

CR SPECTRA DOCUMENTATION

Generated by Doxygen 1.8.13

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	File Index	3
2.1	File List	3
3	Namespace Documentation	5
3.1	background_diffusion_coefficient Namespace Reference	5
3.1.1	Function Documentation	6
3.1.1.1	Duu_Alfven_Slab_Linear_Undamped()	6
3.1.1.2	IonNeutral_Damping()	7
3.1.1.3	J()	7
3.1.1.4	Kappa_zz()	7
3.1.2	Variable Documentation	7
3.1.2.1	a	8
3.1.2.2	b	8
3.1.2.3	B0	8
3.1.2.4	c	8
3.1.2.5	chi	8
3.1.2.6	D	8
3.1.2.7	d_uu	8
3.1.2.8	dmu	8
3.1.2.9	E	9
3.1.2.10	eps	9

3.1.2.11	fj_m	9
3.1.2.12	fj_p	9
3.1.2.13	gamma	9
3.1.2.14	gs0	9
3.1.2.15	l	9
3.1.2.16	ltot	9
3.1.2.17	k_cm	10
3.1.2.18	k_cp	10
3.1.2.19	k_max	10
3.1.2.20	k_min	10
3.1.2.21	k_zz	10
3.1.2.22	m	10
3.1.2.23	mi	10
3.1.2.24	mn	10
3.1.2.25	mu	11
3.1.2.26	ni	11
3.1.2.27	nn	11
3.1.2.28	nu_in	11
3.1.2.29	nu_ni	11
3.1.2.30	Omega	11
3.1.2.31	Omega0	11
3.1.2.32	p	11
3.1.2.33	phase	12
3.1.2.34	q	12
3.1.2.35	T	12
3.1.2.36	v	12
3.1.2.37	VA	12
3.1.2.38	VAi	12
3.2	background_diffusion_coefficient_2 Namespace Reference	12
3.2.1	Function Documentation	13

3.2.1.1	g_s()	13
3.2.1.2	IonNeutral_Damping()	13
3.2.1.3	R()	14
3.2.2	Variable Documentation	14
3.2.2.1	B0	14
3.2.2.2	D_uu	14
3.2.2.3	dk	14
3.2.2.4	E	14
3.2.2.5	gamma	14
3.2.2.6	I	15
3.2.2.7	k	15
3.2.2.8	k_zz	15
3.2.2.9	kmin	15
3.2.2.10	m	15
3.2.2.11	medium_props	15
3.2.2.12	mu	15
3.2.2.13	Omega	15
3.2.2.14	Omega0	16
3.2.2.15	p	16
3.2.2.16	particles_props	16
3.2.2.17	q	16
3.2.2.18	v	16
3.2.2.19	w	16
3.2.2.20	wtot	16
3.3	background_diffusion_coefficient_3 Namespace Reference	16
3.3.1	Function Documentation	18
3.3.1.1	Duu_Alfven_Slab_Linear_Undamped()	18
3.3.1.2	IonNeutral_Damping()	18
3.3.1.3	Kappa_zz()	19
3.3.1.4	kappa_zz_BC()	19

3.3.2	Variable Documentation	19
3.3.2.1	alpha	19
3.3.2.2	ax0	19
3.3.2.3	ax1	20
3.3.2.4	ax2	20
3.3.2.5	ax3	20
3.3.2.6	bbox_to_anchor	20
3.3.2.7	c	20
3.3.2.8	CNM	20
3.3.2.9	custom_lines	20
3.3.2.10	DeC	21
3.3.2.11	DeM	21
3.3.2.12	DiM	21
3.3.2.13	E	21
3.3.2.14	Emax	21
3.3.2.15	Emin	21
3.3.2.16	facecolor	21
3.3.2.17	fig	21
3.3.2.18	GeV	22
3.3.2.19	gs	22
3.3.2.20	handles	22
3.3.2.21	hatch	22
3.3.2.22	HII	22
3.3.2.23	hspace	22
3.3.2.24	I	22
3.3.2.25	K_zz_1	23
3.3.2.26	K_zz_2	23
3.3.2.27	K_zz_bc_1	23
3.3.2.28	K_zz_bc_2	23
3.3.2.29	K_zz_CNM_1	23

3.3.2.30	K_zz_CNM_2	23
3.3.2.31	K_zz_DeC_1	23
3.3.2.32	K_zz_DeC_2	23
3.3.2.33	K_zz_DeM_1	24
3.3.2.34	K_zz_DeM_2	24
3.3.2.35	K_zz_DiM_1	24
3.3.2.36	K_zz_DiM_2	24
3.3.2.37	K_zz_HII_1	24
3.3.2.38	K_zz_HII_2	24
3.3.2.39	K_zz_WIM_1	24
3.3.2.40	K_zz_WIM_2	24
3.3.2.41	K_zz_WNM_1	25
3.3.2.42	K_zz_WNM_2	25
3.3.2.43	kmin	25
3.3.2.44	label	25
3.3.2.45	loc	25
3.3.2.46	ls	25
3.3.2.47	mass	25
3.3.2.48	mp	25
3.3.2.49	ncol	26
3.3.2.50	q	26
3.3.2.51	size_x	26
3.3.2.52	size_y	26
3.3.2.53	sub_x	26
3.3.2.54	sub_y	26
3.3.2.55	WIM	26
3.3.2.56	WNM	26
3.3.2.57	wspace	27
3.3.2.58	xmax	27
3.3.2.59	xmin	27

3.3.2.60	ymax	27
3.3.2.61	ymin	27
3.4	constants Namespace Reference	27
3.4.1	Variable Documentation	28
3.4.1.1	c	28
3.4.1.2	e	28
3.4.1.3	eV	28
3.4.1.4	GeV	28
3.4.1.5	kbolz	28
3.4.1.6	kms	28
3.4.1.7	kpc	28
3.4.1.8	kyr	29
3.4.1.9	mCII	29
3.4.1.10	me	29
3.4.1.11	MeV	29
3.4.1.12	mH2	29
3.4.1.13	mHCOII	29
3.4.1.14	mHel	29
3.4.1.15	mHell	29
3.4.1.16	mHI	30
3.4.1.17	mHII	30
3.4.1.18	mn	30
3.4.1.19	mp	30
3.4.1.20	pc	30
3.4.1.21	TeV	30
3.4.1.22	yr	30
3.5	d1_grid_generator Namespace Reference	30
3.5.1	Function Documentation	31
3.5.1.1	grid()	31
3.6	damping Namespace Reference	31

3.6.1	Function Documentation	31
3.6.1.1	damping_lazarian()	31
3.6.1.2	damping_lazarian_nopos()	31
3.6.1.3	IN_damping_approx_1()	32
3.6.1.4	IN_damping_approx_2()	32
3.6.1.5	indamping_alfven()	32
3.6.1.6	indamping_alfven_nopos()	33
3.6.1.7	IonNeutral_Damping()	33
3.6.1.8	non_linear_landau_damping()	33
3.7	damping_models Namespace Reference	33
3.7.1	Function Documentation	34
3.7.1.1	damprate_to_damptime()	34
3.7.2	Variable Documentation	35
3.7.2.1	alpha	35
3.7.2.2	ax	35
3.7.2.3	bbox_inches	35
3.7.2.4	bbox_to_anchor	35
3.7.2.5	c	35
3.7.2.6	color	35
3.7.2.7	E	35
3.7.2.8	egridtype	36
3.7.2.9	Em	36
3.7.2.10	E _{max}	36
3.7.2.11	E _{min}	36
3.7.2.12	E _p	36
3.7.2.13	facecolor	36
3.7.2.14	fig	36
3.7.2.15	Gamma _{lz}	36
3.7.2.16	Gamma _{nld_inf}	37
3.7.2.17	Gamma _{nld_sup}	37

3.7.2.18	GeV	37
3.7.2.19	gs	37
3.7.2.20	hspace	37
3.7.2.21	linf	37
3.7.2.22	in_damping	37
3.7.2.23	lsup	37
3.7.2.24	label	38
3.7.2.25	loc	38
3.7.2.26	ls	38
3.7.2.27	lw	38
3.7.2.28	lz_damping	38
3.7.2.29	lz_min	38
3.7.2.30	Name	38
3.7.2.31	NE	38
3.7.2.32	pad_inches	39
3.7.2.33	phases	39
3.7.2.34	pos_1	39
3.7.2.35	pos_2	39
3.7.2.36	size_x	39
3.7.2.37	size_y	39
3.7.2.38	sub_x	39
3.7.2.39	sub_y	39
3.7.2.40	wl_Alfven	40
3.7.2.41	wl_Alfven_o1	40
3.7.2.42	wl_Alfven_o2	40
3.7.2.43	wR_Alfven	40
3.7.2.44	wR_Alfven_o1	40
3.7.2.45	wR_Alfven_o2	40
3.7.2.46	wspace	40
3.7.2.47	x	40

3.7.2.48	<code>xlim</code>	41
3.7.2.49	<code>xlims</code>	41
3.7.2.50	<code>y</code>	41
3.7.2.51	<code>ylim</code>	41
3.7.2.52	<code>ylims</code>	41
3.8	Data_reader Namespace Reference	41
3.8.1	Function Documentation	41
3.8.1.1	<code>readAxis()</code>	42
3.8.1.2	<code>readDataXE()</code>	42
3.8.2	Variable Documentation	42
3.8.2.1	<code>data</code>	42
3.8.2.2	<code>E</code>	42
3.8.2.3	<code>figsize</code>	42
3.8.2.4	<code>X</code>	42
3.9	electrons_emax_t Namespace Reference	43
3.9.1	Function Documentation	43
3.9.1.1	<code>B()</code>	43
3.9.1.2	<code>Em_age()</code>	44
3.9.1.3	<code>Em_cool()</code>	44
3.9.1.4	<code>Em_esc()</code>	44
3.9.1.5	<code>Emax_electrons()</code>	44
3.9.1.6	<code>escape_time()</code>	45
3.9.1.7	<code>eta_g()</code>	45
3.9.1.8	<code>InterpolatingSpline()</code>	45
3.9.1.9	<code>InverseTrigonalMatrix()</code>	45
3.9.1.10	<code>ProductMatrix()</code>	45
3.9.1.11	<code>Rsh()</code>	45
3.9.2	Variable Documentation	46
3.9.2.1	<code>alpha</code>	46
3.9.2.2	<code>alpha_B</code>	46

3.9.2.3	beta	46
3.9.2.4	C06	46
3.9.2.5	E	46
3.9.2.6	E51	46
3.9.2.7	EM	46
3.9.2.8	Mej	47
3.9.2.9	nt	47
3.9.2.10	phi_c	47
3.9.2.11	tesc	47
3.9.2.12	tsed	47
3.9.2.13	vej8	47
3.9.2.14	xhi_cr	47
3.9.2.15	xhi_m	48
3.10	EscapeModel_protons Namespace Reference	48
3.10.1	Function Documentation	49
3.10.1.1	df1dx()	49
3.10.1.2	df2dx()	49
3.10.1.3	f1()	50
3.10.1.4	f2()	50
3.10.1.5	Gettesc()	50
3.10.1.6	InterpolatingSpline()	50
3.10.1.7	InverseTrigonalMatrix()	50
3.10.1.8	NewtonRaphson()	50
3.10.1.9	ProductMatrix()	51
3.10.2	Variable Documentation	51
3.10.2.1	a	51
3.10.2.2	b	51
3.10.2.3	beta	51
3.10.2.4	c	51
3.10.2.5	C06	51

3.10.2.6	delta	51
3.10.2.7	E51	52
3.10.2.8	Ecr	52
3.10.2.9	E _{max}	52
3.10.2.10	EMAX	52
3.10.2.11	E _{min}	52
3.10.2.12	eps	52
3.10.2.13	f_SNR	52
3.10.2.14	figsize	53
3.10.2.15	gamma	53
3.10.2.16	GeV	53
3.10.2.17	kyr	53
3.10.2.18	label	53
3.10.2.19	logR	53
3.10.2.20	logr_new	53
3.10.2.21	logt	54
3.10.2.22	logt_new	54
3.10.2.23	lw	54
3.10.2.24	marker	54
3.10.2.25	Mej	54
3.10.2.26	niter	54
3.10.2.27	nt	54
3.10.2.28	pc	54
3.10.2.29	phi_c	55
3.10.2.30	R	55
3.10.2.31	R_free	55
3.10.2.32	R_ini	55
3.10.2.33	R_MCS	55
3.10.2.34	R_merge	55
3.10.2.35	r_new	55

3.10.2.36 R_PDS	56
3.10.2.37 t	56
3.10.2.38 t_new	56
3.10.2.39 tesc	56
3.10.2.40 tfree	56
3.10.2.41 tini	56
3.10.2.42 tmax	56
3.10.2.43 tMCS	57
3.10.2.44 tmerge	57
3.10.2.45 tPDS	57
3.10.2.46 tSed	57
3.10.2.47 u_sh	57
3.10.2.48 vej8	57
3.10.2.49 x0	57
3.10.2.50 xhi_cr	58
3.10.2.51 xhi_m	58
3.11 EscapeModel_protons_2 Namespace Reference	58
3.11.1 Function Documentation	59
3.11.1.1 df1dx()	59
3.11.1.2 df2dx()	59
3.11.1.3 f1()	59
3.11.1.4 f2()	60
3.11.1.5 getEmax()	60
3.11.1.6 getSNR()	60
3.11.1.7 Gettesc()	60
3.11.1.8 InterpolatingSpline()	60
3.11.1.9 InverseTrigonalMatrix()	61
3.11.1.10 NewtonRaphson()	61
3.11.1.11 ProductMatrix()	61
3.11.2 Variable Documentation	61

3.11.2.1	ax0	61
3.11.2.2	ax1	61
3.11.2.3	bbox_to_anchor	61
3.11.2.4	c	62
3.11.2.5	delta	62
3.11.2.6	Ecr	62
3.11.2.7	emax_CNM	62
3.11.2.8	emax_DiM	62
3.11.2.9	emax_HII	62
3.11.2.10	emax_WIM	63
3.11.2.11	emax_WNM	63
3.11.2.12	fig	63
3.11.2.13	GeV	63
3.11.2.14	gs	63
3.11.2.15	hspace	63
3.11.2.16	kyr	64
3.11.2.17	label	64
3.11.2.18	loc	64
3.11.2.19	ls	64
3.11.2.20	ncol	64
3.11.2.21	pad	64
3.11.2.22	size_x	64
3.11.2.23	size_y	64
3.11.2.24	SNR_CNM	65
3.11.2.25	SNR_DiM	65
3.11.2.26	SNR_HII	65
3.11.2.27	SNR_WIM	65
3.11.2.28	SNR_WNM	65
3.11.2.29	sub_x	65
3.11.2.30	sub_y	65

3.11.2.31 tesc_CNM	65
3.11.2.32 tesc_CNM_3	66
3.11.2.33 tesc_DiM	66
3.11.2.34 tesc_DiM_3	66
3.11.2.35 tesc_HII	66
3.11.2.36 tesc_HII_3	66
3.11.2.37 tesc_WIM	66
3.11.2.38 tesc_WIM_3	66
3.11.2.39 tesc_WNM	66
3.11.2.40 tesc_WNM_3	67
3.11.2.41 wspace	67
3.12 freader Namespace Reference	67
3.12.1 Function Documentation	67
3.12.1.1 search()	67
3.13 fwriter Namespace Reference	67
3.13.1 Function Documentation	67
3.13.1.1 fileWrite()	68
3.13.1.2 search()	68
3.13.1.3 write1D()	68
3.13.1.4 write1Daxis()	68
3.13.1.5 write2D()	68
3.14 gaussian_subNormalization Namespace Reference	69
3.14.1 Function Documentation	69
3.14.1.1 gauss()	69
3.14.2 Variable Documentation	69
3.14.2.1 c	69
3.14.2.2 C_a	70
3.14.2.3 C_c	70
3.14.2.4 color	70
3.14.2.5 gauss_a	70

3.14.2.6	gauss_c	70
3.14.2.7	mu	70
3.14.2.8	Nc	70
3.14.2.9	Nv	70
3.14.2.10	r	71
3.14.2.11	sig	71
3.14.2.12	ta	71
3.14.2.13	tc	71
3.14.2.14	tesc	71
3.14.2.15	tmax	71
3.14.2.16	tmin	71
3.15	mathmethods Namespace Reference	72
3.15.1	Function Documentation	72
3.15.1.1	cardano3()	72
3.15.1.2	Cubic3()	72
3.15.1.3	f()	72
3.15.1.4	findF()	73
3.15.1.5	findG()	73
3.15.1.6	findH()	73
3.15.1.7	g()	73
3.15.1.8	g1()	73
3.15.1.9	g2()	73
3.15.1.10	glin()	74
3.15.1.11	histogram()	74
3.15.1.12	multishape()	74
3.15.1.13	shape()	74
3.15.1.14	simpson_lin()	74
3.15.1.15	simpson_log()	75
3.15.1.16	SmoothPhaseTransition()	75
3.16	namelist Namespace Reference	75

3.16.1	Function Documentation	76
3.16.1.1	getDamping()	76
3.16.1.2	getVA()	76
3.16.2	Variable Documentation	76
3.16.2.1	B	76
3.16.2.2	bdiff_model	77
3.16.2.3	box_center	77
3.16.2.4	E	77
3.16.2.5	egridtype	77
3.16.2.6	E _{max}	77
3.16.2.7	E _{min}	77
3.16.2.8	folder_name	77
3.16.2.9	folder_path	78
3.16.2.10	gamma_in	78
3.16.2.11	gamma_lz	78
3.16.2.12	in_damping	78
3.16.2.13	ism_values	78
3.16.2.14	lz_damping	78
3.16.2.15	mi	78
3.16.2.16	mn	79
3.16.2.17	NE	79
3.16.2.18	ni	79
3.16.2.19	nlld_damping	79
3.16.2.20	nn	79
3.16.2.21	nt	79
3.16.2.22	NX	79
3.16.2.23	Pcr_1GeV	80
3.16.2.24	Pe_1GeV	80
3.16.2.25	phases	80
3.16.2.26	smooth_width_transition	80

3.16.2.27 T	80
3.16.2.28 total_path	80
3.16.2.29 va	80
3.16.2.30 X	81
3.16.2.31 xgridtype	81
3.16.2.32 Xi	81
3.16.2.33 Xmax	81
3.16.2.34 Xmin	81
3.17 namelist_adv Namespace Reference	81
3.17.1 Function Documentation	82
3.17.1.1 getVA()	82
3.17.2 Variable Documentation	83
3.17.2.1 B	83
3.17.2.2 bdiff_model	83
3.17.2.3 box_center	83
3.17.2.4 E	83
3.17.2.5 egridtype	83
3.17.2.6 Emax	83
3.17.2.7 Emin	83
3.17.2.8 folder_name	84
3.17.2.9 folder_path	84
3.17.2.10 in_damping	84
3.17.2.11 ism_values	84
3.17.2.12 lz_damping	84
3.17.2.13 mi	84
3.17.2.14 mn	84
3.17.2.15 NE	85
3.17.2.16 ni	85
3.17.2.17 nlld_damping	85
3.17.2.18 nn	85

3.17.2.19 nt	85
3.17.2.20 NX	85
3.17.2.21 Pcr_1GeV	85
3.17.2.22 Pe_1GeV	86
3.17.2.23 phases	86
3.17.2.24 smooth_width_transition	86
3.17.2.25 T	86
3.17.2.26 total_path	86
3.17.2.27 va	86
3.17.2.28 X	86
3.17.2.29 xgridtype	87
3.17.2.30 Xi	87
3.17.2.31 Xmax	87
3.17.2.32 Xmin	87
3.18 namelist_adve Namespace Reference	87
3.18.1 Function Documentation	88
3.18.1.1 getVA()	88
3.18.2 Variable Documentation	88
3.18.2.1 B	88
3.18.2.2 bdiff_model	88
3.18.2.3 box_center	89
3.18.2.4 E	89
3.18.2.5 egridtype	89
3.18.2.6 Emax	89
3.18.2.7 Emin	89
3.18.2.8 folder_name	89
3.18.2.9 folder_path	89
3.18.2.10 in_damping	90
3.18.2.11 ism_values	90
3.18.2.12 lz_damping	90

