadshaw.equation](#) (const Matrix< double > & *x*, Matrix< double > & *F*, const void * *data*)
- 5.32.1.6 int [DebyeHuckel.equation](#) (const Matrix< double > & *x*, Matrix< double > & *F*, const void * *data*)
- 5.32.1.7 int [ideal_solution](#) (const Matrix< double > & *x*, Matrix< double > & *F*, const void * *data*)
- 5.32.1.8 int [linearsolve_choice](#) (const std::string & *input*)
- 5.32.1.9 bool [linsearch_choice](#) (const std::string & *input*)
- 5.32.1.10 void [print2file_shark_header](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.11 void [print2file_shark_info](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.12 void [print2file_shark_results_new](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.13 void [print2file_shark_results_old](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.14 int [read_equilrxn](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.15 int [read_massbalance](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.16 int [read_options](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.17 int [read_scenario](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.18 int [read_species](#) (SHARK_DATA * *shark.dat*)
- 5.32.1.19 int [read_unsteadyrxn](#) (SHARK_DATA * *shark.dat*)

- 5.32.1.20 `int setup_SHARK_DATA (FILE * file, int(*) (const Matrix< double > &x, Matrix< double > &res, const void *data) residual, int(*) (const Matrix< double > &x, Matrix< double > &gama, const void *data) activity, int(*) (const Matrix< double > &r, Matrix< double > &p, const void *data) precond, SHARK_DATA * dat, const void * activity_data, const void * residual_data, const void * precon_data, const void * other_data)`
- 5.32.1.21 `int SHARK (SHARK_DATA * shark_dat)`
- 5.32.1.22 `int shark_add_customResidual (int i, double(*) (const Matrix< double > &x, SHARK_DATA * shark_dat, const void * other_data) other_res, SHARK_DATA * shark_dat)`
- 5.32.1.23 `int shark_energy_calculations (SHARK_DATA * shark_dat)`
- 5.32.1.24 `int shark_executioner (SHARK_DATA * shark_dat)`
- 5.32.1.25 `int shark_guess (SHARK_DATA * shark_dat)`
- 5.32.1.26 `int shark_initial_conditions (SHARK_DATA * shark_dat)`
- 5.32.1.27 `int shark_parameter_check (SHARK_DATA * shark_dat)`
- 5.32.1.28 `int shark_pH_finder (SHARK_DATA * shark_dat)`
- 5.32.1.29 `int shark_postprocesses (SHARK_DATA * shark_dat)`
- 5.32.1.30 `int shark_preprocesses (SHARK_DATA * shark_dat)`
- 5.32.1.31 `int shark_reset (SHARK_DATA * shark_dat)`
- 5.32.1.32 `int shark_residual (const Matrix< double > & x, Matrix< double > & F, const void * data)`
- 5.32.1.33 `int SHARK_SCENARIO (const char * yaml_input)`
- 5.32.1.34 `int shark_solver (SHARK_DATA * shark_dat)`
- 5.32.1.35 `int shark_temperature_calculations (SHARK_DATA * shark_dat)`
- 5.32.1.36 `int SHARK_TESTS ()`
- 5.32.1.37 `int shark_timestep_adapt (SHARK_DATA * shark_dat)`
- 5.32.1.38 `int shark_timestep_const (SHARK_DATA * shark_dat)`

5.33 shark.h File Reference

```
#include "mola.h"
#include "macaw.h"
#include "lark.h"
#include "yaml_wrapper.h"
```

Classes

- class [MasterSpeciesList](#)
- class [Reaction](#)
- class [MassBalance](#)

- class [UnsteadyReaction](#)
- class [Mechanism](#)
- class [Precipitation](#)
- class [UnsteadyPrecipitation](#)
- struct [SHARK_DATA](#)

Macros

- `#define Rstd 8.3144621`

Typedefs

- `typedef struct SHARK_DATA SHARK_DATA`

Enumerations

- `enum valid_act {
 IDEAL, DAVIES, DEBYE_HUCKEL, DAVIES_LADSHAW,
 SIT, PITZER }`

Functions

- `void print2file_shark_info (SHARK_DATA *shark_dat)`
- `void print2file_shark_header (SHARK_DATA *shark_dat)`
- `void print2file_shark_results_new (SHARK_DATA *shark_dat)`
- `void print2file_shark_results_old (SHARK_DATA *shark_dat)`
- `int ideal_solution (const Matrix< double > &x, Matrix< double > &F, const void *data)`
- `int Davies_equation (const Matrix< double > &x, Matrix< double > &F, const void *data)`
- `int DebyeHuckel_equation (const Matrix< double > &x, Matrix< double > &F, const void *data)`
- `int DaviesLadshaw_equation (const Matrix< double > &x, Matrix< double > &F, const void *data)`
- `int act_choice (const std::string &input)`
- `bool linesearch_choice (const std::string &input)`
- `int linearsolve_choice (const std::string &input)`
- `int Convert2LogConcentration (const Matrix< double > &x, Matrix< double > &logx)`
- `int Convert2Concentration (const Matrix< double > &logx, Matrix< double > &x)`
- `int read_scenario (SHARK_DATA *shark_dat)`
- `int read_options (SHARK_DATA *shark_dat)`
- `int read_species (SHARK_DATA *shark_dat)`
- `int read_massbalance (SHARK_DATA *shark_dat)`
- `int read_equilrxn (SHARK_DATA *shark_dat)`
- `int read_unsteadyrxn (SHARK_DATA *shark_dat)`
- `int setup_SHARK_DATA (FILE *file, int(*residual)(const Matrix< double > &x, Matrix< double > &res, const void *data), int(*activity)(const Matrix< double > &x, Matrix< double > &gama, const void *data), int(*precond)(const Matrix< double > &r, Matrix< double > &p, const void *data), SHARK_DATA *dat, const void *activity_data, const void *residual_data, const void *precon_data, const void *other_data)`
- `int shark_add_customResidual (int i, double(*other_res)(const Matrix< double > &x, SHARK_DATA *shark_dat, const void *other_data), SHARK_DATA *shark_dat)`
- `int shark_parameter_check (SHARK_DATA *shark_dat)`
- `int shark_energy_calculations (SHARK_DATA *shark_dat)`
- `int shark_temperature_calculations (SHARK_DATA *shark_dat)`
- `int shark_pH_finder (SHARK_DATA *shark_dat)`
- `int shark_guess (SHARK_DATA *shark_dat)`
- `int shark_initial_conditions (SHARK_DATA *shark_dat)`

- int [shark_executioner](#) (SHARK_DATA *shark_dat)
- int [shark_timestep_const](#) (SHARK_DATA *shark_dat)
- int [shark_timestep_adapt](#) (SHARK_DATA *shark_dat)
- int [shark_preprocesses](#) (SHARK_DATA *shark_dat)
- int [shark_solver](#) (SHARK_DATA *shark_dat)
- int [shark_postprocesses](#) (SHARK_DATA *shark_dat)
- int [shark_reset](#) (SHARK_DATA *shark_dat)
- int [shark_residual](#) (const [Matrix](#)< double > &x, [Matrix](#)< double > &F, const void *data)
- int [SHARK](#) (SHARK_DATA *shark_dat)
- int [SHARK_SCENARIO](#) (const char *yaml_input)
- int [SHARK_TESTS](#) ()

5.33.1 Macro Definition Documentation

5.33.1.1 `#define Rstd 8.3144621`

5.33.2 Typedef Documentation

5.33.2.1 `typedef struct SHARK_DATA SHARK_DATA`

5.33.3 Enumeration Type Documentation

5.33.3.1 `enum valid_act`

Enumerator

IDEAL

DAVIES

DEBYE_HUCKEL

DAVIES_LADSHAW

SIT

PITZER

5.33.4 Function Documentation

5.33.4.1 `int act_choice (const std::string & input)`

5.33.4.2 `int Convert2Concentration (const Matrix< double > &logx, Matrix< double > &x)`

5.33.4.3 `int Convert2LogConcentration (const Matrix< double > &x, Matrix< double > &logx)`

5.33.4.4 `int Davies.equation (const Matrix< double > &x, Matrix< double > &F, const void * data)`

5.33.4.5 `int DaviesLadshaw.equation (const Matrix< double > &x, Matrix< double > &F, const void * data)`

5.33.4.6 `int DebyeHuckel.equation (const Matrix< double > &x, Matrix< double > &F, const void * data)`

5.33.4.7 `int ideal_solution (const Matrix< double > &x, Matrix< double > &F, const void * data)`

5.33.4.8 `int linearsolve_choice (const std::string & input)`

5.33.4.9 `bool linesearch_choice (const std::string & input)`

- 5.33.4.10 void print2file_shark_header (SHARK_DATA * *shark.dat*)
- 5.33.4.11 void print2file_shark_info (SHARK_DATA * *shark.dat*)
- 5.33.4.12 void print2file_shark_results_new (SHARK_DATA * *shark.dat*)
- 5.33.4.13 void print2file_shark_results_old (SHARK_DATA * *shark.dat*)
- 5.33.4.14 int read_equilrxn (SHARK_DATA * *shark.dat*)
- 5.33.4.15 int read_massbalance (SHARK_DATA * *shark.dat*)
- 5.33.4.16 int read_options (SHARK_DATA * *shark.dat*)
- 5.33.4.17 int read_scenario (SHARK_DATA * *shark.dat*)
- 5.33.4.18 int read_species (SHARK_DATA * *shark.dat*)
- 5.33.4.19 int read_unsteadyrxn (SHARK_DATA * *shark.dat*)
- 5.33.4.20 int setup_SHARK_DATA (FILE * *file*, int(*) (const Matrix< double > &x, Matrix< double > &res, const void *data) *residual*, int(*) (const Matrix< double > &x, Matrix< double > &gama, const void *data) *activity*, int(*) (const Matrix< double > &r, Matrix< double > &p, const void *data) *precond*, SHARK_DATA * *dat*, const void * *activity_data*, const void * *residual_data*, const void * *precon_data*, const void * *other_data*)
- 5.33.4.21 int SHARK (SHARK_DATA * *shark.dat*)
- 5.33.4.22 int shark_add_customResidual (int *i*, double(*) (const Matrix< double > &x, SHARK_DATA * *shark.dat*, const void **other_data*) *other_res*, SHARK_DATA * *shark.dat*)
- 5.33.4.23 int shark_energy_calculations (SHARK_DATA * *shark.dat*)
- 5.33.4.24 int shark_executioner (SHARK_DATA * *shark.dat*)
- 5.33.4.25 int shark_guess (SHARK_DATA * *shark.dat*)
- 5.33.4.26 int shark_initial_conditions (SHARK_DATA * *shark.dat*)
- 5.33.4.27 int shark_parameter_check (SHARK_DATA * *shark.dat*)
- 5.33.4.28 int shark_pH_finder (SHARK_DATA * *shark.dat*)
- 5.33.4.29 int shark_postprocesses (SHARK_DATA * *shark.dat*)
- 5.33.4.30 int shark_preprocesses (SHARK_DATA * *shark.dat*)
- 5.33.4.31 int shark_reset (SHARK_DATA * *shark.dat*)
- 5.33.4.32 int shark_residual (const Matrix< double > &x, Matrix< double > &F, const void * *data*)
- 5.33.4.33 int SHARK_SCENARIO (const char * *yaml.input*)
- 5.33.4.34 int shark_solver (SHARK_DATA * *shark.dat*)
- 5.33.4.35 int shark_temperature_calculations (SHARK_DATA * *shark.dat*)

5.33.4.36 int SHARK_TESTS ()

5.33.4.37 int shark_timestep_adapt (SHARK_DATA * shark_dat)

5.33.4.38 int shark_timestep_const (SHARK_DATA * shark_dat)

5.34 skua.cpp File Reference

```
#include "skua.h"
```

Functions

- void [print2file_species_header](#) (FILE *Output, SKUA_DATA *skua_dat, int i)
- void [print2file_SKUA_time_header](#) (FILE *Output, SKUA_DATA *skua_dat, int i)
- void [print2file_SKUA_header](#) (SKUA_DATA *skua_dat)
- void [print2file_SKUA_results_old](#) (SKUA_DATA *skua_dat)
- void [print2file_SKUA_results_new](#) (SKUA_DATA *skua_dat)
- double [default_Dc](#) (int i, int l, const void *data)
- double [default_kf](#) (int i, const void *data)
- double [const_Dc](#) (int i, int l, const void *data)
- double [simple_darken_Dc](#) (int i, int l, const void *data)
- double [theoretical_darken_Dc](#) (int i, int l, const void *data)
- double [empirical_kf](#) (int i, const void *data)
- double [const_kf](#) (int i, const void *data)
- int [molefractionCheck](#) (SKUA_DATA *skua_dat)
- int [setup_SKUA_DATA](#) (FILE *file, double(*eval_Dc)(int i, int l, const void *user_data), double(*eval_Kf)(int i, const void *user_data), const void *user_data, MIXED_GAS *gas_data, SKUA_DATA *skua_dat)
- int [SKUA_Executioner](#) (SKUA_DATA *skua_dat)
- int [set_SKUA_ICs](#) (SKUA_DATA *skua_dat)
- int [set_SKUA_timestep](#) (SKUA_DATA *skua_dat)
- int [SKUA_preprocesses](#) (SKUA_DATA *skua_dat)
- int [set_SKUA_params](#) (const void *user_data)
- int [SKUA_postprocesses](#) (SKUA_DATA *skua_dat)
- int [SKUA_reset](#) (SKUA_DATA *skua_dat)
- int [SKUA](#) (SKUA_DATA *skua_dat)
- int [SKUA_CYCLE_TEST01](#) (SKUA_DATA *skua_dat)
- int [SKUA_CYCLE_TEST02](#) (SKUA_DATA *skua_dat)
- int [SKUA_LOW_TEST03](#) (SKUA_DATA *skua_dat)
- int [SKUA_MID_TEST04](#) (SKUA_DATA *skua_dat)
- int [SKUA_SCENARIOS](#) (const char *scene, const char *sorbent, const char *comp, const char *sorbate)
- int [SKUA_TESTS](#) ()

5.34.1 Function Documentation

5.34.1.1 double [const_Dc](#) (int *i*, int *l*, const void * *data*)

5.34.1.2 double [const_kf](#) (int *i*, const void * *data*)

5.34.1.3 double [default_Dc](#) (int *i*, int *l*, const void * *data*)

5.34.1.4 double [default_kf](#) (int *i*, const void * *data*)

- 5.34.1.5 `double empirical_kf (int i, const void * data)`
- 5.34.1.6 `int molefractionCheck (SKUA_DATA * skua_dat)`
- 5.34.1.7 `void print2file_SKUA_header (SKUA_DATA * skua_dat)`
- 5.34.1.8 `void print2file_SKUA_results_new (SKUA_DATA * skua_dat)`
- 5.34.1.9 `void print2file_SKUA_results_old (SKUA_DATA * skua_dat)`
- 5.34.1.10 `void print2file_SKUA_time_header (FILE * Output, SKUA_DATA * skua_dat, int i)`
- 5.34.1.11 `void print2file_species_header (FILE * Output, SKUA_DATA * skua_dat, int i)`
- 5.34.1.12 `int set_SKUA_ICs (SKUA_DATA * skua_dat)`
- 5.34.1.13 `int set_SKUA_params (const void * user_data)`
- 5.34.1.14 `int set_SKUA_timestep (SKUA_DATA * skua_dat)`
- 5.34.1.15 `int setup_SKUA_DATA (FILE * file, double(*) (int i, int l, const void * user_data) eval_Dc, double(*) (int i, const void * user_data) eval_Kf, const void * user_data, MIXED_GAS * gas_data, SKUA_DATA * skua_dat)`
- 5.34.1.16 `double simple_darken_Dc (int i, int l, const void * data)`
- 5.34.1.17 `int SKUA (SKUA_DATA * skua_dat)`
- 5.34.1.18 `int SKUA_CYCLE_TEST01 (SKUA_DATA * skua_dat)`
- 5.34.1.19 `int SKUA_CYCLE_TEST02 (SKUA_DATA * skua_dat)`
- 5.34.1.20 `int SKUA_Executioner (SKUA_DATA * skua_dat)`
- 5.34.1.21 `int SKUA_LOW_TEST03 (SKUA_DATA * skua_dat)`
- 5.34.1.22 `int SKUA_MID_TEST04 (SKUA_DATA * skua_dat)`
- 5.34.1.23 `int SKUA_postprocesses (SKUA_DATA * skua_dat)`
- 5.34.1.24 `int SKUA_preprocesses (SKUA_DATA * skua_dat)`
- 5.34.1.25 `int SKUA_reset (SKUA_DATA * skua_dat)`
- 5.34.1.26 `int SKUA_SCENARIOS (const char * scene, const char * sorbent, const char * comp, const char * sorbate)`
- 5.34.1.27 `int SKUA_TESTS ()`
- 5.34.1.28 `double theoretical_darken_Dc (int i, int l, const void * data)`

5.35 skua.h File Reference

```
#include "finch.h"
#include "magpie.h"
#include "egret.h"
```

Classes

- struct [SKUA_PARAM](#)
- struct [SKUA_DATA](#)

Macros

- `#define SKUA_HPP_`
- `#define D_inf(Dref, Tref, B, p, T) (Dref * pow(p+sqrt(DBL_EPSILON),(Tref/T)-B))`
- `#define D_o(Diff, E, T) (Diff * exp(-E/(Rstd*T)))`
- `#define D_c(Diff, phi) (Diff * (1.0/((1.0+1.1E-6)-phi)))`

Functions

- void [print2file_species_header](#) (FILE *Output, [SKUA_DATA](#) *skua_dat, int i)
- void [print2file_SKUA_time_header](#) (FILE *Output, [SKUA_DATA](#) *skua_dat, int i)
- void [print2file_SKUA_header](#) ([SKUA_DATA](#) *skua_dat)
- void [print2file_SKUA_results_old](#) ([SKUA_DATA](#) *skua_dat)
- void [print2file_SKUA_results_new](#) ([SKUA_DATA](#) *skua_dat)
- double [default_Dc](#) (int i, int l, const void *data)
- double [default_kf](#) (int i, const void *data)
- double [const_Dc](#) (int i, int l, const void *data)
- double [simple_darken_Dc](#) (int i, int l, const void *data)
- double [theoretical_darken_Dc](#) (int i, int l, const void *data)
- double [empirical_kf](#) (int i, const void *data)
- double [const_kf](#) (int i, const void *data)
- int [molefractionCheck](#) ([SKUA_DATA](#) *skua_dat)
- int [setup_SKUA_DATA](#) (FILE *file, double(*eval_Dc)(int i, int l, const void *user_data), double(*eval_Kf)(int i, const void *user_data), const void *user_data, [MIXED_GAS](#) *gas_data, [SKUA_DATA](#) *skua_dat)
- int [SKUA_Executioner](#) ([SKUA_DATA](#) *skua_dat)
- int [set_SKUA_ICs](#) ([SKUA_DATA](#) *skua_dat)
- int [set_SKUA_timestep](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA_preprocesses](#) ([SKUA_DATA](#) *skua_dat)
- int [set_SKUA_params](#) (const void *user_data)
- int [SKUA_postprocesses](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA_reset](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA_CYCLE_TEST01](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA_CYCLE_TEST02](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA_LOW_TEST03](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA_MID_TEST04](#) ([SKUA_DATA](#) *skua_dat)
- int [SKUA_SCENARIOS](#) (const char *scene, const char *sorbent, const char *comp, const char *sorbate)
- int [SKUA_TESTS](#) ()

5.35.1 Macro Definition Documentation

5.35.1.1 `#define D_c(Diff, phi) (Diff * (1.0/((1.0+1.1E-6)-phi)))`

5.35.1.2 `#define D_inf(Dref, Tref, B, p, T) (Dref * pow(p+sqrt(DBL_EPSILON),(Tref/T)-B))`

5.35.1.3 `#define D_o(Diff, E, T) (Diff * exp(-E/(Rstd*T)))`

5.35.1.4 `#define SKUA_HPP_`

5.35.2 Function Documentation

5.35.2.1 `double const_Dc (int i, int l, const void * data)`

5.35.2.2 `double const_kf (int i, const void * data)`

5.35.2.3 `double default_Dc (int i, int l, const void * data)`

5.35.2.4 `double default_kf (int i, const void * data)`

5.35.2.5 `double empirical_kf (int i, const void * data)`

5.35.2.6 `int molefractionCheck (SKUA_DATA * skua_dat)`

5.35.2.7 `void print2file_SKUA_header (SKUA_DATA * skua_dat)`

5.35.2.8 `void print2file_SKUA_results_new (SKUA_DATA * skua_dat)`

5.35.2.9 `void print2file_SKUA_results_old (SKUA_DATA * skua_dat)`

5.35.2.10 `void print2file_SKUA_time_header (FILE * Output, SKUA_DATA * skua_dat, int i)`

5.35.2.11 `void print2file_species_header (FILE * Output, SKUA_DATA * skua_dat, int i)`

5.35.2.12 `int set_SKUA_ICs (SKUA_DATA * skua_dat)`

5.35.2.13 `int set_SKUA_params (const void * user_data)`

5.35.2.14 `int set_SKUA_timestep (SKUA_DATA * skua_dat)`

5.35.2.15 `int setup_SKUA_DATA (FILE * file, double(*)(int i, int l, const void *user_data) eval_Dc, double(*)(int i, const void *user_data) eval_Kf, const void * user_data, MIXED_GAS * gas_data, SKUA_DATA * skua_dat)`

5.35.2.16 `double simple_darken_Dc (int i, int l, const void * data)`

5.35.2.17 `int SKUA (SKUA_DATA * skua_dat)`

5.35.2.18 `int SKUA_CYCLE_TEST01 (SKUA_DATA * skua_dat)`

5.35.2.19 `int SKUA_CYCLE_TEST02 (SKUA_DATA * skua_dat)`

5.35.2.20 `int SKUA_Executioner (SKUA_DATA * skua_dat)`

5.35.2.21 `int SKUA_LOW_TEST03 (SKUA_DATA * skua_dat)`

5.35.2.22 `int SKUA_MID_TEST04 (SKUA_DATA * skua_dat)`

5.35.2.23 `int SKUA_postprocesses (SKUA_DATA * skua_dat)`

5.35.2.24 `int SKUA_preprocesses (SKUA_DATA * skua_dat)`

5.35.2.25 `int SKUA_reset (SKUA_DATA * skua_dat)`

5.35.2.26 `int SKUA_SCENARIOS (const char * scene, const char * sorbent, const char * comp, const char * sorbate)`

5.35.2.27 `int SKUA_TESTS ()`

5.35.2.28 `double theoretical_darken_Dc (int i, int l, const void * data)`

5.36 skua_opt.cpp File Reference

```
#include "skua_opt.h"
```

Functions

- `int SKUA_OPT_set_y (SKUA_OPT_DATA *skua_opt)`
- `int initial_guess_SKUA (SKUA_OPT_DATA *skua_opt)`
- `void eval_SKUA_Uptake (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `int SKUA_OPTIMIZE (const char *scene, const char *sorbent, const char *comp, const char *sorbate, const char *data)`

5.36.1 Function Documentation

5.36.1.1 `void eval_SKUA_Uptake (const double * par, int m_dat, const void * data, double * fvec, int * info)`

5.36.1.2 `int initial_guess_SKUA (SKUA_OPT_DATA * skua_opt)`

5.36.1.3 `int SKUA_OPT_set_y (SKUA_OPT_DATA * skua_opt)`

5.36.1.4 `int SKUA_OPTIMIZE (const char * scene, const char * sorbent, const char * comp, const char * sorbate, const char * data)`

5.37 skua_opt.h File Reference

```
#include "skua.h"
```

Classes

- struct `SKUA_OPT_DATA`

Functions

- `int SKUA_OPT_set_y (SKUA_OPT_DATA *skua_opt)`
- `int initial_guess_SKUA (SKUA_OPT_DATA *skua_opt)`
- `void eval_SKUA_Uptake (const double *par, int m_dat, const void *data, double *fvec, int *info)`
- `int SKUA_OPTIMIZE (const char *scene, const char *sorbent, const char *comp, const char *sorbate, const char *data)`

5.37.1 Function Documentation

5.37.1.1 `void eval_SKUA_Uptake (const double * par, int m_dat, const void * data, double * fvec, int * info)`

5.37.1.2 `int initial_guess_SKUA (SKUA_OPT_DATA * skua_opt)`

5.37.1.3 `int SKUA_OPT_set_y (SKUA_OPT_DATA * skua_opt)`

5.37.1.4 `int SKUA_OPTIMIZE (const char * scene, const char * sorbent, const char * comp, const char * sorbate, const char * data)`