3.18.2.13 mi	90
3.18.2.14 mn	90
3.18.2.15 NE	90
3.18.2.16 ni	90
3.18.2.17 nlld_damping	91
3.18.2.18 nn	91
3.18.2.19 nt	91
3.18.2.20 NX	91
3.18.2.21 Pcr_1GeV	91
3.18.2.22 Pe_1GeV	91
3.18.2.23 phases	91
3.18.2.24 smooth_width_transition	92
3.18.2.25 T	92
3.18.2.26 total_path	92
3.18.2.27 va	92
3.18.2.28 X	92
3.18.2.29 xgridtype	92
3.18.2.30 Xi	92
3.18.2.31 Xmax	93
3.18.2.32 Xmin	93
3.19 namelist_advst Namespace Reference	93
3.19.1 Function Documentation	94
3.19.1.1 getVA()	94
3.19.2 Variable Documentation	94
3.19.2.1 B	94
3.19.2.2 bdiff_model	94
3.19.2.3 box_center	94
3.19.2.4 E	94
3.19.2.5 egridtype	94
3.19.2.6 Emax	95

3.19.2.7 Emin	95
3.19.2.8 folder_name	95
3.19.2.9 folder_path	95
3.19.2.10 in_damping	95
3.19.2.11 ism_values	95
3.19.2.12 lz_damping	95
3.19.2.13 mi	96
3.19.2.14 mn	96
3.19.2.15 NE	96
3.19.2.16 ni	96
3.19.2.17 nlld_damping	96
3.19.2.18 nn	96
3.19.2.19 nt	96
3.19.2.20 NX	97
3.19.2.21 Pcr_1GeV	97
3.19.2.22 Pe_1GeV	97
3.19.2.23 phases	97
3.19.2.24 smooth_width_transition	97
3.19.2.25 T	97
3.19.2.26 total_path	97
3.19.2.27 va	98
3.19.2.28 X	98
3.19.2.29 xgridtype	98
3.19.2.30 Xi	98
3.19.2.31 Xmax	98
3.19.2.32 Xmin	98
3.20 namelist_diff Namespace Reference	98
3.20.1 Function Documentation	99
3.20.1.1 getVA()	99
3.20.2 Variable Documentation	100

3.20.2.1	B	100
3.20.2.2	bdiff_model	100
3.20.2.3	box_center	100
3.20.2.4	E	100
3.20.2.5	egridtype	100
3.20.2.6	E _{max}	100
3.20.2.7	E _{min}	100
3.20.2.8	folder_name	101
3.20.2.9	folder_path	101
3.20.2.10	in_damping	101
3.20.2.11	ism_values	101
3.20.2.12	lz_damping	101
3.20.2.13	mi	101
3.20.2.14	mn	101
3.20.2.15	NE	102
3.20.2.16	ni	102
3.20.2.17	nld_damping	102
3.20.2.18	nn	102
3.20.2.19	nt	102
3.20.2.20	NX	102
3.20.2.21	P _{cr} _1GeV	102
3.20.2.22	P _e _1GeV	103
3.20.2.23	phases	103
3.20.2.24	smooth_width_transition	103
3.20.2.25	T	103
3.20.2.26	total_path	103
3.20.2.27	va	103
3.20.2.28	X	103
3.20.2.29	xgridtype	104
3.20.2.30	Xi	104

3.20.2.31 Xmax	104
3.20.2.32 Xmin	104
3.21 namelist_uniform Namespace Reference	104
3.21.1 Function Documentation	105
3.21.1.1 getDamping()	105
3.21.1.2 getVA()	105
3.21.2 Variable Documentation	105
3.21.2.1 B	106
3.21.2.2 bdiff_model	106
3.21.2.3 box_center	106
3.21.2.4 E	106
3.21.2.5 egridtype	106
3.21.2.6 Emax	106
3.21.2.7 Emin	106
3.21.2.8 folder_name	107
3.21.2.9 folder_path	107
3.21.2.10 gamma_in	107
3.21.2.11 gamma_lz	107
3.21.2.12 in_damping	107
3.21.2.13 ism_values	107
3.21.2.14 lz_damping	107
3.21.2.15 mi	108
3.21.2.16 mn	108
3.21.2.17 NE	108
3.21.2.18 ni	108
3.21.2.19 nlld_damping	108
3.21.2.20 nn	108
3.21.2.21 nt	108
3.21.2.22 NX	109
3.21.2.23 Pcr_1GeV	109

3.21.2.24	Pe_1GeV	109
3.21.2.25	phases	109
3.21.2.26	smooth_width_transition	109
3.21.2.27	T	109
3.21.2.28	total_path	109
3.21.2.29	va	110
3.21.2.30	X	110
3.21.2.31	xgridtype	110
3.21.2.32	Xi	110
3.21.2.33	Xmax	110
3.21.2.34	Xmin	110
3.22	namelist_vdiff Namespace Reference	110
3.22.1	Function Documentation	111
3.22.1.1	getVA()	111
3.22.2	Variable Documentation	112
3.22.2.1	B	112
3.22.2.2	bdiff_model	112
3.22.2.3	box_center	112
3.22.2.4	E	112
3.22.2.5	egridtype	112
3.22.2.6	Emax	112
3.22.2.7	Emin	112
3.22.2.8	folder_name	113
3.22.2.9	folder_path	113
3.22.2.10	in_damping	113
3.22.2.11	ism_values	113
3.22.2.12	lz_damping	113
3.22.2.13	mi	113
3.22.2.14	mn	113
3.22.2.15	NE	114

3.22.2.16 ni	114
3.22.2.17 nlld_damping	114
3.22.2.18 nn	114
3.22.2.19 nt	114
3.22.2.20 NX	114
3.22.2.21 Pcr_1GeV	114
3.22.2.22 Pe_1GeV	115
3.22.2.23 phases	115
3.22.2.24 smooth_width_transition	115
3.22.2.25 T	115
3.22.2.26 total_path	115
3.22.2.27 va	115
3.22.2.28 X	115
3.22.2.29 xgridtype	116
3.22.2.30 Xi	116
3.22.2.31 Xmax	116
3.22.2.32 Xmin	116
3.23 Output_functions Namespace Reference	116
3.24 pcr_ip_2D Namespace Reference	116
3.24.1 Function Documentation	117
3.24.1.1 getData()	117
3.24.1.2 getTimeID()	117
3.24.1.3 readAxis()	118
3.24.1.4 readDataXE()	118
3.24.2 Variable Documentation	118
3.24.2.1 ax	118
3.24.2.2 ax0	118
3.24.2.3 ax1	118
3.24.2.4 cax	118
3.24.2.5 cmap	118

3.24.2.6	<code>delta_t</code>	119
3.24.2.7	<code>E</code>	119
3.24.2.8	<code>EV</code>	119
3.24.2.9	<code>EV_log</code>	119
3.24.2.10	<code>False</code>	119
3.24.2.11	<code>figsize</code>	119
3.24.2.12	<code>fontsize</code>	119
3.24.2.13	<code>im0</code>	119
3.24.2.14	<code>im1</code>	120
3.24.2.15	<code>indexing</code>	120
3.24.2.16	<code>lp</code>	120
3.24.2.17	<code>label</code>	120
3.24.2.18	<code>ncols</code>	120
3.24.2.19	<code>out_id</code>	120
3.24.2.20	<code>Pcr</code>	120
3.24.2.21	<code>sharey</code>	120
3.24.2.22	<code>sparse</code>	121
3.24.2.23	<code>t_ini</code>	121
3.24.2.24	<code>t_max</code>	121
3.24.2.25	<code>time_test</code>	121
3.24.2.26	<code>X</code>	121
3.24.2.27	<code>x_center</code>	121
3.24.2.28	<code>XV</code>	121
3.24.2.29	<code>XV_log</code>	121
3.25	PDE_solvers Namespace Reference	122
3.25.1	Function Documentation	122
3.25.1.1	<code>A()</code>	122
3.25.1.2	<code>B()</code>	123
3.25.1.3	<code>C()</code>	123
3.25.1.4	<code>CC70Solver()</code>	123

3.25.1.5	<code>finiteDiffSolver()</code>	123
3.25.1.6	<code>Q()</code>	123
3.25.1.7	<code>SimpleImplicitSolver()</code>	123
3.25.1.8	<code>T()</code>	124
3.25.1.9	<code>TDMA()</code>	124
3.25.2	Variable Documentation	124
3.25.2.1	<code>c</code>	124
3.25.2.2	<code>figsize</code>	124
3.25.2.3	<code>kyr</code>	124
3.25.2.4	<code>M</code>	124
3.25.2.5	<code>pc</code>	125
3.25.2.6	<code>Tmax</code>	125
3.25.2.7	<code>u</code>	125
3.25.2.8	<code>u0</code>	125
3.25.2.9	<code>u_0</code>	125
3.25.2.10	<code>u_1</code>	125
3.25.2.11	<code>u_2</code>	126
3.25.2.12	<code>u_end</code>	126
3.25.2.13	<code>u_ini</code>	126
3.25.2.14	<code>uM</code>	126
3.25.2.15	<code>X</code>	126
3.25.2.16	<code>Xmax</code>	126
3.25.2.17	<code>Xmin</code>	126
3.25.2.18	<code>yr</code>	126
3.26	<code>phases_collection</code> Namespace Reference	127
3.26.1	Function Documentation	127
3.26.1.1	<code>ism_phase()</code>	127
3.26.2	Variable Documentation	127
3.26.2.1	<code>CNM</code>	127
3.26.2.2	<code>DeC</code>	127

3.26.2.3	DeM	128
3.26.2.4	DiM	128
3.26.2.5	HII	128
3.26.2.6	WIM	128
3.26.2.7	WNM	128
3.27	physical_models Namespace Reference	128
3.27.1	Function Documentation	128
3.27.1.1	collision_rate()	129
3.27.1.2	cr_escape_radius_model()	129
3.27.1.3	cr_escape_time_model()	129
3.28	setup Namespace Reference	129
3.28.1	Variable Documentation	130
3.28.1.1	B	130
3.28.1.2	D	130
3.28.1.3	d00	130
3.28.1.4	Db	131
3.28.1.5	E	131
3.28.1.6	ext	131
3.28.1.7	g_in	131
3.28.1.8	g_lz	131
3.28.1.9	gamma_in	131
3.28.1.10	gamma_lazarian	131
3.28.1.11	gamma_nlld	131
3.28.1.12	gamma_tot	132
3.28.1.13	I	132
3.28.1.14	Im	132
3.28.1.15	in_damping	132
3.28.1.16	lp	132
3.28.1.17	ism_values	132
3.28.1.18	kmin	132

3.28.1.19 mass	132
3.28.1.20 medium_props	133
3.28.1.21 mi	133
3.28.1.22 mn	133
3.28.1.23 ne	133
3.28.1.24 ni	133
3.28.1.25 nn	133
3.28.1.26 nx	134
3.28.1.27 path	134
3.28.1.28 Pcr	134
3.28.1.29 Pe	134
3.28.1.30 q	134
3.28.1.31 T	134
3.28.1.32 va	134
3.28.1.33 VA	135
3.28.1.34 variable	135
3.28.1.35 variables	135
3.28.1.36 X	135
3.28.1.37 x_center	135
3.28.1.38 x_center_index	135
3.28.1.39 Xi	136
3.29 setup_adv Namespace Reference	136
3.29.1 Function Documentation	136
3.29.1.1 door()	137
3.29.2 Variable Documentation	137
3.29.2.1 B	137
3.29.2.2 D	137
3.29.2.3 d00	137
3.29.2.4 Db	137
3.29.2.5 E	137

3.29.2.6	ext	138
3.29.2.7	gamma_in	138
3.29.2.8	gamma_lazarian	138
3.29.2.9	gamma_nlld	138
3.29.2.10	gamma_tot	138
3.29.2.11	lm	138
3.29.2.12	lp	138
3.29.2.13	ism_values	138
3.29.2.14	mi	139
3.29.2.15	mn	139
3.29.2.16	ne	139
3.29.2.17	ni	139
3.29.2.18	nn	139
3.29.2.19	nx	139
3.29.2.20	path	139
3.29.2.21	Pcr	140
3.29.2.22	Pe	140
3.29.2.23	T	140
3.29.2.24	va	140
3.29.2.25	VA	140
3.29.2.26	variable	140
3.29.2.27	variables	140
3.29.2.28	X	141
3.29.2.29	x_center	141
3.29.2.30	x_center_index	141
3.29.2.31	Xi	141
3.30	setup_adve Namespace Reference	141
3.30.1	Function Documentation	142
3.30.1.1	door()	142
3.30.1.2	spec()	142

3.30.2	Variable Documentation	142
3.30.2.1	B	142
3.30.2.2	D	143
3.30.2.3	d00	143
3.30.2.4	Db	143
3.30.2.5	E	143
3.30.2.6	ext	143
3.30.2.7	gamma_in	143
3.30.2.8	gamma_lazarian	143
3.30.2.9	gamma_nlld	144
3.30.2.10	gamma_tot	144
3.30.2.11	lm	144
3.30.2.12	lp	144
3.30.2.13	ism_values	144
3.30.2.14	mi	144
3.30.2.15	mn	144
3.30.2.16	ne	144
3.30.2.17	ni	145
3.30.2.18	nn	145
3.30.2.19	nx	145
3.30.2.20	path	145
3.30.2.21	Pcr	145
3.30.2.22	Pe	145
3.30.2.23	T	145
3.30.2.24	va	146
3.30.2.25	VA	146
3.30.2.26	variable	146
3.30.2.27	variables	146
3.30.2.28	X	146
3.30.2.29	x_center	146

3.30.2.30 x_center_index	147
3.30.2.31 Xi	147
3.31 setup_advst Namespace Reference	147
3.31.1 Function Documentation	148
3.31.1.1 door()	148
3.31.1.2 spec()	148
3.31.2 Variable Documentation	148
3.31.2.1 B	148
3.31.2.2 D	148
3.31.2.3 d00	148
3.31.2.4 Db	149
3.31.2.5 E	149
3.31.2.6 ext	149
3.31.2.7 gamma_in	149
3.31.2.8 gamma_lazarian	149
3.31.2.9 gamma_nlld	149
3.31.2.10 gamma_tot	149
3.31.2.11 lm	149
3.31.2.12 lp	150
3.31.2.13 ism_values	150
3.31.2.14 mi	150
3.31.2.15 mn	150
3.31.2.16 ne	150
3.31.2.17 ni	150
3.31.2.18 nn	150
3.31.2.19 nx	151
3.31.2.20 path	151
3.31.2.21 Pcr	151
3.31.2.22 Pe	151
3.31.2.23 T	151

3.31.2.24	va	151
3.31.2.25	VA	151
3.31.2.26	variable	152
3.31.2.27	variables	152
3.31.2.28	X	152
3.31.2.29	x_center	152
3.31.2.30	x_center_index	152
3.31.2.31	Xi	152
3.32	setup_diff Namespace Reference	153
3.32.1	Function Documentation	153
3.32.1.1	door()	153
3.32.2	Variable Documentation	154
3.32.2.1	B	154
3.32.2.2	D	154
3.32.2.3	d00	154
3.32.2.4	Db	154
3.32.2.5	E	154
3.32.2.6	ext	154
3.32.2.7	gamma_in	154
3.32.2.8	gamma_lazarian	155
3.32.2.9	gamma_nlld	155
3.32.2.10	gamma_tot	155
3.32.2.11	lm	155
3.32.2.12	lp	155
3.32.2.13	ism_values	155
3.32.2.14	mi	155
3.32.2.15	mn	155
3.32.2.16	ne	156
3.32.2.17	ni	156
3.32.2.18	nn	156

3.32.2.19 nx	156
3.32.2.20 path	156
3.32.2.21 Pcr	156
3.32.2.22 Pe	156
3.32.2.23 T	157
3.32.2.24 va	157
3.32.2.25 VA	157
3.32.2.26 variable	157
3.32.2.27 variables	157
3.32.2.28 X	157
3.32.2.29 x_center	158
3.32.2.30 x_center_index	158
3.32.2.31 Xi	158
3.33 setup_uniform Namespace Reference	158
3.33.1 Variable Documentation	159
3.33.1.1 B	159
3.33.1.2 D	159
3.33.1.3 d00	159
3.33.1.4 Db	159
3.33.1.5 E	159
3.33.1.6 ext	160
3.33.1.7 g_in	160
3.33.1.8 g_lz	160
3.33.1.9 gamma_in	160
3.33.1.10 gamma_lazarian	160
3.33.1.11 gamma_nlld	160
3.33.1.12 gamma_tot	160
3.33.1.13 I	160
3.33.1.14 lm	161
3.33.1.15 in_damping	161

3.33.1.16 lp	161
3.33.1.17 ism_values	161
3.33.1.18 kmin	161
3.33.1.19 mass	161
3.33.1.20 medium_props	161
3.33.1.21 mi	162
3.33.1.22 mn	162
3.33.1.23 ne	162
3.33.1.24 ni	162
3.33.1.25 nn	162
3.33.1.26 nx	162
3.33.1.27 path	162
3.33.1.28 Pcr	163
3.33.1.29 Pe	163
3.33.1.30 q	163
3.33.1.31 T	163
3.33.1.32 va	163
3.33.1.33 VA	163
3.33.1.34 variable	163
3.33.1.35 variables	164
3.33.1.36 X	164
3.33.1.37 x_center	164
3.33.1.38 x_center_index	164
3.33.1.39 Xi	164
3.34 setup_vdiff Namespace Reference	164
3.34.1 Function Documentation	165
3.34.1.1 door()	165
3.34.1.2 f()	166
3.34.2 Variable Documentation	166
3.34.2.1 B	166

3.34.2.2	D	166
3.34.2.3	d00	166
3.34.2.4	Db	166
3.34.2.5	E	166
3.34.2.6	ext	167
3.34.2.7	gamma_in	167
3.34.2.8	gamma_lazarian	167
3.34.2.9	gamma_nlld	167
3.34.2.10	gamma_tot	167
3.34.2.11	lm	167
3.34.2.12	lp	167
3.34.2.13	ism_values	167
3.34.2.14	mi	168
3.34.2.15	mn	168
3.34.2.16	ne	168
3.34.2.17	ni	168
3.34.2.18	nn	168
3.34.2.19	nx	168
3.34.2.20	path	168
3.34.2.21	Pcr	169
3.34.2.22	Pe	169
3.34.2.23	T	169
3.34.2.24	va	169
3.34.2.25	VA	169
3.34.2.26	variable	169
3.34.2.27	variables	169
3.34.2.28	X	170
3.34.2.29	x_center	170
3.34.2.30	x_center_index	170
3.34.2.31	Xi	170

3.35 show_data Namespace Reference	170
3.35.1 Function Documentation	171
3.35.1.1 colorArray()	171
3.35.1.2 colorFader()	171
3.35.1.3 getData()	171
3.35.1.4 readAxis()	171
3.35.1.5 readDataXE()	171
3.35.1.6 readInfo()	172
3.35.1.7 show()	172
3.35.2 Variable Documentation	172
3.35.2.1 elim	172
3.35.2.2 fig_save	172
3.35.2.3 source_center	172
3.35.2.4 vlim	172
3.36 ShowInjectionEvolution Namespace Reference	173
3.36.1 Function Documentation	173
3.36.1.1 readAxis()	173
3.36.1.2 readDataXE()	173
3.36.2 Variable Documentation	173
3.36.2.1 data	173
3.36.2.2 E	174
3.36.2.3 figsize	174
3.36.2.4 index	174
3.36.2.5 loc_id	174
3.36.2.6 X	174
3.37 SNR_evolution Namespace Reference	174
3.37.1 Function Documentation	175
3.37.1.1 InterpolatingSpline()	175
3.37.1.2 InverseTrigonalMatrix()	175
3.37.1.3 ProductMatrix()	175

3.37.1.4	Rsh()	176
3.37.2	Variable Documentation	176
3.37.2.1	ax0	176
3.37.2.2	ax1	176
3.37.2.3	bbox_to_anchor	176
3.37.2.4	c	176
3.37.2.5	fig	176
3.37.2.6	gs	176
3.37.2.7	hspace	177
3.37.2.8	kms	177
3.37.2.9	label	177
3.37.2.10	loc	177
3.37.2.11	lw	177
3.37.2.12	marker	177
3.37.2.13	ncol	177
3.37.2.14	pc	177
3.37.2.15	size_x	178
3.37.2.16	size_y	178
3.37.2.17	sub_x	178
3.37.2.18	sub_y	178
3.37.2.19	wspace	178
3.38	Split_solvers Namespace Reference	178
3.38.1	Function Documentation	179
3.38.1.1	generalized_diffusion()	179
3.38.1.2	TDMA()	179
3.38.2	Variable Documentation	179
3.38.2.1	c	179
3.38.2.2	D	179
3.38.2.3	dt	179
3.38.2.4	NX	180

3.38.2.5	t	180
3.38.2.6	t_ini	180
3.38.2.7	t_max	180
3.38.2.8	u	180
3.38.2.9	u_new_0	180
3.38.2.10	u_new_1	180
3.38.2.11	u_old	180
3.38.2.12	X	181
3.38.2.13	Xmax	181
3.38.2.14	Xmin	181
3.39	test_tesc Namespace Reference	181
3.39.1	Function Documentation	181
3.39.1.1	gauss()	182
3.39.1.2	sig()	182
3.39.1.3	tesc()	182
3.39.2	Variable Documentation	182
3.39.2.1	beta	182
3.39.2.2	c	182
3.39.2.3	e	182
3.39.2.4	E	183
3.39.2.5	Emin	183
3.39.2.6	Esn	183
3.39.2.7	figsize	183
3.39.2.8	GeV	183
3.39.2.9	kyr	183
3.39.2.10	label	183
3.39.2.11	Qcr	183
3.39.2.12	rho_0	184
3.39.2.13	t	184
3.39.2.14	xhi_0	184
3.39.2.15	xhi_cr	184

4	File Documentation	185
4.1	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/data_analysis/pcr_ip_2D.py File Reference	185
4.2	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/data_analysis/show_↔ data.py File Reference	186
4.3	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/namelist.py File Reference	186
4.4	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/fiducial_↔ tests/phases_energy_dependance/namelist_uniform.py File Reference	187
4.5	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/fiducial_↔ tests/phases_energy_dependance/setup_uniform.py File Reference	188
4.6	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_↔ adv.py File Reference	189
4.7	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_↔ adve.py File Reference	190
4.8	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_↔ advst.py File Reference	191
4.9	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_↔ diff.py File Reference	192
4.10	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_↔ vdiff.py File Reference	193
4.11	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_adv.py File Reference	194
4.12	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_↔ adve.py File Reference	195
4.13	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_↔ advst.py File Reference	196
4.14	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_diff.py File Reference	197
4.15	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_vdiff.py File Reference	198
4.16	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/setup.py File Reference	199
4.17	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/constants.h File Refer- ence	201
4.17.1	Variable Documentation	205
4.17.1.1	alpha	205
4.17.1.2	bbeta	205
4.17.1.3	c	205

4.17.1.4 C06	205
4.17.1.5 coherence_length	205
4.17.1.6 delta	205
4.17.1.7 delta_log_output	206
4.17.1.8 e	206
4.17.1.9 Eknee	206
4.17.1.10 electron_injection_rate	206
4.17.1.11 Emin	206
4.17.1.12 Esn	206
4.17.1.13 eta_acc	207
4.17.1.14 eta_gfree	207
4.17.1.15 eV	207
4.17.1.16 gam	207
4.17.1.17 GeV	207
4.17.1.18 inj_exp_alpha	207
4.17.1.19 injection_cutoff	207
4.17.1.20 injection_function_norm	208
4.17.1.21 injection_function_width	208
4.17.1.22 injection_shape_time	208
4.17.1.23 isotropy	208
4.17.1.24 kbolz	208
4.17.1.25 km	208
4.17.1.26 kms	209
4.17.1.27 kpc	209
4.17.1.28 kyr	209
4.17.1.29 log_first_data	209
4.17.1.30 mCII	209
4.17.1.31 me	209
4.17.1.32 Mej	210
4.17.1.33 MeV	210

4.17.1.34 mH2	210
4.17.1.35 mHel	210
4.17.1.36 mHI	210
4.17.1.37 mHII	210
4.17.1.38 mn	211
4.17.1.39 mp	211
4.17.1.40 nproc	211
4.17.1.41 number_out_data	211
4.17.1.42 oh_model	211
4.17.1.43 output_freq	211
4.17.1.44 pc	212
4.17.1.45 phi_c	212
4.17.1.46 pi	212
4.17.1.47 r_snr_thickness	212
4.17.1.48 set_background	212
4.17.1.49 sig_T	212
4.17.1.50 sigma_coherence	213
4.17.1.51 solver_Dilution	213
4.17.1.52 solver_ImAdvection	213
4.17.1.53 solver_ImDampGrowth	213
4.17.1.54 solver_ImSource1	213
4.17.1.55 solver_IpAdvection	213
4.17.1.56 solver_IpDampGrowth	214
4.17.1.57 solver_IpSource1	214
4.17.1.58 solver_PcrAdvection	214
4.17.1.59 solver_PcrAdvection2	214
4.17.1.60 solver_PcrAdvectionE	214
4.17.1.61 solver_PcrDiffusion	214
4.17.1.62 solver_PcrPerpDiff	215
4.17.1.63 solver_PcrSource1	215

4.17.1.64 solver_PcrSource2	215
4.17.1.65 solver_PeAdvection	215
4.17.1.66 solver_PeAdvection2	215
4.17.1.67 solver_PeAdvectionE	215
4.17.1.68 solver_PeAdvectionE1	216
4.17.1.69 solver_PeAdvectionE2	216
4.17.1.70 solver_PeDiffusion	216
4.17.1.71 solver_PePerpDiff	216
4.17.1.72 solver_PeSource1	216
4.17.1.73 solver_PeSource2	216
4.17.1.74 source_terms_exact	217
4.17.1.75 step_implicit	217
4.17.1.76 t_data_out_max	217
4.17.1.77 t_data_out_min	217
4.17.1.78 t_end_injection	217
4.17.1.79 t_start_injection	217
4.17.1.80 tesc_model	218
4.17.1.81 TeV	218
4.17.1.82 time_distrib_of_data	218
4.17.1.83 Tmax	218
4.17.1.84 tau_sat	218
4.17.1.85 verbose	218
4.17.1.86 xhi_0	219
4.17.1.87 xhi_cr	219
4.17.1.88 xi_n	219
4.17.1.89 yr	219
4.18 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/cr_source.h File Reference	219
4.18.1 Function Documentation	221
4.18.1.1 B_sat()	221
4.18.1.2 dNdE()	221

4.18.1.3	ff()	221
4.18.1.4	Finj()	222
4.18.1.5	GetEM()	222
4.18.1.6	GetEM_e()	222
4.18.1.7	GetTSed()	222
4.18.1.8	Pcr_ini()	222
4.18.1.9	Resc()	222
4.18.1.10	RSNR()	223
4.18.1.11	sigm()	223
4.18.1.12	tesc()	224
4.18.1.13	tesc_e()	224
4.18.1.14	theta()	224
4.18.1.15	u_sh()	224
4.18.2	Variable Documentation	224
4.18.2.1	Bcenter	224
4.18.2.2	m_neutral	224
4.18.2.3	ne	225
4.18.2.4	ni	225
4.18.2.5	nt	225
4.18.2.6	nx	225
4.18.2.7	parameters	225
4.18.2.8	sBcenter	225
4.18.2.9	scenter	225
4.18.2.10	scenter_index	225
4.18.2.11	smn	226
4.18.2.12	sne	226
4.18.2.13	sni	226
4.18.2.14	snx	226
4.18.2.15	sT	226
4.18.2.16	sX	226

4.18.2.17 T	226
4.18.2.18 x_center	226
4.18.2.19 x_center_index	227
4.18.2.20 Xi	227
4.19 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/freader.h File Reference	227
4.19.1 Function Documentation	228
4.19.1.1 readAxis()	228
4.19.1.2 search()	228
4.20 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/fwriter.h File Reference	228
4.20.1 Function Documentation	229
4.20.1.1 writeInfo()	229
4.20.1.2 writeXE()	229
4.21 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/logmaker.h File Reference	230
4.21.1 Function Documentation	230
4.21.1.1 showLog_0()	231
4.22 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/main.cpp File Reference	231
4.22.1 Function Documentation	231
4.22.1.1 main()	232
4.23 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/mathematics.h File Reference	232
4.23.1 Function Documentation	233
4.23.1.1 df1dx()	233
4.23.1.2 df2dx()	233
4.23.1.3 f1()	233
4.23.1.4 f2()	233
4.23.1.5 GetMax()	234
4.23.1.6 InterpolatingSpline()	234
4.23.1.7 InverseTrigonalMatrix()	234
4.23.1.8 minmod()	234
4.23.1.9 NewtonRaphson()	234
4.23.1.10 ProductMatrix()	234

4.23.1.11 TDMA()	235
4.23.1.12 transpose()	235
4.24 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/cr_escape.cpp File Reference	235
4.24.1 Function Documentation	236
4.24.1.1 df1dx()	236
4.24.1.2 df2dx()	236
4.24.1.3 f1()	236
4.24.1.4 f2()	236
4.24.1.5 GetMax()	236
4.24.1.6 GetTesc()	236
4.24.1.7 InterpolatingSpline()	237
4.24.1.8 InverseTrigonalMatrix()	237
4.24.1.9 main()	237
4.24.1.10 NewtonRaphson()	237
4.24.1.11 ProductMatrix()	237
4.25 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/PDE_solvers.py File Reference	237
4.26 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/Split_solvers.py File Reference	238
4.27 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/out.h File Reference	239
4.27.1 Function Documentation	240
4.27.1.1 getLogOutput()	240
4.27.1.2 getOutput()	240
4.27.1.3 outputData()	241
4.27.1.4 setTmax()	241
4.27.1.5 specificOutputData()	241
4.28 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/read2D.h File Reference	241
4.28.1 Function Documentation	242
4.28.1.1 parse2DCsvFile()	242
4.29 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/solver1D.h File Reference	242
4.29.1 Function Documentation	244

4.29.1.1	advectionSolverE()	244
4.29.1.2	advectionSolverE1()	244
4.29.1.3	advectionSolverE2()	245
4.29.1.4	advectionSolverX()	245
4.29.1.5	CRsInjectionSourceSolver()	245
4.29.1.6	dilute_solver()	245
4.29.1.7	electron_source()	246
4.29.1.8	NotMove()	246
4.29.1.9	perpendicular_diffusion_solver()	246
4.29.1.10	sourceGrowthDampRateSolver()	246
4.29.1.11	sourceSolver()	247
4.29.1.12	thetaDiffusionSolver()	247
4.30	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/tools.h File Reference	247
4.30.1	Function Documentation	248
4.30.1.1	absmaxElement2D()	248
4.30.1.2	maxElement1D()	248
4.30.1.3	maxElement2D()	248
4.30.1.4	minElement1D()	249
4.30.1.5	minElement2D()	249
4.31	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/background_↔ diffusion_coefficient.py File Reference	249
4.32	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/obsolete/background_↔ _diffusion_coefficient.py File Reference	249
4.33	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/constants.py File Ref- erence	250
4.34	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/d1_grid_generator.py File Reference	251
4.35	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/damping.py File Ref- erence	251
4.36	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/Data_reader.py File Reference	252
4.37	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/freader.py File Refer- ence	252

4.38	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/fwriter.py File Reference	252
4.39	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/gaussian_sub↔ Normalization.py File Reference	253
4.40	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/mathmethods.py File Reference	253
4.41	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/Output_functions.py File Reference	254
4.42	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/phases_collection.py File Reference	254
4.43	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/physical_models.py File Reference	255
4.44	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/background↔ _diffusion_coefficient_3.py File Reference	255
4.45	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/damping↔ _models.py File Reference	256
4.46	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/electrons↔ _emax_t.py File Reference	258
4.47	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/↔ EscapeModel_protons.py File Reference	258
4.48	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/↔ EscapeModel_protons_2.py File Reference	260
4.49	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/obsolete/background↔ _diffusion_coefficient_2.py File Reference	261
4.50	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/SNR↔ _evolution.py File Reference	262
4.51	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/ShowInjection↔ Evolution.py File Reference	263
4.52	/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/test_tesc.py File Ref- erence	263
Index		265

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

background_diffusion_coefficient	5
background_diffusion_coefficient_2	12
background_diffusion_coefficient_3	16
constants	27
d1_grid_generator	30
damping	31
damping_models	33
Data_reader	41
electrons_emax_t	43
EscapeModel_protons	48
EscapeModel_protons_2	58
freader	67
fwriter	67
gaussian_subNormalization	69
mathmethods	72
namelist	75
namelist_adv	81
namelist_adve	87
namelist_advst	93
namelist_diff	98
namelist_uniform	104
namelist_vdiff	110
Output_functions	116
pcr_ip_2D	116
PDE_solvers	122
phases_collection	127
physical_models	128
setup	129
setup_adv	136
setup_adve	141
setup_advst	147
setup_diff	153
setup_uniform	158
setup_vdiff	164
show_data	170

ShowInjectionEvolution	173
SNR_evolution	174
Split_solvers	178
test_tesc	181

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/namelist.py	186
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/setup.py	199
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/data_analysis/pcr_ip_2D.py . . .	185
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/data_analysis/show_data.py . .	186
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_adv.py .	189
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_adve.py .	190
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_advst.py	191
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_diff.py . .	192
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_vdiff.py .	193
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_adv.py . . .	194
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_adve.py . .	195
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_advst.py . .	196
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_diff.py . . .	197
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_vdiff.py . . .	198
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/fiducial_tests/phases↵ _energy_dependance/namelist_uniform.py	187
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/fiducial_tests/phases↵ _energy_dependance/setup_uniform.py	188
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/constants.h	201
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/cr_source.h	219
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/freader.h	227
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/fwritter.h	228
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/logmaker.h	230
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/main.cpp	231
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/mathematics.h	232
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/out.h	239
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/read2D.h	241
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/solver1D.h	242
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/tools.h	247
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/cr_escape.cpp	235
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/PDE_solvers.py . . .	237
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/Split_solvers.py . . .	238
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/background_diffusion_↵ coefficient.py	249
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/constants.py	250

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/d1_grid_generator.py . . .	251
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/damping.py	251
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/Data_reader.py	252
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/freader.py	252
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/fwriter.py	252
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/gaussian_subNormalization.py	253
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/mathmethods.py	253
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/Output_functions.py	254
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/phases_collection.py	254
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/physical_models.py	255
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/ShowInjectionEvolution.py	263
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/test_tesc.py	263
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/background_← _diffusion_coefficient_3.py	255
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/damping_← models.py	256
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/electrons_← emax_t.py	258
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/Escape_← Model_protons.py	258
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/Escape_← Model_protons_2.py	260
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/SNR_← evolution.py	262
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/obsolete/background_← _diffusion_coefficient.py	249
/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/obsolete/background_← _diffusion_coefficient_2.py	261

Chapter 3

Namespace Documentation

3.1 background_diffusion_coefficient Namespace Reference

Functions

- def `IonNeutral_Damping` (k, medium_props, nu_n=0, theta=0)
- def `Duu_Alfven_Slab_Linear_Undamped` (mu, E, medium_props, mass=cst.mp, kmin=1e-20, q=1.5, l=1e-4)
- def `Kappa_zz` (E, medium_props, mass=cst.mp, kmin=(50.*cst.pc) **(-1), q=5./3, l=1e-4)
- def `J` (a, b, c, D, q, x)

Variables

- `phase` = ism.CNM
- `B0` = phase.get('B')
- `mi` = phase.get("mi")
- `mn` = phase.get("mn")
- `ni` = phase.get("ni")
- `nn` = phase.get("nn")
- `T` = phase.get("T")
- tuple `chi` = (mn*nn)/(mi*ni)
- `VAi` = B0/np.sqrt(4*np.pi*mi*ni)
- `VA` = B0/np.sqrt(4*np.pi*(mi*ni + mn*nn))
- int `nu_in` = 2*nn*8.4e-9*(50/1e4)**0.4
- tuple `nu_ni` = chi**(-1.)*nu_in
- int `k_min` = 1e-20
- int `k_cm` = 1e-15
- int `k_cp` = 1e-20
- int `k_max` = 2*nu_ni/VA
- int `E` = 10*cst.GeV
- `m` = cst.mp
- int `gamma` = 1 + (E/(m*cst.c**2))
- `v` = cst.c*np.sqrt(1 - (1/(E/(m*cst.c**2) + 1))**2)
- int `p` = gamma*m*v
- `Omega0` = cst.e*B0/(m*cst.c)
- int `Omega` = Omega0/gamma
- `mu` = np.linspace(-0.99, 0.99, 100)
- `d_uu` = np.zeros(len(mu))

- `int k_zz = 0.`
- `dmu = mu[1] - mu[0]`
- `int ltot = 1e-1`
- `float q = 1.5`
- `tuple a = (v*mu[iii] - VA)**2`
- `int b = 2*Omega*(v*mu[iii] - VA)`
- `int c = Omega**2 + (- nu_in/2)**2`
- `int D = b**2 - 4*a*c`
- `eps = VA/v`
- `int gs0 = 2*(q-1)*(B0**2/(8*np.pi))*ltot*k_cp**2*(q - 1)`
- `def fj_p = J(a, b, c, D, q, k_max) - J(a, b, c, D, q, k_cp)`
- `def fj_m = J(a, -b, c, D, q, k_max) - J(a, -b, c, D, q, k_cp)`
- `int l = -2*gs0*(1 - mu[iii]*eps)**2*(- nu_in/2.)*fj_p`

3.1.1 Function Documentation

3.1.1.1 Duu_Alfven_Slab_Linear_Undamped()

```
def background_diffusion_coefficient.Duu_Alfven_Slab_Linear_Undamped (
    mu,
    E,
    medium_props,
    mass = cst.mp,
    kmin = 1e-20,
    q = 1.5,
    I = 1e-4 )
```

See. Schlickeiser (2002, Chap 13.1.3.1, p.318)

Parameters

```
mu : TYPE
    DESCRIPTION.
medium_props : TYPE
    DESCRIPTION.
particles_props : TYPE
    DESCRIPTION.
kmin : TYPE, optional
    DESCRIPTION. The default is 1e-20.
q : TYPE, optional
    DESCRIPTION. The default is 1.5.
```

Returns

```
D : TYPE
    DESCRIPTION.
```


3.1.1.2 IonNeutral_Damping()

```
def background_diffusion_coefficient.IonNeutral_Damping (
    k,
    medium_props,
    nu_n = 0,
    theta = 0 )
```

3.1.1.3 J()

```
def background_diffusion_coefficient.J (
    a,
    b,
    c,
    D,
    q,
    x )
```

3.1.1.4 Kappa_zz()

```
def background_diffusion_coefficient.Kappa_zz (
    E,
    medium_props,
    mass = cst.mp,
    kmin = (50.*cst.pc)**(-1),
    q = 5./3,
    I = 1e-4 )
```

Parameters

```
-----
E : TYPE
    DESCRIPTION.
medium_props : TYPE
    DESCRIPTION.
mass : TYPE float, optional
    Mass of the diffusin particle. The default is m_proton
kmin : TYPE float, optional
    Minumun length in cm^-1 for the turbulence spectra -> Injection length. The default is 50pc**(-1)
q : TYPE float, optional
    Spectral index of the Kolmogorov-like turbulence
    spectrum. The default is 5./3.
I : TYPE float, optional
    Turbulence average level. Defaut is 10^-4
```

Returns

```
-----
TYPE
    DESCRIPTION.
```

3.1.2 Variable Documentation

3.1.2.1 a

```
tuple background_diffusion_coefficient.a = (v*mu[ii] - VA)**2
```

3.1.2.2 b

```
int background_diffusion_coefficient.b = 2*Omega*(v*mu[ii] - VA)
```

3.1.2.3 B0

```
background_diffusion_coefficient.B0 = phase.get('B')
```

3.1.2.4 c

```
int background_diffusion_coefficient.c = Omega**2 + (- nu_in/2)**2
```

3.1.2.5 chi

```
tuple background_diffusion_coefficient.chi = (mn*nn)/(mi*ni)
```

3.1.2.6 D

```
int background_diffusion_coefficient.D = b**2 - 4*a*c
```

3.1.2.7 d_uu

```
background_diffusion_coefficient.d_uu = np.zeros(len(mu))
```

3.1.2.8 dmu

```
background_diffusion_coefficient.dmu = mu[1] - mu[0]
```

3.1.2.9 E

```
int background_diffusion_coefficient.E = 10*cst.GeV
```

3.1.2.10 eps

```
background_diffusion_coefficient.eps = VA/v
```

3.1.2.11 fj_m

```
def background_diffusion_coefficient.fj_m = J(a, -b, c, D, q, k_max) - J(a, -b, c, D, q, k_cp)
```

3.1.2.12 fj_p

```
def background_diffusion_coefficient.fj_p = J(a, b, c, D, q, k_max) - J(a, b, c, D, q, k_cp)
```

3.1.2.13 gamma

```
int background_diffusion_coefficient.gamma = 1 + (E / (m*cst.c**2))
```

3.1.2.14 gs0

```
int background_diffusion_coefficient.gs0 = 2*(q-1)*(B0**2/(8*np.pi))*Itot*k_cp**(q - 1)
```