5.38 Trajectory.cpp File Reference

```
#include "Trajectory.h"
```

Functions

- `double Magnetic_R (const Matrix< double > &dX, const Matrix< double > &dY, int i, double b, double mu_0, double chi_p, double M, double H0, double a)`
- `double Magnetic_T (const Matrix< double > &dX, const Matrix< double > &dY, int i, double b, double mu_0, double chi_p, double M, double H0, double a)`
- `double Grav_R (const Matrix< double > &dX, int i, double b, double rho_p, double rho_f)`
- `double Grav_T (const Matrix< double > &dX, int i, double b, double rho_p, double rho_f)`
- `double Van_R (const Matrix< double > &dX, const Matrix< double > &dY, int i, double Hamaker, double b, double a)`
- `double V_RAD (const Matrix< double > &dX, const Matrix< double > &dY, int i, double V0, double rho_f, double a, double eta)`
- `double V_THETA (const Matrix< double > &dX, const Matrix< double > &dY, int i, double V0, double rho_f, double a, double eta)`
- `double Brown_RAD (double n_rand, double m_rand, double sigma_n, double sigma_m)`
- `double Brown_THETA (double s_rand, double t_rand, double sigma_n, double sigma_m)`
- `int POLAR (Matrix< double > &POL, const Matrix< double > &dX, const Matrix< double > &dY, const void *data, int i)`
- `double RADIAL_FORCE (const Matrix< double > &POL, double eta, double b, double mp, double t, double a)`
- `double TANGENTIAL_FORCE (const Matrix< double > &POL, const Matrix< double > &dY, double eta, double b, double mp, double t, double a, int i)`
- `int CARTESIAN (const Matrix< double > &POL, Matrix< double > &H, const Matrix< double > &dY, double i, const void *data)`
- `int DISPLACEMENT (Matrix< double > &dX, Matrix< double > &dY, const Matrix< double > &H, int i)`
- `int LOCATION (const Matrix< double > &dY, const Matrix< double > &dX, Matrix< double > &X, Matrix< double > &Y, int i)`
- `double Removal_Efficiency (double Sum_Cap, const void *data)`
- `int Trajectory_SetupConstants (TRAJECTORY_DATA *dat)`
- `int Number_Generator (TRAJECTORY_DATA *dat)`
- `int Run_Trajectory ()`

5.38.1 Function Documentation

5.38.1.1 `double Brown_RAD (double n_rand, double m_rand, double sigma_n, double sigma_m)`

5.38.1.2 `double Brown_THETA (double s_rand, double t_rand, double sigma_n, double sigma_m)`

5.38.1.3 `int CARTESIAN (const Matrix< double > & POL, Matrix< double > & H, const Matrix< double > & dY, double i, const void * data)`

5.38.1.4 `int DISPLACEMENT (Matrix< double > & dX, Matrix< double > & dY, const Matrix< double > & H, int i)`

5.38.1.5 `double Grav_R (const Matrix< double > & dX, int i, double b, double rho_p, double rho_f)`

- 5.38.1.6 `double Grav_T (const Matrix< double > & dX, int i, double b, double rho_p, double rho_f)`
- 5.38.1.7 `int LOCATION (const Matrix< double > & dY, const Matrix< double > & dX, Matrix< double > & X, Matrix< double > & Y, int i)`
- 5.38.1.8 `double Magnetic_R (const Matrix< double > & dX, const Matrix< double > & dY, int i, double b, double mu_0, double chi_p, double M, double H0, double a)`
- 5.38.1.9 `double Magnetic_T (const Matrix< double > & dX, const Matrix< double > & dY, int i, double b, double mu_0, double chi_p, double M, double H0, double a)`
- 5.38.1.10 `int Number_Generator (TRAJECTORY_DATA * dat)`
- 5.38.1.11 `int POLAR (Matrix< double > & POL, const Matrix< double > & dX, const Matrix< double > & dY, const void * data, int i)`
- 5.38.1.12 `double RADIAL_FORCE (const Matrix< double > & POL, double eta, double b, double mp, double t, double a)`
- 5.38.1.13 `double Removal_Efficiency (double Sum_Cap, const void * data)`
- 5.38.1.14 `int Run_Trajectory ()`
- 5.38.1.15 `double TANGENTIAL_FORCE (const Matrix< double > & POL, const Matrix< double > & dY, double eta, double b, double mp, double t, double a, int i)`
- 5.38.1.16 `int Trajectory_SetupConstants (TRAJECTORY_DATA * dat)`
- 5.38.1.17 `double V_RAD (const Matrix< double > & dX, const Matrix< double > & dY, int i, double V0, double rho_f, double a, double eta)`
- 5.38.1.18 `double V_THETA (const Matrix< double > & dX, const Matrix< double > & dY, int i, double V0, double rho_f, double a, double eta)`
- 5.38.1.19 `double Van_R (const Matrix< double > & dX, const Matrix< double > & dY, int i, double Hamaker, double b, double a)`

5.39 Trajectory.h File Reference

```
#include "macaw.h"
#include <random>
#include <chrono>
```

Classes

- struct [TRAJECTORY_DATA](#)

Functions

- double [Magnetic_R](#) (const [Matrix](#)< double > &dX, const [Matrix](#)< double > &dY, int i, double b, double mu_0, double chi_p, double M, double H0, double a)
- double [Magnetic_T](#) (const [Matrix](#)< double > &dX, const [Matrix](#)< double > &dY, int i, double b, double mu_0, double chi_p, double M, double H0, double a)
- double [Grav_R](#) (const [Matrix](#)< double > &dX, int i, double b, double rho_p, double rho_f)
- double [Grav_T](#) (const [Matrix](#)< double > &dX, int i, double b, double rho_p, double rho_f)

- double `Van_R` (const `Matrix< double >` &dX, const `Matrix< double >` &dY, int i, double Hamaker, double b, double a)
- double `V_RAD` (const `Matrix< double >` &dX, const `Matrix< double >` &dY, int i, double V0, double rho_f, double a, double eta)
- double `V_THETA` (const `Matrix< double >` &dX, const `Matrix< double >` &dY, int i, double V0, double rho_f, double a, double eta)
- double `Brown_RAD` (double n_rand, double m_rand, double sigma_n, double sigma_m)
- double `Brown_THETA` (double s_rand, double t_rand, double sigma_n, double sigma_m)
- int `POLAR` (`Matrix< double >` &POL, const `Matrix< double >` &dX, const `Matrix< double >` &dY, const void *data, int i)
- double `RADIAL_FORCE` (const `Matrix< double >` &POL, double eta, double b, double mp, double t, double a)
- double `TANGENTIAL_FORCE` (const `Matrix< double >` &POL, const `Matrix< double >` &dY, double eta, double b, double mp, double t, double a, int i)
- int `CARTESIAN` (const `Matrix< double >` &POL, `Matrix< double >` &H, const `Matrix< double >` &dY, double i, const void *data)
- int `DISPLACEMENT` (`Matrix< double >` &dX, `Matrix< double >` &dY, const `Matrix< double >` &H, int i)
- int `LOCATION` (const `Matrix< double >` &dY, const `Matrix< double >` &dX, `Matrix< double >` &X, `Matrix< double >` &Y, int i)
- double `Removal_Efficiency` (double Sum_Cap, const void *data)
- int `Trajectory_SetupConstants` (`TRAJECTORY_DATA` *dat)
- int `Number_Generator` (`TRAJECTORY_DATA` *dat)
- int `Run_Trajectory` ()

5.39.1 Function Documentation

5.39.1.1 double `Brown_RAD` (double *n_rand*, double *m_rand*, double *sigma_n*, double *sigma_m*)

5.39.1.2 double `Brown_THETA` (double *s_rand*, double *t_rand*, double *sigma_n*, double *sigma_m*)

5.39.1.3 int `CARTESIAN` (const `Matrix< double >` & *POL*, `Matrix< double >` & *H*, const `Matrix< double >` & *dY*, double *i*, const void * *data*)

5.39.1.4 int `DISPLACEMENT` (`Matrix< double >` & *dX*, `Matrix< double >` & *dY*, const `Matrix< double >` & *H*, int *i*)

5.39.1.5 double `Grav_R` (const `Matrix< double >` & *dX*, int *i*, double *b*, double *rho_p*, double *rho_f*)

5.39.1.6 double `Grav_T` (const `Matrix< double >` & *dX*, int *i*, double *b*, double *rho_p*, double *rho_f*)

5.39.1.7 int `LOCATION` (const `Matrix< double >` & *dY*, const `Matrix< double >` & *dX*, `Matrix< double >` & *X*, `Matrix< double >` & *Y*, int *i*)

5.39.1.8 double `Magnetic_R` (const `Matrix< double >` & *dX*, const `Matrix< double >` & *dY*, int *i*, double *b*, double *mu_0*, double *chi_p*, double *M*, double *H0*, double *a*)

5.39.1.9 double `Magnetic_T` (const `Matrix< double >` & *dX*, const `Matrix< double >` & *dY*, int *i*, double *b*, double *mu_0*, double *chi_p*, double *M*, double *H0*, double *a*)

5.39.1.10 int `Number_Generator` (`TRAJECTORY_DATA` * *dat*)

5.39.1.11 int `POLAR` (`Matrix< double >` & *POL*, const `Matrix< double >` & *dX*, const `Matrix< double >` & *dY*, const void * *data*, int *i*)

5.39.1.12 double `RADIAL_FORCE` (const `Matrix< double >` & *POL*, double *eta*, double *b*, double *mp*, double *t*, double *a*)

- 5.39.1.13 `double Removal_Efficiency (double Sum_Cap, const void * data)`
- 5.39.1.14 `int Run_Trajectory ()`
- 5.39.1.15 `double TANGENTIAL_FORCE (const Matrix< double > & POL, const Matrix< double > & dY, double eta, double b, double mp, double t, double a, int i)`
- 5.39.1.16 `int Trajectory_SetupConstants (TRAJECTORY_DATA * dat)`
- 5.39.1.17 `double V_RAD (const Matrix< double > & dX, const Matrix< double > & dY, int i, double V0, double rho_f, double a, double eta)`
- 5.39.1.18 `double V.THETA (const Matrix< double > & dX, const Matrix< double > & dY, int i, double V0, double rho_f, double a, double eta)`
- 5.39.1.19 `double Van_R (const Matrix< double > & dX, const Matrix< double > & dY, int i, double Hamaker, double b, double a)`

5.40 ui.cpp File Reference

User Interface for Ecosystem.

```
#include "ui.h"
```

Functions

- void `au_i_help` ()
Function to display help for Advanced User Interface.
- void `bui_help` ()
Function to display help for Basic User Interface.
- std::string `allLower` (const std::string &`input`)
Function to return an all lower case string based on the passed argument.
- bool `exit` (const std::string &`input`)
Function returns true if user requests exit.
- bool `help` (const std::string &`input`)
Function returns true if the user requests help.
- bool `version` (const std::string &`input`)
Function returns true if user requests to know the executable version.
- bool `test` (const std::string &`input`)
Function returns true if user requests to run a test.
- bool `exec` (const std::string &`input`)
Function returns true if the user requests to run a simulation/executable.
- bool `path` (const std::string &`input`)
Function returns true if the user indicates that input files share a common path.
- bool `input` (const std::string &`input`)
Function returns true if the user indicates that the next arguments are input files.
- bool `valid_test_string` (const std::string &`input`, UI_DATA *`ui_dat`)
Function returns true if the user gave a valid test option.
- bool `valid_exec_string` (const std::string &`input`, UI_DATA *`ui_dat`)
Function returns true if the user gave a valid execution option.
- int `number_files` (UI_DATA *`ui_dat`)
Function returns the number of expected input files for the user's run option.

- bool [valid_addon_options](#) (UI_DATA *ui_dat)
Function returns true if the user has chosen a valid additional runtime option.
- void [display_help](#) (UI_DATA *ui_dat)
Function to call the appropriate help menu based on type of interface.
- void [display_version](#) (UI_DATA *ui_dat)
Function to display ecosystem version information to the console.
- int [invalid_input](#) (int count, int max)
Function returns a CONTINUE or EXIT when invalid input is given.
- bool [valid_input_main](#) (UI_DATA *ui_dat)
Function returns true if user gave valid input in Basic UI.
- bool [valid_input_tests](#) (UI_DATA *ui_dat)
Function returns true if user gave a valid test function to run.
- bool [valid_input_execute](#) (UI_DATA *ui_dat)
Function returns true if user gave a valid executable function to run.
- int [test_loop](#) (UI_DATA *ui_dat)
Function that loops the Basic UI until a valid test option was selected.
- int [exec_loop](#) (UI_DATA *ui_dat)
Function that loops the Basic UI until a valid executable option was selected.
- int [run_test](#) (UI_DATA *ui_dat)
Function will call the user requested test function.
- int [run_exec](#) (UI_DATA *ui_dat)
Function will call the user requested executable function.
- int [run_executable](#) (int argc, const char *argv[])
Function called by the main and runs both user interfaces for the program.

5.40.1 Detailed Description

User Interface for Ecosystem. [ui.h](#)

Author

Austin Ladshaw

Version

0.0 beta

Date

08/25/2015

Copyright

This software was designed and built at the Georgia Institute of Technology by Austin Ladshaw for PhD research in the area of adsorption and surface science. Copyright (c) 2015, all rights reserved.

5.40.2 Function Documentation

5.40.2.1 `std::string allLower (const std::string & input)`

Function to return an all lower case string based on the passed argument.

This function will copy the input paramter and convert that copy to all lower case. The copy is then returned and can be checked against valid or allowed strings.

Parameters

<i>input</i>	string to copy and convert to lower case
--------------	--

5.40.2.2 void aui_help ()

Function to display help for Advanced User Interface.

The Advanced User Interface help screen is accessed by including run option -h or -help when executing the program from command line.

5.40.2.3 void bui_help ()

Function to display help for Basic User Interface.

The Basic User Interface help screen is accessed by running the executable, then typing "help" at any point during the console prompts. Exception to this occurs when the console prompts you to provide input files for your chosen routine. In this circumstance, the executable always assumes that what the user types in will be an input file.

5.40.2.4 void display_help (UI_DATA * ui_dat)

Function to call the appropriate help menu based on type of interface.

This function looks at the ui_dat structure and the user's OS files to determine what help menu to display and how to display it. There are two different types of help menus that can be displayed: (i) Advanced Help and (ii) Basic Help. Additionally, this function checks the OS file system for the existence of installed help files. If it finds those files, then it instructs the command terminal to read the contents of those files with the "less" command. Otherwise, it will just print the appropriate help menu to the console window.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.5 void display_version (UI_DATA * ui_dat)

Function to display ecosystem version information to the console.

This function will check the ui_dat structure to see which type of interface the user is using, then print out the version information for the executable being run.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.6 bool exec (const std::string & input)

Function returns true if the user requests to run a simulation/executable.

This function will check the input string for "-e" or "–execute" and determine whether or not the user requests to run an ecosystem executable function.

Parameters

<i>input</i>	input string the user gives to the console
--------------	--

5.40.2.7 `int exec_loop (UI_DATA * ui_dat)`

Function that loops the Basic UI until a valid executable option was selected.

This function loops the Basic UI menu for running an executable until a valid executable is selected by the user. If a valid executable is not selected, and the maximum number of loops has been reached, then this function will cause the program to force quit.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.8 `bool exit (const std::string & input)`

Function returns true if user requests exit.

This function will check the input string for "exit" or "quit" and terminate the executable. Only checked if using the Basic User Interface.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.40.2.9 `bool help (const std::string & input)`

Function returns true if the user requests help.

This function will check the input string for "help", "-h", or "–help" and will tell the executable to display the help menu. The help menu that gets displayed depends on how the executable was run to begin with.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.40.2.10 `bool input (const std::string & input)`

Function returns true if the user indicates that the next arguments are input files.

This function will check the input string for "-i" or "–input" and determine whether or not the user's next arguments are input files for a specific simulation. Only used in Advanced User Interface.

Parameters

<i>input</i>	input string the user gives to the console
--------------	--

5.40.2.11 `int invalid_input (int count, int max)`

Function returns a CONTINUE or EXIT when invalid input is given.

This function looks at the current count and the max iterations and determines whether or not to force the executable to terminate. If the user provides too many incorrect options during the Basic User Interface, then the executable will force quit.

Parameters

<i>count</i>	number of times the user has provided a bad option
<i>max</i>	maximum allowable bad options before force quit

5.40.2.12 int number_files (UI_DATA * ui_dat)

Function returns the number of expected input files for the user's run option.

This function will check the option variable in the ui_dat structure to determine the number of input files that is expected to be given. Running different executable functions in ecosystem may require various number of input files.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.13 bool path (const std::string & input)

Function returns true if the user indicates that input files share a common path.

This function will check the input string for "-p" or "–path" and determine whether or not the user will give a common path to all input files needed for the specified simulation. Only used in Advanced User Interface.

Parameters

<i>input</i>	input string the user gives to the console
--------------	--

5.40.2.14 int run_exec (UI_DATA * ui_dat)

Function will call the user requested executable function.

This function checks the option variable of the ui_dat structure and runs the corresponding executable function.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.15 int run_executable (int argc, const char * argv[])

Function called by the main and runs both user interfaces for the program.

This function is called in the [main.cpp](#) file and passes the console arguments given at run time.

Parameters

<i>argc</i>	number of arguments provided by the user at the time of execution
<i>argv</i>	list of C-strings that was provided by the user at the time of execution

5.40.2.16 int run_test (UI_DATA * ui_dat)

Function will call the user requested test function.

This function checks the option variable of the ui_dat structure and runs the corresponding test function.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.17 bool test (const std::string & input)

Function returns true if user requests to run a test.

This function will check the input string for "-t" or "–test" and determine whether or not the user requests to run an ecosystem test function.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.40.2.18 int test_loop (UI_DATA * ui_dat)

Function that loops the Basic UI until a valid test option was selected.

This function loops the Basic UI menu for running a test until a valid test is selected by the user. If a valid test is not selected, and the maximum number of loops has been reached, then this function will cause the program to force quit.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.19 bool valid_addon_options (UI_DATA * ui_dat)

Function returns true if the user has choosen a valid additional runtime option.

This function will check all additional input options in the user_input variable of ui_dat to determine if the user requests any additional options during runtime. Valid additional options are -p or -path and -i or -input.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.20 bool valid_exec_string (const std::string & input, UI_DATA * ui_dat)

Function returns true if the user gave a valid execution option.

This function will check the input string given by the user and determine whether that string denotes a valid execution option. Then, it will mark the option variable in ui_dat with the appropriate option from the valid_options enum.

Parameters

<i>input</i>	input string the user gives to the console
<i>ui_dat</i>	pointer to the data structure for the ui object

5.40.2.21 bool valid_input_execute (UI_DATA * ui_dat)

Function returns true if user gave a valid executable function to run.

This function checks the user_input argument of ui_dat for a valid executable option. If no valid executable was given, then this function returns false.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.22 `bool valid_input_main (UI_DATA * ui_dat)`

Function returns true if user gave valid input in Basic UI.

This function is only called if the user is running the Basic UI. It checks the given console argument stored in `user_input` of `ui_dat` for a valid option. If no valid option is given, then this function returns false.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.23 `bool valid_input_tests (UI_DATA * ui_dat)`

Function returns true if user gave a valid test function to run.

This function checks the `user_input` argument of `ui_dat` for a valid test option. If no valid test was given, then this function returns false.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.40.2.24 `bool valid_test_string (const std::string & input, UI_DATA * ui_dat)`

Function returns true if the user gave a valid test option.

This function will check the input string given by the user and determine whether that string denotes a valid test. Then, it will mark the option variable in `ui_dat` with the appropriate option from the `valid_options` enum.

Parameters

<i>input</i>	input string the user gives to the console
<i>ui_dat</i>	pointer to the data structure for the ui object

5.40.2.25 `bool version (const std::string & input)`

Function returns true if user requests to know the executable version.

This function will check the input string for "version", "-v", or "–version" and will tell the executable to display version information about the executable.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.41 ui.h File Reference

User Interface for Ecosystem.

```
#include <fstream>
#include <string>
#include <iostream>
#include "error.h"
#include "yaml_wrapper.h"
#include "flock.h"
#include "school.h"
#include "sandbox.h"
#include "Trajectory.h"
```

Classes

- struct [UI_DATA](#)
Data structure holding the UI arguments.

Macros

- #define [UI_HPP_](#)
- #define [ECO_VERSION](#) "0.0 alpha"
Macro expansion for executable current version number.
- #define [ECO_EXECUTABLE](#) "eco0"
Macro expansion for executable current name.

Enumerations

- enum [valid_options](#) {
 [TEST](#), [EXECUTE](#), [EXIT](#), [CONTINUE](#),
 [HELP](#), [dogfish](#), [eel](#), [egret](#),
 [finch](#), [lark](#), [macaw](#), [mola](#),
 [monkfish](#), [sandbox](#), [scopsowl](#), [shark](#),
 [skua](#), [gsta_opt](#), [magpie](#), [scops_opt](#),
 [skua_opt](#), [trajectory](#) }
Valid options available upon execution of the code.

Functions

- void [aui_help](#) ()
Function to display help for Advanced User Interface.
- void [bui_help](#) ()
Function to display help for Basic User Interface.
- std::string [allLower](#) (const std::string &[input](#))
Function to return an all lower case string based on the passed argument.
- bool [exit](#) (const std::string &[input](#))
Function returns true if user requests exit.
- bool [help](#) (const std::string &[input](#))
Function returns true if the user requests help.
- bool [version](#) (const std::string &[input](#))
Function returns true if user requests to know the executable version.
- bool [test](#) (const std::string &[input](#))
Function returns true if user requests to run a test.
- bool [exec](#) (const std::string &[input](#))

- Function returns true if the user requests to run a simulation/executable.*

 - bool [path](#) (const std::string &input)

Function returns true if the user indicates that input files share a common path.
- bool [input](#) (const std::string &input)

Function returns true if the user indicates that the next arguments are input files.
- bool [valid_test_string](#) (const std::string &input, UI_DATA *ui_dat)

Function returns true if the user gave a valid test option.
- bool [valid_exec_string](#) (const std::string &input, UI_DATA *ui_dat)

Function returns true if the user gave a valid execution option.
- int [number_files](#) (UI_DATA *ui_dat)

Function returns the number of expected input files for the user's run option.
- bool [valid_addon_options](#) (UI_DATA *ui_dat)

Function returns true if the user has chosen a valid additional runtime option.
- void [display_help](#) (UI_DATA *ui_dat)

Function to call the appropriate help menu based on type of interface.
- void [display_version](#) (UI_DATA *ui_dat)

Function to display ecosystem version information to the console.
- int [invalid_input](#) (int count, int max)

Function returns a CONTINUE or EXIT when invalid input is given.
- bool [valid_input_main](#) (UI_DATA *ui_dat)

Function returns true if user gave valid input in Basic UI.
- bool [valid_input_tests](#) (UI_DATA *ui_dat)

Function returns true if user gave a valid test function to run.
- bool [valid_input_execute](#) (UI_DATA *ui_dat)

Function returns true if user gave a valid executable function to run.
- int [test_loop](#) (UI_DATA *ui_dat)

Function that loops the Basic UI until a valid test option was selected.
- int [exec_loop](#) (UI_DATA *ui_dat)

Function that loops the Basic UI until a valid executable option was selected.
- int [run_test](#) (UI_DATA *ui_dat)

Function will call the user requested test function.
- int [run_exec](#) (UI_DATA *ui_dat)

Function will call the user requested executable function.
- int [run_executable](#) (int argc, const char *argv[])

Function called by the main and runs both user interfaces for the program.

5.41.1 Detailed Description

User Interface for Ecosystem. [ui.cpp](#)

These routines define how the user will interface with the software

Author

Austin Ladshaw

Version

0.0 beta

Date

08/25/2015

Copyright

This software was designed and built at the Georgia Institute of Technology by Austin Ladshaw for PhD research in the area of adsorption and surface science. Copyright (c) 2015, all rights reserved.

5.41.2 Macro Definition Documentation

5.41.2.1 `#define ECO_EXECUTABLE "eco0"`

Macro expansion for executable current name.

5.41.2.2 `#define ECO_VERSION "0.0 alpha"`

Macro expansion for executable current version number.

5.41.2.3 `#define UI_HPP_`

5.41.3 Enumeration Type Documentation

5.41.3.1 `enum valid_options`

Valid options available upon execution of the code.

Enumeration of valid options for executing the ecosystem code. More options become available as the code updates. Some options that appear here may not be viewable in the "help" screen of the executable. Those options are hidden, but are still valid entries.

Enumerator

TEST

EXECUTE

EXIT

CONTINUE

HELP

dogfish

eel

egret

finch

lark

macaw

mola

monkfish

sandbox

scopsowl

shark

skua

gsta_opt

magpie

scops_opt

skua_opt

trajectory

5.41.4 Function Documentation

5.41.4.1 `std::string allLower (const std::string & input)`

Function to return an all lower case string based on the passed argument.

This function will copy the input paramter and convert that copy to all lower case. The copy is then returned and can be checked against valid or allowed strings.

Parameters

<i>input</i>	string to copy and convert to lower case
--------------	--

5.41.4.2 `void aui_help ()`

Function to display help for Advanced User Interface.

The Advanved User Interface help screen is accessed by including run option -h or --help when executing the program from command line.

5.41.4.3 `void bui_help ()`

Function to display help for Basic User Interface.

The Basic User Interface help screen is accessed by running the executable, then typing "help" at any point during the console prompts. Exception to this occurs when the console prompts you to provide input files for your choosen routine. In this circumstance, the executable always assumes that what the user types in will be an input file.

5.41.4.4 `void display_help (UI_DATA * ui_dat)`

Function to call the appropriate help menu based on type of interface.

This function looks at the ui_dat structure and the user's OS files to determine what help menu to display and how to display it. There are two different types of help menus that can be displayed: (i) Advanced Help and (ii) Basic Help. Additionally, this function checks the OS file system for the existence of installed help files. If it finds those files, then it instructs the command terminal to read the contents of those files with the "less" command. Otherwise, it will just print the appropriate help menu to the console window.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.5 `void display_version (UI_DATA * ui_dat)`

Function to display ecosystem version information to the console.

This function will check the ui_dat structure to see which type of interface the user is using, then print out the version information for the executable being run.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.6 `bool exec (const std::string & input)`

Function returns true if the user requests to run a simulation/executable.

This function will check the input string for "-e" or "–execute" and determine whether or not the user requests to run an ecosystem executable function.

Parameters

<i>input</i>	input string the user gives to the console
--------------	--

5.41.4.7 int exec_loop (UI_DATA * ui_dat)

Function that loops the Basic UI until a valid executable option was selected.

This function loops the Basic UI menu for running an executable until a valid executable is selected by the user. If a valid executable is not selected, and the maximum number of loops has been reached, then this function will cause the program to force quit.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.8 bool exit (const std::string & input)

Function returns true if user requests exit.

This function will check the input string for "exit" or "quit" and terminate the executable. Only checked if using the Basic User Interface.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.41.4.9 bool help (const std::string & input)

Function returns true if the user requests help.

This function will check the input string for "help", "-h", or "–help" and will tell the executable to display the help menu. The help menu that gets displayed depends on how the executable was run to begin with.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.41.4.10 bool input (const std::string & input)

Function returns true if the user indicates that the next arguments are input files.

This function will check the input string for "-i" or "–input" and determine whether or not the user's next arguments are input files for a specific simulation. Only used in Advanced User Interface.

Parameters

<i>input</i>	input string the user gives to the console
--------------	--

5.41.4.11 int invalid_input (int count, int max)

Function returns a CONTINUE or EXIT when invalid input is given.

This function looks at the current count and the max iterations and determines whether or not to force the executable to terminate. If the user provides too many incorrect options during the Basic User Interface, then the executable will force quit.

Parameters

<i>count</i>	number of times the user has provided a bad option
<i>max</i>	maximum allowable bad options before force quit

5.41.4.12 int number_files (UI_DATA * ui_dat)

Function returns the number of expected input files for the user's run option.

This function will check the option variable in the ui_dat structure to determine the number of input files that is expected to be given. Running different executable functions in ecosystem may require various number of input files.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.13 bool path (const std::string & input)

Function returns true if the user indicates that input files share a common path.

This function will check the input string for "-p" or "–path" and determine whether or not the user will give a common path to all input files needed for the specified simulation. Only used in Advanced User Interface.

Parameters

<i>input</i>	input string the user gives to the console
--------------	--

5.41.4.14 int run_exec (UI_DATA * ui_dat)

Function will call the user requested executable function.

This function checks the option variable of the ui_dat structure and runs the corresponding executable function.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.15 int run_executable (int argc, const char * argv[])

Function called by the main and runs both user interfaces for the program.

This function is called in the [main.cpp](#) file and passes the console arguments given at run time.

Parameters

<i>argc</i>	number of arguments provided by the user at the time of execution
<i>argv</i>	list of C-strings that was provided by the user at the time of execution

5.41.4.16 `int run_test (UI_DATA * ui_dat)`

Function will call the user requested test function.

This function checks the option variable of the ui_dat structure and runs the corresponding test function.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.17 `bool test (const std::string & input)`

Function returns true if user requests to run a test.

This function will check the input string for "-t" or "‐test" and determine whether or not the user requests to run an ecosystem test function.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.41.4.18 `int test_loop (UI_DATA * ui_dat)`

Function that loops the Basic UI until a valid test option was selected.

This function loops the Basic UI menu for running a test until a valid test is selected by the user. If a valid test is not selected, and the maximum number of loops has been reached, then this function will cause the program to force quit.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.19 `bool valid_addon_options (UI_DATA * ui_dat)`

Function returns true if the user has chosen a valid additional runtime option.

This function will check all additional input options in the user_input variable of ui_dat to determine if the user requests any additional options during runtime. Valid additional options are -p or -path and -i or -input.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.20 `bool valid_exec_string (const std::string & input, UI_DATA * ui_dat)`

Function returns true if the user gave a valid execution option.

This function will check the input string given by the user and determine whether that string denotes a valid execution option. Then, it will mark the option variable in ui_dat with the appropriate option from the valid_options enum.

Parameters

<i>input</i>	input string the user gives to the console
<i>ui_dat</i>	pointer to the data structure for the ui object

5.41.4.21 `bool valid_input_execute (UI_DATA * ui_dat)`

Function returns true if user gave a valid executable function to run.

This function checks the `user_input` argument of `ui_dat` for a valid executable option. If no valid executable was given, then this function returns false.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.22 `bool valid_input_main (UI_DATA * ui_dat)`

Function returns true if user gave valid input in Basic UI.

This function is only called if the user is running the Basic UI. It checks the given console argument stored in `user_input` of `ui_dat` for a valid option. If no valid option is given, then this function returns false.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.23 `bool valid_input_tests (UI_DATA * ui_dat)`

Function returns true if user gave a valid test function to run.

This function checks the `user_input` argument of `ui_dat` for a valid test option. If no valid test was given, then this function returns false.

Parameters

<i>ui_dat</i>	pointer to the data structure for the ui object
---------------	---

5.41.4.24 `bool valid_test_string (const std::string & input, UI_DATA * ui_dat)`

Function returns true if the user gave a valid test option.

This function will check the input string given by the user and determine whether that string denotes a valid test. Then, it will mark the option variable in `ui_dat` with the appropriate option from the `valid_options` enum.

Parameters

<i>input</i>	input string the user gives to the console
<i>ui_dat</i>	pointer to the data structure for the ui object

5.41.4.25 `bool version (const std::string & input)`

Function returns true if user requests to know the executable version.

This function will check the input string for "version", "-v", or "–version" and will tell the executable to display version information about the executable.

Parameters

<i>input</i>	input string user gives to the console
--------------	--

5.42 yaml_wrapper.cpp File Reference

```
#include "yaml_wrapper.h"
```

Functions

- int [YAML_WRAPPER_TESTS](#) ()
- int [YAML_CPP_TEST](#) (const char *file)

5.42.1 Function Documentation

5.42.1.1 int [YAML_CPP_TEST](#) (const char * *file*)

5.42.1.2 int [YAML_WRAPPER_TESTS](#) ()

5.43 yaml_wrapper.h File Reference

```
#include "yaml.h"  
#include "error.h"  
#include <map>  
#include <string>  
#include <iostream>  
#include <utility>  
#include <stdexcept>
```

Classes

- class [ValueTypePair](#)
- class [KeyValueMap](#)
- class [SubHeader](#)
- class [Header](#)
- class [Document](#)
- class [YamlWrapper](#)
- class [yaml_cpp_class](#)

Typedefs

- typedef enum [data_type](#) [data_type](#)
- typedef enum [header_state](#) [header_state](#)

Enumerations

- enum [data_type](#) {
 [STRING](#), [BOOLEAN](#), [DOUBLE](#), [INT](#),
 [UNKNOWN](#) }
- enum [header_state](#) { [ANCHOR](#), [ALIAS](#), [NONE](#) }

Functions

- int [YAML_WRAPPER_TESTS](#) ()
- int [YAML_CPP_TEST](#) (const char *file)

5.43.1 Typedef Documentation

5.43.1.1 typedef enum data_type data_type

5.43.1.2 typedef enum header_state header_state

5.43.2 Enumeration Type Documentation

5.43.2.1 enum data_type

Enumerator

STRING
BOOLEAN
DOUBLE
INT
UNKNOWN

5.43.2.2 enum header_state

Enumerator

ANCHOR
ALIAS
NONE

5.43.3 Function Documentation

5.43.3.1 int [YAML_CPP_TEST](#) (const char * *file*)

5.43.3.2 int [YAML_WRAPPER_TESTS](#) ()

Index

- ~Atom
 - Atom, [10](#)
- ~Document
 - Document, [21](#)
- ~Header
 - Header, [46](#)
- ~KeyValueMap
 - KeyValueMap, [49](#)
- ~MassBalance
 - MassBalance, [51](#)
- ~MasterSpeciesList
 - MasterSpeciesList, [53](#)
- ~Matrix
 - Matrix, [55](#)
- ~Molecule
 - Molecule, [60](#)
- ~PeriodicTable
 - PeriodicTable, [71](#)
- ~Reaction
 - Reaction, [80](#)
- ~SubHeader
 - SubHeader, [96](#)
- ~UnsteadyReaction
 - UnsteadyReaction, [105](#)
- ~ValueTypePair
 - ValueTypePair, [108](#)
- ~YamlWrapper
 - YamlWrapper, [111](#)
- ~yaml_cpp_class
 - yaml_cpp_class, [110](#)
- A
 - magpie.h, [139](#)
- a
 - TRAJECTORY_DATA, [100](#)
- ALIAS
 - yaml_wrapper.h, [180](#)
- ANCHOR
 - yaml_wrapper.h, [180](#)
- A_separator
 - TRAJECTORY_DATA, [100](#)
- A_wire
 - TRAJECTORY_DATA, [100](#)
- ARNOLDI_DATA, [7](#)
 - beta, [8](#)
 - e1, [8](#)
 - Hkp1, [8](#)
 - hp1, [8](#)
 - iter, [8](#)
 - k, [8](#)
 - Output, [8](#)
 - sum, [8](#)
 - v, [8](#)
 - Vk, [8](#)
 - w, [8](#)
 - yk, [9](#)
- abs_tol_bias
 - SCOPSOWL_OPT_DATA, [84](#)
 - SKUA_OPT_DATA, [92](#)
- act_choice
 - shark.cpp, [151](#)
 - shark.h, [154](#)
- act_fun
 - SHARK_DATA, [88](#)
- activation_energy
 - SCOPSOWL_PARAM_DATA, [86](#)
 - SKUA_PARAM, [94](#)
 - UnsteadyReaction, [107](#)
- activity_data
 - SHARK_DATA, [88](#)
- activity_new
 - SHARK_DATA, [88](#)
- activity_old
 - SHARK_DATA, [88](#)
- addDocKey
 - YamlWrapper, [111](#)
- addHeadKey
 - Document, [21](#)
- addKey
 - KeyValueMap, [49](#)
- addPair
 - Document, [21](#)
 - Header, [46](#)
 - KeyValueMap, [49](#)
 - SubHeader, [96](#)
- addSubKey
 - Header, [46](#)
- adjoint
 - Matrix, [55](#)
- adsorb_index
 - SCOPSOWL_OPT_DATA, [84](#)
 - SKUA_OPT_DATA, [92](#)
- Adsorbable
 - SCOPSOWL_PARAM_DATA, [86](#)
 - SKUA_PARAM, [94](#)
- affinity
 - SCOPSOWL_PARAM_DATA, [86](#)
 - SKUA_PARAM, [94](#)
- Ai

- OPTRANS_DATA, 67
- alias
 - SubHeader, 97
- alkalinity
 - MasterSpeciesList, 53
- all_pars
 - GSTA_OPT_DATA, 44
- allLower
 - ui.cpp, 165
 - ui.h, 174
- alpha
 - BACKTRACK_DATA, 12
 - BiCGSTAB_DATA, 14
 - CGS_DATA, 18
 - GCR_DATA, 32
 - PCG_DATA, 68
- anchor_alias_dne
 - error.h, 120
- Ap
 - PCG_DATA, 68
- arg
 - GMRESR_DATA, 37
- arg_matrix_same
 - error.h, 120
- argc
 - UI_DATA, 102
- argv
 - UI_DATA, 102
- arnoldi
 - lark.cpp, 130
 - lark.h, 135
- arnoldi_dat
 - GMRESLP_DATA, 35
- As
 - SYSTEM_DATA, 98
- assertType
 - KeyValueMap, 49
 - ValueTypePair, 108
- Atom, 9
 - ~Atom, 10
 - Atom, 10
 - AtomCategory, 10
 - AtomName, 10
 - AtomState, 10
 - AtomSymbol, 10
 - atomic_number, 11
 - atomic_weight, 11
 - AtomicNumber, 10
 - AtomicWeight, 10
 - BondingElectrons, 10
 - Category, 11
 - DisplayInfo, 10
 - editAtomicWeight, 10
 - editElectrons, 10
 - editNeutrons, 10
 - editOxidationState, 10
 - editProtons, 10
 - editValence, 10
 - Electrons, 11
 - electrons, 11
 - Name, 11
 - NaturalState, 11
 - Neutrons, 11
 - neutrons, 11
 - oxidation_state, 11
 - OxidationState, 11
 - Protons, 11
 - protons, 11
 - Register, 11
 - removeElectron, 11
 - removeNeutron, 11
 - removeProton, 11
 - Symbol, 11
 - valence_e, 11
- AtomCategory
 - Atom, 10
- AtomName
 - Atom, 10
- AtomState
 - Atom, 10
- AtomSymbol
 - Atom, 10
- atomic_number
 - Atom, 11
- atomic_weight
 - Atom, 11
- AtomicNumber
 - Atom, 10
- AtomicWeight
 - Atom, 10
- atoms
 - Molecule, 61
- au_help
 - ui.cpp, 166
 - ui.h, 174
- avg_fiber_density
 - MONKFISH_DATA, 63
- avg_norm
 - SYSTEM_DATA, 98
- avg_sorption
 - MONKFISH_PARAM, 65
- avg_sorption_old
 - MONKFISH_PARAM, 65
- avgDp
 - scopsowl.h, 148
- avgPar
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- avgValue
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- b
 - TRAJECTORY_DATA, 100
- B0
 - TRAJECTORY_DATA, 100
- BOOLEAN

- yaml_wrapper.h, 180
- BACKTRACK_DATA, 11
 - alpha, 12
 - constRho, 12
 - Fk, 12
 - lambdaMin, 12
 - normFkp1, 12
 - rho, 12
 - xk, 12
- backtrack_dat
 - PJFNK_DATA, 75
- backtrackLineSearch
 - lark.cpp, 130
 - lark.h, 135
- BasicUI
 - UI_DATA, 102
- begin
 - Document, 21
 - Header, 46
 - KeyValueMap, 49
 - YamlWrapper, 111
- best_par
 - GSTA_OPT_DATA, 44
- bestres
 - BiCGSTAB_DATA, 14
 - CGS_DATA, 18
 - GCR_DATA, 32
 - GMRESLP_DATA, 35
 - GMRESRP_DATA, 40
 - PCG_DATA, 68
 - PICARD_DATA, 72
- bestx
 - BiCGSTAB_DATA, 14
 - CGS_DATA, 18
 - GCR_DATA, 32
 - GMRESLP_DATA, 35
 - GMRESRP_DATA, 40
 - PCG_DATA, 68
 - PICARD_DATA, 72
 - PJFNK_DATA, 75
- beta
 - ARNOLDI_DATA, 8
 - BiCGSTAB_DATA, 14
 - CGS_DATA, 18
 - FINCH_DATA, 27
 - GCR_DATA, 32
 - PCG_DATA, 69
 - TRAJECTORY_DATA, 100
- BiCGSTAB
 - lark.h, 134
- BiCGSTAB_DATA, 13
 - alpha, 14
 - bestres, 14
 - bestx, 14
 - beta, 14
 - breakdown, 14
 - iter, 14
 - maxit, 15
 - omega, 15
 - omega_old, 15
 - Output, 15
 - p, 15
 - r, 15
 - r0, 15
 - relres, 15
 - relres_base, 15
 - res, 15
 - rho, 15
 - rho_old, 15
 - s, 16
 - t, 16
 - tol_abs, 16
 - tol_rel, 16
 - v, 16
 - x, 16
 - y, 16
 - z, 16
- bicgstab
 - lark.cpp, 130
 - lark.h, 135
- bicgstab_dat
 - PJFNK_DATA, 75
- binary_diffusion
 - MIXED_GAS, 58
- binder_fraction
 - SCOPSOWL_DATA, 82
- binder_poresize
 - SCOPSOWL_DATA, 82
- binder_porosity
 - SCOPSOWL_DATA, 82
- BondingElectrons
 - Atom, 10
- Bounce
 - PJFNK_DATA, 75
- breakdown
 - BiCGSTAB_DATA, 14
 - CGS_DATA, 18
 - GCR_DATA, 32
- Brown_RAD
 - Trajectory.cpp, 161
 - Trajectory.h, 163
- Brown_THETA
 - Trajectory.cpp, 161
 - Trajectory.h, 163
- buckley_leverett_ic
 - finch.cpp, 121
 - finch.h, 124
- buckley_leverett_params
 - finch.cpp, 121
 - finch.h, 124
- bui_help
 - ui.cpp, 166
 - ui.h, 174
- burgers_bcs
 - finch.cpp, 121
 - finch.h, 124

- burgers_ic
 - finch.cpp, 121
 - finch.h, 124
- burgers_params
 - finch.cpp, 121
 - finch.h, 124
- C
 - Speciation_Test01_Data, 95
- c
 - CGS_DATA, 18
 - GCR_DATA, 32
- CGS
 - lark.h, 135
- CONTINUE
 - ui.h, 173
- c_temp
 - GCR_DATA, 32
- CARTESIAN
 - Trajectory.cpp, 161
 - Trajectory.h, 163
- CC_E
 - FINCH_DATA, 27
- CC_I
 - FINCH_DATA, 27
- CE3
 - egret.h, 117
- CGS_DATA, 16
 - alpha, 18
 - bestres, 18
 - bestx, 18
 - beta, 18
 - breakdown, 18
 - c, 18
 - iter, 18
 - maxit, 18
 - Output, 18
 - p, 18
 - r, 18
 - r0, 19
 - relres, 19
 - relres_base, 19
 - res, 19
 - rho, 19
 - sigma, 19
 - tol_abs, 19
 - tol_rel, 19
 - u, 19
 - v, 19
 - w, 19
 - x, 19
 - z, 20
- CL_E
 - FINCH_DATA, 27
- CL_I
 - FINCH_DATA, 27
- CN
 - FINCH_DATA, 27
- CR_E
 - FINCH_DATA, 27
- CR_I
 - FINCH_DATA, 27
- CT
 - Speciation_Test01_Data, 95
- CURVE_TEST03
 - scopsowl.cpp, 145
 - scopsowl.h, 148
- CURVE_TEST04
 - scopsowl.cpp, 145
 - scopsowl.h, 148
- CURVE_TEST05
 - scopsowl.cpp, 145
 - scopsowl.h, 148
- calculate_properties
 - egret.cpp, 116
 - egret.h, 118
- calculateAvgOxiState
 - Molecule, 60
- calculateEnergies
 - Reaction, 80
 - UnsteadyReaction, 105
- calculateEquilibrium
 - Reaction, 80
 - UnsteadyReaction, 105
- calculateRate
 - UnsteadyReaction, 105
- callroutine
 - FINCH_DATA, 27
- CanCalcG
 - Reaction, 80
- CanCalcHS
 - Reaction, 80
- Cap
 - TRAJECTORY_DATA, 100
- Carrier
 - SYSTEM_DATA, 98
- Cartesian
 - finch.h, 124
- Category
 - Atom, 11
- cgs
 - lark.cpp, 130
 - lark.h, 135
- cgs_dat
 - PJFNK_DATA, 75
- changeKey
 - Document, 21
 - Header, 46
 - YamlWrapper, 111
- char_length
 - MIXED_GAS, 58
- char_macro
 - SCOPSOWL_DATA, 82
- char_measure
 - SKUA_DATA, 91
- char_micro
 - SCOPSOWL_DATA, 82

- Charge
 - Molecule, 60
- charge
 - MasterSpeciesList, 53
 - Molecule, 61
- check_Mass
 - finch.cpp, 122
 - finch.h, 124
- CheckMass
 - FINCH_DATA, 27
- CheckMolefractions
 - MIXED_GAS, 58
- checkSpeciesEnergies
 - Reaction, 80
 - UnsteadyReaction, 105
- chi_p
 - TRAJECTORY_DATA, 100
- cleanup
 - yaml_cpp_class, 110
- clear
 - Document, 21
 - Header, 46
 - KeyValueMap, 49
 - SubHeader, 96
 - YamlWrapper, 111
- cofactor
 - Matrix, 55
- columnExtend
 - Matrix, 55
- columnExtract
 - Matrix, 55
- columnProjection
 - Matrix, 55
- columnReplace
 - Matrix, 55
- columnShrink
 - Matrix, 55
- columnVectorFill
 - Matrix, 55
- columns
 - Matrix, 55
- CompareFile
 - SCOPSOWL_OPT_DATA, 84
 - SKUA_OPT_DATA, 92
- Conc_new
 - SHARK_DATA, 88
- Conc_old
 - SHARK_DATA, 88
- Console_Output
 - SHARK_DATA, 88
- const_Dc
 - skua.cpp, 156
 - skua.h, 159
- const_filmMassTransfer
 - scopsowl.cpp, 145
 - scopsowl.h, 148
- const_kf
 - skua.cpp, 156
- skua.h, 159
- const_pH
 - SHARK_DATA, 88
- const_pore_diffusion
 - scopsowl.cpp, 145
 - scopsowl.h, 148
- constRho
 - BACKTRACK_DATA, 12
- ConstantICFill
 - Matrix, 55
- Contains_pH
 - SHARK_DATA, 88
- Contains_pOH
 - SHARK_DATA, 88
- Converged
 - SHARK_DATA, 88
- Convert2Concentration
 - shark.cpp, 151
 - shark.h, 154
- Convert2LogConcentration
 - shark.cpp, 151
 - shark.h, 154
- coord
 - SKUA_DATA, 91
- coord_macro
 - SCOPSOWL_DATA, 82
- coord_micro
 - SCOPSOWL_DATA, 82
- copyAnchor2Alias
 - Document, 22
 - Header, 46
 - YamlWrapper, 112
- count
 - UI_DATA, 102
- crystal_radius
 - SCOPSOWL_DATA, 82
- Cstd
 - egret.h, 117
- current_equil
 - SCOPSOWL_OPT_DATA, 84
 - SKUA_OPT_DATA, 92
- current_points
 - SCOPSOWL_OPT_DATA, 84
 - SKUA_OPT_DATA, 92
- current_press
 - SCOPSOWL_OPT_DATA, 84
 - SKUA_OPT_DATA, 92
- current_temp
 - SCOPSOWL_OPT_DATA, 84
 - SKUA_OPT_DATA, 92
- current_token
 - yaml_cpp_class, 110
- Cylindrical
 - finch.h, 124
- d
 - FINCH_DATA, 27
- DAVIES
 - shark.h, 154