3.1.2.15 I

```
int background_diffusion_coefficient.I = -2*gs0*(1 - mu[ii]*eps)**2*(- nu_in/2.)*fj_p
```

3.1.2.16 Itot

```
int background_diffusion_coefficient.Itot = 1e-1
```

3.1.2.17 k_cm

```
int background_diffusion_coefficient.k_cm = 1e-15
```

3.1.2.18 k_cp

```
int background_diffusion_coefficient.k_cp = 1e-20
```

3.1.2.19 k_max

```
int background_diffusion_coefficient.k_max = 2*nu\_ni/VA
```

3.1.2.20 k_min

```
int background_diffusion_coefficient.k_min = 1e-20
```

3.1.2.21 k_zz

```
int background_diffusion_coefficient.k_zz = 0.
```

3.1.2.22 m

```
background_diffusion_coefficient.m = cst.mp
```

3.1.2.23 mi

```
background_diffusion_coefficient.mi = phase.get("mi")
```

3.1.2.24 mn

```
background_diffusion_coefficient.mn = phase.get("mn")
```

3.1.2.25 mu

```
background_diffusion_coefficient.mu = np.linspace(-0.99, 0.99, 100)
```

3.1.2.26 ni

```
background_diffusion_coefficient.ni = phase.get("ni")
```

3.1.2.27 nn

```
background_diffusion_coefficient.nn = phase.get("nn")
```

3.1.2.28 nu_in

```
int background_diffusion_coefficient.nu_in = 2*nn*8.4e-9*(50/1e4)**0.4
```

3.1.2.29 nu_ni

```
tuple background_diffusion_coefficient.nu_ni = chi**(-1.)*nu_in
```

3.1.2.30 Omega

```
int background_diffusion_coefficient.Omega = Omega0/gamma
```

3.1.2.31 Omega0

```
background_diffusion_coefficient.Omega0 = cst.e*B0/(m*cst.c)
```

3.1.2.32 p

```
int background_diffusion_coefficient.p = gamma*m*v
```

3.1.2.33 phase

```
background_diffusion_coefficient.phase = ism.CNM
```

3.1.2.34 q

```
float background_diffusion_coefficient.q = 1.5
```

3.1.2.35 T

```
background_diffusion_coefficient.T = phase.get("T")
```

3.1.2.36 v

```
background_diffusion_coefficient.v = cst.c*np.sqrt(1 - (1/(E/(m*cst.c**2) + 1))**2)
```

3.1.2.37 VA

```
background_diffusion_coefficient.VA = B0/np.sqrt(4*np.pi*(mi*ni + mn*nn))
```

3.1.2.38 VAi

```
background_diffusion_coefficient.VAi = B0/np.sqrt(4*np.pi*mi*ni)
```

3.2 background_diffusion_coefficient_2 Namespace Reference

Functions

- def `IonNeutral_Damping` (`k`, `medium_props`, `nu_n=0`, `theta=0`)
- def `R` (`k`, `medium_props`, `particles_props`, `mu`, `kind="+"`)
- def `g_s` (`k`, `kmin`, `q`, `medium_props`)

Variables

- `medium_props` = `ism.WNM`
- `float E` = `0.001*cst.GeV`
- `m` = `cst.mp`
- `int gamma` = `1 + (E/(m*cst.c**2))`
- `v` = `cst.c*np.sqrt(1 - (1/(E/(m*cst.c**2) + 1))**2)`
- `int p` = `gamma*m*v`
- `Omega0` = `cst.e*medium_props.get("B")/(m*cst.c)`
- `int Omega` = `Omega0/gamma`
- dictionary `particles_props` = `{"v":v, "Omega":Omega}`
- `mu` = `np.linspace(-0.99, 0.99, 100)`
- `k` = `np.logspace(-20, -10, 100)`
- `D_uu` = `np.zeros(len(mu))`
- `int kmin` = `1e-17`
- `float q` = `1.5`
- `int k_zz` = `0.`
- `float dk` = `0.5*(k[ii+1] - k[ii-1])`
- `B0` = `medium_props.get("B")`
- `def w` = `IonNeutral_Damping(k[ii], medium_props)`
- `def wtot` = `w.get("wr") + 1j*w.get("wi")`
- `float l` = `dk*g_s(k[ii], kmin, q, medium_props)*(1 - (mu[jj]*wtot)/(k[ii]*v))**2`

3.2.1 Function Documentation

3.2.1.1 `g_s()`

```
def background_diffusion_coefficient_2.g_s (
    k,
    kmin,
    q,
    medium_props )
```

3.2.1.2 `IonNeutral_Damping()`

```
def background_diffusion_coefficient_2.IonNeutral_Damping (
    k,
    medium_props,
    nu_n = 0,
    theta = 0 )
```

3.2.1.3 R()

```
def background_diffusion_coefficient_2.R (
    k,
    medium_props,
    particles_props,
    mu,
    kind = "+" )
```

3.2.2 Variable Documentation

3.2.2.1 B0

```
background_diffusion_coefficient_2.B0 = medium_props.get("B")
```

3.2.2.2 D_uu

```
background_diffusion_coefficient_2.D_uu = np.zeros(len(mu))
```

3.2.2.3 dk

```
float background_diffusion_coefficient_2.dk = 0.5*(k[ii+1] - k[ii-1])
```

3.2.2.4 E

```
float background_diffusion_coefficient_2.E = 0.001*cst.GeV
```

3.2.2.5 gamma

```
int background_diffusion_coefficient_2.gamma = 1 + (E / (m*cst.c**2))
```


3.2.2.6 I

```
float background_diffusion_coefficient_2.I = dk*g_s(k[ii], kmin, q, medium_props)*(1 - (mu[jj]*wtot)/(k[ii]*v))
```

3.2.2.7 k

```
background_diffusion_coefficient_2.k = np.logspace(-20, -10, 100)
```

3.2.2.8 k_zz

```
int background_diffusion_coefficient_2.k_zz = 0.
```

3.2.2.9 kmin

```
int background_diffusion_coefficient_2.kmin = 1e-17
```

3.2.2.10 m

```
background_diffusion_coefficient_2.m = cst.mp
```

3.2.2.11 medium_props

```
background_diffusion_coefficient_2.medium_props = ism.WNM
```

3.2.2.12 mu

```
background_diffusion_coefficient_2.mu = np.linspace(-0.99, 0.99, 100)
```

3.2.2.13 Omega

```
dictionary background_diffusion_coefficient_2.Omega = Omega0/gamma
```

3.2.2.14 Omega0

```
background_diffusion_coefficient_2.Omega0 = cst.e*medium_props.get("B")/(m*cst.c)
```

3.2.2.15 p

```
int background_diffusion_coefficient_2.p = gamma*m*v
```

3.2.2.16 particles_props

```
dictionary background_diffusion_coefficient_2.particles_props = {"v":v, "Omega":Omega}
```

3.2.2.17 q

```
float background_diffusion_coefficient_2.q = 1.5
```

3.2.2.18 v

```
background_diffusion_coefficient_2.v = cst.c*np.sqrt(1 - (1/(E/(m*cst.c**2) + 1))**2)
```

3.2.2.19 w

```
def background_diffusion_coefficient_2.w = IonNeutral_Damping(k[ii], medium_props)
```

3.2.2.20 wtot

```
def background_diffusion_coefficient_2.wtot = w.get("wr") + 1j*w.get("wi")
```

3.3 background_diffusion_coefficient_3 Namespace Reference

Functions

- def [IonNeutral_Damping](#) (k, medium_props, nu_n=0, theta=0)
- def [Duu_Alfven_Slab_Linear_Undamped](#) (mu, E, medium_props, mass=cst.mp, kmin=1e-20, q=1.5, l=1e-4)
- def [Kappa_zz](#) (E, medium_props, mass=cst.mp, kmin=1e-20, q=5./3, l=1e28)
- def [kappa_zz_BC](#) (E, d00, delta)

Variables

- float `Emin` = 0.1*cst.GeV
- int `Emax` = 100*cst.TeV
- `E` = np.logspace(np.log10(`Emin`), np.log10(`Emax`), 100)
- `K_zz_bc_1` = np.zeros(len(`E`))
- `K_zz_bc_2` = np.zeros(len(`E`))
- `K_zz_1` = np.zeros(len(`E`))
- `K_zz_2` = np.zeros(len(`E`))
- `K_zz_HII_1` = np.zeros(len(`E`))
- `K_zz_WIM_1` = np.zeros(len(`E`))
- `K_zz_WNM_1` = np.zeros(len(`E`))
- `K_zz_CNM_1` = np.zeros(len(`E`))
- `K_zz_DiM_1` = np.zeros(len(`E`))
- `K_zz_DeM_1` = np.zeros(len(`E`))
- `K_zz_DeC_1` = np.zeros(len(`E`))
- `K_zz_HII_2` = np.zeros(len(`E`))
- `K_zz_WIM_2` = np.zeros(len(`E`))
- `K_zz_WNM_2` = np.zeros(len(`E`))
- `K_zz_CNM_2` = np.zeros(len(`E`))
- `K_zz_DiM_2` = np.zeros(len(`E`))
- `K_zz_DeM_2` = np.zeros(len(`E`))
- `K_zz_DeC_2` = np.zeros(len(`E`))
- `HII`
- `mass`
- `mp`
- `kmin`
- `q`
- `l`
- `WIM`
- `WNM`
- `CNM`
- `DiM`
- `DeM`
- `DeC`
- int `size_x` = 4
- int `size_y` = 3
- int `sub_x` = 2
- int `sub_y` = 2
- `fig` = plt.figure(figsize=(`size_x`*`sub_x`,`size_y`*`sub_y`))
- `gs` = gridspec.GridSpec(ncols= `sub_x`, nrows = `sub_y`, figure = `fig`)
- `wspace`
- `hspace`
- `ax0` = fig.add_subplot(`gs`[0, 0])
- `GeV`
- `c`
- `ls`
- `facecolor`
- `alpha`
- `hatch`
- `label`
- `loc`
- `ax1` = fig.add_subplot(`gs`[0, 1])
- `ax2` = fig.add_subplot(`gs`[1, 0])
- `ax3` = fig.add_subplot(`gs`[1, 1])

- list [custom_lines](#)
- [handles](#)
- [bbox_to_anchor](#)
- [ncol](#)
- int [ymin](#) = 1e27
- int [ymax](#) = 1e33
- int [xmin](#) = 1e-1
- int [xmax](#) = 1e5

3.3.1 Function Documentation

3.3.1.1 Duu_Alfven_Slab_Linear_Undamped()

```
def background_diffusion_coefficient_3.Duu_Alfven_Slab_Linear_Undamped (
    mu,
    E,
    medium_props,
    mass = cst.mp,
    kmin = 1e-20,
    q = 1.5,
    I = 1e-4 )
```

See. Schlickeiser (2002, Chap 13.1.3.1, p.318)

Parameters

```
-----
mu : TYPE
    DESCRIPTION.
medium_props : TYPE
    DESCRIPTION.
particles_props : TYPE
    DESCRIPTION.
kmin : TYPE, optional
    DESCRIPTION. The default is 1e-20.
q : TYPE, optional
    DESCRIPTION. The default is 1.5.
```

Returns

```
-----
D : TYPE
    DESCRIPTION.
```

3.3.1.2 IonNeutral_Damping()

```
def background_diffusion_coefficient_3.IonNeutral_Damping (
    k,
    medium_props,
    nu_n = 0,
    theta = 0 )
```

3.3.1.3 Kappa_zz()

```
def background_diffusion_coefficient_3.Kappa_zz (
    E,
    medium_props,
    mass = cst.mp,
    kmin = 1e-20,
    q = 5./3,
    I = 1e28 )

Parameters
-----
E : TYPE
    DESCRIPTION.
medium_props : TYPE
    DESCRIPTION.
mass : TYPE float, optional
    Mass of the diffusin particle. The default is m_proton
kmin : TYPE float, optional
    Minimun length in cm^-1 for the turbulence spectra. The default is 1e-20.
q : TYPE float, optional
    Spectral index of the Kolmogorov-like turbulence
    spectrum. The default is 5./3.
I : TYPE float, optional
    Diffusion coefficient normalization value for 1GeV particle and 5./3 spectrum. The default is 1e28.

Returns
-----
TYPE
    DESCRIPTION.
```

3.3.1.4 kappa_zz_BC()

```
def background_diffusion_coefficient_3.kappa_zz_BC (
    E,
    d00,
    delta )
```

3.3.2 Variable Documentation

3.3.2.1 alpha

```
background_diffusion_coefficient_3.alpha
```

3.3.2.2 ax0

```
background_diffusion_coefficient_3.ax0 = fig.add_subplot(gs[0, 0])
```

3.3.2.3 ax1

```
background_diffusion_coefficient_3.ax1 = fig.add_subplot(gs[0, 1])
```

3.3.2.4 ax2

```
background_diffusion_coefficient_3.ax2 = fig.add_subplot(gs[1, 0])
```

3.3.2.5 ax3

```
background_diffusion_coefficient_3.ax3 = fig.add_subplot(gs[1, 1])
```

3.3.2.6 bbox_to_anchor

```
background_diffusion_coefficient_3.bbox_to_anchor
```

3.3.2.7 c

```
background_diffusion_coefficient_3.c
```

3.3.2.8 CNM

```
background_diffusion_coefficient_3.CNM
```

3.3.2.9 custom_lines

```
background_diffusion_coefficient_3.custom_lines
```

Initial value:

```
1 = [Line2D([0],[0], color = "white", label = "ISM independant"),
2     Line2D([0],[0], color = "white", label = "ISM dependant"),
3     Line2D([0],[0], color = "black", ls = '--', label = "$\\delta = 0.5$, $d_{00} = 10^{28}$
cm$^{2}$s$^{-1}$"),
4     Line2D([0],[0], color = "black", ls = '--', label = "$q = 3/2$, $I_{\\rm} = 10^{-4}$"),
5     Line2D([0],[0], color = "black", ls = '-', label = "$\\delta = 2/3$, $d_{00} = 10^{29}$
cm$^{2}$s$^{-1}$"),
6     Line2D([0],[0], color = "black", ls = '-', label = "$q = 5/3$, $I_{\\rm} = 10^{-4}$")
7 ]
```

3.3.2.10 DeC

```
background_diffusion_coefficient_3.DeC
```

3.3.2.11 DeM

```
background_diffusion_coefficient_3.DeM
```

3.3.2.12 DiM

```
background_diffusion_coefficient_3.DiM
```

3.3.2.13 E

```
background_diffusion_coefficient_3.E = np.logspace(np.log10(Emin), np.log10(Emax), 100)
```

3.3.2.14 Emax

```
int background_diffusion_coefficient_3.Emax = 100*cst.TeV
```

3.3.2.15 Emin

```
float background_diffusion_coefficient_3.Emin = 0.1*cst.GeV
```

3.3.2.16 facecolor

```
background_diffusion_coefficient_3.facecolor
```

3.3.2.17 fig

```
background_diffusion_coefficient_3.fig = plt.figure(figsize=(size_x*sub_x, size_y*sub_y))
```

3.3.2.18 GeV

`background_diffusion_coefficient_3.GeV`

3.3.2.19 gs

```
background_diffusion_coefficient_3.gs = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure
= fig )
```

3.3.2.20 handles

`background_diffusion_coefficient_3.handles`

3.3.2.21 hatch

`background_diffusion_coefficient_3.hatch`

3.3.2.22 HII

`background_diffusion_coefficient_3.HII`

3.3.2.23 hspace

`background_diffusion_coefficient_3.hspace`

3.3.2.24 I

`background_diffusion_coefficient_3.I`

3.3.2.25 K_zz_1

```
background_diffusion_coefficient_3.K_zz_1 = np.zeros(len(E))
```

3.3.2.26 K_zz_2

```
background_diffusion_coefficient_3.K_zz_2 = np.zeros(len(E))
```

3.3.2.27 K_zz_bc_1

```
background_diffusion_coefficient_3.K_zz_bc_1 = np.zeros(len(E))
```

3.3.2.28 K_zz_bc_2

```
background_diffusion_coefficient_3.K_zz_bc_2 = np.zeros(len(E))
```

3.3.2.29 K_zz_CNM_1

```
background_diffusion_coefficient_3.K_zz_CNM_1 = np.zeros(len(E))
```

3.3.2.30 K_zz_CNM_2

```
background_diffusion_coefficient_3.K_zz_CNM_2 = np.zeros(len(E))
```

3.3.2.31 K_zz_DeC_1

```
background_diffusion_coefficient_3.K_zz_DeC_1 = np.zeros(len(E))
```

3.3.2.32 K_zz_DeC_2

```
background_diffusion_coefficient_3.K_zz_DeC_2 = np.zeros(len(E))
```

3.3.2.33 K_zz_DeM_1

```
background_diffusion_coefficient_3.K_zz_DeM_1 = np.zeros(len(E))
```

3.3.2.34 K_zz_DeM_2

```
background_diffusion_coefficient_3.K_zz_DeM_2 = np.zeros(len(E))
```

3.3.2.35 K_zz_DiM_1

```
background_diffusion_coefficient_3.K_zz_DiM_1 = np.zeros(len(E))
```

3.3.2.36 K_zz_DiM_2

```
background_diffusion_coefficient_3.K_zz_DiM_2 = np.zeros(len(E))
```

3.3.2.37 K_zz_HII_1

```
background_diffusion_coefficient_3.K_zz_HII_1 = np.zeros(len(E))
```

3.3.2.38 K_zz_HII_2

```
background_diffusion_coefficient_3.K_zz_HII_2 = np.zeros(len(E))
```

3.3.2.39 K_zz_WIM_1

```
background_diffusion_coefficient_3.K_zz_WIM_1 = np.zeros(len(E))
```

3.3.2.40 K_zz_WIM_2

```
background_diffusion_coefficient_3.K_zz_WIM_2 = np.zeros(len(E))
```

3.3.2.41 K_zz_WNM_1

```
background_diffusion_coefficient_3.K_zz_WNM_1 = np.zeros(len(E))
```

3.3.2.42 K_zz_WNM_2

```
background_diffusion_coefficient_3.K_zz_WNM_2 = np.zeros(len(E))
```

3.3.2.43 kmin

```
background_diffusion_coefficient_3.kmin
```

3.3.2.44 label

```
background_diffusion_coefficient_3.label
```

3.3.2.45 loc

```
background_diffusion_coefficient_3.loc
```

3.3.2.46 ls

```
background_diffusion_coefficient_3.ls
```

3.3.2.47 mass

```
background_diffusion_coefficient_3.mass
```

3.3.2.48 mp

```
background_diffusion_coefficient_3.mp
```

3.3.2.49 ncol

```
background_diffusion_coefficient_3.ncol
```

3.3.2.50 q

```
background_diffusion_coefficient_3.q
```

3.3.2.51 size_x

```
int background_diffusion_coefficient_3.size_x = 4
```

3.3.2.52 size_y

```
int background_diffusion_coefficient_3.size_y = 3
```

3.3.2.53 sub_x

```
int background_diffusion_coefficient_3.sub_x = 2
```

3.3.2.54 sub_y

```
int background_diffusion_coefficient_3.sub_y = 2
```

3.3.2.55 WIM

```
background_diffusion_coefficient_3.WIM
```

3.3.2.56 WNM

```
background_diffusion_coefficient_3.WNM
```

3.3.2.57 wspace

```
background_diffusion_coefficient_3.wspace
```

3.3.2.58 xmax

```
int background_diffusion_coefficient_3.xmax = 1e5
```

3.3.2.59 xmin

```
int background_diffusion_coefficient_3.xmin = 1e-1
```

3.3.2.60 ymax

```
int background_diffusion_coefficient_3.ymax = 1e33
```

3.3.2.61 ymin

```
int background_diffusion_coefficient_3.ymin = 1e27
```

3.4 constants Namespace Reference

Variables

- float `yr` = 3.154e+7
- int `kyr` = 1e3*yr
- float `pc` = 3.086e18
- int `kpc` = 1.e3*pc
- float `GeV` = 0.00160218
- int `TeV` = 1e3*GeV
- float `eV` = `GeV`*1e-9
- int `MeV` = 1e-3*GeV
- float `me` = 9.10938e-28
- float `mp` = 1.6726219e-24
- float `mn` = 1.6749286e-24
- float `mHl` = `mp`
- float `mHll` = `mp`
- int `mHel` = 2*`mp` + 2*`mn`
- int `mHell` = 2*`mp` + 2*`mn`
- int `mCll` = 6*`mp` + 6*`mn`
- int `mHCOll` = 8*`mp` + 8*`mn` + 6*`mp` + 6*`mn` + `mp` + `mn`
- int `mH2` = 2*`mp`
- float `e` = 4.80326e-10
- int `c` = 29979245800.
- float `kbolz` = 1.3807e-16
- int `kms` = 1e5

3.4.1 Variable Documentation

3.4.1.1 c

```
int constants.c = 29979245800.
```

3.4.1.2 e

```
float constants.e = 4.80326e-10
```

3.4.1.3 eV

```
float constants.eV = GeV*1e-9
```

3.4.1.4 GeV

```
float constants.GeV = 0.00160218
```

3.4.1.5 kbolz

```
float constants.kbolz = 1.3807e-16
```

3.4.1.6 kms

```
int constants.kms = 1e5
```

3.4.1.7 kpc

```
int constants.kpc = 1.e3*pc
```

3.4.1.8 kyr

```
int constants.kyr = 1e3*yr
```

3.4.1.9 mCII

```
int constants.mCII = 6*mp + 6*mn
```

3.4.1.10 me

```
float constants.me = 9.10938e-28
```

3.4.1.11 MeV

```
int constants.MeV = 1e-3*GeV
```

3.4.1.12 mH2

```
int constants.mH2 = 2*mp
```

3.4.1.13 mHCOII

```
int constants.mHCOII = 8*mp + 8*mn + 6*mp + 6*mn + mp + mn
```

3.4.1.14 mHeI

```
int constants.mHeI = 2*mp + 2*mn
```

3.4.1.15 mHeII

```
int constants.mHeII = 2*mp + 2*mn
```

3.4.1.16 mHI

```
float constants.mHI = mp
```

3.4.1.17 mHII

```
float constants.mHII = mp
```

3.4.1.18 mn

```
float constants.mn = 1.6749286e-24
```

3.4.1.19 mp

```
float constants.mp = 1.6726219e-24
```

3.4.1.20 pc

```
float constants.pc = 3.086e18
```

3.4.1.21 TeV

```
int constants.TeV = 1e3*GeV
```

3.4.1.22 yr

```
float constants.yr = 3.154e+7
```

3.5 d1_grid_generator Namespace Reference

Functions

- def [grid](#) (Smin, Smax, Ns, name, s_center=None, width=None, smooth=None, dXmin=0.01 *[cst.pc](#))

3.5.1 Function Documentation

3.5.1.1 grid()

```
def dl_grid_generator.grid (
    Smin,
    Smax,
    Ns,
    name,
    s_center = None,
    width = None,
    smooth = None,
    dXmin = 0.01*cst.pc )
```

3.6 damping Namespace Reference

Functions

- def [IN_damping_approx_2](#) (E, medium_props, [theta](#)=0)
- def [IN_damping_approx_1](#) (E, medium_props, nu_n=0, [theta](#)=0)
- def [IonNeutral_Damping](#) (E, medium_props, nu_n=0, [theta](#)=0)
- def [indamping_alfven](#) (position_index, E, medium_props)
- def [indamping_alfven_nopos](#) (E, medium_props)
- def [damping_lazarian](#) (position_index, E, medium_props)
- def [damping_lazarian_nopos](#) (E, medium_props)
- def [non_linear_landau_damping](#) (T, lp, lm, mi, q, B0, Ecr)

3.6.1 Function Documentation

3.6.1.1 damping_lazarian()

```
def damping.damping_lazarian (
    position_index,
    E,
    medium_props )
```

3.6.1.2 damping_lazarian_nopos()

```
def damping.damping_lazarian_nopos (
    E,
    medium_props )
```

3.6.1.3 IN_damping_approx_1()

```
def damping.IN_damping_approx_1 (
    E,
    medium_props,
    nu_n = 0,
    theta = 0 )
```

From Xu et al. (2016), M2 internship p. 20

```
Parameters
-----
E : TYPE
    DESCRIPTION.
medium_props : TYPE
    DESCRIPTION.
nu_n : TYPE, optional
    DESCRIPTION. The default is 0.
theta : TYPE, optional
    DESCRIPTION. The default is 0.

Returns
-----
None.
```

3.6.1.4 IN_damping_approx_2()

```
def damping.IN_damping_approx_2 (
    E,
    medium_props,
    theta = 0 )
```

From Xu et al. (2016), M2 internship p. 20, asymptotic regime

```
Parameters
-----
E : TYPE
    DESCRIPTION.
medium_props : TYPE
    DESCRIPTION.
theta : TYPE, optional
    DESCRIPTION. The default is 0.

Returns
-----
None.
```

3.6.1.5 indamping_alfven()

```
def damping.indamping_alfven (
    position_index,
    E,
    medium_props )
```

3.6.1.6 indamping_alfven_nopos()

```
def damping.indamping_alfven_nopos (
    E,
    medium_props )
```

3.6.1.7 IonNeutral_Damping()

```
def damping.IonNeutral_Damping (
    E,
    medium_props,
    nu_n = 0,
    theta = 0 )
```

3.6.1.8 non_linear_landau_damping()

```
def damping.non_linear_landau_damping (
    T,
    Ip,
    Im,
    mi,
    q,
    B0,
    Ecr )
```

3.7 damping_models Namespace Reference

Functions

- def [damprate_to_damptime](#) (gamma)

Variables

- int [NE](#) = 10
- float [Emin](#) = 0.99*cst.GeV
- float [Emax](#) = 100.01*cst.TeV
- string [egridtype](#) = "logspace"
- [E](#) = grid.grid([Emin](#), [Emax](#), 2**[NE](#), [egridtype](#))
- list [phases](#) = [ism.HII, ism.WIM, ism.WNM, ism.CNM, ism.DiM, ism.DeM, ism.DeC]
- list [xlim](#) = [[Emin/cst.GeV](#), [Emax/cst.GeV](#)]
- list [xlims](#) = [[xlim](#), [xlim](#), [xlim](#), [xlim](#), [xlim](#), [xlim](#), [xlim](#)]
- list [ylim](#) = [1e-14, 1e-3]
- list [ylims](#) = [[ylim](#), [ylim](#), [ylim](#), [ylim](#), [ylim](#), [ylim](#), [ylim](#)]
- list [Name](#) = ["HII", "WIM", "WNM", "CNM", "DiM", "DeM", "DeC"]
- int [size_x](#) = 4

- `int size_y = 3`
- `int sub_x = 2`
- `int sub_y = 4`
- `fig = plt.figure(figsize=(size_x*sub_x,size_y*sub_y))`
- `gs = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig)`
- `wspace`
- `hspace`
- `list pos_1 = [0, 0, 1, 1, 2, 2, 3, 3]`
- `list pos_2 = [0, 1, 0, 1, 0, 1, 0, 1]`
- `wR_Alfven = np.zeros(len(E))`
- `wI_Alfven = np.zeros(len(E))`
- `wR_Alfven_o1 = np.zeros(len(E))`
- `wI_Alfven_o1 = np.zeros(len(E))`
- `wR_Alfven_o2 = np.zeros(len(E))`
- `wI_Alfven_o2 = np.zeros(len(E))`
- `Ep = np.NaN`
- `Em = np.NaN`
- `Gamma_lz = np.zeros(len(E))`
- `Gamma_nlld_inf = np.zeros(len(E))`
- `Gamma_nlld_sup = np.zeros(len(E))`
- `in_damping = dp.IonNeutral_Damping(E[e], phases[pi], nu_n = 0, theta = 0)`
- `lz_damping = dp.damping_lazarian_nopos(E[e], phases[pi])`
- `lz_min = lz_damping[1]`
- `int linf = 1e-4`
- `int lsup = 1e-1`
- `ax = fig.add_subplot(gs[pos_1[pi], pos_2[pi]])`
- `GeV`
- `c`
- `ls`
- `lw`
- `alpha`
- `color`
- `facecolor`
- `x`
- `y`
- `label`
- `loc`
- `bbox_to_anchor`
- `bbox_inches`
- `pad_inches`

3.7.1 Function Documentation

3.7.1.1 `damprate_to_damptime()`

```
def damping_models.damprate_to_damptime (
    gamma )
```

3.7.2 Variable Documentation

3.7.2.1 alpha

damping_models.alpha

3.7.2.2 ax

damping_models.ax = fig.add_subplot([gs\[pos_1\[pi\]](#), [pos_2\[pi\]](#)])

3.7.2.3 bbox_inches

damping_models.bbox_inches

3.7.2.4 bbox_to_anchor

damping_models.bbox_to_anchor

3.7.2.5 c

damping_models.c

3.7.2.6 color

damping_models.color

3.7.2.7 E

damping_models.E = grid.grid([Emin](#), [Emax](#), 2**[NE](#), [egridtype](#))

3.7.2.8 egridtype

```
string damping_models.egridtype = "logspace"
```

3.7.2.9 Em

```
damping_models.Em = np.NaN
```

3.7.2.10 Emax

```
float damping_models.Emax = 100.01*cst.TeV
```

3.7.2.11 Emin

```
float damping_models.Emin = 0.99*cst.GeV
```

3.7.2.12 Ep

```
damping_models.Ep = np.NaN
```

3.7.2.13 facecolor

```
damping_models.facecolor
```

3.7.2.14 fig

```
damping_models.fig = plt.figure(figsize=(size_x*sub_x, size_y*sub_y))
```

3.7.2.15 Gamma_lz

```
damping_models.Gamma_lz = np.zeros(len(E))
```

3.7.2.16 Gamma_nlld_inf

```
damping_models.Gamma_nlld_inf = np.zeros(len(E))
```

3.7.2.17 Gamma_nlld_sup

```
damping_models.Gamma_nlld_sup = np.zeros(len(E))
```

3.7.2.18 GeV

```
damping_models.GeV
```

3.7.2.19 gs

```
damping_models.gs = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig )
```

3.7.2.20 hspace

```
damping_models.hspace
```

3.7.2.21 linf

```
int damping_models.linf = 1e-4
```

3.7.2.22 in_damping

```
damping_models.in_damping = dp.IonNeutral_Damping(E[e], phases[pi], nu_n = 0, theta = 0)
```

3.7.2.23 Isup

```
int damping_models.Isup = 1e-1
```

3.7.2.24 label

```
damping_models.label
```

3.7.2.25 loc

```
damping_models.loc
```

3.7.2.26 ls

```
damping_models.ls
```

3.7.2.27 lw

```
damping_models.lw
```

3.7.2.28 lz_damping

```
damping_models.lz_damping = dp.damping_lazarian_nopos(E[e], phases[pi])
```

3.7.2.29 lz_min

```
damping_models.lz_min = lz_damping[1]
```

3.7.2.30 Name

```
list damping_models.Name = ["HII", "WIM", "WNM", "CNM", "DiM", "DeM", "DeC"]
```

3.7.2.31 NE

```
int damping_models.NE = 10
```


3.7.2.32 pad_inches

```
damping_models.pad_inches
```

3.7.2.33 phases

```
list damping_models.phases = [ism.HII, ism.WIM, ism.WNM, ism.CNM, ism.DiM, ism.DeM, ism.DeC]
```

3.7.2.34 pos_1

```
list damping_models.pos_1 = [0, 0, 1, 1, 2, 2, 3, 3]
```

3.7.2.35 pos_2

```
list damping_models.pos_2 = [0, 1, 0, 1, 0, 1, 0, 1]
```

3.7.2.36 size_x

```
int damping_models.size_x = 4
```

3.7.2.37 size_y

```
int damping_models.size_y = 3
```

3.7.2.38 sub_x

```
int damping_models.sub_x = 2
```

3.7.2.39 sub_y

```
int damping_models.sub_y = 4
```

3.7.2.40 `wI_Alfven`

```
damping_models.wI_Alfven = np.zeros(len(E))
```

3.7.2.41 `wI_Alfven_o1`

```
damping_models.wI_Alfven_o1 = np.zeros(len(E))
```

3.7.2.42 `wI_Alfven_o2`

```
damping_models.wI_Alfven_o2 = np.zeros(len(E))
```

3.7.2.43 `wR_Alfven`

```
damping_models.wR_Alfven = np.zeros(len(E))
```

3.7.2.44 `wR_Alfven_o1`

```
damping_models.wR_Alfven_o1 = np.zeros(len(E))
```

3.7.2.45 `wR_Alfven_o2`

```
damping_models.wR_Alfven_o2 = np.zeros(len(E))
```

3.7.2.46 `wspace`

```
damping_models.wspace
```

3.7.2.47 `x`

```
damping_models.x
```

3.7.2.48 xlim

```
list damping_models.xlim = [Emin/cst.GeV, Emax/cst.GeV]
```

3.7.2.49 xlims

```
list damping_models.xlims = [xlim, xlim, xlim, xlim, xlim, xlim, xlim]
```

3.7.2.50 y

```
damping_models.y
```

3.7.2.51 ylim

```
list damping_models.ylim = [1e-14, 1e-3]
```

3.7.2.52 ylims

```
list damping_models.ylims = [ylim, ylim, ylim, ylim, ylim, ylim, ylim]
```

3.8 Data_reader Namespace Reference

Functions

- def [readDataXE](#) (file_name, NX, NE)
- def [readAxis](#) (file_name)

Variables

- def [data](#) = [readDataXE](#)("../data_out/Pcr_0165.dat", 2**11, 2**5)
- def [X](#) = [readAxis](#)("../data_ini/X.dat")
- def [E](#) = [readAxis](#)("../data_ini/E.dat")
- [figsize](#)

3.8.1 Function Documentation

3.8.1.1 readAxis()

```
def Data_reader.readAxis (  
    file_name )
```

3.8.1.2 readDataXE()

```
def Data_reader.readDataXE (  
    file_name,  
    NX,  
    NE )
```

3.8.2 Variable Documentation

3.8.2.1 data

```
def Data_reader.data = readDataXE("../data_out/Pcr_0165.dat", 2**11, 2**5)
```

3.8.2.2 E

```
def Data_reader.E = readAxis("../data_ini/E.dat")
```

3.8.2.3 figsize

```
Data_reader.figsize
```

3.8.2.4 X

```
def Data_reader.X = readAxis("../data_ini/X.dat")
```

3.9 electrons_emax_t Namespace Reference

Functions

- def `InverseTrigonalMatrix` (T)
FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- def `ProductMatrix` (A, B)
- def `InterpolatingSpline` (X, Y)
- def `Rsh` (nt)
- def `B` (t, ts, tB, `alpha_B`, BISM, Bfree)
ELECTRON ESCAPE MODEL (from Ohira et al.
- def `eta_g` (t, ts, tB, `alpha`, `alpha_B`, eta_free)
- def `Em_age` (t, ts, tB, `alpha_B`, `alpha`, BISM, Bfree, `eta_acc`, eta_free, Rs)
- def `Em_cool` (t, ts, tB, `alpha_B`, `alpha`, BISM, Bfree, Ems)
- def `Em_esc` (t, ts, tB, `alpha_B`, `alpha`, BISM, Bfree, `eta_acc`, eta_free, eta_esc, Rs)
- def `Emax_electrons` (time)
- def `escape_time` (E, tSed, EM, delta)

Variables

- float `alpha` = 2.6
- int `alpha_B` = 9./10
- int `xhi_m` = 1
- float `xhi_cr` = 0.1
- int `E51` = 1
- int `Mej` = 1
- int `C06` = 1
- int `beta` = 1
- int `phi_c` = 1
- int `vej8` = 10.*(E51/Mej)**(0.5)
- float `nt` = 0.35
- float `tsed` = 0.3*E51**(-0.5)*Mej*nt**(-1./3)*cst.kyr
FUNCTIONS IN ORDER TO MAKE OUR SNR EXPAND IN THE ISM #.
- def `EM` = `Emax_electrons`(tsed)
- `E` = np.logspace(np.log10(1*cst.GeV), np.log10(100*cst.TeV), num=100)
- `tesc` = np.zeros(len(E))

3.9.1 Function Documentation

3.9.1.1 B()

```
def electrons_emax_t.B (
    t,
    ts,
    tB,
    alpha_B,
    BISM,
    Bfree )
```

ELECTRON ESCAPE MODEL (from Ohira et al.

(2012)) #

3.9.1.2 Em_age()

```
def electrons_emax_t.Em_age (  
    t,  
    ts,  
    tB,  
    alpha_B,  
    alpha,  
    BISM,  
    Bfree,  
    eta_acc,  
    eta_free,  
    Rs )
```

3.9.1.3 Em_cool()

```
def electrons_emax_t.Em_cool (  
    t,  
    ts,  
    tB,  
    alpha_B,  
    alpha,  
    BISM,  
    Bfree,  
    Ems )
```

3.9.1.4 Em_esc()

```
def electrons_emax_t.Em_esc (  
    t,  
    ts,  
    tB,  
    alpha_B,  
    alpha,  
    BISM,  
    Bfree,  
    eta_acc,  
    eta_free,  
    eta_esc,  
    Rs )
```

3.9.1.5 Emax_electrons()

```
def electrons_emax_t.Emax_electrons (  
    time )
```

3.9.1.6 escape_time()

```
def electrons_emax_t.escape_time (
    E,
    tSed,
    EM,
    delta )
```

3.9.1.7 eta_g()

```
def electrons_emax_t.eta_g (
    t,
    ts,
    tB,
    alpha,
    alpha_B,
    eta_free )
```

3.9.1.8 InterpolatingSpline()

```
def electrons_emax_t.InterpolatingSpline (
    X,
    Y )
```

3.9.1.9 InverseTrigonalMatrix()

```
def electrons_emax_t.InverseTrigonalMatrix (
    T )
```

FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.

TriDiagonal matrix inversion function

3.9.1.10 ProductMatrix()

```
def electrons_emax_t.ProductMatrix (
    A,
    B )
```

3.9.1.11 Rsh()

```
def electrons_emax_t.Rsh (
    nt )
```

3.9.2 Variable Documentation

3.9.2.1 alpha

```
float electrons_emax_t.alpha = 2.6
```

3.9.2.2 alpha_B

```
int electrons_emax_t.alpha_B = 9./10
```

3.9.2.3 beta

```
int electrons_emax_t.beta = 1
```

3.9.2.4 C06

```
int electrons_emax_t.C06 = 1
```

3.9.2.5 E

```
electrons_emax_t.E = np.logspace(np.log10(1*cst.GeV), np.log10(100*cst.TeV), num=100)
```

3.9.2.6 E51

```
int electrons_emax_t.E51 = 1
```

3.9.2.7 EM

```
def electrons_emax_t.EM = Emax_electrons(tsed)
```


3.9.2.8 Mej

```
int electrons_emax_t.Mej = 1
```

3.9.2.9 nt

```
float electrons_emax_t.nt = 0.35
```

3.9.2.10 phi_c

```
int electrons_emax_t.phi_c = 1
```

3.9.2.11 tesc

```
electrons_emax_t.tesc = np.zeros(len(E))
```

3.9.2.12 tsed

```
float electrons_emax_t.tsed = 0.3*E51**(-0.5)*Mej*nt**(-1./3)*cst.kyr
```

FUNCTIONS IN ORDER TO MAKE OUR SNR EXPAND IN THE ISM #.