- DAVIES_LADSHAW
 - shark.h, [154](#)
- DEBYE_HUCKEL
 - shark.h, [154](#)
- DOUBLE
 - yaml_wrapper.h, [180](#)
- D_c
 - skua.h, [158](#)
- D_ii
 - egret.h, [117](#)
- D_ij
 - egret.h, [117](#)
- D_inf
 - skua.h, [158](#)
- D_o
 - skua.h, [158](#)
- DBL_EPSILON
 - magpie.h, [139](#)
- dHo
 - GSTA_DATA, [43](#)
- DIC
 - FINCH_DATA, [27](#)
- DISPLACEMENT
 - Trajectory.cpp, [161](#)
 - Trajectory.h, [163](#)
- DOGFISH
 - dogfish.cpp, [113](#)
 - dogfish.h, [115](#)
- DOGFISH_DATA, [23](#)
 - DirichletBC, [23](#)
 - end_time, [23](#)
 - eval_DI, [23](#)
 - eval_R, [23](#)
 - eval_kf, [23](#)
 - eval_qs, [23](#)
 - fiber_diameter, [23](#)
 - fiber_length, [23](#)
 - finch_dat, [24](#)
 - NonLinear, [24](#)
 - NumComp, [24](#)
 - OutputFile, [24](#)
 - param_dat, [24](#)
 - Print2Console, [24](#)
 - Print2File, [24](#)
 - t_counter, [24](#)
 - t_print, [24](#)
 - time, [24](#)
 - time_old, [24](#)
 - total_sorption, [24](#)
 - total_sorption_old, [24](#)
 - total_steps, [24](#)
 - user_data, [24](#)
- DOGFISH_Executioner
 - dogfish.cpp, [114](#)
 - dogfish.h, [115](#)
- DOGFISH_PARAM, [24](#)
 - film_transfer_coeff, [24](#)
 - initial_sorption, [24](#)
 - intraparticle_diffusion, [25](#)
 - sorbed_molefraction, [25](#)
 - species, [25](#)
 - surface_concentration, [25](#)
- DOGFISH_TESTS
 - dogfish.cpp, [114](#)
 - dogfish.h, [115](#)
- DOGFISH_postprocesses
 - dogfish.cpp, [114](#)
 - dogfish.h, [115](#)
- DOGFISH_preprocesses
 - dogfish.cpp, [114](#)
 - dogfish.h, [115](#)
- DOGFISH_reset
 - dogfish.cpp, [114](#)
 - dogfish.h, [115](#)
- dSo
 - GSTA_DATA, [43](#)
- dX
 - TRAJECTORY_DATA, [100](#)
- dY
 - TRAJECTORY_DATA, [100](#)
- Data
 - Matrix, [57](#)
- Data_Map
 - SubHeader, [97](#)
- data_type
 - yaml_wrapper.h, [180](#)
- Davies_equation
 - shark.cpp, [151](#)
 - shark.h, [154](#)
- DaviesLadshaw_equation
 - shark.cpp, [151](#)
 - shark.h, [154](#)
- DebyeHuckel_equation
 - shark.cpp, [151](#)
 - shark.h, [154](#)
- default_Dc
 - skua.cpp, [156](#)
 - skua.h, [159](#)
- default_FilmMTCoeff
 - dogfish.cpp, [113](#)
 - dogfish.h, [115](#)
- default_IntraDiffusion
 - dogfish.cpp, [113](#)
 - dogfish.h, [115](#)
- default_Retardation
 - dogfish.cpp, [113](#)
 - dogfish.h, [115](#)
- default_SurfaceConcentration
 - dogfish.cpp, [113](#)
 - dogfish.h, [115](#)
- default_adsorption
 - scopsowl.cpp, [145](#)
 - scopsowl.h, [148](#)
- default_bcs
 - finch.cpp, [122](#)
 - finch.h, [124](#)

default_density
 monkfish.cpp, 142
 monkfish.h, 143
default_effective_diffusion
 scopsowl.cpp, 146
 scopsowl.h, 148
default_execution
 finch.cpp, 122
 finch.h, 124
default_exterior_concentration
 monkfish.cpp, 142
 monkfish.h, 143
default_film_transfer
 monkfish.cpp, 142
 monkfish.h, 143
default_filmMassTransfer
 scopsowl.cpp, 146
 scopsowl.h, 148
default_ic
 finch.cpp, 122
 finch.h, 125
default_interparticle_diffusion
 monkfish.cpp, 142
 monkfish.h, 143
default_kf
 skua.cpp, 156
 skua.h, 159
default_monk_adsorption
 monkfish.cpp, 142
 monkfish.h, 143
default_monk_equilibrium
 monkfish.cpp, 142
 monkfish.h, 143
default_monkfish_retardation
 monkfish.cpp, 142
 monkfish.h, 143
default_params
 finch.cpp, 122
 finch.h, 125
default_pore_diffusion
 scopsowl.cpp, 146
 scopsowl.h, 148
default_porosity
 monkfish.cpp, 142
 monkfish.h, 143
default_postprocess
 finch.cpp, 122
 finch.h, 125
default_precon
 finch.cpp, 122
 finch.h, 125
default_preprocess
 finch.cpp, 122
 finch.h, 125
default_reset
 finch.cpp, 122
 finch.h, 125
default_retardation
 scopsowl.cpp, 146
 scopsowl.h, 148
default_solve
 finch.cpp, 122
 finch.h, 125
default_surf_diffusion
 scopsowl.cpp, 146
 scopsowl.h, 148
default_timestep
 finch.cpp, 122
 finch.h, 125
Delta
 MassBalance, 52
density
 PURE_GAS, 78
determinate
 Matrix, 55
diagonalSolve
 Matrix, 55
dielectric_const
 SHARK_DATA, 88
diffusion_type
 SCOPSOWL_OPT_DATA, 84
 SKUA_OPT_DATA, 92
dim_mis_match
 error.h, 120
Dirichlet
 FINCH_DATA, 27
DirichletBC
 DOGFISH_DATA, 23
 MONKFISH_DATA, 63
 SCOPSOWL_DATA, 82
 SKUA_DATA, 91
dirichletBCFill
 Matrix, 55
discretize
 FINCH_DATA, 27
Display
 Matrix, 56
Display_Info
 MassBalance, 51
 Reaction, 80
 UnsteadyReaction, 106
display_help
 ui.cpp, 166
 ui.h, 174
display_version
 ui.cpp, 166
 ui.h, 174
DisplayAll
 MasterSpeciesList, 53
DisplayConcentrations
 MasterSpeciesList, 53
DisplayContents
 Document, 22

- Header, 46
- SubHeader, 97
- yaml_cpp_class, 110
- YamlWrapper, 112
- DisplayInfo
 - Atom, 10
 - MasterSpeciesList, 53
 - Molecule, 60
- DisplayMap
 - KeyValueMap, 49
- DisplayPair
 - ValueTypePair, 109
- DisplayTable
 - PeriodicTable, 71
- Dk
 - scopsowl.h, 148
- Dn
 - FINCH_DATA, 27
- Dnp1
 - FINCH_DATA, 27
- Do
 - FINCH_DATA, 27
- Doc_Map
 - YamlWrapper, 112
- Document, 20
 - ~Document, 21
 - addHeadKey, 21
 - addPair, 21
 - begin, 21
 - changeKey, 21
 - clear, 21
 - copyAnchor2Alias, 22
 - DisplayContents, 22
 - Document, 21
 - end, 22
 - getAlias, 22
 - getAnchoredHeader, 22
 - getDataMap, 22
 - getHeadFromSubAlias, 22
 - getHeadMap, 22
 - getHeader, 22
 - getName, 22
 - getState, 22
 - Head_Map, 22
 - isAlias, 22
 - isAnchor, 22
 - operator(), 22
 - operator=, 22
 - resetKeys, 22
 - revalidateAllKeys, 22
 - setAlias, 22
 - setName, 22
 - setNameAliasPair, 22
 - setState, 22
 - size, 22
- dog_dat
 - MONKFISH_DATA, 63
- dogfish
 - ui.h, 173
- dogfish.cpp, 113
 - DOGFISH, 113
 - DOGFISH_Executioner, 114
 - DOGFISH_TESTS, 114
 - DOGFISH_postprocesses, 114
 - DOGFISH_preprocesses, 114
 - DOGFISH_reset, 114
 - default_FilmMTCoeff, 113
 - default_IntraDiffusion, 113
 - default_Retardation, 113
 - default_SurfaceConcentration, 113
 - print2file_DOGFISH_header, 114
 - print2file_DOGFISH_result_new, 114
 - print2file_DOGFISH_result_old, 114
 - print2file_species_header, 114
 - set_DOGFISH_ICs, 114
 - set_DOGFISH_params, 114
 - set_DOGFISH_timestep, 114
 - setup_DOGFISH_DATA, 114
- dogfish.h, 114
 - DOGFISH, 115
 - DOGFISH_Executioner, 115
 - DOGFISH_TESTS, 115
 - DOGFISH_postprocesses, 115
 - DOGFISH_preprocesses, 115
 - DOGFISH_reset, 115
 - default_FilmMTCoeff, 115
 - default_IntraDiffusion, 115
 - default_Retardation, 115
 - default_SurfaceConcentration, 115
 - print2file_DOGFISH_header, 115
 - print2file_DOGFISH_result_new, 115
 - print2file_DOGFISH_result_old, 115
 - print2file_species_header, 115
 - set_DOGFISH_ICs, 115
 - set_DOGFISH_params, 115
 - set_DOGFISH_timestep, 115
 - setup_DOGFISH_DATA, 115
- domain_diameter
 - MONKFISH_DATA, 63
- Dp
 - scopsowl.h, 148
- Dp_ij
 - egret.h, 118
- dq_dc
 - SCOPSOWL_PARAM_DATA, 86
- dq_dco
 - SCOPSOWL_PARAM_DATA, 86
- dq_dp
 - magpie.cpp, 137
 - magpie.h, 139
- dt
 - FINCH_DATA, 27
 - SHARK_DATA, 88
 - TRAJECTORY_DATA, 100
- dt_min
 - SHARK_DATA, 88

- dt_old
 - FINCH_DATA, 27
- duplicate_variable
 - error.h, 120
- dxj
 - NUM_JAC_DATA, 66
- dynamic_viscosity
 - PURE_GAS, 78
- dz
 - FINCH_DATA, 27
- e0
 - GMRESRP_DATA, 40
- e0_bar
 - GMRESRP_DATA, 40
- e1
 - ARNOLDI_DATA, 8
- EXECUTE
 - ui.h, 173
- EXIT
 - ui.h, 173
- e_norm
 - SCOPSOWL_OPT_DATA, 84
 - SKUA_OPT_DATA, 93
- e_norm_old
 - SCOPSOWL_OPT_DATA, 84
 - SKUA_OPT_DATA, 93
- ECO_EXECUTABLE
 - ui.h, 173
- ECO_VERSION
 - ui.h, 173
- EEL_TESTS
 - eel.cpp, 116
 - eel.h, 116
- EGRET_TESTS
 - egret.cpp, 116
 - egret.h, 118
- eMax
 - magpie.cpp, 137
 - magpie.h, 139
 - mSPD_DATA, 65
- edit
 - Matrix, 56
- editAlloOxidationStates
 - Molecule, 60
- editAtomicWeight
 - Atom, 10
- editCharge
 - Molecule, 61
- editElectrons
 - Atom, 10
- editEnergy
 - Molecule, 61
- editEnthalpy
 - Molecule, 61
- editEntropy
 - Molecule, 61
- editHS
 - Molecule, 61
- editNeutrons
 - Atom, 10
- editOneOxidationState
 - Molecule, 61
- editOxidationState
 - Atom, 10
- editPair
 - ValueTypePair, 109
- editProtons
 - Atom, 10
- editValence
 - Atom, 10
- editValue
 - ValueTypePair, 109
- editValue4Key
 - KeyValueMap, 49
- eduGuess
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- eel
 - ui.h, 173
- eel.cpp, 115
 - EEL_TESTS, 116
- eel.h, 116
 - EEL_TESTS, 116
- egret
 - ui.h, 173
- egret.cpp, 116
 - calculate_properties, 116
 - EGRET_TESTS, 116
 - initialize_data, 117
 - set_variables, 117
- egret.h, 117
 - CE3, 117
 - calculate_properties, 118
 - Cstd, 117
 - D_ij, 117
 - D_ij, 117
 - Dp_ij, 118
 - EGRET_TESTS, 118
 - FilmMTCoeff, 118
 - initialize_data, 118
 - Mu, 118
 - Nu, 118
 - PE3, 118
 - PSI, 118
 - Po, 118
 - Pstd, 118
 - RE3, 118
 - ReNum, 118
 - Rstd, 118
 - ScNum, 118
 - set_variables, 118
- Electrons
 - Atom, 11
- electrons
 - Atom, 11
- empirical_kf

- skua.cpp, 156
- skua.h, 159
- empty_matrix
 - error.h, 120
- end
 - Document, 22
 - Header, 46, 47
 - KeyValueMap, 49
 - YamlWrapper, 112
- end_time
 - DOGFISH_DATA, 23
 - MONKFISH_DATA, 63
- Energy
 - Molecule, 61
- energy
 - Reaction, 81
- Enthalpy
 - Molecule, 61
- enthalpy
 - Reaction, 81
- Entropy
 - Molecule, 61
- entropy
 - Reaction, 81
- eps
 - NUM_JAC_DATA, 66
 - PJFNK_DATA, 75
- Equilibrium
 - Reaction, 81
- error
 - error.cpp, 118
 - error.h, 120
 - gsta_opt.h, 128
- error.h
 - anchor_alias_dne, 120
 - arg_matrix_same, 120
 - dim_mis_match, 120
 - duplicate_variable, 120
 - empty_matrix, 120
 - file_dne, 119
 - generic_error, 119
 - indexing_error, 119
 - initial_error, 120
 - invalid_atom, 120
 - invalid_boolean, 119
 - invalid_components, 119
 - invalid_console_input, 120
 - invalid_electron, 120
 - invalid_fraction, 120
 - invalid_gas_sum, 119
 - invalid_molefraction, 119
 - invalid_neutron, 120
 - invalid_norm, 120
 - invalid_proton, 120
 - invalid_size, 120
 - invalid_solid_sum, 119
 - invalid_species, 120
 - invalid_type, 120
 - invalid_valence, 120
 - key_not_found, 120
 - magpie_reverse_error, 119
 - matrix_too_small, 120
 - matvec_mis_match, 120
 - missing_information, 120
 - negative_mass, 120
 - negative_time, 120
 - no_diffusion, 120
 - non_real_edge, 120
 - non_square_matrix, 119
 - not_a_token, 120
 - nullptr_error, 120
 - nullptr_func, 120
 - opt_no_support, 120
 - ortho_check_fail, 120
 - out_of_bounds, 119
 - read_error, 120
 - rxn_rate_error, 120
 - scenario_fail, 119
 - simulation_fail, 119
 - singular_matrix, 120
 - string_parse_error, 120
 - tensor_out_of_bounds, 120
 - unregistered_name, 120
 - unstable_matrix, 120
 - vector_out_of_bounds, 120
 - zero_vector, 120
- error.cpp, 118
 - error, 118
- error.h, 118
 - error, 120
 - error_type, 119
 - mError, 119
- error_type
 - error.h, 119
- eta
 - mSPD_DATA, 65
 - TRAJECTORY_DATA, 100
- eval_Cex
 - MONKFISH_DATA, 63
- Eval_ChargeResidual
 - MasterSpeciesList, 53
- eval_DI
 - DOGFISH_DATA, 23
- eval_Dex
 - MONKFISH_DATA, 63
- eval_GPAST
 - magpie.cpp, 137
 - magpie.h, 139
- eval_GSTA
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- Eval_IC_Residual
 - UnsteadyReaction, 106
- eval_R
 - DOGFISH_DATA, 23
- Eval_ReactionRate

- UnsteadyReaction, [106](#)
- Eval_Residual
 - MassBalance, [51](#)
 - Reaction, [80](#)
 - UnsteadyReaction, [106](#)
- eval_Ret
 - MONKFISH_DATA, [63](#)
- eval_SCOPSOWL_Uptake
 - scopsowl_opt.cpp, [149](#)
 - scopsowl_opt.h, [150](#)
- eval_SKUA_Uptake
 - skua_opt.cpp, [160](#)
 - skua_opt.h, [160](#)
- eval_ads
 - MONKFISH_DATA, [63](#)
 - SCOPSOWL_DATA, [82](#)
- eval_diff
 - SCOPSOWL_DATA, [82](#)
 - SKUA_DATA, [91](#)
- eval_eps
 - MONKFISH_DATA, [63](#)
- eval_eta
 - magpie.cpp, [137](#)
 - magpie.h, [139](#)
- eval_kf
 - DOGFISH_DATA, [23](#)
 - MONKFISH_DATA, [63](#)
 - SCOPSOWL_DATA, [82](#)
 - SKUA_DATA, [91](#)
- eval_po
 - magpie.cpp, [137](#)
 - magpie.h, [140](#)
- eval_po_PI
 - magpie.cpp, [137](#)
 - magpie.h, [140](#)
- eval_po_qo
 - magpie.cpp, [137](#)
 - magpie.h, [140](#)
- eval_qs
 - DOGFISH_DATA, [23](#)
- eval_retard
 - SCOPSOWL_DATA, [82](#)
- eval_rho
 - MONKFISH_DATA, [63](#)
- eval_surfDiff
 - SCOPSOWL_DATA, [82](#)
- EvalActivity
 - SHARK_DATA, [89](#)
- evalprecon
 - FINCH_DATA, [28](#)
- evalres
 - FINCH_DATA, [28](#)
- evaluation
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- exec
 - ui.cpp, [166](#)
 - ui.h, [174](#)
- exec_loop
 - ui.cpp, [166](#)
 - ui.h, [175](#)
- executeYamlRead
 - yaml_cpp_class, [110](#)
- exit
 - ui.cpp, [167](#)
 - ui.h, [175](#)
- Explicit_Eval
 - UnsteadyReaction, [106](#)
- ExplicitFlux
 - FINCH_DATA, [28](#)
- exterior_concentration
 - MONKFISH_PARAM, [65](#)
- exterior_transfer_coeff
 - MONKFISH_PARAM, [65](#)
- F
 - PJFNK_DATA, [75](#)
- FOM
 - lark.h, [135](#)
- f_bias
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- f_bias_old
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- fC_E
 - FINCH_DATA, [28](#)
- fC_I
 - FINCH_DATA, [28](#)
- FINCH_DATA, [25](#)
 - beta, [27](#)
 - CC_E, [27](#)
 - CC_I, [27](#)
 - CL_E, [27](#)
 - CL_I, [27](#)
 - CN, [27](#)
 - CR_E, [27](#)
 - CR_I, [27](#)
 - callroutine, [27](#)
 - CheckMass, [27](#)
 - d, [27](#)
 - DIC, [27](#)
 - Dirichlet, [27](#)
 - discretize, [27](#)
 - Dn, [27](#)
 - Dnp1, [27](#)
 - Do, [27](#)
 - dt, [27](#)
 - dt_old, [27](#)
 - dz, [27](#)
 - evalprecon, [28](#)
 - evalres, [28](#)
 - ExplicitFlux, [28](#)
 - fC_E, [28](#)
 - fC_I, [28](#)
 - fL_E, [28](#)
 - fL_I, [28](#)

- fR_E, 28
- fR_I, 28
- Fn, 28
- Fnp1, 28
- gE, 28
- gl, 28
- Iterative, 28
- kIC, 28
- kfn, 28
- kfnp1, 28
- kn, 28
- knp1, 28
- ko, 28
- L, 28
- LN, 28
- lambda_E, 28
- lambda_I, 28
- ME, 28
- MI, 28
- max_iter, 28
- NE, 28
- NI, 29
- nl_method, 29
- NormTrack, 29
- OE, 29
- OI, 29
- param_data, 29
- picard_dat, 29
- pjfnk_dat, 29
- pres, 29
- RIC, 29
- res, 29
- resettime, 29
- Rn, 29
- Rnp1, 29
- Ro, 29
- s, 29
- setbcs, 29
- setic, 29
- setparams, 29
- setpostprocess, 29
- setpreprocess, 29
- settime, 29
- Sn, 29
- Snp1, 29
- solve, 29
- SteadyState, 29
- T, 29
- t, 29
- t_old, 30
- tol_abs, 30
- tol_rel, 30
- total_iter, 30
- u_star, 30
- uAvg, 30
- uAvg_old, 30
- uIC, 30
- uT, 30
- uT_old, 30
- ubest, 30
- un, 30
- unm1, 30
- unp1, 30
- uo, 30
- Update, 30
- uz_I_E, 30
- uz_I_I, 30
- uz_lm1_E, 30
- uz_lm1_I, 30
- uz_lp1_E, 30
- uz_lp1_I, 30
- vIC, 30
- vn, 30
- vnp1, 30
- vo, 30
- FINCH_Picard
 - finch.h, 124
- FINCH_TESTS
 - finch.cpp, 122
 - finch.h, 125
- fL_E
 - FINCH_DATA, 28
- fL_I
 - FINCH_DATA, 28
- fR_E
 - FINCH_DATA, 28
- fR_I
 - FINCH_DATA, 28
- fiber_diameter
 - DOGFISH_DATA, 23
- fiber_length
 - DOGFISH_DATA, 23
- file_dne
 - error.h, 119
- File_Output
 - SHARK_DATA, 89
- file_name
 - yaml_cpp_class, 110
- Files
 - UI_DATA, 103
- film_transfer
 - SCOPSOWL_PARAM_DATA, 86
 - SKUA_PARAM, 94
- film_transfer_coeff
 - DOGFISH_PARAM, 24
 - MONKFISH_PARAM, 65
- FilmMTCoeff
 - egret.h, 118
- finch
 - ui.h, 173
- finch.cpp, 120
 - buckley_leverett_ic, 121
 - buckley_leverett_params, 121
 - burgers_bcs, 121
 - burgers_ic, 121
 - burgers_params, 121

- check_Mass, [122](#)
- default_bcs, [122](#)
- default_execution, [122](#)
- default_ic, [122](#)
- default_params, [122](#)
- default_postprocess, [122](#)
- default_precon, [122](#)
- default_preprocess, [122](#)
- default_res, [122](#)
- default_reset, [122](#)
- default_solve, [122](#)
- default_timestep, [122](#)
- FINCH_TESTS, [122](#)
- I_direct, [122](#)
- lark_picard_step, [122](#)
- max, [122](#)
- min, [122](#)
- minmod, [122](#)
- minmod_discretization, [122](#)
- nl_picard, [122](#)
- ospre_discretization, [122](#)
- print2file_dim_header, [122](#)
- print2file_newline, [122](#)
- print2file_result_new, [122](#)
- print2file_result_old, [122](#)
- print2file_tab, [122](#)
- print2file_time_header, [122](#)
- setup_FINCH_DATA, [122](#)
- uAverage, [123](#)
- uTotal, [123](#)
- vanAlbada_discretization, [123](#)
- finch.h, [123](#)
 - buckley_leverett_ic, [124](#)
 - buckley_leverett_params, [124](#)
 - burgers_bcs, [124](#)
 - burgers_ic, [124](#)
 - burgers_params, [124](#)
 - Cartesian, [124](#)
 - check_Mass, [124](#)
 - Cylindrical, [124](#)
 - default_bcs, [124](#)
 - default_execution, [124](#)
 - default_ic, [125](#)
 - default_params, [125](#)
 - default_postprocess, [125](#)
 - default_precon, [125](#)
 - default_preprocess, [125](#)
 - default_res, [125](#)
 - default_reset, [125](#)
 - default_solve, [125](#)
 - default_timestep, [125](#)
 - FINCH_Picard, [124](#)
 - FINCH_TESTS, [125](#)
 - I_direct, [125](#)
 - LARK_PJFNK, [124](#)
 - LARK_Picard, [124](#)
 - lark_picard_step, [125](#)
 - max, [125](#)
 - min, [125](#)
 - minmod, [125](#)
 - minmod_discretization, [125](#)
 - nl_picard, [125](#)
 - ospre_discretization, [125](#)
 - print2file_dim_header, [125](#)
 - print2file_newline, [125](#)
 - print2file_result_new, [125](#)
 - print2file_result_old, [125](#)
 - print2file_tab, [125](#)
 - print2file_time_header, [125](#)
 - setup_FINCH_DATA, [125](#)
 - Spherical, [124](#)
 - uAverage, [126](#)
 - uTotal, [126](#)
 - vanAlbada_discretization, [126](#)
- finch_dat
 - DOGFISH_DATA, [24](#)
 - MONKFISH_DATA, [63](#)
 - SCOPSOWL_DATA, [82](#)
 - SKUA_DATA, [91](#)
- findAllTypes
 - KeyValueMap, [49](#)
- findType
 - KeyValueMap, [49](#)
 - ValueTypePair, [109](#)
- Fk
 - BACKTRACK_DATA, [12](#)
- flock.h, [126](#)
- Fn
 - FINCH_DATA, [28](#)
- Fnp1
 - FINCH_DATA, [28](#)
- Fobj
 - GSTA_OPT_DATA, [44](#)
- fom
 - lark.cpp, [130](#)
 - lark.h, [135](#)
- formation_energy
 - Molecule, [61](#)
- formation_enthalpy
 - Molecule, [61](#)
- formation_entropy
 - Molecule, [62](#)
- Formula
 - Molecule, [62](#)
- forward_rate
 - UnsteadyReaction, [107](#)
- forward_ref_rate
 - UnsteadyReaction, [107](#)
- funeval
 - PJFNK_DATA, [75](#)
- Fv
 - PJFNK_DATA, [75](#)
- Fx
 - NUM_JAC_DATA, [66](#)
- Fxp
 - NUM_JAC_DATA, [66](#)

- GCR
 - lark.h, [135](#)
- GMRESLP
 - lark.h, [134](#)
- GMRESR
 - lark.h, [135](#)
- GMRESRP
 - lark.h, [135](#)
- GCR_DATA, [31](#)
 - alpha, [32](#)
 - bestres, [32](#)
 - bestx, [32](#)
 - beta, [32](#)
 - breakdown, [32](#)
 - c, [32](#)
 - c_temp, [32](#)
 - iter_inner, [32](#)
 - iter_outer, [32](#)
 - maxit, [32](#)
 - Output, [32](#)
 - r, [33](#)
 - relres, [33](#)
 - relres_base, [33](#)
 - res, [33](#)
 - restart, [33](#)
 - tol_abs, [33](#)
 - tol_rel, [33](#)
 - total_iter, [33](#)
 - transpose_dat, [33](#)
 - u, [33](#)
 - u_temp, [33](#)
 - x, [33](#)
- GCR_Output
 - GMRESR_DATA, [37](#)
- gE
 - FINCH_DATA, [28](#)
- gl
 - FINCH_DATA, [28](#)
- GMRES_Output
 - GMRESR_DATA, [38](#)
- GMRESLP_DATA, [34](#)
 - arnoldi_dat, [35](#)
 - bestres, [35](#)
 - bestx, [35](#)
 - iter, [35](#)
 - maxit, [35](#)
 - Output, [35](#)
 - r, [35](#)
 - relres, [35](#)
 - relres_base, [35](#)
 - res, [35](#)
 - restart, [35](#)
 - steps, [36](#)
 - tol_abs, [36](#)
 - tol_rel, [36](#)
 - x, [36](#)
- GMRESR_DATA, [36](#)
 - arg, [37](#)
 - GCR_Output, [37](#)
 - GMRES_Output, [38](#)
 - gcr_abs_tol, [37](#)
 - gcr_dat, [37](#)
 - gcr_maxit, [37](#)
 - gcr_rel_tol, [38](#)
 - gcr_restart, [38](#)
 - gmres_dat, [38](#)
 - gmres_maxit, [38](#)
 - gmres_restart, [38](#)
 - gmres_tol, [38](#)
 - iter_inner, [38](#)
 - iter_outer, [38](#)
 - matvec, [38](#)
 - matvec_data, [38](#)
 - N, [38](#)
 - term_precon, [39](#)
 - terminal_precon, [39](#)
 - total_iter, [39](#)
- GMRESRP_DATA, [39](#)
 - bestres, [40](#)
 - bestx, [40](#)
 - e0, [40](#)
 - e0_bar, [40](#)
 - H, [41](#)
 - H_bar, [41](#)
 - iter_inner, [41](#)
 - iter_outer, [41](#)
 - iter_total, [41](#)
 - maxit, [41](#)
 - Output, [41](#)
 - r, [41](#)
 - relres, [41](#)
 - relres_base, [41](#)
 - res, [41](#)
 - restart, [41](#)
 - sum, [42](#)
 - tol_abs, [42](#)
 - tol_rel, [42](#)
 - v, [42](#)
 - Vk, [42](#)
 - w, [42](#)
 - x, [42](#)
 - y, [42](#)
- GPAST_DATA, [42](#)
 - gama_inf, [43](#)
 - He, [43](#)
 - Plo, [43](#)
 - po, [43](#)
 - poi, [43](#)
 - present, [43](#)
 - q, [43](#)
 - qo, [43](#)
 - x, [43](#)
 - y, [43](#)
- GSTA_DATA, [43](#)
 - dHo, [43](#)
 - dSo, [43](#)