We define the characteristic times of our problem

3.9.2.13 vej8

```
int electrons_emax_t.vej8 = 10.*(E51/Mej)**(0.5)
```

3.9.2.14 xhi_cr

```
float electrons_emax_t.xhi_cr = 0.1
```

3.9.2.15 xhi_m

```
int electrons_emax_t.xhi_m = 1
```

3.10 EscapeModel_protons Namespace Reference

Functions

- def [InverseTrigonalMatrix](#) (T)
FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- def [ProductMatrix](#) (A, B)
- def [InterpolatingSpline](#) (X, Y)
- def [f1](#) (x, const)
- def [df1dx](#) (x, const)
- def [f2](#) (x, const)
- def [df2dx](#) (x, const)
- def [NewtonRaphson](#) (f, df, x0, eps, const)
- def [Gettesc](#) (E, delta)

Variables

- float [nt](#) = 0.35
- int [xhi_m](#) = 1
- float [xhi_cr](#) = 0.1
- int [E51](#) = 1
- int [Mej](#) = 1
- int [C06](#) = 1
- int [beta](#) = 1
- int [phi_c](#) = 1
- int [vej8](#) = 10.*(E51/Mej)**(0.5)
- int [tini](#) = 1e-4*cst.kyr
FUNCTIONS IN ORDER TO MAKE OUR SNR EXPAND IN THE ISM #.
- float [tfree](#) = 0.3*E51**(-0.5)*Mej*nt**(-1./3)*cst.kyr
- float [tPDS](#) = np.exp(-1.)*3.61e4*E51**(3./14)/(xhi_m**(5./14)*nt**(4./7))*cst.yr
- float [tMCS](#) = min(61*vej8**3/(xhi_m**(9./14)*nt**(3./7)*E51**(3./14)), 476./(xhi_m*phi_c)**(9./14))*tPDS
- int [tmerge](#) = 153.*(E51**(1./14)*nt**(1./7)*xhi_m**(3./14)/(beta*C06))**(10./7)*tPDS
- [tmax](#) = min(tMCS, tmerge)
- float [R_free](#) = 5.0*(E51/nt)**(1./5)*(1 - (0.05*Mej**(5./6))/(E51**0.5*nt**(1./3)*(tfree/cst.kyr)))*(2./5)*(tfree/cst.kyr)**(2./5)*cst.pc
- float [R_ini](#) = [R_free](#)*(tini/tfree)**(1.)
- float [R_PDS](#) = 5.0*(E51/nt)**(1./5)*(1 - (0.05*Mej**(5./6))/(E51**0.5*nt**(1./3)*(tPDS/cst.kyr)))*(2./5)*(tPDS/cst.kyr)**(2./5)*cst.pc
- float [R_MCS](#) = [R_PDS](#)*(tMCS/tPDS)**(3./10)
- float [R_merge](#) = [R_MCS](#)*(tmerge/tMCS)**(1./4)
- [t](#) = np.array([tini, tfree, tPDS, tMCS, tmerge])
- [R](#) = np.array([R_ini, R_free, R_PDS, R_MCS, R_merge])
- [logt](#) = np.empty(len(t))
- [logR](#) = np.empty(len(R))
- def [f_SNR](#) = [InterpolatingSpline](#)(logt, logR)
- [logt_new](#) = np.linspace(logt[0], logt[-1], 100)
- [logr_new](#) = np.empty(len(logt_new))

- `t_new` = `np.empty(len(logt_new))`
- `r_new` = `np.empty(len(logr_new))`
- `u_sh` = `np.empty(len(r_new))`
- float `gamma` = 2.2
- float `Emin` = `0.1*cst.GeV`
- `Emax` = `np.empty(len(t_new))`
- int `eps` = `1e-4`
- int `x0` = `10.*cst.GeV`
- float `a` = `Emin`
- int `b` = `beta`
- tuple `c` = `(beta/(1+beta))*cst.e*np.sqrt(4*np.pi*nt*cst.mp)/(10.*cst.c)*xhi_cr*u_sh[ii]**2*r_new[iii]`
- `niter`
- `EMAX` = `max(Emax)`
- int `delta` = 2.
- float `tSed` = `tfree`
- `Ecr` = `np.logspace(np.log10(0.1*cst.GeV), np.log10(100.*cst.TeV), 100)`
- `tesc` = `np.empty(len(Ecr))`
- `figsize`

Model figure #.

- `pc`
- `marker`
- `lw`
- `label`
- `GeV`
- `kyr`

3.10.1 Function Documentation

3.10.1.1 `df1dx()`

```
def EscapeModel_protons.df1dx (
    x,
    const )
```

3.10.1.2 `df2dx()`

```
def EscapeModel_protons.df2dx (
    x,
    const )
```

3.10.1.3 f1()

```
def EscapeModel_protons.f1 (
    x,
    const )
```

3.10.1.4 f2()

```
def EscapeModel_protons.f2 (
    x,
    const )
```

3.10.1.5 Gettesc()

```
def EscapeModel_protons.Gettesc (
    E,
    delta )
```

3.10.1.6 InterpolatingSpline()

```
def EscapeModel_protons.InterpolatingSpline (
    X,
    Y )
```

3.10.1.7 InverseTrigonalMatrix()

```
def EscapeModel_protons.InverseTrigonalMatrix (
    T )
```

FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.

TriDiagonal matrix inversion function

3.10.1.8 NewtonRaphson()

```
def EscapeModel_protons.NewtonRaphson (
    f,
    df,
    x0,
    eps,
    const )
```

3.10.1.9 ProductMatrix()

```
def EscapeModel_protons.ProductMatrix (
    A,
    B )
```

3.10.2 Variable Documentation

3.10.2.1 a

```
float EscapeModel_protons.a = Emin
```

3.10.2.2 b

```
float EscapeModel_protons.b = beta
```

3.10.2.3 beta

```
float EscapeModel_protons.beta = 1
```

3.10.2.4 c

```
EscapeModel_protons.c = (beta/(1+beta))*cst.e*np.sqrt(4*np.pi*nt*cst.mp)/(10.*cst.c)*xhi_cr*u↵_sh[ii]**2*r_new[ii]
```

3.10.2.5 C06

```
int EscapeModel_protons.C06 = 1
```

3.10.2.6 delta

```
int EscapeModel_protons.delta = 2.
```

3.10.2.7 E51

```
int EscapeModel_protons.E51 = 1
```

3.10.2.8 Ecr

```
EscapeModel_protons.Ecr = np.logspace(np.log10(0.1*cst.GeV), np.log10(100.*cst.TeV), 100)
```

3.10.2.9 Emax

```
EscapeModel_protons.Emax = np.empty(len(t_new))
```

3.10.2.10 EMAX

```
EscapeModel_protons.EMAX = max(Emax)
```

3.10.2.11 Emin

```
float EscapeModel_protons.Emin = 0.1*cst.GeV
```

3.10.2.12 eps

```
int EscapeModel_protons.eps = 1e-4
```

3.10.2.13 f_SNR

```
def EscapeModel_protons.f_SNR = InterpolatingSpline(logt, logR)
```

3.10.2.14 figsize

EscapeModel_protons.figsize

Model figure #.

3.10.2.15 gamma

float EscapeModel_protons.gamma = 2.2

3.10.2.16 GeV

EscapeModel_protons.GeV

3.10.2.17 kyr

EscapeModel_protons.kyr

3.10.2.18 label

EscapeModel_protons.label

3.10.2.19 logR

EscapeModel_protons.logR = np.empty(len(R))

3.10.2.20 logr_new

EscapeModel_protons.logr_new = np.empty(len(logt_new))

3.10.2.21 logt

```
EscapeModel_protons.logt = np.empty(len(t))
```

3.10.2.22 logt_new

```
EscapeModel_protons.logt_new = np.linspace(logt[0], logt[-1], 100)
```

3.10.2.23 lw

```
EscapeModel_protons.lw
```

3.10.2.24 marker

```
EscapeModel_protons.marker
```

3.10.2.25 Mej

```
int EscapeModel_protons.Mej = 1
```

3.10.2.26 niter

```
EscapeModel_protons.niter
```

3.10.2.27 nt

```
float EscapeModel_protons.nt = 0.35
```

3.10.2.28 pc

```
EscapeModel_protons.pc
```


3.10.2.29 phi_c

```
int EscapeModel_protons.phi_c = 1
```

3.10.2.30 R

```
EscapeModel_protons.R = np.array([R_ini, R_free, R_PDS, R_MCS, R_merge])
```

3.10.2.31 R_free

```
float EscapeModel_protons.R_free = 5.0*(E51/nt)**(1./5)*(1 - (0.05*Mej**(5./6)))/(E51**0.4↵  
5*nt**(1./3)*(tfree/cst.kyr))**(2./5)*(tfree/cst.kyr)**(2./5)*cst.pc
```

3.10.2.32 R_ini

```
float EscapeModel_protons.R_ini = R_free*(tini/tfree)**(1.)
```

3.10.2.33 R_MCS

```
float EscapeModel_protons.R_MCS = R_PDS*(tMCS/tPDS)**(3./10)
```

3.10.2.34 R_merge

```
float EscapeModel_protons.R_merge = R_MCS*(tmerge/tMCS)**(1./4)
```

3.10.2.35 r_new

```
EscapeModel_protons.r_new = np.empty(len(logr_new))
```

3.10.2.36 R_PDS

```
float EscapeModel_protons.R_PDS = 5.0*(E51/nt)**(1./5)*(1 - (0.05*Mej**(5./6))/(E51**0.5*nt**(1./3)*(tPDS/cst.kyr)))**(2./5)*(tPDS/cst.kyr)**(2./5)*cst.pc
```

3.10.2.37 t

```
EscapeModel_protons.t = np.array([tini, tfree, tPDS, tMCS, tmerge])
```

3.10.2.38 t_new

```
EscapeModel_protons.t_new = np.empty(len(logt_new))
```

3.10.2.39 tesc

```
EscapeModel_protons.tesc = np.empty(len(Ecr))
```

3.10.2.40 tfree

```
float EscapeModel_protons.tfree = 0.3*E51**(-0.5)*Mej*nt**(-1./3)*cst.kyr
```

3.10.2.41 tini

```
int EscapeModel_protons.tini = 1e-4*cst.kyr
```

FUNCTIONS IN ORDER TO MAKE OUR SNR EXPAND IN THE ISM #.

We define the characteristic times of our problem

3.10.2.42 tmax

```
EscapeModel_protons.tmax = min(tMCS, tmerge)
```

3.10.2.43 tMCS

```
float EscapeModel_protons.tMCS = min(61*vej8**3/(xhi_m**(9./14)*nt**(3./7)*E51**(3./14)), 476./(xhi_↵
_m*phi_c)**(9./14))*tPDS
```

3.10.2.44 tmerge

```
int EscapeModel_protons.tmerge = 153.*(E51**(1./14)*nt**(1./7)*xhi_m**(3./14)/(beta*C06))**(10./7)*t↵
PDS
```

3.10.2.45 tPDS

```
float EscapeModel_protons.tPDS = np.exp(-1.)*3.61e4*E51**(3./14)/(xhi_m**(5./14)*nt**(4./7))*cst.↵
yr
```

3.10.2.46 tSed

```
float EscapeModel_protons.tSed = tfree
```

3.10.2.47 u_sh

```
EscapeModel_protons.u_sh = np.empty(len(r_new))
```

3.10.2.48 vej8

```
int EscapeModel_protons.vej8 = 10.*(E51/Mej)**(0.5)
```

3.10.2.49 x0

```
int EscapeModel_protons.x0 = 10.*cst.GeV
```

3.10.2.50 xhi_cr

```
float EscapeModel_protons.xhi_cr = 0.1
```

3.10.2.51 xhi_m

```
int EscapeModel_protons.xhi_m = 1
```

3.11 EscapeModel_protons_2 Namespace Reference

Functions

- def [InverseTrigonalMatrix](#) (T)
 FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- def [ProductMatrix](#) (A, B)
- def [InterpolatingSpline](#) (X, Y)
- def [f1](#) (x, const)
- def [df1dx](#) (x, const)
- def [f2](#) (x, const)
- def [df2dx](#) (x, const)
- def [NewtonRaphson](#) (f, df, x0, eps, const)
- def [getSNR](#) (phase, size=100)
- def [Gettesc](#) (E, [delta](#), tSed, EMAX, [Emin](#)=0.1 *[cst.GeV](#))
- def [getEmax](#) (t_new, [u_sh](#), r_new, phase, gamma=2.2, [Emin](#)=0.1 *[cst.GeV](#), eps=1e-4, x0=10.*[cst.GeV](#))

Variables

- [Ecr](#) = np.logspace(np.log10(0.1*[cst.GeV](#)), np.log10(100.*[cst.TeV](#)), 1000)
- def [SNR_HII](#) = [getSNR](#)(ism.HII, size = 100)
- def [emax_HII](#)
- def [SNR_WIM](#) = [getSNR](#)(ism.WIM, size = 100)
- def [emax_WIM](#)
- def [SNR_WNM](#) = [getSNR](#)(ism.WNM, size = 100)
- def [emax_WNM](#)
- def [SNR_CNM](#) = [getSNR](#)(ism.CNM, size = 100)
- def [emax_CNM](#)
- def [SNR_DiM](#) = [getSNR](#)(ism.DiM, size = 100)
- def [emax_DiM](#)
- int [delta](#) = 2.
- [tesc_HII](#) = np.empty(len([Ecr](#)))
- [tesc_WIM](#) = np.empty(len([Ecr](#)))
- [tesc_WNM](#) = np.empty(len([Ecr](#)))
- [tesc_CNM](#) = np.empty(len([Ecr](#)))
- [tesc_DiM](#) = np.empty(len([Ecr](#)))
- [tesc_HII_3](#) = np.empty(len([Ecr](#)))
- [tesc_WIM_3](#) = np.empty(len([Ecr](#)))
- [tesc_WNM_3](#) = np.empty(len([Ecr](#)))
- [tesc_CNM_3](#) = np.empty(len([Ecr](#)))

- `tesc_DiM_3` = `np.empty(len(Ecr))`
- `int size_x` = 4
- `float size_y` = 3.5
- `int sub_x` = 2
- `int sub_y` = 1
- `fig` = `plt.figure(figsize=(size_x*sub_x,size_y*sub_y))`
- `gs` = `gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig)`
- `wspace`
- `hspace`
- `ax0` = `fig.add_subplot(gs[0])`
- `GeV`
- `c`
- `label`
- `loc`
- `ncol`
- `bbox_to_anchor`
- `ax1` = `fig.add_subplot(gs[1])`
- `kyr`
- `ls`
- `pad`

3.11.1 Function Documentation

3.11.1.1 `df1dx()`

```
def EscapeModel_protons_2.df1dx (
    x,
    const )
```

3.11.1.2 `df2dx()`

```
def EscapeModel_protons_2.df2dx (
    x,
    const )
```

3.11.1.3 `f1()`

```
def EscapeModel_protons_2.f1 (
    x,
    const )
```

3.11.1.4 f2()

```
def EscapeModel_protons_2.f2 (
    x,
    const )
```

3.11.1.5 getEmax()

```
def EscapeModel_protons_2.getEmax (
    t_new,
    u_sh,
    r_new,
    phase,
    gamma = 2.2,
    Emin = 0.1*cst.GeV,
    eps = 1e-4,
    x0 = 10.*cst.GeV )
```

3.11.1.6 getSNR()

```
def EscapeModel_protons_2.getSNR (
    phase,
    size = 100 )
```

3.11.1.7 Gettesc()

```
def EscapeModel_protons_2.Gettesc (
    E,
    delta,
    tSed,
    EMAX,
    Emin = 0.1*cst.GeV )
```

3.11.1.8 InterpolatingSpline()

```
def EscapeModel_protons_2.InterpolatingSpline (
    X,
    Y )
```

3.11.1.9 InverseTrigonalMatrix()

```
def EscapeModel_protons_2.InverseTrigonalMatrix (
    T )
```

FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.

TriDiagonal matrix inversion function

3.11.1.10 NewtonRaphson()

```
def EscapeModel_protons_2.NewtonRaphson (
    f,
    df,
    x0,
    eps,
    const )
```

3.11.1.11 ProductMatrix()

```
def EscapeModel_protons_2.ProductMatrix (
    A,
    B )
```

3.11.2 Variable Documentation

3.11.2.1 ax0

```
EscapeModel_protons_2.ax0 = fig.add_subplot(gs[0])
```

3.11.2.2 ax1

```
EscapeModel_protons_2.ax1 = fig.add_subplot(gs[1])
```

3.11.2.3 bbox_to_anchor

```
EscapeModel_protons_2.bbox_to_anchor
```

3.11.2.4 c

```
EscapeModel_protons_2.c
```

3.11.2.5 delta

```
int EscapeModel_protons_2.delta = 2.
```

3.11.2.6 Ecr

```
EscapeModel_protons_2.Ecr = np.logspace(np.log10(0.1*cst.GeV), np.log10(100.*cst.TeV), 1000)
```

3.11.2.7 emax_CNM

```
def EscapeModel_protons_2.emax_CNM
```

Initial value:

```
1 = getEmax(SNR_CNM.get("t_SNR"), SNR_CNM.get("u_sh"), SNR_CNM.get("R_SNR"), ism.CNM,
2           gamma = 2.2, Emin = 0.1*cst.GeV, eps = 1e-4, x0 = 10.*cst.GeV)
```

3.11.2.8 emax_DiM

```
def EscapeModel_protons_2.emax_DiM
```

Initial value:

```
1 = getEmax(SNR_DiM.get("t_SNR"), SNR_DiM.get("u_sh"), SNR_DiM.get("R_SNR"), ism.DiM,
2           gamma = 2.2, Emin = 0.1*cst.GeV, eps = 1e-4, x0 = 10.*cst.GeV)
```

3.11.2.9 emax_HII

```
def EscapeModel_protons_2.emax_HII
```

Initial value:

```
1 = getEmax(SNR_HII.get("t_SNR"), SNR_HII.get("u_sh"), SNR_HII.get("R_SNR"), ism.HII,
2           gamma = 2.2, Emin = 0.1*cst.GeV, eps = 1e-4, x0 = 10.*cst.GeV)
```


3.11.2.10 `emax_WIM`

```
def EscapeModel_protons_2.emax_WIM
```

Initial value:

```
1 = getEmax(SNR_WIM.get("t_SNR"), SNR_WIM.get("u_sh"), SNR_WIM.get("R_SNR"), ism.WIM,  
2          gamma = 2.2, Emin = 0.1*cst.GeV, eps = 1e-4, x0 = 10.*cst.GeV)
```

3.11.2.11 `emax_WNM`

```
def EscapeModel_protons_2.emax_WNM
```

Initial value:

```
1 = getEmax(SNR_WNM.get("t_SNR"), SNR_WNM.get("u_sh"), SNR_WNM.get("R_SNR"), ism.WNM,  
2          gamma = 2.2, Emin = 0.1*cst.GeV, eps = 1e-4, x0 = 10.*cst.GeV)
```

3.11.2.12 `fig`

```
EscapeModel_protons_2.fig = plt.figure(figsize=(size_x*sub_x,size_y*sub_y))
```

3.11.2.13 `GeV`

```
EscapeModel_protons_2.GeV
```

3.11.2.14 `gs`

```
EscapeModel_protons_2.gs = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig )
```

3.11.2.15 `hspace`

```
EscapeModel_protons_2.hspace
```

3.11.2.16 kyr

```
EscapeModel_protons_2.kyr
```

3.11.2.17 label

```
EscapeModel_protons_2.label
```

3.11.2.18 loc

```
EscapeModel_protons_2.loc
```

3.11.2.19 ls

```
EscapeModel_protons_2.ls
```

3.11.2.20 ncol

```
EscapeModel_protons_2.ncol
```

3.11.2.21 pad

```
EscapeModel_protons_2.pad
```

3.11.2.22 size_x

```
int EscapeModel_protons_2.size_x = 4
```

3.11.2.23 size_y

```
float EscapeModel_protons_2.size_y = 3.5
```

3.11.2.24 SNR_CNM

```
def EscapeModel_protons_2.SNR_CNM = getSNR(ism.CNM, size = 100)
```

3.11.2.25 SNR_DiM

```
def EscapeModel_protons_2.SNR_DiM = getSNR(ism.DiM, size = 100)
```

3.11.2.26 SNR_HII

```
def EscapeModel_protons_2.SNR_HII = getSNR(ism.HII, size = 100)
```

3.11.2.27 SNR_WIM

```
def EscapeModel_protons_2.SNR_WIM = getSNR(ism.WIM, size = 100)
```

3.11.2.28 SNR_WNM

```
def EscapeModel_protons_2.SNR_WNM = getSNR(ism.WNM, size = 100)
```

3.11.2.29 sub_x

```
int EscapeModel_protons_2.sub_x = 2
```

3.11.2.30 sub_y

```
int EscapeModel_protons_2.sub_y = 1
```

3.11.2.31 tesc_CNM

```
EscapeModel_protons_2.tesc_CNM = np.empty(len(Ecr))
```

3.11.2.32 tesc_CNM_3

```
EscapeModel_protons_2.tesc_CNM_3 = np.empty(len(Ecr))
```

3.11.2.33 tesc_DiM

```
EscapeModel_protons_2.tesc_DiM = np.empty(len(Ecr))
```

3.11.2.34 tesc_DiM_3

```
EscapeModel_protons_2.tesc_DiM_3 = np.empty(len(Ecr))
```

3.11.2.35 tesc_HII

```
EscapeModel_protons_2.tesc_HII = np.empty(len(Ecr))
```

3.11.2.36 tesc_HII_3

```
EscapeModel_protons_2.tesc_HII_3 = np.empty(len(Ecr))
```

3.11.2.37 tesc_WIM

```
EscapeModel_protons_2.tesc_WIM = np.empty(len(Ecr))
```

3.11.2.38 tesc_WIM_3

```
EscapeModel_protons_2.tesc_WIM_3 = np.empty(len(Ecr))
```

3.11.2.39 tesc_WNM

```
EscapeModel_protons_2.tesc_WNM = np.empty(len(Ecr))
```

3.11.2.40 tesc_WNM_3

```
EscapeModel_protons_2.tesc_WNM_3 = np.empty(len(Ecr))
```

3.11.2.41 wspace

```
EscapeModel_protons_2.wspace
```

3.12 freader Namespace Reference

Functions

- def [search](#) (file_name, variable)

3.12.1 Function Documentation

3.12.1.1 [search\(\)](#)

```
def freader.search (
    file_name,
    variable )
```

3.13 fwriter Namespace Reference

Functions

- def [search](#) (file_name, variable)
- def [fileWrite](#) (file_name, variables={}, path='./', ext='.dat')
- def [write1D](#) (file_name, nx=None, ne=None, variable=None, path=".")
- def [write2D](#) (file_name, nx=None, ny=None, ne=None, variable=None, path=".")
- def [write1Daxis](#) (file_name, variable=None, nx=None, path=".")