- m, [43](#)
 - qmax, [43](#)
- GSTA_OPT_DATA, [44](#)
 - all_pars, [44](#)
 - best_par, [44](#)
 - Fobj, [44](#)
 - iso, [44](#)
 - Kno, [44](#)
 - n_par, [44](#)
 - norms, [44](#)
 - opt_qmax, [44](#)
 - P, [44](#)
 - q, [44](#)
 - qmax, [44](#)
 - total_eval, [45](#)
- gama
 - mSPD_DATA, [65](#)
- gama_inf
 - GPAST_DATA, [43](#)
- gas_dat
 - SCOPSOWL_DATA, [82](#)
 - SKUA_DATA, [91](#)
- gas_temperature
 - MIXED_GAS, [58](#)
 - SCOPSOWL_DATA, [82](#)
- gas_velocity
 - SCOPSOWL_DATA, [82](#)
 - SKUA_DATA, [91](#)
- gcr
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- gcr_abs_tol
 - GMRESR_DATA, [37](#)
- gcr_dat
 - GMRESR_DATA, [37](#)
 - PJFNK_DATA, [75](#)
- gcr_maxit
 - GMRESR_DATA, [37](#)
- gcr_rel_tol
 - GMRESR_DATA, [38](#)
- gcr_restart
 - GMRESR_DATA, [38](#)
- generic_error
 - error.h, [119](#)
- Get_ActivationEnergy
 - UnsteadyReaction, [106](#)
- Get_Affinity
 - UnsteadyReaction, [106](#)
- Get_Delta
 - MassBalance, [51](#)
- Get_Energy
 - Reaction, [80](#)
 - UnsteadyReaction, [106](#)
- Get_Enthalpy
 - Reaction, [80](#)
 - UnsteadyReaction, [106](#)
- Get_Entropy
 - Reaction, [80](#)
 - UnsteadyReaction, [106](#)
- Get_Equilibrium
 - Reaction, [80](#)
 - UnsteadyReaction, [106](#)
- Get_Forward
 - UnsteadyReaction, [106](#)
- Get_ForwardRef
 - UnsteadyReaction, [106](#)
- Get_InitialValue
 - UnsteadyReaction, [106](#)
- Get_MaximumValue
 - UnsteadyReaction, [106](#)
- Get_Name
 - MassBalance, [51](#)
- Get_Reverse
 - UnsteadyReaction, [106](#)
- Get_ReverseRef
 - UnsteadyReaction, [106](#)
- Get_Species_Index
 - UnsteadyReaction, [106](#)
- Get_Stoichiometric
 - Reaction, [80](#)
 - UnsteadyReaction, [106](#)
- Get_TimeStep
 - UnsteadyReaction, [106](#)
- Get_TotalConcentration
 - MassBalance, [51](#)
- get_index
 - MasterSpeciesList, [53](#)
- get_species
 - MasterSpeciesList, [53](#)
- getAlias
 - Document, [22](#)
 - Header, [47](#)
 - SubHeader, [97](#)
- getAnchoredDoc
 - YamlWrapper, [112](#)
- getAnchoredHeader
 - Document, [22](#)
- getAnchoredSub
 - Header, [47](#)
- getBool
 - KeyValueMap, [49](#)
 - ValueTypePair, [109](#)
- getDataMap
 - Document, [22](#)
 - Header, [47](#)
- getDocFromHeadAlias
 - YamlWrapper, [112](#)
- getDocFromSubAlias
 - YamlWrapper, [112](#)
- getDocMap
 - YamlWrapper, [112](#)
- getDocument
 - YamlWrapper, [112](#)
- getDouble
 - KeyValueMap, [49](#)
 - ValueTypePair, [109](#)

- getHeadFromSubAlias
 - Document, [22](#)
- getHeadMap
 - Document, [22](#)
- getHeader
 - Document, [22](#)
- getInt
 - KeyValueMap, [49](#)
 - ValueTypePair, [109](#)
- getMap
 - KeyValueMap, [49](#)
 - SubHeader, [97](#)
- getName
 - Document, [22](#)
 - Header, [47](#)
 - SubHeader, [97](#)
- getPair
 - KeyValueMap, [49](#)
 - ValueTypePair, [109](#)
- getState
 - Document, [22](#)
 - Header, [47](#)
 - SubHeader, [97](#)
- getString
 - KeyValueMap, [49](#)
 - ValueTypePair, [109](#)
- getSubHeader
 - Header, [47](#)
- getSubMap
 - Header, [47](#)
- getType
 - KeyValueMap, [49](#)
 - ValueTypePair, [109](#)
- getValue
 - KeyValueMap, [50](#)
 - ValueTypePair, [109](#)
- getYamlWrapper
 - yaml_cpp_class, [110](#)
- gmres_dat
 - GMRESR_DATA, [38](#)
- gmres_maxit
 - GMRESR_DATA, [38](#)
- gmres_restart
 - GMRESR_DATA, [38](#)
- gmres_tol
 - GMRESR_DATA, [38](#)
- gmresLeftPreconditioned
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- gmresPreconditioner
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- gmresRightPreconditioned
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- gmreslp_dat
 - PJFNK_DATA, [75](#)
- gmresr
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- gmresr_dat
 - PJFNK_DATA, [76](#)
- gmresrp_dat
 - PJFNK_DATA, [76](#)
- gpast_dat
 - MAGPIE_DATA, [50](#)
- grad_mSPD
 - magpie.cpp, [137](#)
 - magpie.h, [140](#)
- Grav_R
 - Trajectory.cpp, [161](#)
 - Trajectory.h, [163](#)
- Grav_T
 - Trajectory.cpp, [161](#)
 - Trajectory.h, [163](#)
- gsta_opt
 - ui.h, [173](#)
- gsta_dat
 - MAGPIE_DATA, [50](#)
- gsta_opt.cpp, [126](#)
 - avgPar, [127](#)
 - avgValue, [127](#)
 - eduGuess, [127](#)
 - eval_GSTA, [127](#)
 - gsta_optimize, [127](#)
 - gstaFunc, [127](#)
 - gstaObjFunc, [127](#)
 - isSmooth, [127](#)
 - minIndex, [127](#)
 - minValue, [127](#)
 - orderMag, [127](#)
 - orthoLinReg, [127](#)
 - rSq, [127](#)
 - roundIt, [127](#)
 - twoFifths, [127](#)
 - weightedAvg, [127](#)
- gsta_opt.h, [127](#)
 - avgPar, [128](#)
 - avgValue, [128](#)
 - eduGuess, [128](#)
 - error, [128](#)
 - eval_GSTA, [128](#)
 - gsta_optimize, [128](#)
 - gstaFunc, [128](#)
 - gstaObjFunc, [128](#)
 - isSmooth, [128](#)
 - minIndex, [128](#)
 - minValue, [129](#)
 - Na, [128](#)
 - orderMag, [129](#)
 - orthoLinReg, [129](#)
 - Po, [128](#)
 - R, [128](#)
 - rSq, [129](#)
 - roundIt, [129](#)
 - twoFifths, [129](#)

- weightedAvg, 129
- gsta_optimize
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- gstaFunc
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- gstaObjFunc
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- H
 - GMRESRP_DATA, 41
 - TRAJECTORY_DATA, 100
- H0
 - TRAJECTORY_DATA, 100
- HELP
 - ui.h, 173
- H_bar
 - GMRESRP_DATA, 41
- Hamaker
 - TRAJECTORY_DATA, 100
- HaveEnergy
 - Molecule, 61
- HaveEquil
 - Reaction, 81
- haveEquilibrium
 - Reaction, 80
 - UnsteadyReaction, 106
- HaveForRef
 - UnsteadyReaction, 107
- HaveForward
 - UnsteadyReaction, 107
- HaveG
 - Reaction, 81
- haveG
 - Molecule, 62
- HaveHS
 - Molecule, 61
 - Reaction, 81
- haveHS
 - Molecule, 62
- haveMinMax
 - MONKFISH_DATA, 63
- haveRate
 - UnsteadyReaction, 106
- HaveRevRef
 - UnsteadyReaction, 107
- HaveReverse
 - UnsteadyReaction, 107
- He
 - GPAST_DATA, 43
 - magpie.h, 139
- Head_Map
 - Document, 22
- Header, 45
 - ~Header, 46
 - addPair, 46
 - addSubKey, 46
 - begin, 46
 - changeKey, 46
 - clear, 46
 - copyAnchor2Alias, 46
 - DisplayContents, 46
 - end, 46, 47
 - getAlias, 47
 - getAnchoredSub, 47
 - getDataMap, 47
 - getName, 47
 - getState, 47
 - getSubHeader, 47
 - getSubMap, 47
 - Header, 46
 - isAlias, 47
 - isAnchor, 47
 - operator(), 47
 - operator=, 47
 - resetKeys, 47
 - setAlias, 47
 - setName, 47
 - setNameAliasPair, 47
 - setState, 47
 - size, 47
 - Sub_Map, 47
- header_state
 - yaml_wrapper.h, 180
- help
 - ui.cpp, 167
 - ui.h, 175
- Heterogeneous
 - SCOPSOWL_DATA, 83
- Hkp1
 - ARNOLDI_DATA, 8
- hp1
 - ARNOLDI_DATA, 8
- I
 - SYSTEM_DATA, 98
- IDEAL
 - shark.h, 154
- INT
 - yaml_wrapper.h, 180
- Ideal
 - SYSTEM_DATA, 98
- ideal_solution
 - shark.cpp, 151
 - shark.h, 154
- li
 - OPTRANS_DATA, 67
- indexing_error
 - error.h, 119
- initial_error
 - error.h, 120
- initial_guess_SCOPSOWL
 - scopsowl_opt.cpp, 149
 - scopsowl_opt.h, 150
- initial_guess_SKUA
 - skua_opt.cpp, 160

- skua_opt.h, 160
- initial_sorption
 - DOGFISH_PARAM, 24
 - MONKFISH_PARAM, 65
- initial_value
 - UnsteadyReaction, 107
- initialGuess_mSPD
 - magpie.cpp, 138
 - magpie.h, 140
- Initialize_List
 - MassBalance, 51
 - Reaction, 80
 - UnsteadyReaction, 106
- initialize_data
 - egret.cpp, 117
 - egret.h, 118
- inner_product
 - Matrix, 56
- input
 - ui.cpp, 167
 - ui.h, 175
- input_file
 - yaml_cpp_class, 110
- input_files
 - UI_DATA, 103
- IntegralAvg
 - Matrix, 56
- IntegralTotal
 - Matrix, 56
- interparticle_diffusion
 - MONKFISH_PARAM, 65
- intraparticle_diffusion
 - DOGFISH_PARAM, 25
 - MONKFISH_PARAM, 65
- invalid_atom
 - error.h, 120
- invalid_boolean
 - error.h, 119
- invalid_components
 - error.h, 119
- invalid_console_input
 - error.h, 120
- invalid_electron
 - error.h, 120
- invalid_fraction
 - error.h, 120
- invalid_gas_sum
 - error.h, 119
- invalid_molefraction
 - error.h, 119
- invalid_neutron
 - error.h, 120
- invalid_norm
 - error.h, 120
- invalid_proton
 - error.h, 120
- invalid_size
 - error.h, 120
- invalid_solid_sum
 - error.h, 119
- invalid_species
 - error.h, 120
- invalid_type
 - error.h, 120
- invalid_valence
 - error.h, 120
- invalid_input
 - ui.cpp, 167
 - ui.h, 175
- inverse
 - Matrix, 56
- isAlias
 - Document, 22
 - Header, 47
 - SubHeader, 97
- isAnchor
 - Document, 22
 - Header, 47
 - SubHeader, 97
- isRegistered
 - Molecule, 61
- isSmooth
 - gsta_opt.cpp, 127
 - gsta_opt.h, 128
- iso
 - GSTA_OPT_DATA, 44
- iter
 - ARNOLDI_DATA, 8
 - BiCGSTAB_DATA, 14
 - CGS_DATA, 18
 - GMRESLP_DATA, 35
 - PCG_DATA, 69
 - PICARD_DATA, 72
- iter_inner
 - GCR_DATA, 32
 - GMRESR_DATA, 38
 - GMRESRP_DATA, 41
- iter_outer
 - GCR_DATA, 32
 - GMRESR_DATA, 38
 - GMRESRP_DATA, 41
- iter_total
 - GMRESRP_DATA, 41
- Iterative
 - FINCH_DATA, 28
- J
 - SYSTEM_DATA, 98
- Jacobian
 - Speciation_Test01_Data, 95
- jacvec
 - lark.cpp, 131
 - lark.h, 135
- K
 - SYSTEM_DATA, 98
- k

- ARNOLDI_DATA, [8](#)
- TRAJECTORY_DATA, [100](#)
- kB
 - magpie.h, [139](#)
- kIC
 - FINCH_DATA, [28](#)
- key_not_found
 - error.h, [120](#)
- Key_Value
 - KeyValueMap, [50](#)
- KeyValueMap, [48](#)
 - ~KeyValueMap, [49](#)
 - addKey, [49](#)
 - addPair, [49](#)
 - assertType, [49](#)
 - begin, [49](#)
 - clear, [49](#)
 - DisplayMap, [49](#)
 - editValue4Key, [49](#)
 - end, [49](#)
 - findAllTypes, [49](#)
 - findType, [49](#)
 - getBool, [49](#)
 - getDouble, [49](#)
 - getInt, [49](#)
 - getMap, [49](#)
 - getPair, [49](#)
 - getString, [49](#)
 - getType, [49](#)
 - getValue, [50](#)
 - Key_Value, [50](#)
 - KeyValueMap, [49](#)
 - KeyValueMap, [49](#)
 - operator=, [50](#)
 - size, [50](#)
- kfn
 - FINCH_DATA, [28](#)
- kfnp1
 - FINCH_DATA, [28](#)
- kinematic_viscosity
 - MIXED_GAS, [58](#)
- kn
 - FINCH_DATA, [28](#)
- Kno
 - GSTA_OPT_DATA, [44](#)
- knp1
 - FINCH_DATA, [28](#)
- ko
 - FINCH_DATA, [28](#)
- krylov_method
 - lark.h, [134](#)
- L
 - FINCH_DATA, [28](#)
 - TRAJECTORY_DATA, [100](#)
- L_Output
 - PJFNK_DATA, [76](#)
- l_direct
 - finch.cpp, [122](#)
- finch.h, [125](#)
- l_iter
 - PJFNK_DATA, [76](#)
- L_wire
 - TRAJECTORY_DATA, [100](#)
- LARGE_CYCLE_TEST01
 - scopsowl.cpp, [146](#)
 - scopsowl.h, [148](#)
- LARK_PJFNK
 - finch.h, [124](#)
- LARK_Picard
 - finch.h, [124](#)
- LARK_TESTS
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- LN
 - FINCH_DATA, [28](#)
- LOCATION
 - Trajectory.cpp, [162](#)
 - Trajectory.h, [163](#)
- ladshawSolve
 - Matrix, [56](#)
- lambda_E
 - FINCH_DATA, [28](#)
- lambda_I
 - FINCH_DATA, [28](#)
- lambdaMin
 - BACKTRACK_DATA, [12](#)
- lark
 - ui.h, [173](#)
- lark.h
 - BiCGSTAB, [134](#)
 - CGS, [135](#)
 - FOM, [135](#)
 - GCR, [135](#)
 - GMRESLP, [134](#)
 - GMRESR, [135](#)
 - GMRESRP, [135](#)
 - PCG, [134](#)
- lark.cpp, [129](#)
 - arnoldi, [130](#)
 - backtrackLineSearch, [130](#)
 - bicgstab, [130](#)
 - cgs, [130](#)
 - fom, [130](#)
 - gcr, [131](#)
 - gmresLeftPreconditioned, [131](#)
 - gmresPreconditioner, [131](#)
 - gmresRightPreconditioned, [131](#)
 - gmresr, [131](#)
 - jacvec, [131](#)
 - LARK_TESTS, [131](#)
 - NumericalJacobian, [131](#)
 - operatorTranspose, [131](#)
 - pcg, [131](#)
 - picard, [131](#)
 - pjfnk, [131](#)
 - update_arnoldi_solution, [131](#)

- lark.h, 131
 - arnoldi, 135
 - backtrackLineSearch, 135
 - bigstab, 135
 - cgs, 135
 - fom, 135
 - gcr, 135
 - gmresLeftPreconditioned, 135
 - gmresPreconditioner, 135
 - gmresRightPreconditioned, 135
 - gmresr, 135
 - jacvec, 135
 - krylov_method, 134
 - LARK_TESTS, 135
 - NumericalJacobian, 135
 - operatorTranspose, 135
 - pcg, 136
 - picard, 136
 - pjfnk, 136
 - update_arnoldi_solution, 136
- lark_picard_step
 - finch.cpp, 122
 - finch.h, 125
- level
 - MONKFISH_DATA, 63
 - SCOPSOWL_DATA, 83
- lin_precon
 - SHARK_DATA, 89
- lin_tol_abs
 - PJFNK_DATA, 76
- lin_tol_rel
 - PJFNK_DATA, 76
- LineSearch
 - PJFNK_DATA, 76
- linear_solver
 - PJFNK_DATA, 76
- linearsolve_choice
 - shark.cpp, 151
 - shark.h, 154
- linesearch_choice
 - shark.cpp, 151
 - shark.h, 154
- List
 - MassBalance, 52
 - Mechanism, 58
 - Reaction, 81
- list_size
 - MasterSpeciesList, 53
- InKo
 - magpie.h, 139
- Inact_mSPD
 - magpie.cpp, 138
 - magpie.h, 140
- logC
 - Speciation_Test01_Data, 95
- logKa1
 - Speciation_Test01_Data, 95
- logKa2
 - Speciation_Test01_Data, 95
- logKw
 - Speciation_Test01_Data, 95
- lowerHessenberg2Triangular
 - Matrix, 56
- lowerHessenbergSolve
 - Matrix, 56
- lowerTriangularSolve
 - Matrix, 56
- M
 - TRAJECTORY_DATA, 100
- m
 - GSTA_DATA, 43
- M_PI
 - macaw.h, 137
- m_rand
 - TRAJECTORY_DATA, 100
- MACAW_TESTS
 - macaw.cpp, 136
 - macaw.h, 137
- MAGPIE
 - magpie.cpp, 138
 - magpie.h, 140
- MAGPIE_DATA, 50
 - gpast_dat, 50
 - gsta_dat, 50
 - mspd_dat, 50
 - sys_dat, 50
- MAGPIE_SCENARIOS
 - magpie.cpp, 138
 - magpie.h, 140
- ME
 - FINCH_DATA, 28
- mError
 - error.h, 119
- MI
 - FINCH_DATA, 28
- MIXED_GAS, 58
 - binary_diffusion, 58
 - char_length, 58
 - CheckMolefractions, 58
 - gas_temperature, 58
 - kinematic_viscosity, 58
 - molefraction, 58
 - N, 58
 - Reynolds, 58
 - species_dat, 59
 - total_density, 59
 - total_dyn_vis, 59
 - total_molecular_weight, 59
 - total_pressure, 59
 - total_specific_heat, 59
 - velocity, 59
- MOLA_TESTS
 - mola.cpp, 141
 - mola.h, 141
- MONKFISH_DATA, 62
 - avg_fiber_density, 63

- DirichletBC, [63](#)
- dog_dat, [63](#)
- domain_diameter, [63](#)
- end_time, [63](#)
- eval_Cex, [63](#)
- eval_Dex, [63](#)
- eval_Ret, [63](#)
- eval_ads, [63](#)
- eval_eps, [63](#)
- eval_kf, [63](#)
- eval_rho, [63](#)
- finch_dat, [63](#)
- haveMinMax, [63](#)
- level, [63](#)
- max_fiber_density, [63](#)
- max_porosity, [63](#)
- min_fiber_density, [63](#)
- min_porosity, [63](#)
- MultiScale, [63](#)
- NonLinear, [63](#)
- NumComp, [63](#)
- Output, [64](#)
- param_dat, [64](#)
- Print2Console, [64](#)
- Print2File, [64](#)
- single_fiber_density, [64](#)
- t_counter, [64](#)
- t_print, [64](#)
- time, [64](#)
- time_old, [64](#)
- total_sorption, [64](#)
- total_sorption_old, [64](#)
- total_steps, [64](#)
- user_data, [64](#)
- MONKFISH_PARAM, [64](#)
 - avg_sorption, [65](#)
 - avg_sorption_old, [65](#)
 - exterior_concentration, [65](#)
 - exterior_transfer_coeff, [65](#)
 - film_transfer_coeff, [65](#)
 - initial_sorption, [65](#)
 - interparticle_diffusion, [65](#)
 - intraparticle_diffusion, [65](#)
 - sorbed_molefraction, [65](#)
 - sorption_bc, [65](#)
 - species, [65](#)
- MONKFISH_TESTS
 - monkfish.cpp, [142](#)
 - monkfish.h, [143](#)
- mSPD_DATA, [65](#)
 - eMax, [65](#)
 - eta, [65](#)
 - gama, [65](#)
 - s, [65](#)
 - v, [65](#)
- macaw
 - ui.h, [173](#)
- macaw.cpp, [136](#)
- MACAW_TESTS, [136](#)
- macaw.h, [136](#)
 - M_PI, [137](#)
 - MACAW_TESTS, [137](#)
- Magnetic_R
 - Trajectory.cpp, [162](#)
 - Trajectory.h, [163](#)
- Magnetic_T
 - Trajectory.cpp, [162](#)
 - Trajectory.h, [163](#)
- magpie
 - ui.h, [173](#)
- magpie.cpp, [137](#)
 - dq_dp, [137](#)
 - eMax, [137](#)
 - eval_GPAST, [137](#)
 - eval_eta, [137](#)
 - eval_po, [137](#)
 - eval_po_PI, [137](#)
 - eval_po_qo, [137](#)
 - grad_mSPD, [137](#)
 - initialGuess_mSPD, [138](#)
 - lnact_mSPD, [138](#)
 - MAGPIE, [138](#)
 - MAGPIE_SCENARIOS, [138](#)
 - PI, [138](#)
 - q_p, [138](#)
 - qT, [138](#)
 - qo, [138](#)
 - Qst, [138](#)
- magpie.h, [138](#)
 - A, [139](#)
 - DBL_EPSILON, [139](#)
 - dq_dp, [139](#)
 - eMax, [139](#)
 - eval_GPAST, [139](#)
 - eval_eta, [139](#)
 - eval_po, [140](#)
 - eval_po_PI, [140](#)
 - eval_po_qo, [140](#)
 - grad_mSPD, [140](#)
 - He, [139](#)
 - initialGuess_mSPD, [140](#)
 - kB, [139](#)
 - lnKo, [139](#)
 - lnact_mSPD, [140](#)
 - MAGPIE, [140](#)
 - MAGPIE_SCENARIOS, [140](#)
 - Na, [139](#)
 - PI, [140](#)
 - Po, [139](#)
 - q_p, [140](#)
 - qT, [140](#)
 - qo, [140](#)
 - Qst, [140](#)
 - R, [139](#)
 - shapeFactor, [139](#)
 - V, [139](#)

- Z, 139
- magpie_reverse_error
 - error.h, 119
- magpie_dat
 - SCOPSOWL_DATA, 83
 - SKUA_DATA, 91
- main
 - main.cpp, 141
- main.cpp, 140
 - main, 141
- MassBalance, 50
 - ~MassBalance, 51
 - Delta, 52
 - Display_Info, 51
 - Eval_Residual, 51
 - Get_Delta, 51
 - Get_Name, 51
 - Get_TotalConcentration, 51
 - Initialize_List, 51
 - List, 52
 - MassBalance, 51
 - MassBalance, 51
 - Name, 52
 - Set_Delta, 51
 - Set_Name, 51
 - Set_TotalConcentration, 51
 - Sum_Delta, 51
 - TotalConcentration, 52
- MassBalanceList
 - SHARK_DATA, 89
- MasterList
 - SHARK_DATA, 89
- MasterSpeciesList, 52
 - ~MasterSpeciesList, 53
 - alkalinity, 53
 - charge, 53
 - DisplayAll, 53
 - DisplayConcentrations, 53
 - DisplayInfo, 53
 - Eval_ChargeResidual, 53
 - get_index, 53
 - get_species, 53
 - list_size, 53
 - MasterSpeciesList, 53
 - MasterSpeciesList, 53
 - operator=, 53
 - residual_alkalinity, 53
 - set_alkalinity, 53
 - set_list_size, 53
 - set_species, 53
 - size, 53
 - species, 53
 - speciesName, 53
- Matrix
 - ~Matrix, 55
 - adjoint, 55
 - cofactor, 55
 - columnExtend, 55
 - columnExtract, 55
 - columnProjection, 55
 - columnReplace, 55
 - columnShrink, 55
 - columnVectorFill, 55
 - columns, 55
 - ConstantICFill, 55
 - Data, 57
 - determinate, 55
 - diagonalSolve, 55
 - dirichletBCFill, 55
 - Display, 56
 - edit, 56
 - inner_product, 56
 - IntegralAvg, 56
 - IntegralTotal, 56
 - inverse, 56
 - ladshawSolve, 56
 - lowerHessenberg2Triangular, 56
 - lowerHessenbergSolve, 56
 - lowerTriangularSolve, 56
 - Matrix, 55
 - naturalLaplacian3D, 56
 - norm, 56
 - num_cols, 57
 - num_rows, 57
 - operator*, 56
 - operator(), 56
 - operator+, 56
 - operator-, 56
 - operator/, 56
 - operator=, 56
 - rowExtend, 56
 - rowExtract, 56
 - rowReplace, 56
 - rowShrink, 56
 - rows, 56
 - set_size, 56
 - SolnTransform, 56
 - sphericalAvg, 57
 - sphericalBCFill, 57
 - sum, 57
 - transpose, 57
 - transpose_multiply, 57
 - tridiagonalFill, 57
 - tridiagonalSolve, 57
 - tridiagonalVectorFill, 57
 - upperHessenberg2Triangular, 57
 - upperHessenbergSolve, 57
 - upperTriangularSolve, 57
 - zeros, 57
- Matrix< T >, 54
- matrix_too_small
 - error.h, 120
- matvec
 - GMRESR_DATA, 38
- matvec_mis_match
 - error.h, 120

- matvec_data
 - GMRESR_DATA, [38](#)
- max
 - finch.cpp, [122](#)
 - finch.h, [125](#)
 - UI_DATA, [103](#)
- max_bias
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- max_fiber_density
 - MONKFISH_DATA, [63](#)
- max_guess_iter
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- max_iter
 - FINCH_DATA, [28](#)
- max_norm
 - SYSTEM_DATA, [98](#)
- max_porosity
 - MONKFISH_DATA, [63](#)
- max_value
 - UnsteadyReaction, [107](#)
- maxit
 - BiCGSTAB_DATA, [15](#)
 - CGS_DATA, [18](#)
 - GCR_DATA, [32](#)
 - GMRESLP_DATA, [35](#)
 - GMRESRP_DATA, [41](#)
 - PCG_DATA, [69](#)
 - PICARD_DATA, [72](#)
- Mechanism, [57](#)
 - List, [58](#)
 - reactions, [58](#)
 - species_index, [58](#)
 - weight, [58](#)
- min
 - finch.cpp, [122](#)
 - finch.h, [125](#)
- min_bias
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- min_fiber_density
 - MONKFISH_DATA, [63](#)
- min_porosity
 - MONKFISH_DATA, [63](#)
- minIndex
 - gsta_opt.cpp, [127](#)
 - gsta_opt.h, [128](#)
- minValue
 - gsta_opt.cpp, [127](#)
 - gsta_opt.h, [129](#)
- minmod
 - finch.cpp, [122](#)
 - finch.h, [125](#)
- minmod_discretization
 - finch.cpp, [122](#)
 - finch.h, [125](#)
- missing_information
 - error.h, [120](#)
- MissingArg
 - UI_DATA, [103](#)
- mola
 - ui.h, [173](#)
- mola.cpp, [141](#)
 - MOLA_TESTS, [141](#)
- mola.h, [141](#)
 - MOLA_TESTS, [141](#)
- molar_weight
 - Molecule, [62](#)
- MolarWeight
 - Molecule, [61](#)
- molecular_diffusion
 - PURE_GAS, [78](#)
- molecular_weight
 - PURE_GAS, [78](#)
- MolecularFormula
 - Molecule, [61](#)
- Molecule, [59](#)
 - ~Molecule, [60](#)
 - atoms, [61](#)
 - calculateAvgOxiState, [60](#)
 - Charge, [60](#)
 - charge, [61](#)
 - DisplayInfo, [60](#)
 - editAllOxidationStates, [60](#)
 - editCharge, [61](#)
 - editEnergy, [61](#)
 - editEnthalpy, [61](#)
 - editEntropy, [61](#)
 - editHS, [61](#)
 - editOneOxidationState, [61](#)
 - Energy, [61](#)
 - Enthalpy, [61](#)
 - Entropy, [61](#)
 - formation_energy, [61](#)
 - formation_enthalpy, [61](#)
 - formation_entropy, [62](#)
 - Formula, [62](#)
 - HaveEnergy, [61](#)
 - haveG, [62](#)
 - HaveHS, [61](#)
 - haveHS, [62](#)
 - isRegistered, [61](#)
 - molar_weight, [62](#)
 - MolarWeight, [61](#)
 - MolecularFormula, [61](#)
 - Molecule, [60](#)
 - MoleculeName, [61](#)
 - MoleculePhase, [61](#)
 - Name, [62](#)
 - Phase, [62](#)
 - recalculateMolarWeight, [61](#)
 - Register, [61](#)
 - registered, [62](#)
 - removeAllAtoms, [61](#)
 - removeOneAtom, [61](#)

- setFormula, [61](#)
 - setMolarWeigth, [61](#)
- MoleculeName
 - Molecule, [61](#)
- MoleculePhase
 - Molecule, [61](#)
- molefraction
 - MIXED_GAS, [58](#)
- molefractionCheck
 - skua.cpp, [157](#)
 - skua.h, [159](#)
- monkfish
 - ui.h, [173](#)
- monkfish.cpp, [141](#)
 - default_density, [142](#)
 - default_exterior_concentration, [142](#)
 - default_film_transfer, [142](#)
 - default_interparticle_diffusion, [142](#)
 - default_monk_adsorption, [142](#)
 - default_monk_equilibrium, [142](#)
 - default_monkfish_retardation, [142](#)
 - default_porosity, [142](#)
 - MONKFISH_TESTS, [142](#)
- monkfish.h, [142](#)
 - default_density, [143](#)
 - default_exterior_concentration, [143](#)
 - default_film_transfer, [143](#)
 - default_interparticle_diffusion, [143](#)
 - default_monk_adsorption, [143](#)
 - default_monk_equilibrium, [143](#)
 - default_monkfish_retardation, [143](#)
 - default_porosity, [143](#)
 - MONKFISH_TESTS, [143](#)
 - setup_MONKFISH_DATA, [143](#)
- mp
 - TRAJECTORY_DATA, [100](#)
- Ms
 - TRAJECTORY_DATA, [100](#)
- mspd_dat
 - MAGPIE_DATA, [50](#)
- Mu
 - egret.h, [118](#)
- mu_0
 - TRAJECTORY_DATA, [100](#)
- MultiScale
 - MONKFISH_DATA, [63](#)
- N
 - GMRESR_DATA, [38](#)
 - MIXED_GAS, [58](#)
 - Speciation_Test01_Data, [95](#)
 - SYSTEM_DATA, [98](#)
- NONE
 - yaml_wrapper.h, [180](#)
- n_par
 - GSTA_OPT_DATA, [44](#)
- n_rand
 - TRAJECTORY_DATA, [100](#)
- NE
 - FINCH_DATA, [28](#)
- NI
 - FINCH_DATA, [29](#)
- NL_Output
 - PJFNK_DATA, [76](#)
- NUM_JAC_DATA, [66](#)
 - dxj, [66](#)
 - eps, [66](#)
 - Fx, [66](#)
 - Fxp, [66](#)
- Na
 - gsta_opt.h, [128](#)
 - magpie.h, [139](#)
- NaT
 - Speciation_Test01_Data, [95](#)
- Name
 - Atom, [11](#)
 - MassBalance, [52](#)
 - Molecule, [62](#)
- name
 - SubHeader, [97](#)
- naturalLaplacian3D
 - Matrix, [56](#)
- NaturalState
 - Atom, [11](#)
- negative_mass
 - error.h, [120](#)
- negative_time
 - error.h, [120](#)
- Neutrons
 - Atom, [11](#)
- neutrons
 - Atom, [11](#)
- Newton_data
 - SHARK_DATA, [89](#)
- nl_bestres
 - PJFNK_DATA, [76](#)
- nl_iter
 - PJFNK_DATA, [76](#)
- nl_maxit
 - PJFNK_DATA, [76](#)
- nl_method
 - FINCH_DATA, [29](#)
- nl_picard
 - finch.cpp, [122](#)
 - finch.h, [125](#)
- nl_relres
 - PJFNK_DATA, [77](#)
- nl_res
 - PJFNK_DATA, [77](#)
- nl_res_base
 - PJFNK_DATA, [77](#)
- nl_tol_abs
 - PJFNK_DATA, [77](#)
- nl_tol_rel
 - PJFNK_DATA, [77](#)
- no_diffusion
 - error.h, [120](#)

- non_real_edge
 - error.h, [120](#)
- non_square_matrix
 - error.h, [119](#)
- NonLinear
 - DOGFISH_DATA, [24](#)
 - MONKFISH_DATA, [63](#)
 - SCOPSOWL_DATA, [83](#)
 - SKUA_DATA, [91](#)
- Norm
 - SHARK_DATA, [89](#)
- norm
 - Matrix, [56](#)
- normFkp1
 - BACKTRACK_DATA, [12](#)
- NormTrack
 - FINCH_DATA, [29](#)
- norms
 - GSTA_OPT_DATA, [44](#)
- not_a_token
 - error.h, [120](#)
- Nu
 - egret.h, [118](#)
- nullptr_error
 - error.h, [120](#)
- nullptr_func
 - error.h, [120](#)
- num_cols
 - Matrix, [57](#)
- num_curves
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- num_mbe
 - SHARK_DATA, [89](#)
- num_other
 - SHARK_DATA, [89](#)
- num_params
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- num_rows
 - Matrix, [57](#)
- num_ssr
 - SHARK_DATA, [89](#)
- num_usr
 - SHARK_DATA, [89](#)
- NumComp
 - DOGFISH_DATA, [24](#)
 - MONKFISH_DATA, [63](#)
- NumJac
 - Speciation_Test01_Data, [95](#)
- Number_Generator
 - Trajectory.cpp, [162](#)
 - Trajectory.h, [163](#)
- number_elements
 - PeriodicTable, [71](#)
- number_files
 - ui.cpp, [168](#)
 - ui.h, [176](#)
- NumericalJacobian
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- numvar
 - SHARK_DATA, [89](#)
- OE
 - FINCH_DATA, [29](#)
- OI
 - FINCH_DATA, [29](#)
- OPTRANS_DATA, [67](#)
 - Ai, [67](#)
 - li, [67](#)
- omega
 - BiCGSTAB_DATA, [15](#)
- omega_old
 - BiCGSTAB_DATA, [15](#)
- operator*
 - Matrix, [56](#)
- operator()
 - Document, [22](#)
 - Header, [47](#)
 - Matrix, [56](#)
 - YamlWrapper, [112](#)
- operator+
 - Matrix, [56](#)
- operator-
 - Matrix, [56](#)
- operator/
 - Matrix, [56](#)
- operator=
 - Document, [22](#)
 - Header, [47](#)
 - KeyValueMap, [50](#)
 - MasterSpeciesList, [53](#)
 - Matrix, [56](#)
 - SubHeader, [97](#)
 - ValueTypePair, [109](#)
 - YamlWrapper, [112](#)
- operatorTranspose
 - lark.cpp, [131](#)
 - lark.h, [135](#)
- opt_no_support
 - error.h, [120](#)
- opt_qmax
 - GSTA_OPT_DATA, [44](#)
- Optimize
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- option
 - UI_DATA, [103](#)
- orderMag
 - gsta_opt.cpp, [127](#)
 - gsta_opt.h, [129](#)
- ortho_check_fail
 - error.h, [120](#)
- orthoLinReg
 - gsta_opt.cpp, [127](#)
 - gsta_opt.h, [129](#)

- ospre_discretization
 - finch.cpp, [122](#)
 - finch.h, [125](#)
- other_data
 - SHARK_DATA, [89](#)
- OtherList
 - SHARK_DATA, [89](#)
- out_of_bounds
 - error.h, [119](#)
- Output
 - ARNOLDI_DATA, [8](#)
 - BiCGSTAB_DATA, [15](#)
 - CGS_DATA, [18](#)
 - GCR_DATA, [32](#)
 - GMRESLP_DATA, [35](#)
 - GMRESRP_DATA, [41](#)
 - MONKFISH_DATA, [64](#)
 - PCG_DATA, [69](#)
 - PICARD_DATA, [72](#)
 - SYSTEM_DATA, [98](#)
- OutputFile
 - DOGFISH_DATA, [24](#)
 - SCOPSOWL_DATA, [83](#)
 - SHARK_DATA, [89](#)
 - SKUA_DATA, [91](#)
- owl_dat
 - SCOPSOWL_OPT_DATA, [85](#)
- oxidation_state
 - Atom, [11](#)
- OxidationState
 - Atom, [11](#)
- P
 - GSTA_OPT_DATA, [44](#)
- p
 - BiCGSTAB_DATA, [15](#)
 - CGS_DATA, [18](#)
 - PCG_DATA, [69](#)
- PCG
 - lark.h, [134](#)
- PITZER
 - shark.h, [154](#)
- PCG_DATA, [67](#)
 - alpha, [68](#)
 - Ap, [68](#)
 - bestres, [68](#)
 - bestx, [68](#)
 - beta, [69](#)
 - iter, [69](#)
 - maxit, [69](#)
 - Output, [69](#)
 - p, [69](#)
 - r, [69](#)
 - r_old, [69](#)
 - relres, [69](#)
 - relres_base, [69](#)
 - res, [69](#)
 - tol_abs, [69](#)
 - tol_rel, [69](#)
 - x, [70](#)
 - z, [70](#)
 - z_old, [70](#)
- PE3
 - egret.h, [118](#)
- pH
 - SHARK_DATA, [89](#)
- pH_index
 - SHARK_DATA, [89](#)
- PI
 - maggie.cpp, [138](#)
 - maggie.h, [140](#)
 - SYSTEM_DATA, [98](#)
- PICARD_DATA, [71](#)
 - bestres, [72](#)
 - bestx, [72](#)
 - iter, [72](#)
 - maxit, [72](#)
 - Output, [72](#)
 - r, [72](#)
 - relres, [72](#)
 - relres_base, [72](#)
 - res, [72](#)
 - tol_abs, [73](#)
 - tol_rel, [73](#)
 - x0, [73](#)
- Plo
 - GPAST_DATA, [43](#)
- PJFNK_DATA, [73](#)
 - backtrack_dat, [75](#)
 - bestx, [75](#)
 - bicgstab_dat, [75](#)
 - Bounce, [75](#)
 - cgs_dat, [75](#)
 - eps, [75](#)
 - F, [75](#)
 - funeval, [75](#)
 - Fv, [75](#)
 - gcr_dat, [75](#)
 - gmreslp_dat, [75](#)
 - gmresr_dat, [76](#)
 - gmresrp_dat, [76](#)
 - L_Output, [76](#)
 - l_iter, [76](#)
 - lin_tol_abs, [76](#)
 - lin_tol_rel, [76](#)
 - LineSearch, [76](#)
 - linear_solver, [76](#)
 - NL_Output, [76](#)
 - nl_bestres, [76](#)
 - nl_iter, [76](#)
 - nl_maxit, [76](#)
 - nl_relres, [77](#)
 - nl_res, [77](#)
 - nl_res_base, [77](#)
 - nl_tol_abs, [77](#)
 - nl_tol_rel, [77](#)
 - pcg_dat, [77](#)

- precon, [77](#)
- precon_data, [77](#)
- res_data, [77](#)
- v, [77](#)
- x, [77](#)
- pOH_index
 - SHARK_DATA, [89](#)
- POL
 - TRAJECTORY_DATA, [101](#)
- POLAR
 - Trajectory.cpp, [162](#)
 - Trajectory.h, [163](#)
- PSI
 - egret.h, [118](#)
- PT
 - SYSTEM_DATA, [98](#)
- PURE_GAS, [78](#)
 - density, [78](#)
 - dynamic_viscosity, [78](#)
 - molecular_diffusion, [78](#)
 - molecular_weight, [78](#)
 - Schmidt, [78](#)
 - specific_heat, [78](#)
 - Sutherland_Const, [78](#)
 - Sutherland_Temp, [78](#)
 - Sutherland_Viscosity, [79](#)
- Par
 - SYSTEM_DATA, [98](#)
- param_dat
 - DOGFISH_DATA, [24](#)
 - MONKFISH_DATA, [64](#)
 - SCOPSOWL_DATA, [83](#)
 - SKUA_DATA, [91](#)
- param_data
 - FINCH_DATA, [29](#)
- param_guess
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- param_guess_old
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- ParamFile
 - SCOPSOWL_OPT_DATA, [85](#)
 - SKUA_OPT_DATA, [93](#)
- Path
 - UI_DATA, [103](#)
- path
 - ui.cpp, [168](#)
 - ui.h, [176](#)
 - UI_DATA, [103](#)
- pcg
 - lark.cpp, [131](#)
 - lark.h, [136](#)
- pcg_dat
 - PJFNK_DATA, [77](#)
- pellet_density
 - SCOPSOWL_DATA, [83](#)
- pellet_radius
 - SCOPSOWL_DATA, [83](#)
 - SKUA_DATA, [91](#)
- PeriodicTable, [70](#)
 - ~PeriodicTable, [71](#)
 - DisplayTable, [71](#)
 - number_elements, [71](#)
 - PeriodicTable, [71](#)
 - PeriodicTable, [71](#)
 - Table, [71](#)
- Phase
 - Molecule, [62](#)
- pi
 - SYSTEM_DATA, [98](#)
- picard
 - lark.cpp, [131](#)
 - lark.h, [136](#)
- picard_dat
 - FINCH_DATA, [29](#)
- pjfnk
 - lark.cpp, [131](#)
 - lark.h, [136](#)
- pjfnk_dat
 - FINCH_DATA, [29](#)
- Po
 - egret.h, [118](#)
 - gsta_opt.h, [128](#)
 - magpie.h, [139](#)
- po
 - GPAST_DATA, [43](#)
- poi
 - GPAST_DATA, [43](#)
- pore_diffusion
 - SCOPSOWL_PARAM_DATA, [86](#)
- porosity
 - TRAJECTORY_DATA, [101](#)
- Precipitation, [78](#)
- precon
 - PJFNK_DATA, [77](#)
- precon_data
 - PJFNK_DATA, [77](#)
 - SHARK_DATA, [89](#)
- pres
 - FINCH_DATA, [29](#)
- present
 - GPAST_DATA, [43](#)
- previous_token
 - yaml_cpp_class, [110](#)
- Print2Console
 - DOGFISH_DATA, [24](#)
 - MONKFISH_DATA, [64](#)
 - SCOPSOWL_DATA, [83](#)
 - SKUA_DATA, [91](#)
- Print2File
 - DOGFISH_DATA, [24](#)
 - MONKFISH_DATA, [64](#)
 - SCOPSOWL_DATA, [83](#)
 - SKUA_DATA, [91](#)
- print2file_DOGFISH_header

- dogfish.cpp, 114
- dogfish.h, 115
- print2file_DOGFISH_result_new
 - dogfish.cpp, 114
 - dogfish.h, 115
- print2file_DOGFISH_result_old
 - dogfish.cpp, 114
 - dogfish.h, 115
- print2file_SCOPSOWL_header
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- print2file_SCOPSOWL_result_new
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- print2file_SCOPSOWL_result_old
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- print2file_SCOPSOWL_time_header
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- print2file_SKUA_header
 - skua.cpp, 157
 - skua.h, 159
- print2file_SKUA_results_new
 - skua.cpp, 157
 - skua.h, 159
- print2file_SKUA_results_old
 - skua.cpp, 157
 - skua.h, 159
- print2file_SKUA_time_header
 - skua.cpp, 157
 - skua.h, 159
- print2file_dim_header
 - finch.cpp, 122
 - finch.h, 125
- print2file_newline
 - finch.cpp, 122
 - finch.h, 125
- print2file_result_new
 - finch.cpp, 122
 - finch.h, 125
- print2file_result_old
 - finch.cpp, 122
 - finch.h, 125
- print2file_shark_header
 - shark.cpp, 151
 - shark.h, 154
- print2file_shark_info
 - shark.cpp, 151
 - shark.h, 155
- print2file_shark_results_new
 - shark.cpp, 151
 - shark.h, 155
- print2file_shark_results_old
 - shark.cpp, 151
 - shark.h, 155
- print2file_species_header
 - dogfish.cpp, 114
- dogfish.h, 115
- scopsowl.cpp, 146
- scopsowl.h, 148
- skua.cpp, 157
- skua.h, 159
- print2file_tab
 - finch.cpp, 122
 - finch.h, 125
- print2file_time_header
 - finch.cpp, 122
 - finch.h, 125
- Protons
 - Atom, 11
- protons
 - Atom, 11
- Pstd
 - egret.h, 118
- q
 - GPAST_DATA, 43
 - GSTA_OPT_DATA, 44
- q_bar
 - TRAJECTORY_DATA, 101
- q_data
 - SCOPSOWL_OPT_DATA, 85
 - SKUA_OPT_DATA, 93
- Q_in
 - TRAJECTORY_DATA, 101
- q_p
 - magpie.cpp, 138
 - magpie.h, 140
- q_sim
 - SCOPSOWL_OPT_DATA, 85
 - SKUA_OPT_DATA, 93
- qAvg
 - SCOPSOWL_PARAM_DATA, 86
- qAvg_old
 - SCOPSOWL_PARAM_DATA, 86
- qIntegralAvg
 - SCOPSOWL_PARAM_DATA, 86
- qIntegralAvg_old
 - SCOPSOWL_PARAM_DATA, 86
- qT
 - magpie.cpp, 138
 - magpie.h, 140
 - SYSTEM_DATA, 98
- qTn
 - SKUA_DATA, 91
- qTnp1
 - SKUA_DATA, 91
- qmax
 - GSTA_DATA, 43
 - GSTA_OPT_DATA, 44
- qo
 - GPAST_DATA, 43
 - magpie.cpp, 138
 - magpie.h, 140
 - SCOPSOWL_PARAM_DATA, 86
- Qst

- magpie.cpp, 138
- magpie.h, 140
- SCOPSOWL_PARAM_DATA, 86
- Qst_old
 - SCOPSOWL_PARAM_DATA, 86
- QstAvg
 - SCOPSOWL_PARAM_DATA, 86
- QstAvg_old
 - SCOPSOWL_PARAM_DATA, 86
- Qstn
 - SKUA_PARAM, 94
- Qstnp1
 - SKUA_PARAM, 94
- Qsto
 - SCOPSOWL_PARAM_DATA, 87
- R
 - gsta_opt.h, 128
 - magpie.h, 139
- r
 - BiCGSTAB_DATA, 15
 - CGS_DATA, 18
 - GCR_DATA, 33
 - GMRESLP_DATA, 35
 - GMRESRP_DATA, 41
 - PCG_DATA, 69
 - PICARD_DATA, 72
- r0
 - BiCGSTAB_DATA, 15
 - CGS_DATA, 19
- r_old
 - PCG_DATA, 69
- RADIAL_FORCE
 - Trajectory.cpp, 162
 - Trajectory.h, 163
- RE3
 - egret.h, 118
- RIC
 - FINCH_DATA, 29
- rSq
 - gsta_opt.cpp, 127
 - gsta_opt.h, 129
- RUN_SANDBOX
 - sandbox.cpp, 143
 - sandbox.h, 144
- ReNum
 - egret.h, 118
- Reaction, 79
 - ~Reaction, 80
 - calculateEnergies, 80
 - calculateEquilibrium, 80
 - CanCalcG, 80
 - CanCalcHS, 80
 - checkSpeciesEnergies, 80
 - Display_Info, 80
 - energy, 81
 - enthalpy, 81
 - entropy, 81
 - Equilibrium, 81
 - Eval_Residual, 80
 - Get_Energy, 80
 - Get_Enthalpy, 80
 - Get_Entropy, 80
 - Get_Equilibrium, 80
 - Get_Stoichiometric, 80
 - HaveEquil, 81
 - haveEquilibrium, 80
 - HaveG, 81
 - HaveHS, 81
 - Initialize_List, 80
 - List, 81
 - Reaction, 80
 - Set_Energy, 80
 - Set_Enthalpy, 80
 - Set_EnthalpyANDEntropy, 80
 - Set_Entropy, 80
 - Set_Equilibrium, 80
 - Set_Stoichiometric, 80
 - Stoichiometric, 81
- ReactionList
 - SHARK_DATA, 89
- reactions
 - Mechanism, 58
- read_error
 - error.h, 120
- read_equilrxn
 - shark.cpp, 151
 - shark.h, 155
- read_massbalance
 - shark.cpp, 151
 - shark.h, 155
- read_options
 - shark.cpp, 151
 - shark.h, 155
- read_scenario
 - shark.cpp, 151
 - shark.h, 155
- read_species
 - shark.cpp, 151
 - shark.h, 155
- read_unsteadyrxn
 - shark.cpp, 151
 - shark.h, 155
- readInputFile
 - yaml_cpp_class, 110
- recalculateMolarWeight
 - Molecule, 61
- Recover
 - SYSTEM_DATA, 98
- ref_diffusion
 - SCOPSOWL_PARAM_DATA, 87
 - SKUA_PARAM, 94
- ref_pressure
 - SCOPSOWL_PARAM_DATA, 87
 - SKUA_PARAM, 94
- ref_temperature
 - SCOPSOWL_PARAM_DATA, 87