3.13.1 Function Documentation

3.13.1.1 fileWrite()

```
def fwritter.fileWrite (  
    file_name,  
    variables = {},  
    path = './',  
    ext = '.dat' )
```

3.13.1.2 search()

```
def fwritter.search (  
    file_name,  
    variable )
```

3.13.1.3 write1D()

```
def fwritter.write1D (  
    file_name,  
    nx = None,  
    ne = None,  
    variable = None,  
    path = "./" )
```

3.13.1.4 write1Daxis()

```
def fwritter.write1Daxis (  
    file_name,  
    variable = None,  
    nx = None,  
    path = "./" )
```

3.13.1.5 write2D()

```
def fwritter.write2D (  
    file_name,  
    nx = None,  
    ny = None,  
    ne = None,  
    variable = None,  
    path = "./" )
```

3.14 gaussian_subNormalization Namespace Reference

Functions

- def `gauss` (`t`, `sig`, `mu`)

Variables

- float `tmin` = 0.01
- float `tesc` = 25.23
- int `tmax` = 2*`tesc` - `tmin`
- int `sig` = 2
- int `mu` = 25
- int `Nc` = 1e6
- `tc` = `np.linspace(tmin, tmax, Nc)`
- int `Nv` = 100
- `ta` = `np.linspace(tmin, tmax, Nv)`
- int `r` = `Nc/Nv`
- `gauss_c` = `np.zeros(len(tc))`
- `gauss_a` = `np.zeros(len(ta))`
- int `C_c` = 0.
- int `C_a` = 0.
- `color`
- `c`

3.14.1 Function Documentation

3.14.1.1 `gauss()`

```
def gaussian_subNormalization.gauss (  
    t,  
    sig,  
    mu )
```

3.14.2 Variable Documentation

3.14.2.1 `c`

```
gaussian_subNormalization.c
```

3.14.2.2 C_a

```
int gaussian_subNormalization.C_a = 0.
```

3.14.2.3 C_c

```
int gaussian_subNormalization.C_c = 0.
```

3.14.2.4 color

```
gaussian_subNormalization.color
```

3.14.2.5 gauss_a

```
gaussian_subNormalization.gauss_a = np.zeros(len(ta))
```

3.14.2.6 gauss_c

```
gaussian_subNormalization.gauss_c = np.zeros(len(tc))
```

3.14.2.7 mu

```
int gaussian_subNormalization.mu = 25
```

3.14.2.8 Nc

```
int gaussian_subNormalization.Nc = 1e6
```

3.14.2.9 Nv

```
int gaussian_subNormalization.Nv = 100
```


3.14.2.10 r

```
int gaussian_subNormalization.r = Nc/Nv
```

3.14.2.11 sig

```
int gaussian_subNormalization.sig = 2
```

3.14.2.12 ta

```
gaussian_subNormalization.ta = np.linspace(tmin, tmax, Nv)
```

3.14.2.13 tc

```
gaussian_subNormalization.tc = np.linspace(tmin, tmax, Nc)
```

3.14.2.14 tesc

```
gaussian_subNormalization.tesc = 25.23
```

3.14.2.15 tmax

```
int gaussian_subNormalization.tmax = 2*tesc - tmin
```

3.14.2.16 tmin

```
float gaussian_subNormalization.tmin = 0.01
```

3.15 mathmethods Namespace Reference

Functions

- def [Cubic3](#) (a, b, [c](#), d)
- def [findF](#) (a, b, [c](#))
- def [findG](#) (a, b, [c](#), d)
- def [findH](#) (g, f)
- def [cardano3](#) (a, b, [c](#))
- def [histogram](#) (data, xi, xf, nbins, scale, normalization)
- def [g](#) (f, t)
- def [simpson_log](#) (f, a, b, N)
- def [glin](#) (f, t)
- def [simpson_lin](#) (f, a, b, N)
- def [g1](#) (x, xt, l)
- def [g2](#) (x, xt, l)
- def [f](#) (x, xt, l, v1, v2)
- def [shape](#) (X, Amp, Xmin=0., Xmax=1., sig_Xmin=0.1, sig_Xmax=0.2)
- def [multishape](#) (X, Amp, Xmin, Xmax, sig)
- def [SmoothPhaseTransition](#) (X, E, phases, smooth_width)

3.15.1 Function Documentation

3.15.1.1 [cardano3\(\)](#)

```
def mathmethods.cardano3 (
    a,
    b,
    c )
```

3.15.1.2 [Cubic3\(\)](#)

```
def mathmethods.Cubic3 (
    a,
    b,
    c,
    d )
```

3.15.1.3 [f\(\)](#)

```
def mathmethods.f (
    x,
    xt,
    l,
    v1,
    v2 )
```

3.15.1.4 findF()

```
def mathmethods.findF (  
    a,  
    b,  
    c )
```

3.15.1.5 findG()

```
def mathmethods.findG (  
    a,  
    b,  
    c,  
    d )
```

3.15.1.6 findH()

```
def mathmethods.findH (  
    g,  
    f )
```

3.15.1.7 g()

```
def mathmethods.g (  
    f,  
    t )
```

3.15.1.8 g1()

```
def mathmethods.g1 (  
    x,  
    xt,  
    l )
```

3.15.1.9 g2()

```
def mathmethods.g2 (  
    x,  
    xt,  
    l )
```

3.15.1.10 `glin()`

```
def mathmethods.glin (
    f,
    t )
```

3.15.1.11 `histogram()`

```
def mathmethods.histogram (
    data,
    xi,
    xf,
    nbin,
    scale,
    normalization )
```

3.15.1.12 `multishape()`

```
def mathmethods.multishape (
    X,
    Amp,
    Xmin,
    Xmax,
    sig )
```

3.15.1.13 `shape()`

```
def mathmethods.shape (
    X,
    Amp,
    Xmin = 0.,
    Xmax = 1.,
    sig_Xmin = 0.1,
    sig_Xmax = 0.2 )
```

3.15.1.14 `simpson_lin()`

```
def mathmethods.simpson_lin (
    f,
    a,
    b,
    N )
```

3.15.1.15 simpson_log()

```
def mathmethods.simpson_log (
    f,
    a,
    b,
    N )
```

3.15.1.16 SmoothPhaseTransition()

```
def mathmethods.SmoothPhaseTransition (
    X,
    E,
    phases,
    smooth_width )
```

3.16 namelist Namespace Reference

Functions

- def [getVA](#) (E, phase)
- def [getDamping](#) (E, phase)

Variables

- string [folder_name](#) = "Test_standard_full"
OUTPUT FOLDER CREATOR #.
- string [folder_path](#) = "../WorkFolder/"
- string [total_path](#) = [folder_path](#)+[folder_name](#)
- int [NX](#) = 10
GRID PARAMETERS #.
- int [NE](#) = 7
- int [Xmin](#) = 0.*[cst.pc](#)
- int [Xmax](#) = 2000.*[cst.pc](#)
- string [xgridtype](#) = "cartesian"
- float [Emin](#) = 0.99*[cst.GeV](#)
- float [Emax](#) = 50.01*[cst.TeV](#)
- string [egridtype](#) = "logspace"
- int [box_center](#) = 1000.*[cst.pc](#)
- [X](#) = grid.grid([Xmin](#), [Xmax](#), 2**[NX](#), [xgridtype](#), s_center = [box_center](#))
- [E](#) = grid.grid([Emin](#), [Emax](#), 2**[NE](#), [egridtype](#))
- bool [in_damping](#) = True
OTHER TERMS #.
- bool [lz_damping](#) = True
- bool [nlld_damping](#) = True
- int [Pcr_1GeV](#) = 1*[cst.eV](#)
- int [Pe_1GeV](#) = 1e-2*[cst.eV](#)
- string [bdiff_model](#) = "ISM_independant"

- list `phases` = []
ISM STRUCTURE #.
- list `smooth_width_transition` = [10.*`cst.pc`, 3.*`cst.pc`, 3.*`cst.pc`, 10.*`cst.pc`, 10.*`cst.pc`, 3.*`cst.pc`, 3.*`cst.pc`, 10.*`cst.pc`]
- `T`
- `B`
- `ni`
- `nn`
- `nt`
- `Xi`
- `mi`
- `mn`
- `va`
- `gamma_in`
- `gamma_lz`
- `ism_values` = dict(`T=T`, `B=B`, `ni=ni`, `nn=nn`, `nt=nt`, `X=Xi`, `mi=mi`, `mn=mn`, `VA=va`, `gamma_in` = `gamma_in`, `gamma_lz` = `gamma_lz`)

3.16.1 Function Documentation

3.16.1.1 `getDamping()`

```
def namelist.getDamping (
    E,
    phase )
```

3.16.1.2 `getVA()`

```
def namelist.getVA (
    E,
    phase )
```

3.16.2 Variable Documentation

3.16.2.1 `B`

```
namelist.B
```

3.16.2.2 bdiff_model

```
string namelist.bdiff_model = "ISM_independant"
```

3.16.2.3 box_center

```
int namelist.box_center = 1000.*cst.pc
```

3.16.2.4 E

```
namelist.E = grid.grid(Emin, Emax, 2**NE, egridtype)
```

3.16.2.5 egridtype

```
string namelist.egridtype = "logspace"
```

3.16.2.6 Emax

```
float namelist.Emax = 50.01*cst.TeV
```

3.16.2.7 Emin

```
float namelist.Emin = 0.99*cst.GeV
```

3.16.2.8 folder_name

```
string namelist.folder_name = "Test_standard_full"
```

OUTPUT FOLDER CREATOR #.

Relative position of the ouput folder

3.16.2.9 folder_path

```
string namelist.folder_path = "../WorkFolder/"
```

3.16.2.10 gamma_in

```
namelist.gamma_in
```

3.16.2.11 gamma_lz

```
namelist.gamma_lz
```

3.16.2.12 in_damping

```
bool namelist.in_damping = True
```

OTHER TERMS #.

3.16.2.13 ism_values

```
namelist.ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va, gamma_in  
= gamma_in, gamma_lz = gamma_lz)
```

3.16.2.14 lz_damping

```
bool namelist.lz_damping = True
```

3.16.2.15 mi

```
namelist.mi
```


3.16.2.16 mn

```
namelist.mn
```

3.16.2.17 NE

```
int namelist.NE = 7
```

3.16.2.18 ni

```
namelist.ni
```

3.16.2.19 nlld_damping

```
bool namelist.nlld_damping = True
```

3.16.2.20 nn

```
namelist.nn
```

3.16.2.21 nt

```
namelist.nt
```

3.16.2.22 NX

```
int namelist.NX = 10
```

GRID PARAMETERS #.

3.16.2.23 Pcr_1GeV

```
int namelist.Pcr_1GeV = 1*cst.eV
```

3.16.2.24 Pe_1GeV

```
int namelist.Pe_1GeV = 1e-2*cst.eV
```

3.16.2.25 phases

```
list namelist.phases = []
```

ISM STRUCTURE #.

3.16.2.26 smooth_width_transition

```
list namelist.smooth_width_transition = [10.*cst.pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc, 10.*cst.↵  
pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc]
```

3.16.2.27 T

```
namelist.T
```

3.16.2.28 total_path

```
string namelist.total_path = folder_path+folder_name
```

3.16.2.29 va

```
namelist.va
```

3.16.2.30 X

```
namelist.X = grid.grid(Xmin, Xmax, 2**NX, xgridtype, s_center = box_center)
```

3.16.2.31 xgridtype

```
string namelist.xgridtype = "cartesian"
```

3.16.2.32 Xi

```
namelist.Xi
```

3.16.2.33 Xmax

```
int namelist.Xmax = 2000.*cst.pc
```

3.16.2.34 Xmin

```
int namelist.Xmin = 0.*cst.pc
```

3.17 namelist_adv Namespace Reference

Functions

- def [getVA](#) (E, phase)

Variables

- string `folder_name` = "advc_z_X12"
OUTPUT FOLDER CREATOR #.
- string `folder_path` = ".././WorkFolder/Fiducial_tests_for_thesis/"
- string `total_path` = `folder_path`+`folder_name`
- int `NX` = 12
GRID PARAMETERS #.
- int `NE` = 4
- int `Xmin` = 0.*`cst.pc`
- int `Xmax` = 2000.*`cst.pc`
- string `xgridtype` = "cartesian"
- int `Emin` = 10.*`cst.GeV`
- int `Emax` = 10.*`cst.TeV`
- string `egridtype` = "logspace"
- int `box_center` = 1000.*`cst.pc`
- `X`
- `E` = `grid.grid(Emin, Emax, 2**NE, egridtype)`
- bool `in_damping` = True
OTHER TERMS #.
- bool `lz_damping` = True
- bool `nlld_damping` = True
- int `Pcr_1GeV` = 1.*`cst.eV`
- int `Pe_1GeV` = 1.*`cst.eV`
- string `bdiff_model` = "ISM_independant"
- list `phases` = []
ISM STRUCTURE #.
- int `smooth_width_transition` = 10.*`cst.pc`
- `T`
- `B`
- `ni`
- `nn`
- `nt`
- `Xi`
- `mi`
- `mn`
- `va`
- `ism_values` = `dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)`

3.17.1 Function Documentation

3.17.1.1 `getVA()`

```
def namelist_adv.getVA (
    E,
    phase )
```

3.17.2 Variable Documentation

3.17.2.1 B

```
namelist_adv.B
```

3.17.2.2 bdiff_model

```
string namelist_adv.bdiff_model = "ISM_independant"
```

3.17.2.3 box_center

```
int namelist_adv.box_center = 1000.*cst.pc
```

3.17.2.4 E

```
namelist_adv.E = grid.grid(Emin, Emax, 2**NE, egridtype)
```

3.17.2.5 egridtype

```
string namelist_adv.egridtype = "logspace"
```

3.17.2.6 Emax

```
int namelist_adv.Emax = 10.*cst.TeV
```

3.17.2.7 Emin

```
int namelist_adv.Emin = 10.*cst.GeV
```

3.17.2.8 folder_name

```
string namelist_adv.folder_name = "advc_z_X12"
```

OUTPUT FOLDER CREATOR #.

Relative position of the ouput folder

3.17.2.9 folder_path

```
string namelist_adv.folder_path = "../../../WorkFolder/Fiducial_tests_for_thesis/"
```

3.17.2.10 in_damping

```
bool namelist_adv.in_damping = True
```

OTHER TERMS #.

3.17.2.11 ism_values

```
namelist_adv.ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)
```

3.17.2.12 lz_damping

```
bool namelist_adv.lz_damping = True
```

3.17.2.13 mi

```
namelist_adv.mi
```

3.17.2.14 mn

```
namelist_adv.mn
```

3.17.2.15 NE

```
int namelist_adv.NE = 4
```

3.17.2.16 ni

```
namelist_adv.ni
```

3.17.2.17 nlld_damping

```
bool namelist_adv.nlld_damping = True
```

3.17.2.18 nn

```
namelist_adv.nn
```

3.17.2.19 nt

```
namelist_adv.nt
```

3.17.2.20 NX

```
int namelist_adv.NX = 12
```

GRID PARAMETERS #.

3.17.2.21 Pcr_1GeV

```
int namelist_adv.Pcr_1GeV = 1*cst.eV
```

3.17.2.22 Pe_1GeV

```
int namelist_adv.Pe_1GeV = 1*cst.ev
```

3.17.2.23 phases

```
list namelist_adv.phases = [ ]
```

ISM STRUCTURE #.

3.17.2.24 smooth_width_transition

```
int namelist_adv.smooth_width_transition = 10.*cst.pc
```

3.17.2.25 T

```
namelist_adv.T
```

3.17.2.26 total_path

```
string namelist_adv.total_path = folder_path+folder_name
```

3.17.2.27 va

```
namelist_adv.va
```

3.17.2.28 X

```
namelist_adv.X
```

Initial value:

```
1 = grid.grid(Xmin, Xmax, 2**NX, xgridtype,  
2           s_center = box_center, width = 200.*cst.pc, smooth = 20.*cst.pc, dXmin = 1.*cst.pc)
```


3.17.2.29 xgridtype

```
string namelist_adv.xgridtype = "cartesian"
```

3.17.2.30 Xi

```
namelist_adv.Xi
```

3.17.2.31 Xmax

```
int namelist_adv.Xmax = 2000.*cst.pc
```

3.17.2.32 Xmin

```
int namelist_adv.Xmin = 0.*cst.pc
```

3.18 namelist_adve Namespace Reference

Functions

- def [getVA](#) (E, phase)

Variables

- string [folder_name](#) = "adv_e_X6"
OUTPUT FOLDER CREATOR #.
- string [folder_path](#) = "../WorkFolder/Fiducial_tests_for_thesis/"
- string [total_path](#) = [folder_path](#)+[folder_name](#)
- int [NX](#) = 6
GRID PARAMETERS #.
- int [NE](#) = 6
- int [Xmin](#) = 0.*[cst.pc](#)
- int [Xmax](#) = 2000.*[cst.pc](#)
- string [xgridtype](#) = "cartesian"
- int [Emin](#) = 10.*[cst.GeV](#)
- int [Emax](#) = 10.*[cst.TeV](#)
- string [egridtype](#) = "logspace"
- int [box_center](#) = 1000.*[cst.pc](#)
- [X](#)
- [E](#) = grid.grid([Emin](#), [Emax](#), 2**[NE](#), [egridtype](#))
- bool [in_damping](#) = True

```

    OTHER TERMS #.
    • bool lz_damping = True
    • bool nlld_damping = True
    • int Pcr_1GeV = 1*cst.eV
    • int Pe_1GeV = 1*cst.eV
    • string bdiff_model = "ISM_independant"
    • list phases = []

    ISM STRUCTURE #.
    • int smooth_width_transition = 10.*cst.pc
    • T
    • B
    • ni
    • nn
    • nt
    • Xi
    • mi
    • mn
    • va
    • ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)

```

3.18.1 Function Documentation

3.18.1.1 `getVA()`

```

def namelist_adve.getVA (
    E,
    phase )

```

3.18.2 Variable Documentation

3.18.2.1 `B`

```
namelist_adve.B
```

3.18.2.2 `bdiff_model`

```
string namelist_adve.bdiff_model = "ISM_independant"
```

3.18.2.3 box_center

```
int namelist_adve.box_center = 1000.*cst.pc
```

3.18.2.4 E

```
namelist_adve.E = grid.grid(Emin, Emax, 2**NE, egridtype)
```

3.18.2.5 egridtype

```
string namelist_adve.egridtype = "logspace"
```

3.18.2.6 Emax

```
int namelist_adve.Emax = 10.*cst.TeV
```

3.18.2.7 Emin

```
int namelist_adve.Emin = 10.*cst.GeV
```

3.18.2.8 folder_name

```
string namelist_adve.folder_name = "adv_e_X6"
```

OUTPUT FOLDER CREATOR #.

Relative position of the ouput folder

3.18.2.9 folder_path

```
string namelist_adve.folder_path = "../../../WorkFolder/Fiducial_tests_for_thesis/"
```

3.18.2.10 in_damping

```
bool namelist_adve.in_damping = True
```

OTHER TERMS #.

3.18.2.11 ism_values

```
namelist_adve.ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)
```

3.18.2.12 lz_damping

```
bool namelist_adve.lz_damping = True
```

3.18.2.13 mi

```
namelist_adve.mi
```

3.18.2.14 mn

```
namelist_adve.mn
```

3.18.2.15 NE

```
int namelist_adve.NE = 6
```

3.18.2.16 ni

```
namelist_adve.ni
```

3.18.2.17 nlld_damping

```
bool namelist_adve.nlld_damping = True
```

3.18.2.18 nn

```
namelist_adve.nn
```

3.18.2.19 nt

```
namelist_adve.nt
```

3.18.2.20 NX

```
int namelist_adve.NX = 6
```

GRID PARAMETERS #.