- SKUA_PARAM, 94
- Register
 - Atom, 11
 - Molecule, 61
- registered
 - Molecule, 62
- rel_tol_norm
 - SCOPSOWL_OPT_DATA, 85
 - SKUA_OPT_DATA, 93
- relres
 - BiCGSTAB_DATA, 15
 - CGS_DATA, 19
 - GCR_DATA, 33
 - GMRESLP_DATA, 35
 - GMRESRP_DATA, 41
 - PCG_DATA, 69
 - PICARD_DATA, 72
- relres_base
 - BiCGSTAB_DATA, 15
 - CGS_DATA, 19
 - GCR_DATA, 33
 - GMRESLP_DATA, 35
 - GMRESRP_DATA, 41
 - PCG_DATA, 69
 - PICARD_DATA, 72
- Removal_Efficiency
 - Trajectory.cpp, 162
 - Trajectory.h, 163
- removeAllAtoms
 - Molecule, 61
- removeElectron
 - Atom, 11
- removeNeutron
 - Atom, 11
- removeOneAtom
 - Molecule, 61
- removeProton
 - Atom, 11
- res
 - BiCGSTAB_DATA, 15
 - CGS_DATA, 19
 - FINCH_DATA, 29
 - GCR_DATA, 33
 - GMRESLP_DATA, 35
 - GMRESRP_DATA, 41
 - PCG_DATA, 69
 - PICARD_DATA, 72
- res_data
 - PJFNK_DATA, 77
- resetKeys
 - Document, 22
 - Header, 47
 - YamlWrapper, 112
- resettime
 - FINCH_DATA, 29
- Residual
 - SHARK_DATA, 89
- residual_alkalinity
 - MasterSpeciesList, 53
- residual_data
 - SHARK_DATA, 89
- restart
 - GCR_DATA, 33
 - GMRESLP_DATA, 35
 - GMRESRP_DATA, 41
- revalidateAllKeys
 - Document, 22
 - YamlWrapper, 112
- reverse_rate
 - UnsteadyReaction, 107
- reverse_ref_rate
 - UnsteadyReaction, 107
- Reynolds
 - MIXED_GAS, 58
- rho
 - BACKTRACK_DATA, 12
 - BiCGSTAB_DATA, 15
 - CGS_DATA, 19
- rho_f
 - TRAJECTORY_DATA, 101
- rho_old
 - BiCGSTAB_DATA, 15
- rho_p
 - TRAJECTORY_DATA, 101
- Rn
 - FINCH_DATA, 29
- Rnp1
 - FINCH_DATA, 29
- Ro
 - FINCH_DATA, 29
- Rough
 - SCOPSOWL_OPT_DATA, 85
 - SKUA_OPT_DATA, 93
- roundIt
 - gsta_opt.cpp, 127
 - gsta_opt.h, 129
- rowExtend
 - Matrix, 56
- rowExtract
 - Matrix, 56
- rowReplace
 - Matrix, 56
- rowShrink
 - Matrix, 56
- rows
 - Matrix, 56
- Rs
 - TRAJECTORY_DATA, 101
- Rstd
 - egret.h, 118
 - shark.h, 154
- Run_Trajectory
 - Trajectory.cpp, 162
 - Trajectory.h, 164
- run_exec
 - ui.cpp, 168

- ui.h, 176
- run_executable
 - ui.cpp, 168
 - ui.h, 176
- run_test
 - ui.cpp, 168
 - ui.h, 176
- rxn_rate_error
 - error.h, 120
- s
 - BiCGSTAB_DATA, 16
 - FINCH_DATA, 29
 - mSPD_DATA, 65
- SIT
 - shark.h, 154
- STRING
 - yaml_wrapper.h, 180
- s_rand
 - TRAJECTORY_DATA, 101
- SCOPSOWL
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- SCOPSOWL_DATA, 81
 - binder_fraction, 82
 - binder_poresize, 82
 - binder_porosity, 82
 - char_macro, 82
 - char_micro, 82
 - coord_macro, 82
 - coord_micro, 82
 - crystal_radius, 82
 - DirichletBC, 82
 - eval_ads, 82
 - eval_diff, 82
 - eval_kf, 82
 - eval_retard, 82
 - eval_surfDiff, 82
 - finch_dat, 82
 - gas_dat, 82
 - gas_temperature, 82
 - gas_velocity, 82
 - Heterogeneous, 83
 - level, 83
 - magpie_dat, 83
 - NonLinear, 83
 - OutputFile, 83
 - param_dat, 83
 - pellet_density, 83
 - pellet_radius, 83
 - Print2Console, 83
 - Print2File, 83
 - sim_time, 83
 - skua_dat, 83
 - SurfDiff, 83
 - t, 83
 - t_counter, 83
 - t_old, 83
 - t_print, 83
 - tempy, 83
 - total_pressure, 83
 - total_steps, 83
 - user_data, 83
 - y, 83
- SCOPSOWL_Executioner
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- SCOPSOWL_HPP_
 - scopsowl.h, 148
- SCOPSOWL_OPT_DATA, 83
 - adsorb_index, 84
 - CompareFile, 84
 - current_equil, 84
 - current_points, 84
 - current_press, 84
 - current_temp, 84
 - diffusion_type, 84
 - e_norm, 84
 - evaluation, 85
 - f_bias, 85
 - max_bias, 85
 - min_bias, 85
 - num_curves, 85
 - num_params, 85
 - Optimize, 85
 - owl_dat, 85
 - param_guess, 85
 - ParamFile, 85
 - q_data, 85
 - q_sim, 85
 - Rough, 85
 - simulation_equil, 85
 - t, 85
 - total_eval, 85
 - y_base, 85
- SCOPSOWL_OPT_set_y
 - scopsowl_opt.cpp, 149
 - scopsowl_opt.h, 150
- SCOPSOWL_OPTIMIZE
 - scopsowl_opt.cpp, 149
 - scopsowl_opt.h, 150
- SCOPSOWL_PARAM_DATA, 85
 - Adsorbable, 86
 - affinity, 86
 - qAvg, 86
 - qo, 86
 - Qst, 86
 - QstAvg, 86
 - Qsto, 87
 - speciesName, 87
- SCOPSOWL_SCENARIOS
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- SCOPSOWL_TESTS
 - scopsowl.cpp, 146
 - scopsowl.h, 149
- SCOPSOWL_postprocesses

- scopsowl.cpp, 146
- scopsowl.h, 148
- SCOPSOWL_preprocesses
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- SCOPSOWL_reset
 - scopsowl.cpp, 146
 - scopsowl.h, 148
- SHARK
 - shark.cpp, 152
 - shark.h, 155
- SHARK_DATA, 87
 - act_fun, 88
 - activity_data, 88
 - activity_new, 88
 - activity_old, 88
 - Conc_new, 88
 - Conc_old, 88
 - Console_Output, 88
 - const_pH, 88
 - Contains_pH, 88
 - Contains_pOH, 88
 - Converged, 88
 - dielectric_const, 88
 - dt, 88
 - dt_min, 88
 - EvalActivity, 89
 - File_Output, 89
 - lin_precon, 89
 - MassBalanceList, 89
 - MasterList, 89
 - Newton_data, 89
 - Norm, 89
 - num_mbe, 89
 - num_other, 89
 - num_ssr, 89
 - num_usr, 89
 - numvar, 89
 - other_data, 89
 - OtherList, 89
 - OutputFile, 89
 - pH, 89
 - pH_index, 89
 - pOH_index, 89
 - precon_data, 89
 - ReactionList, 89
 - Residual, 89
 - residual_data, 89
 - shark.h, 154
 - simulationtime, 89
 - SpeciationCurve, 89
 - steadystate, 89
 - t_count, 89
 - t_out, 89
 - temperature, 89
 - time, 90
 - time_old, 90
 - TimeAdaptivity, 90
 - timesteps, 90
 - totalsteps, 90
 - UnsteadyList, 90
 - X_new, 90
 - X_old, 90
 - yaml_object, 90
- SHARK_SCENARIO
 - shark.cpp, 152
 - shark.h, 155
- SHARK_TESTS
 - shark.cpp, 152
 - shark.h, 155
- SKUA
 - skua.cpp, 157
 - skua.h, 159
- SKUA_CYCLE_TEST01
 - skua.cpp, 157
 - skua.h, 159
- SKUA_CYCLE_TEST02
 - skua.cpp, 157
 - skua.h, 159
- SKUA_DATA, 90
 - char_measure, 91
 - coord, 91
 - DirichletBC, 91
 - eval_diff, 91
 - eval_kf, 91
 - finch_dat, 91
 - gas_dat, 91
 - gas_velocity, 91
 - magpie_dat, 91
 - NonLinear, 91
 - OutputFile, 91
 - param_dat, 91
 - pellet_radius, 91
 - Print2Console, 91
 - Print2File, 91
 - qTn, 91
 - qTnp1, 91
 - sim_time, 91
 - t, 91
 - t_counter, 91
 - t_old, 91
 - t_print, 91
 - total_steps, 91
 - user_data, 91
 - y, 91
- SKUA_Executioner
 - skua.cpp, 157
 - skua.h, 159
- SKUA_HPP_
 - skua.h, 158
- SKUA_LOW_TEST03
 - skua.cpp, 157
 - skua.h, 159
- SKUA_MID_TEST04
 - skua.cpp, 157
 - skua.h, 159

- SKUA_OPT_DATA, 92
 - abs_tol_bias, 92
 - adsorb_index, 92
 - CompareFile, 92
 - current_equil, 92
 - current_points, 92
 - current_press, 92
 - current_temp, 92
 - diffusion_type, 92
 - e_norm, 93
 - e_norm_old, 93
 - evaluation, 93
 - f_bias, 93
 - f_bias_old, 93
 - max_bias, 93
 - max_guess_iter, 93
 - min_bias, 93
 - num_curves, 93
 - num_params, 93
 - Optimize, 93
 - param_guess, 93
 - param_guess_old, 93
 - ParamFile, 93
 - q_data, 93
 - q_sim, 93
 - rel_tol_norm, 93
 - Rough, 93
 - simulation_equil, 93
 - skua_dat, 93
 - t, 93
 - total_eval, 93
 - y_base, 93
- SKUA_OPT_set_y
 - skua_opt.cpp, 160
 - skua_opt.h, 160
- SKUA_OPTIMIZE
 - skua_opt.cpp, 160
 - skua_opt.h, 160
- SKUA_PARAM, 93
 - activation_energy, 94
 - Adsorbable, 94
 - affinity, 94
 - film_transfer, 94
 - Qstn, 94
 - Qstnp1, 94
 - ref_diffusion, 94
 - ref_pressure, 94
 - ref_temperature, 94
 - speciesName, 94
 - xC, 94
 - xn, 94
 - xnp1, 94
 - y_eff, 94
- SKUA_SCENARIOS
 - skua.cpp, 157
 - skua.h, 159
- SKUA_TESTS
 - skua.cpp, 157
 - skua.h, 159
- SKUA_postprocesses
 - skua.cpp, 157
 - skua.h, 159
- SKUA_preprocesses
 - skua.cpp, 157
 - skua.h, 159
- SKUA_reset
 - skua.cpp, 157
 - skua.h, 159
- SMALL_CYCLE_TEST02
 - scopsowl.cpp, 146
 - scopsowl.h, 149
- SYSTEM_DATA, 97
 - As, 98
 - avg_norm, 98
 - Carrier, 98
 - I, 98
 - Ideal, 98
 - J, 98
 - K, 98
 - max_norm, 98
 - N, 98
 - Output, 98
 - PI, 98
 - PT, 98
 - Par, 98
 - pi, 98
 - qT, 98
 - Recover, 98
 - Sys, 98
 - T, 98
 - total_eval, 99
- sandbox
 - ui.h, 173
- sandbox.cpp, 143
 - RUN_SANDBOX, 143
 - Speciation_Test01_Function, 143
 - Speciation_Test01_Guess, 144
 - Speciation_Test01_Jacobian, 144
 - Speciation_Test01_MatVec, 144
- sandbox.h, 144
 - RUN_SANDBOX, 144
 - Speciation_Test01_Function, 144
 - Speciation_Test01_Guess, 144
 - Speciation_Test01_Jacobian, 144
 - Speciation_Test01_MatVec, 144
- ScNum
 - egret.h, 118
- scenario_fail
 - error.h, 119
- Schmidt
 - PURE_GAS, 78
- school.h, 144
- scops_opt
 - ui.h, 173
- scopsowl
 - ui.h, 173

- scopsowl.cpp, 145
 - CURVE_TEST03, 145
 - CURVE_TEST04, 145
 - CURVE_TEST05, 145
 - const_filmMassTransfer, 145
 - const_pore_diffusion, 145
 - default_adsorption, 145
 - default_effective_diffusion, 146
 - default_filmMassTransfer, 146
 - default_pore_diffusion, 146
 - default_retardation, 146
 - default_surf_diffusion, 146
 - LARGE_CYCLE_TEST01, 146
 - print2file_SCOPSOWL_header, 146
 - print2file_SCOPSOWL_result_new, 146
 - print2file_SCOPSOWL_result_old, 146
 - print2file_SCOPSOWL_time_header, 146
 - print2file_species_header, 146
 - SCOPSOWL, 146
 - SCOPSOWL_Executioner, 146
 - SCOPSOWL_SCENARIOS, 146
 - SCOPSOWL_TESTS, 146
 - SCOPSOWL_postprocesses, 146
 - SCOPSOWL_preprocesses, 146
 - SCOPSOWL_reset, 146
 - SMALL_CYCLE_TEST02, 146
 - set_SCOPSOWL_ICs, 146
 - set_SCOPSOWL_params, 146
 - set_SCOPSOWL_timestep, 146
 - setup_SCOPSOWL_DATA, 146
- scopsowl.h, 146
 - avgDp, 148
 - CURVE_TEST03, 148
 - CURVE_TEST04, 148
 - CURVE_TEST05, 148
 - const_filmMassTransfer, 148
 - const_pore_diffusion, 148
 - default_adsorption, 148
 - default_effective_diffusion, 148
 - default_filmMassTransfer, 148
 - default_pore_diffusion, 148
 - default_retardation, 148
 - default_surf_diffusion, 148
 - Dk, 148
 - Dp, 148
 - LARGE_CYCLE_TEST01, 148
 - print2file_SCOPSOWL_header, 148
 - print2file_SCOPSOWL_result_new, 148
 - print2file_SCOPSOWL_result_old, 148
 - print2file_SCOPSOWL_time_header, 148
 - print2file_species_header, 148
 - SCOPSOWL, 148
 - SCOPSOWL_Executioner, 148
 - SCOPSOWL_HPP_, 148
 - SCOPSOWL_SCENARIOS, 148
 - SCOPSOWL_TESTS, 149
 - SCOPSOWL_postprocesses, 148
 - SCOPSOWL_preprocesses, 148
 - SCOPSOWL_reset, 148
 - SMALL_CYCLE_TEST02, 149
 - set_SCOPSOWL_ICs, 149
 - set_SCOPSOWL_params, 149
 - set_SCOPSOWL_timestep, 149
 - setup_SCOPSOWL_DATA, 149
- scopsowl_opt.cpp, 149
 - eval_SCOPSOWL_Uptake, 149
 - initial_guess_SCOPSOWL, 149
 - SCOPSOWL_OPT_set_y, 149
 - SCOPSOWL_OPTIMIZE, 149
- scopsowl_opt.h, 149
 - eval_SCOPSOWL_Uptake, 150
 - initial_guess_SCOPSOWL, 150
 - SCOPSOWL_OPT_set_y, 150
 - SCOPSOWL_OPTIMIZE, 150
- Set_ActivationEnergy
 - UnsteadyReaction, 106
- Set_Affinity
 - UnsteadyReaction, 106
- set_DOGFISH_ICs
 - dogfish.cpp, 114
 - dogfish.h, 115
- set_DOGFISH_params
 - dogfish.cpp, 114
 - dogfish.h, 115
- set_DOGFISH_timestep
 - dogfish.cpp, 114
 - dogfish.h, 115
- Set_Delta
 - MassBalance, 51
- Set_Energy
 - Reaction, 80
 - UnsteadyReaction, 106
- Set_Enthalpy
 - Reaction, 80
 - UnsteadyReaction, 106
- Set_EnthalpyANDEntropy
 - Reaction, 80
 - UnsteadyReaction, 107
- Set_Entropy
 - Reaction, 80
 - UnsteadyReaction, 107
- Set_Equilibrium
 - Reaction, 80
 - UnsteadyReaction, 107
- Set_Forward
 - UnsteadyReaction, 107
- Set_ForwardRef
 - UnsteadyReaction, 107
- Set_InitialValue
 - UnsteadyReaction, 107
- Set_MaximumValue
 - UnsteadyReaction, 107
- Set_Name
 - MassBalance, 51
- Set_Reverse
 - UnsteadyReaction, 107

- Set_ReverseRef
 - UnsteadyReaction, [107](#)
- set_SCOPSOWL_ICs
 - scopsowl.cpp, [146](#)
 - scopsowl.h, [149](#)
- set_SCOPSOWL_params
 - scopsowl.cpp, [146](#)
 - scopsowl.h, [149](#)
- set_SCOPSOWL_timestep
 - scopsowl.cpp, [146](#)
 - scopsowl.h, [149](#)
- set_SKUA_ICs
 - skua.cpp, [157](#)
 - skua.h, [159](#)
- set_SKUA_params
 - skua.cpp, [157](#)
 - skua.h, [159](#)
- set_SKUA_timestep
 - skua.cpp, [157](#)
 - skua.h, [159](#)
- Set_Species_Index
 - UnsteadyReaction, [107](#)
- Set_Stoichiometric
 - Reaction, [80](#)
 - UnsteadyReaction, [107](#)
- Set_TimeStep
 - UnsteadyReaction, [107](#)
- Set_TotalConcentration
 - MassBalance, [51](#)
- set_alkalinity
 - MasterSpeciesList, [53](#)
- set_list_size
 - MasterSpeciesList, [53](#)
- set_size
 - Matrix, [56](#)
- set_species
 - MasterSpeciesList, [53](#)
- set_variables
 - egret.cpp, [117](#)
 - egret.h, [118](#)
- setAlias
 - Document, [22](#)
 - Header, [47](#)
 - SubHeader, [97](#)
- setFormula
 - Molecule, [61](#)
- setInputFile
 - yaml_cpp_class, [110](#)
- setMolarWeigth
 - Molecule, [61](#)
- setName
 - Document, [22](#)
 - Header, [47](#)
 - SubHeader, [97](#)
- setNameAliasPair
 - Document, [22](#)
 - Header, [47](#)
 - SubHeader, [97](#)
- setState
 - Document, [22](#)
 - Header, [47](#)
 - SubHeader, [97](#)
- setbcs
 - FINCH_DATA, [29](#)
- setic
 - FINCH_DATA, [29](#)
- setparams
 - FINCH_DATA, [29](#)
- setpostprocess
 - FINCH_DATA, [29](#)
- setpreprocess
 - FINCH_DATA, [29](#)
- settime
 - FINCH_DATA, [29](#)
- setup_DOGFISH_DATA
 - dogfish.cpp, [114](#)
 - dogfish.h, [115](#)
- setup_FINCH_DATA
 - finch.cpp, [122](#)
 - finch.h, [125](#)
- setup_MONKFISH_DATA
 - monkfish.h, [143](#)
- setup_SCOPSOWL_DATA
 - scopsowl.cpp, [146](#)
 - scopsowl.h, [149](#)
- setup_SHARK_DATA
 - shark.cpp, [151](#)
 - shark.h, [155](#)
- setup_SKUA_DATA
 - skua.cpp, [157](#)
 - skua.h, [159](#)
- shapeFactor
 - magpie.h, [139](#)
- shark
 - ui.h, [173](#)
- shark.h
 - DAVIES, [154](#)
 - DAVIES_LADSHAW, [154](#)
 - DEBYE_HUCKEL, [154](#)
 - IDEAL, [154](#)
 - PITZER, [154](#)
 - SIT, [154](#)
- shark.cpp, [150](#)
 - act_choice, [151](#)
 - Convert2Concentration, [151](#)
 - Convert2LogConcentration, [151](#)
 - Davies_equation, [151](#)
 - DaviesLadshaw_equation, [151](#)
 - DebyeHuckel_equation, [151](#)
 - ideal_solution, [151](#)
 - linearsolve_choice, [151](#)
 - linsearch_choice, [151](#)
 - print2file_shark_header, [151](#)
 - print2file_shark_info, [151](#)
 - print2file_shark_results_new, [151](#)
 - print2file_shark_results_old, [151](#)

- read_equilrxn, 151
- read_massbalance, 151
- read_options, 151
- read_scenario, 151
- read_species, 151
- read_unsteadyrxn, 151
- SHARK, 152
- SHARK_SCENARIO, 152
- SHARK_TESTS, 152
- setup_SHARK_DATA, 151
- shark_add_customResidual, 152
- shark_energy_calculations, 152
- shark_executioner, 152
- shark_guess, 152
- shark_initial_conditions, 152
- shark_pH_finder, 152
- shark_parameter_check, 152
- shark_postprocesses, 152
- shark_preprocesses, 152
- shark_reset, 152
- shark_residual, 152
- shark_solver, 152
- shark_temperature_calculations, 152
- shark_timestep_adapt, 152
- shark_timestep_const, 152
- shark.h, 152
 - act_choice, 154
 - Convert2Concentration, 154
 - Convert2LogConcentration, 154
 - Davies_equation, 154
 - DaviesLadshaw_equation, 154
 - DebyeHuckel_equation, 154
 - ideal_solution, 154
 - linearsolve_choice, 154
 - linesearch_choice, 154
 - print2file_shark_header, 154
 - print2file_shark_info, 155
 - print2file_shark_results_new, 155
 - print2file_shark_results_old, 155
 - read_equilrxn, 155
 - read_massbalance, 155
 - read_options, 155
 - read_scenario, 155
 - read_species, 155
 - read_unsteadyrxn, 155
 - Rstd, 154
 - SHARK, 155
 - SHARK_DATA, 154
 - SHARK_SCENARIO, 155
 - SHARK_TESTS, 155
 - setup_SHARK_DATA, 155
 - shark_add_customResidual, 155
 - shark_energy_calculations, 155
 - shark_executioner, 155
 - shark_guess, 155
 - shark_initial_conditions, 155
 - shark_pH_finder, 155
 - shark_parameter_check, 155
 - shark_postprocesses, 155
 - shark_preprocesses, 155
 - shark_reset, 155
 - shark_residual, 155
 - shark_solver, 155
 - shark_temperature_calculations, 155
 - shark_timestep_adapt, 155
 - shark_timestep_const, 155
 - valid_act, 154
- shark_add_customResidual
 - shark.cpp, 152
 - shark.h, 155
- shark_energy_calculations
 - shark.cpp, 152
 - shark.h, 155
- shark_executioner
 - shark.cpp, 152
 - shark.h, 155
- shark_guess
 - shark.cpp, 152
 - shark.h, 155
- shark_initial_conditions
 - shark.cpp, 152
 - shark.h, 155
- shark_pH_finder
 - shark.cpp, 152
 - shark.h, 155
- shark_parameter_check
 - shark.cpp, 152
 - shark.h, 155
- shark_postprocesses
 - shark.cpp, 152
 - shark.h, 155
- shark_preprocesses
 - shark.cpp, 152
 - shark.h, 155
- shark_reset
 - shark.cpp, 152
 - shark.h, 155
- shark_residual
 - shark.cpp, 152
 - shark.h, 155
- shark_solver
 - shark.cpp, 152
 - shark.h, 155
- shark_temperature_calculations
 - shark.cpp, 152
 - shark.h, 155
- shark_timestep_adapt
 - shark.cpp, 152
 - shark.h, 156
- shark_timestep_const
 - shark.cpp, 152
 - shark.h, 156
- sigma
 - CGS_DATA, 19
- sigma_m
 - TRAJECTORY_DATA, 101

- sigma_n
 - TRAJECTORY_DATA, 101
- sigma_v
 - TRAJECTORY_DATA, 101
- sigma_vz
 - TRAJECTORY_DATA, 101
- sigma_z
 - TRAJECTORY_DATA, 101
- sim_time
 - SCOPSOWL_DATA, 83
 - SKUA_DATA, 91
- simple_darken_Dc
 - skua.cpp, 157
 - skua.h, 159
- simulation_fail
 - error.h, 119
- simulation_equil
 - SCOPSOWL_OPT_DATA, 85
 - SKUA_OPT_DATA, 93
- simulationtime
 - SHARK_DATA, 89
- single_fiber_density
 - MONKFISH_DATA, 64
- singular_matrix
 - error.h, 120
- size
 - Document, 22
 - Header, 47
 - KeyValueMap, 50
 - MasterSpeciesList, 53
 - YamlWrapper, 112
- skua
 - ui.h, 173
- skua.cpp, 156
 - const_Dc, 156
 - const_kf, 156
 - default_Dc, 156
 - default_kf, 156
 - empirical_kf, 156
 - molefractionCheck, 157
 - print2file_SKUA_header, 157
 - print2file_SKUA_results_new, 157
 - print2file_SKUA_results_old, 157
 - print2file_SKUA_time_header, 157
 - print2file_species_header, 157
 - SKUA, 157
 - SKUA_CYCLE_TEST01, 157
 - SKUA_CYCLE_TEST02, 157
 - SKUA_Executioner, 157
 - SKUA_LOW_TEST03, 157
 - SKUA_MID_TEST04, 157
 - SKUA_SCENARIOS, 157
 - SKUA_TESTS, 157
 - SKUA_postprocesses, 157
 - SKUA_preprocesses, 157
 - SKUA_reset, 157
 - set_SKUA_ICs, 157
 - set_SKUA_params, 157
- set_SKUA_timestep, 157
- setup_SKUA_DATA, 157
- simple_darken_Dc, 157
- theoretical_darken_Dc, 157
- skua.h, 157
 - const_Dc, 159
 - const_kf, 159
 - D_c, 158
 - D_inf, 158
 - D_o, 158
 - default_Dc, 159
 - default_kf, 159
 - empirical_kf, 159
 - molefractionCheck, 159
 - print2file_SKUA_header, 159
 - print2file_SKUA_results_new, 159
 - print2file_SKUA_results_old, 159
 - print2file_SKUA_time_header, 159
 - print2file_species_header, 159
 - SKUA, 159
 - SKUA_CYCLE_TEST01, 159
 - SKUA_CYCLE_TEST02, 159
 - SKUA_Executioner, 159
 - SKUA_HPP_, 158
 - SKUA_LOW_TEST03, 159
 - SKUA_MID_TEST04, 159
 - SKUA_SCENARIOS, 159
 - SKUA_TESTS, 159
 - SKUA_postprocesses, 159
 - SKUA_preprocesses, 159
 - SKUA_reset, 159
 - set_SKUA_ICs, 159
 - set_SKUA_params, 159
 - set_SKUA_timestep, 159
 - setup_SKUA_DATA, 159
 - simple_darken_Dc, 159
 - theoretical_darken_Dc, 159
- skua_opt
 - ui.h, 173
- skua_dat
 - SCOPSOWL_DATA, 83
 - SKUA_OPT_DATA, 93
- skua_opt.cpp, 160
 - eval_SKUA_Uptake, 160
 - initial_guess_SKUA, 160
 - SKUA_OPT_set_y, 160
 - SKUA_OPTIMIZE, 160
- skua_opt.h, 160
 - eval_SKUA_Uptake, 160
 - initial_guess_SKUA, 160
 - SKUA_OPT_set_y, 160
 - SKUA_OPTIMIZE, 160
- Sn
 - FINCH_DATA, 29
- Snp1
 - FINCH_DATA, 29
- SolnTransform
 - Matrix, 56

- solve
 - FINCH_DATA, 29
- sorbed_molefraction
 - DOGFISH_PARAM, 25
 - MONKFISH_PARAM, 65
- sorption_bc
 - MONKFISH_PARAM, 65
- Speciation_Test01_Data, 94
 - C, 95
 - CT, 95
 - Jacobian, 95
 - logC, 95
 - logKa1, 95
 - logKa2, 95
 - logKw, 95
 - N, 95
 - NaT, 95
 - NumJac, 95
 - x, 95
- Speciation_Test01_Function
 - sandbox.cpp, 143
 - sandbox.h, 144
- Speciation_Test01_Guess
 - sandbox.cpp, 144
 - sandbox.h, 144
- Speciation_Test01_Jacobian
 - sandbox.cpp, 144
 - sandbox.h, 144
- Speciation_Test01_MatVec
 - sandbox.cpp, 144
 - sandbox.h, 144
- SpeciationCurve
 - SHARK_DATA, 89
- species
 - DOGFISH_PARAM, 25
 - MasterSpeciesList, 53
 - MONKFISH_PARAM, 65
- species_dat
 - MIXED_GAS, 59
- species_index
 - Mechanism, 58
 - UnsteadyReaction, 107
- speciesName
 - MasterSpeciesList, 53
 - SCOPSOWL_PARAM_DATA, 87
 - SKUA_PARAM, 94
- specific_heat
 - PURE_GAS, 78
- Spherical
 - finch.h, 124
- sphericalAvg
 - Matrix, 57
- sphericalBCFill
 - Matrix, 57
- state
 - SubHeader, 97
- SteadyState
 - FINCH_DATA, 29
- steadystate
 - SHARK_DATA, 89
- steps
 - GMRESLP_DATA, 36
- Stoichiometric
 - Reaction, 81
- string_parse_error
 - error.h, 120
- Sub_Map
 - Header, 47
- SubHeader, 95
 - ~SubHeader, 96
 - addPair, 96
 - alias, 97
 - clear, 96
 - Data_Map, 97
 - DisplayContents, 97
 - getAlias, 97
 - getMap, 97
 - getName, 97
 - getState, 97
 - isAlias, 97
 - isAnchor, 97
 - name, 97
 - operator=, 97
 - setAlias, 97
 - setName, 97
 - setNameAliasPair, 97
 - setState, 97
 - state, 97
 - SubHeader, 96
 - SubHeader, 96
- sum
 - ARNOLDI_DATA, 8
 - GMRESRP_DATA, 42
 - Matrix, 57
- Sum_Delta
 - MassBalance, 51
- SurfDiff
 - SCOPSOWL_DATA, 83
- surface_concentration
 - DOGFISH_PARAM, 25
- Sutherland_Const
 - PURE_GAS, 78
- Sutherland_Temp
 - PURE_GAS, 78
- Sutherland_Viscosity
 - PURE_GAS, 79
- Symbol
 - Atom, 11
- Sys
 - SYSTEM_DATA, 98
- sys_dat
 - MAGPIE_DATA, 50
- T
 - FINCH_DATA, 29
 - SYSTEM_DATA, 98
- t

- BiCGSTAB_DATA, 16
- FINCH_DATA, 29
- SCOPSOWL_DATA, 83
- SCOPSOWL_OPT_DATA, 85
- SKUA_DATA, 91
- SKUA_OPT_DATA, 93
- TEST
 - ui.h, 173
- t_count
 - SHARK_DATA, 89
- t_counter
 - DOGFISH_DATA, 24
 - MONKFISH_DATA, 64
 - SCOPSOWL_DATA, 83
 - SKUA_DATA, 91
- t_old
 - FINCH_DATA, 30
 - SCOPSOWL_DATA, 83
 - SKUA_DATA, 91
- t_out
 - SHARK_DATA, 89
- t_print
 - DOGFISH_DATA, 24
 - MONKFISH_DATA, 64
 - SCOPSOWL_DATA, 83
 - SKUA_DATA, 91
- t_rand
 - TRAJECTORY_DATA, 101
- TANGENTIAL_FORCE
 - Trajectory.cpp, 162
 - Trajectory.h, 164
- TRAJECTORY_DATA, 99
 - a, 100
 - A_separator, 100
 - A_wire, 100
 - b, 100
 - B0, 100
 - beta, 100
 - Cap, 100
 - chi_p, 100
 - dX, 100
 - dY, 100
 - dt, 100
 - eta, 100
 - H, 100
 - H0, 100
 - Hamaker, 100
 - k, 100
 - L, 100
 - L_wire, 100
 - M, 100
 - m_rand, 100
 - mp, 100
 - Ms, 100
 - mu_0, 100
 - n_rand, 100
 - POL, 101
 - porosity, 101
 - q_bar, 101
 - Q_in, 101
 - rho_f, 101
 - rho_p, 101
 - Rs, 101
 - s_rand, 101
 - sigma_m, 101
 - sigma_n, 101
 - sigma_v, 101
 - sigma_vz, 101
 - sigma_z, 101
 - t_rand, 101
 - Temp, 101
 - V0, 101
 - V_separator, 101
 - V_wire, 101
 - X, 101
 - Y, 101
 - Y_initial, 101
- Table
 - PeriodicTable, 71
- Temp
 - TRAJECTORY_DATA, 101
- temperature
 - SHARK_DATA, 89
- temperature_affinity
 - UnsteadyReaction, 107
- tempy
 - SCOPSOWL_DATA, 83
- tensor_out_of_bounds
 - error.h, 120
- term_precon
 - GMRESR_DATA, 39
- terminal_precon
 - GMRESR_DATA, 39
- test
 - ui.cpp, 168
 - ui.h, 177
- test_loop
 - ui.cpp, 169
 - ui.h, 177
- theoretical_darken_Dc
 - skua.cpp, 157
 - skua.h, 159
- time
 - DOGFISH_DATA, 24
 - MONKFISH_DATA, 64
 - SHARK_DATA, 90
- time_old
 - DOGFISH_DATA, 24
 - MONKFISH_DATA, 64
 - SHARK_DATA, 90
- time_step
 - UnsteadyReaction, 107
- TimeAdaptivity
 - SHARK_DATA, 90
- timesteps
 - SHARK_DATA, 90

- token_parser
 - yaml_cpp_class, 110
- tol_abs
 - BiCGSTAB_DATA, 16
 - CGS_DATA, 19
 - FINCH_DATA, 30
 - GCR_DATA, 33
 - GMRESLP_DATA, 36
 - GMRESRP_DATA, 42
 - PCG_DATA, 69
 - PICARD_DATA, 73
- tol_rel
 - BiCGSTAB_DATA, 16
 - CGS_DATA, 19
 - FINCH_DATA, 30
 - GCR_DATA, 33
 - GMRESLP_DATA, 36
 - GMRESRP_DATA, 42
 - PCG_DATA, 69
 - PICARD_DATA, 73
- total_density
 - MIXED_GAS, 59
- total_dyn_vis
 - MIXED_GAS, 59
- total_eval
 - GSTA_OPT_DATA, 45
 - SCOPSOWL_OPT_DATA, 85
 - SKUA_OPT_DATA, 93
 - SYSTEM_DATA, 99
- total_iter
 - FINCH_DATA, 30
 - GCR_DATA, 33
 - GMRESR_DATA, 39
- total_molecular_weight
 - MIXED_GAS, 59
- total_pressure
 - MIXED_GAS, 59
 - SCOPSOWL_DATA, 83
- total_sorption
 - DOGFISH_DATA, 24
 - MONKFISH_DATA, 64
- total_sorption_old
 - DOGFISH_DATA, 24
 - MONKFISH_DATA, 64
- total_specific_heat
 - MIXED_GAS, 59
- total_steps
 - DOGFISH_DATA, 24
 - MONKFISH_DATA, 64
 - SCOPSOWL_DATA, 83
 - SKUA_DATA, 91
- TotalConcentration
 - MassBalance, 52
- totalsteps
 - SHARK_DATA, 90
- trajectory
 - ui.h, 173
- Trajectory.cpp, 161
 - Brown_RAD, 161
 - Brown_THETA, 161
 - CARTESIAN, 161
 - DISPLACEMENT, 161
 - Grav_R, 161
 - Grav_T, 161
 - LOCATION, 162
 - Magnetic_R, 162
 - Magnetic_T, 162
 - Number_Generator, 162
 - POLAR, 162
 - RADIAL_FORCE, 162
 - Removal_Efficiency, 162
 - Run_Trajectory, 162
 - TANGENTIAL_FORCE, 162
 - Trajectory_SetupConstants, 162
 - V_RAD, 162
 - V_THETA, 162
 - Van_R, 162
- Trajectory.h, 162
 - Brown_RAD, 163
 - Brown_THETA, 163
 - CARTESIAN, 163
 - DISPLACEMENT, 163
 - Grav_R, 163
 - Grav_T, 163
 - LOCATION, 163
 - Magnetic_R, 163
 - Magnetic_T, 163
 - Number_Generator, 163
 - POLAR, 163
 - RADIAL_FORCE, 163
 - Removal_Efficiency, 163
 - Run_Trajectory, 164
 - TANGENTIAL_FORCE, 164
 - Trajectory_SetupConstants, 164
 - V_RAD, 164
 - V_THETA, 164
 - Van_R, 164
- Trajectory_SetupConstants
 - Trajectory.cpp, 162
 - Trajectory.h, 164
- transpose
 - Matrix, 57
- transpose_dat
 - GCR_DATA, 33
- transpose_multiply
 - Matrix, 57
- tridiagonalFill
 - Matrix, 57
- tridiagonalSolve
 - Matrix, 57
- tridiagonalVectorFill
 - Matrix, 57
- twoFifths
 - gsta_opt.cpp, 127
 - gsta_opt.h, 129
- type

- ValueTypePair, 109
- u
 - CGS_DATA, 19
 - GCR_DATA, 33
- UNKNOWN
 - yaml_wrapper.h, 180
- u_star
 - FINCH_DATA, 30
- u_temp
 - GCR_DATA, 33
- uAverage
 - finch.cpp, 123
 - finch.h, 126
- uAvg
 - FINCH_DATA, 30
- uAvg_old
 - FINCH_DATA, 30
- UI_DATA, 101
 - argc, 102
 - argv, 102
 - BasicUI, 102
 - count, 102
 - Files, 103
 - input_files, 103
 - max, 103
 - MissingArg, 103
 - option, 103
 - Path, 103
 - path, 103
 - user_input, 103
 - value_type, 103
- UI_HPP_
 - ui.h, 173
- uIC
 - FINCH_DATA, 30
- uT
 - FINCH_DATA, 30
- uT_old
 - FINCH_DATA, 30
- uTotal
 - finch.cpp, 123
 - finch.h, 126
- ubest
 - FINCH_DATA, 30
- ui.h
 - CONTINUE, 173
 - dogfish, 173
 - EXECUTE, 173
 - EXIT, 173
 - eel, 173
 - egret, 173
 - finch, 173
 - gsta_opt, 173
 - HELP, 173
 - lark, 173
 - macaw, 173
 - magpie, 173
 - mola, 173
 - monkfish, 173
 - sandbox, 173
 - scops_opt, 173
 - scopsowl, 173
 - shark, 173
 - skua, 173
 - skua_opt, 173
 - TEST, 173
 - trajectory, 173
- ui.cpp, 164
 - allLower, 165
 - ai_help, 166
 - bui_help, 166
 - display_help, 166
 - display_version, 166
 - exec, 166
 - exec_loop, 166
 - exit, 167
 - help, 167
 - input, 167
 - invalid_input, 167
 - number_files, 168
 - path, 168
 - run_exec, 168
 - run_executable, 168
 - run_test, 168
 - test, 168
 - test_loop, 169
 - valid_addon_options, 169
 - valid_exec_string, 169
 - valid_input_execute, 169
 - valid_input_main, 169
 - valid_input_tests, 170
 - valid_test_string, 170
 - version, 170
- ui.h, 170
 - allLower, 174
 - ai_help, 174
 - bui_help, 174
 - display_help, 174
 - display_version, 174
 - ECO_EXECUTABLE, 173
 - ECO_VERSION, 173
 - exec, 174
 - exec_loop, 175
 - exit, 175
 - help, 175
 - input, 175
 - invalid_input, 175
 - number_files, 176
 - path, 176
 - run_exec, 176
 - run_executable, 176
 - run_test, 176
 - test, 177
 - test_loop, 177
 - UI_HPP_, 173
 - valid_addon_options, 177

- valid_exec_string, [177](#)
- valid_input_execute, [177](#)
- valid_input_main, [178](#)
- valid_input_tests, [178](#)
- valid_options, [173](#)
- valid_test_string, [178](#)
- version, [178](#)
- un
 - FINCH_DATA, [30](#)
- unm1
 - FINCH_DATA, [30](#)
- unp1
 - FINCH_DATA, [30](#)
- unregistered_name
 - error.h, [120](#)
- unstable_matrix
 - error.h, [120](#)
- UnsteadyList
 - SHARK_DATA, [90](#)
- UnsteadyPrecipitation, [103](#)
- UnsteadyReaction, [104](#)
 - ~UnsteadyReaction, [105](#)
 - activation_energy, [107](#)
 - calculateEnergies, [105](#)
 - calculateEquilibrium, [105](#)
 - calculateRate, [105](#)
 - checkSpeciesEnergies, [105](#)
 - Display_Info, [106](#)
 - Eval_IC_Residual, [106](#)
 - Eval_ReactionRate, [106](#)
 - Eval_Residual, [106](#)
 - Explicit_Eval, [106](#)
 - forward_rate, [107](#)
 - forward_ref_rate, [107](#)
 - Get_ActivationEnergy, [106](#)
 - Get_Affinity, [106](#)
 - Get_Energy, [106](#)
 - Get_Enthalpy, [106](#)
 - Get_Entropy, [106](#)
 - Get_Equilibrium, [106](#)
 - Get_Forward, [106](#)
 - Get_ForwardRef, [106](#)
 - Get_InitialValue, [106](#)
 - Get_MaximumValue, [106](#)
 - Get_Reverse, [106](#)
 - Get_ReverseRef, [106](#)
 - Get_Species_Index, [106](#)
 - Get_Stoichiometric, [106](#)
 - Get_TimeStep, [106](#)
 - haveEquilibrium, [106](#)
 - HaveForRef, [107](#)
 - HaveForward, [107](#)
 - haveRate, [106](#)
 - HaveRevRef, [107](#)
 - HaveReverse, [107](#)
 - initial_value, [107](#)
 - Initialize_List, [106](#)
 - max_value, [107](#)
 - reverse_rate, [107](#)
 - reverse_ref_rate, [107](#)
 - Set_ActivationEnergy, [106](#)
 - Set_Affinity, [106](#)
 - Set_Energy, [106](#)
 - Set_Enthalpy, [106](#)
 - Set_EnthalpyANDEntropy, [107](#)
 - Set_Entropy, [107](#)
 - Set_Equilibrium, [107](#)
 - Set_Forward, [107](#)
 - Set_ForwardRef, [107](#)
 - Set_InitialValue, [107](#)
 - Set_MaximumValue, [107](#)
 - Set_Reverse, [107](#)
 - Set_ReverseRef, [107](#)
 - Set_Species_Index, [107](#)
 - Set_Stoichiometric, [107](#)
 - Set_TimeStep, [107](#)
 - species_index, [107](#)
 - temperature_affinity, [107](#)
 - time_step, [107](#)
 - UnsteadyReaction, [105](#)
 - UnsteadyReaction, [105](#)
- uo
 - FINCH_DATA, [30](#)
- Update
 - FINCH_DATA, [30](#)
- update_arnoldi_solution
 - lark.cpp, [131](#)
 - lark.h, [136](#)
- upperHessenberg2Triangular
 - Matrix, [57](#)
- upperHessenbergSolve
 - Matrix, [57](#)
- upperTriangularSolve
 - Matrix, [57](#)
- user_data
 - DOGFISH_DATA, [24](#)
 - MONKFISH_DATA, [64](#)
 - SCOPSOWL_DATA, [83](#)
 - SKUA_DATA, [91](#)
- user_input
 - UI_DATA, [103](#)
- uz_I_E
 - FINCH_DATA, [30](#)
- uz_I_I
 - FINCH_DATA, [30](#)
- uz_lm1_E
 - FINCH_DATA, [30](#)
- uz_lm1_I
 - FINCH_DATA, [30](#)
- uz_lp1_E
 - FINCH_DATA, [30](#)
- uz_lp1_I
 - FINCH_DATA, [30](#)
- V
 - magpie.h, [139](#)
- v

- ARNOLDI_DATA, 8
- BiCGSTAB_DATA, 16
- CGS_DATA, 19
- GMRESRP_DATA, 42
- mSPD_DATA, 65
- PJFNK_DATA, 77
- V0
 - TRAJECTORY_DATA, 101
- V_RAD
 - Trajectory.cpp, 162
 - Trajectory.h, 164
- V_THETA
 - Trajectory.cpp, 162
 - Trajectory.h, 164
- V_separator
 - TRAJECTORY_DATA, 101
- V_wire
 - TRAJECTORY_DATA, 101
- vIC
 - FINCH_DATA, 30
- valence_e
 - Atom, 11
- valid_act
 - shark.h, 154
- valid_addon_options
 - ui.cpp, 169
 - ui.h, 177
- valid_exec_string
 - ui.cpp, 169
 - ui.h, 177
- valid_input_execute
 - ui.cpp, 169
 - ui.h, 177
- valid_input_main
 - ui.cpp, 169
 - ui.h, 178
- valid_input_tests
 - ui.cpp, 170
 - ui.h, 178
- valid_options
 - ui.h, 173
- valid_test_string
 - ui.cpp, 170
 - ui.h, 178
- Value_Type
 - ValueTypePair, 109
- value_type
 - UI_DATA, 103
- ValueTypePair, 108
 - ~ValueTypePair, 108
 - assertType, 108
 - DisplayPair, 109
 - editPair, 109
 - editValue, 109
 - findType, 109
 - getBool, 109
 - getDouble, 109
 - getInt, 109
 - getPair, 109
 - getString, 109
 - getType, 109
 - getValue, 109
 - operator=, 109
 - type, 109
 - Value_Type, 109
 - ValueTypePair, 108
 - ValueTypePair, 108
- Van_R
 - Trajectory.cpp, 162
 - Trajectory.h, 164
- vanAlbada_discretization
 - finch.cpp, 123
 - finch.h, 126
- vector_out_of_bounds
 - error.h, 120
- velocity
 - MIXED_GAS, 59
- version
 - ui.cpp, 170
 - ui.h, 178
- Vk
 - ARNOLDI_DATA, 8
 - GMRESRP_DATA, 42
- vn
 - FINCH_DATA, 30
- vnp1
 - FINCH_DATA, 30
- vo
 - FINCH_DATA, 30
- w
 - ARNOLDI_DATA, 8
 - CGS_DATA, 19
 - GMRESRP_DATA, 42
- weight
 - Mechanism, 58
- weightedAvg
 - gsta_opt.cpp, 127
 - gsta_opt.h, 129
- X
 - TRAJECTORY_DATA, 101
- x
 - BiCGSTAB_DATA, 16
 - CGS_DATA, 19
 - GCR_DATA, 33
 - GMRESLP_DATA, 36
 - GMRESRP_DATA, 42
 - GPAST_DATA, 43
 - PCG_DATA, 70
 - PJFNK_DATA, 77
 - Speciation_Test01_Data, 95
- x0
 - PICARD_DATA, 73
- X_new
 - SHARK_DATA, 90
- X_old

- SHARK_DATA, 90
- xiC
 - SCOPSOWL_PARAM_DATA, 87
 - SKUA_PARAM, 94
- xk
 - BACKTRACK_DATA, 12
- xn
 - SKUA_PARAM, 94
- xnp1
 - SKUA_PARAM, 94
- Y
 - TRAJECTORY_DATA, 101
- y
 - BiCGSTAB_DATA, 16
 - GMRESRP_DATA, 42
 - GPAST_DATA, 43
 - SCOPSOWL_DATA, 83
 - SKUA_DATA, 91
- y_base
 - SCOPSOWL_OPT_DATA, 85
 - SKUA_OPT_DATA, 93
- y_eff
 - SKUA_PARAM, 94
- Y_initial
 - TRAJECTORY_DATA, 101
- YAML_CPP_TEST
 - yaml_wrapper.cpp, 179
 - yaml_wrapper.h, 180
- YAML_WRAPPER_TESTS
 - yaml_wrapper.cpp, 179
 - yaml_wrapper.h, 180
- yaml_wrapper.h
 - ALIAS, 180
 - ANCHOR, 180
 - BOOLEAN, 180
 - DOUBLE, 180
 - INT, 180
 - NONE, 180
 - STRING, 180
 - UNKNOWN, 180
- yaml_cpp_class, 109
 - ~yaml_cpp_class, 110
 - cleanup, 110
 - current_token, 110
 - DisplayContents, 110
 - executeYamlRead, 110
 - file_name, 110
 - getYamlWrapper, 110
 - input_file, 110
 - previous_token, 110
 - readInputFile, 110
 - setInputFile, 110
 - token_parser, 110
 - yaml_cpp_class, 110
 - yaml_wrapper, 110
 - yaml_cpp_class, 110
- yaml_object
 - SHARK_DATA, 90
- yaml_wrapper
 - yaml_cpp_class, 110
- yaml_wrapper.cpp, 179
 - YAML_CPP_TEST, 179
- yaml_wrapper.h, 179
 - data_type, 180
 - header_state, 180
 - YAML_CPP_TEST, 180
- YamlWrapper, 110
 - ~YamlWrapper, 111
 - addDocKey, 111
 - begin, 111
 - changeKey, 111
 - clear, 111
 - copyAnchor2Alias, 112
 - DisplayContents, 112
 - Doc_Map, 112
 - end, 112
 - getAnchoredDoc, 112
 - getDocFromHeadAlias, 112
 - getDocFromSubAlias, 112
 - getDocMap, 112
 - getDocument, 112
 - operator(), 112
 - operator=, 112
 - resetKeys, 112
 - revalidateAllKeys, 112
 - size, 112
 - YamlWrapper, 111
 - YamlWrapper, 111
- yk
 - ARNOLDI_DATA, 9
- Z
 - maggie.h, 139
- z
 - BiCGSTAB_DATA, 16
 - CGS_DATA, 20
 - PCG_DATA, 70
- z_old
 - PCG_DATA, 70
- zero_vector
 - error.h, 120
- zeros
 - Matrix, 57