3.18.2.21 Pcr_1GeV

```
int namelist_adve.Pcr_1GeV = 1*cst.eV
```

3.18.2.22 Pe_1GeV

```
int namelist_adve.Pe_1GeV = 1*cst.eV
```

3.18.2.23 phases

```
list namelist_adve.phases = []
```

ISM STRUCTURE #.

3.18.2.24 smooth_width_transition

```
int namelist_adve.smooth_width_transition = 10.*cst.pc
```

3.18.2.25 T

```
namelist_adve.T
```

3.18.2.26 total_path

```
string namelist_adve.total_path = folder_path+folder_name
```

3.18.2.27 va

```
namelist_adve.va
```

3.18.2.28 X

```
namelist_adve.X
```

Initial value:

```
1 = grid.grid(Xmin, Xmax, 2**NX, xgridtype,  
2           s_center = box_center, width = 200.*cst.pc, smooth = 20.*cst.pc, dXmin = 1.*cst.pc)
```

3.18.2.29 xgridtype

```
string namelist_adve.xgridtype = "cartesian"
```

3.18.2.30 Xi

```
namelist_adve.Xi
```

3.18.2.31 Xmax

```
int namelist_adve.Xmax = 2000.*cst.pc
```

3.18.2.32 Xmin

```
int namelist_adve.Xmin = 0.*cst.pc
```

3.19 namelist_advst Namespace Reference

Functions

- def `getVA` (`E`, `phase`)

Variables

- string `folder_name` = "adv_sss_E6"
OUTPUT FOLDER CREATOR #.
- string `folder_path` = "../WorkFolder/Fiducial_tests_for_thesis/"
- string `total_path` = `folder_path`+`folder_name`
- int `NX` = 4
GRID PARAMETERS #.
- int `NE` = 6
- int `Xmin` = 0.*`cst.pc`
- int `Xmax` = 2000.*`cst.pc`
- string `xgridtype` = "cartesian"
- int `Emin` = 10.*`cst.GeV`
- int `Emax` = 10.*`cst.TeV`
- string `egridtype` = "logspace"
- int `box_center` = 1000.*`cst.pc`
- `X`
- `E` = `grid.grid(Emin, Emax, 2**NE, egridtype)`
- bool `in_damping` = True
OTHER TERMS #.
- bool `lz_damping` = True
- bool `nlld_damping` = True
- int `Pcr_1GeV` = 1.*`cst.eV`
- int `Pe_1GeV` = 1.*`cst.eV`
- string `bdiff_model` = "ISM_independant"
- list `phases` = []
ISM STRUCTURE #.
- int `smooth_width_transition` = 10.*`cst.pc`
- `T`
- `B`
- `ni`
- `nn`
- `nt`
- `Xi`
- `mi`
- `mn`
- `va`
- `ism_values` = dict(`T=T`, `B=B`, `ni=ni`, `nn=nn`, `nt=nt`, `X=Xi`, `mi=mi`, `mn=mn`, `VA=va`)

3.19.1 Function Documentation

3.19.1.1 `getVA()`

```
def namelist_advst.getVA (
    E,
    phase )
```

3.19.2 Variable Documentation

3.19.2.1 `B`

```
namelist_advst.B
```

3.19.2.2 `bdiff_model`

```
string namelist_advst.bdiff_model = "ISM_independant"
```

3.19.2.3 `box_center`

```
int namelist_advst.box_center = 1000.*cst.pc
```

3.19.2.4 `E`

```
namelist_advst.E = grid.grid(Emin, Emax, 2**NE, egridtype)
```

3.19.2.5 `egridtype`

```
string namelist_advst.egridtype = "logspace"
```


3.19.2.6 Emax

```
int namelist_advst.Emax = 10.*cst.Tev
```

3.19.2.7 Emin

```
int namelist_advst.Emin = 10.*cst.GeV
```

3.19.2.8 folder_name

```
string namelist_advst.folder_name = "adv_sss_E6"
```

OUTPUT FOLDER CREATOR #.

Relative position of the ouput folder

3.19.2.9 folder_path

```
string namelist_advst.folder_path = "../..//WorkFolder/Fiducial_tests_for_thesis/"
```

3.19.2.10 in_damping

```
bool namelist_advst.in_damping = True
```

OTHER TERMS #.

3.19.2.11 ism_values

```
namelist_advst.ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)
```

3.19.2.12 lz_damping

```
bool namelist_advst.lz_damping = True
```

3.19.2.13 mi

```
namelist_advst.mi
```

3.19.2.14 mn

```
namelist_advst.mn
```

3.19.2.15 NE

```
int namelist_advst.NE = 6
```

3.19.2.16 ni

```
namelist_advst.ni
```

3.19.2.17 nlld_damping

```
bool namelist_advst.nlld_damping = True
```

3.19.2.18 nn

```
namelist_advst.nn
```

3.19.2.19 nt

```
namelist_advst.nt
```

3.19.2.20 NX

```
int namelist_advst.NX = 4
```

GRID PARAMETERS #.

3.19.2.21 Pcr_1GeV

```
int namelist_advst.Pcr_1GeV = 1*cst.eV
```

3.19.2.22 Pe_1GeV

```
int namelist_advst.Pe_1GeV = 1*cst.eV
```

3.19.2.23 phases

```
list namelist_advst.phases = []
```

ISM STRUCTURE #.

3.19.2.24 smooth_width_transition

```
int namelist_advst.smooth_width_transition = 10.*cst.pc
```

3.19.2.25 T

```
namelist_advst.T
```

3.19.2.26 total_path

```
string namelist_advst.total_path = folder_path+folder_name
```

3.19.2.27 va

```
namelist_advst.va
```

3.19.2.28 X

```
namelist_advst.X
```

Initial value:

```
1 = grid.grid(Xmin, Xmax, 2*NX, xgridtype,  
2           s_center = box_center, width = 200.*cst.pc, smooth = 20.*cst.pc, dXmin = 1.*cst.pc)
```

3.19.2.29 xgridtype

```
string namelist_advst.xgridtype = "cartesian"
```

3.19.2.30 Xi

```
namelist_advst.Xi
```

3.19.2.31 Xmax

```
int namelist_advst.Xmax = 2000.*cst.pc
```

3.19.2.32 Xmin

```
int namelist_advst.Xmin = 0.*cst.pc
```

3.20 namelist_diff Namespace Reference

Functions

- def [getVA](#) (E, phase)

Variables

- string `folder_name` = "diff_z_X12"
OUTPUT FOLDER CREATOR #.
- string `folder_path` = ".././WorkFolder/Fiducial_tests_for_thesis/"
- string `total_path` = `folder_path+folder_name`
- int `NX` = 12
GRID PARAMETERS #.
- int `NE` = 4
- int `Xmin` = 0.*`cst.pc`
- int `Xmax` = 2000.*`cst.pc`
- string `xgridtype` = "cartesian"
- int `Emin` = 10.*`cst.GeV`
- int `Emax` = 10.*`cst.TeV`
- string `egridtype` = "logspace"
- int `box_center` = 1000.*`cst.pc`
- `X`
- `E` = `grid.grid(Emin, Emax, 2**NE, egridtype)`
- bool `in_damping` = True
OTHER TERMS #.
- bool `lz_damping` = True
- bool `nlld_damping` = True
- int `Pcr_1GeV` = 1.*`cst.eV`
- int `Pe_1GeV` = 1.*`cst.eV`
- string `bdiff_model` = "ISM_independant"
- list `phases` = []
ISM STRUCTURE #.
- int `smooth_width_transition` = 10.*`cst.pc`
- `T`
- `B`
- `ni`
- `nn`
- `nt`
- `Xi`
- `mi`
- `mn`
- `va`
- `ism_values` = `dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)`

3.20.1 Function Documentation

3.20.1.1 `getVA()`

```
def namelist_diff.getVA (
    E,
    phase )
```

3.20.2 Variable Documentation

3.20.2.1 B

```
namelist_diff.B
```

3.20.2.2 bdiff_model

```
string namelist_diff.bdiff_model = "ISM_independant"
```

3.20.2.3 box_center

```
int namelist_diff.box_center = 1000.*cst.pc
```

3.20.2.4 E

```
namelist_diff.E = grid.grid(Emin, Emax, 2**NE, egridtype)
```

3.20.2.5 egridtype

```
string namelist_diff.egridtype = "logspace"
```

3.20.2.6 Emax

```
int namelist_diff.Emax = 10.*cst.TeV
```

3.20.2.7 Emin

```
int namelist_diff.Emin = 10.*cst.GeV
```

3.20.2.8 folder_name

```
string namelist_diff.folder_name = "diff_z_X12"
```

OUTPUT FOLDER CREATOR #.

Relative position of the ouput folder

3.20.2.9 folder_path

```
string namelist_diff.folder_path = "../../../WorkFolder/Fiducial_tests_for_thesis/"
```

3.20.2.10 in_damping

```
bool namelist_diff.in_damping = True
```

OTHER TERMS #.

3.20.2.11 ism_values

```
namelist_diff.ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)
```

3.20.2.12 lz_damping

```
bool namelist_diff.lz_damping = True
```

3.20.2.13 mi

```
namelist_diff.mi
```

3.20.2.14 mn

```
namelist_diff.mn
```

3.20.2.15 NE

```
int namelist_diff.NE = 4
```

3.20.2.16 ni

```
namelist_diff.ni
```

3.20.2.17 nlld_damping

```
bool namelist_diff.nlld_damping = True
```

3.20.2.18 nn

```
namelist_diff.nn
```

3.20.2.19 nt

```
namelist_diff.nt
```

3.20.2.20 NX

```
int namelist_diff.NX = 12
```

GRID PARAMETERS #.

3.20.2.21 Pcr_1GeV

```
int namelist_diff.Pcr_1GeV = 1*cst.eV
```


3.20.2.22 Pe_1GeV

```
int namelist_diff.Pe_1GeV = 1*cst.ev
```

3.20.2.23 phases

```
list namelist_diff.phases = []
```

ISM STRUCTURE #.

3.20.2.24 smooth_width_transition

```
int namelist_diff.smooth_width_transition = 10.*cst.pc
```

3.20.2.25 T

```
namelist_diff.T
```

3.20.2.26 total_path

```
string namelist_diff.total_path = folder_path+folder_name
```

3.20.2.27 va

```
namelist_diff.va
```

3.20.2.28 X

```
namelist_diff.X
```

Initial value:

```
1 = grid.grid(Xmin, Xmax, 2*NX, xgridtype,  
2           s_center = box_center, width = 200.*cst.pc, smooth = 20.*cst.pc, dXmin = 1.*cst.pc)
```

3.20.2.29 xgridtype

```
string namelist_diff.xgridtype = "cartesian"
```

3.20.2.30 Xi

```
namelist_diff.Xi
```

3.20.2.31 Xmax

```
int namelist_diff.Xmax = 2000.*cst.pc
```

3.20.2.32 Xmin

```
int namelist_diff.Xmin = 0.*cst.pc
```

3.21 namelist_uniform Namespace Reference

Functions

- def [getVA](#) (E, phase)
- def [getDamping](#) (E, phase)

Variables

- string [folder_name](#) = "WNM-CNM-DiM_all_dependant"
OUTPUT FOLDER CREATOR #.
- string [folder_path](#) = ".././../WorkFolder/Fiducial_tests_for_thesis_2/"
- string [total_path](#) = [folder_path](#)+[folder_name](#)
- int [NX](#) = 12
GRID PARAMETERS #.
- int [NE](#) = 8
- int [Xmin](#) = 0.*[cst.pc](#)
- int [Xmax](#) = 2000.*[cst.pc](#)
- string [xgridtype](#) = "cartesian"
- float [Emin](#) = 0.99*[cst.GeV](#)
- float [Emax](#) = 50.01*[cst.TeV](#)
- string [egridtype](#) = "logspace"
- int [box_center](#) = 1000.*[cst.pc](#)
- [X](#) = grid.grid([Xmin](#), [Xmax](#), 2**[NX](#), [xgridtype](#), s_center = [box_center](#))
- [E](#) = grid.grid([Emin](#), [Emax](#), 2**[NE](#), [egridtype](#))

- bool `in_damping` = True
- OTHER TERMS #.*
- bool `lz_damping` = True
- bool `nlld_damping` = True
- int `Pcr_1GeV` = 1*cst.eV
- int `Pe_1GeV` = 1e-2*cst.eV
- string `bdiff_model` = "ISM_dependant"
- list `phases` = []
- ISM STRUCTURE #.*
- list `smooth_width_transition` = [10.*cst.pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc, 10.*cst.pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc]
- `T`
- `B`
- `ni`
- `nn`
- `nt`
- `Xi`
- `mi`
- `mn`
- `va`
- `gamma_in`
- `gamma_lz`
- `ism_values` = dict(`T=T`, `B=B`, `ni=ni`, `nn=nn`, `nt=nt`, `X=Xi`, `mi=mi`, `mn=mn`, `VA=va`, `gamma_in` = `gamma_in`, `gamma_lz` = `gamma_lz`)

3.21.1 Function Documentation

3.21.1.1 getDamping()

```
def namelist_uniform.getDamping (
    E,
    phase )
```

3.21.1.2 getVA()

```
def namelist_uniform.getVA (
    E,
    phase )
```

3.21.2 Variable Documentation

3.21.2.1 B

```
namelist_uniform.B
```

3.21.2.2 bdiff_model

```
string namelist_uniform.bdiff_model = "ISM_dependant"
```

3.21.2.3 box_center

```
int namelist_uniform.box_center = 1000.*cst.pc
```

3.21.2.4 E

```
namelist_uniform.E = grid.grid(Emin, Emax, 2**NE, egridtype)
```

3.21.2.5 egridtype

```
string namelist_uniform.egridtype = "logspace"
```

3.21.2.6 Emax

```
float namelist_uniform.Emax = 50.01*cst.TeV
```

3.21.2.7 Emin

```
float namelist_uniform.Emin = 0.99*cst.GeV
```

3.21.2.8 folder_name

```
string namelist_uniform.folder_name = "WNM-CNM-DiM_all_dependant"
```

OUTPUT FOLDER CREATOR #.

Relative position of the ouput folder

3.21.2.9 folder_path

```
string namelist_uniform.folder_path = "../../../WorkFolder/Fiducial_tests_for_thesis_2/"
```

3.21.2.10 gamma_in

```
namelist_uniform.gamma_in
```

3.21.2.11 gamma_lz

```
namelist_uniform.gamma_lz
```

3.21.2.12 in_damping

```
bool namelist_uniform.in_damping = True
```

OTHER TERMS #.

3.21.2.13 ism_values

```
namelist_uniform.ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va,  
gamma_in = gamma_in, gamma_lz = gamma_lz)
```

3.21.2.14 lz_damping

```
bool namelist_uniform.lz_damping = True
```

3.21.2.15 mi

```
namelist_uniform.mi
```

3.21.2.16 mn

```
namelist_uniform.mn
```

3.21.2.17 NE

```
int namelist_uniform.NE = 8
```

3.21.2.18 ni

```
namelist_uniform.ni
```

3.21.2.19 nlld_damping

```
bool namelist_uniform.nlld_damping = True
```

3.21.2.20 nn

```
namelist_uniform.nn
```

3.21.2.21 nt

```
namelist_uniform.nt
```

3.21.2.22 NX

```
int namelist_uniform.NX = 12
```

GRID PARAMETERS #.

3.21.2.23 Pcr_1GeV

```
int namelist_uniform.Pcr_1GeV = 1*cst.eV
```

3.21.2.24 Pe_1GeV

```
int namelist_uniform.Pe_1GeV = 1e-2*cst.eV
```

3.21.2.25 phases

```
list namelist_uniform.phases = []
```

ISM STRUCTURE #.

3.21.2.26 smooth_width_transition

```
list namelist_uniform.smooth_width_transition = [10.*cst.pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc,  
10.*cst.pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc]
```

3.21.2.27 T

```
namelist_uniform.T
```

3.21.2.28 total_path

```
string namelist_uniform.total_path = folder_path+folder_name
```

3.21.2.29 va

```
namelist_uniform.va
```

3.21.2.30 X

```
namelist_uniform.X = grid.grid(Xmin, Xmax, 2**NX, xgridtype, s_center = box_center)
```

3.21.2.31 xgridtype

```
string namelist_uniform.xgridtype = "cartesian"
```

3.21.2.32 Xi

```
namelist_uniform.Xi
```

3.21.2.33 Xmax

```
int namelist_uniform.Xmax = 2000.*cst.pc
```

3.21.2.34 Xmin

```
int namelist_uniform.Xmin = 0.*cst.pc
```

3.22 namelist_vdiff Namespace Reference

Functions

- def [getVA](#) (E, phase)

Variables

- string `folder_name` = "vdiff_z_X14"
OUTPUT FOLDER CREATOR #.
- string `folder_path` = "../WorkFolder/Fiducial_tests_for_thesis/"
- string `total_path` = `folder_path`+`folder_name`
- int `NX` = 14
GRID PARAMETERS #.
- int `NE` = 4
- int `Xmin` = 0.*`cst.pc`
- int `Xmax` = 2000.*`cst.pc`
- string `xgridtype` = "cartesian"
- int `Emin` = 10.*`cst.GeV`
- int `Emax` = 10.*`cst.TeV`
- string `egridtype` = "logspace"
- int `box_center` = 1000.*`cst.pc`
- `X`
- `E` = `grid.grid(Emin, Emax, 2**NE, egridtype)`
- bool `in_damping` = True
OTHER TERMS #.
- bool `lz_damping` = True
- bool `nlld_damping` = True
- int `Pcr_1GeV` = 1.*`cst.eV`
- int `Pe_1GeV` = 1.*`cst.eV`
- string `bdiff_model` = "ISM_independant"
- list `phases` = []
ISM STRUCTURE #.
- int `smooth_width_transition` = 10.*`cst.pc`
- `T`
- `B`
- `ni`
- `nn`
- `nt`
- `Xi`
- `mi`
- `mn`
- `va`
- `ism_values` = `dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)`

3.22.1 Function Documentation

3.22.1.1 `getVA()`

```
def namelist_vdiff.getVA (
    E,
    phase )
```

3.22.2 Variable Documentation

3.22.2.1 B

```
namelist_vdiff.B
```

3.22.2.2 bdiff_model

```
string namelist_vdiff.bdiff_model = "ISM_independant"
```

3.22.2.3 box_center

```
int namelist_vdiff.box_center = 1000.*cst.pc
```

3.22.2.4 E

```
namelist_vdiff.E = grid.grid(Emin, Emax, 2**NE, egridtype)
```

3.22.2.5 egridtype

```
string namelist_vdiff.egridtype = "logspace"
```

3.22.2.6 Emax

```
int namelist_vdiff.Emax = 10.*cst.TeV
```

3.22.2.7 Emin

```
int namelist_vdiff.Emin = 10.*cst.GeV
```

3.22.2.8 folder_name

```
string namelist_vdiff.folder_name = "vdiff_z_X14"
```

OUTPUT FOLDER CREATOR #.

Relative position of the ouput folder

3.22.2.9 folder_path

```
string namelist_vdiff.folder_path = "../../../WorkFolder/Fiducial_tests_for_thesis/"
```

3.22.2.10 in_damping

```
bool namelist_vdiff.in_damping = True
```

OTHER TERMS #.

3.22.2.11 ism_values

```
namelist_vdiff.ism_values = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)
```

3.22.2.12 lz_damping

```
bool namelist_vdiff.lz_damping = True
```

3.22.2.13 mi

```
namelist_vdiff.mi
```

3.22.2.14 mn

```
namelist_vdiff.mn
```

3.22.2.15 NE

```
int namelist_vdiff.NE = 4
```

3.22.2.16 ni

```
namelist_vdiff.ni
```

3.22.2.17 nlld_damping

```
bool namelist_vdiff.nlld_damping = True
```

3.22.2.18 nn

```
namelist_vdiff.nn
```

3.22.2.19 nt

```
namelist_vdiff.nt
```

3.22.2.20 NX

```
int namelist_vdiff.NX = 14
```

GRID PARAMETERS #.

3.22.2.21 Pcr_1GeV

```
int namelist_vdiff.Pcr_1GeV = 1*cst.eV
```

3.22.2.22 Pe_1GeV

```
int namelist_vdiff.Pe_1GeV = 1*cst.eV
```

3.22.2.23 phases

```
list namelist_vdiff.phases = []
```

ISM STRUCTURE #.

3.22.2.24 smooth_width_transition

```
int namelist_vdiff.smooth_width_transition = 10.*cst.pc
```

3.22.2.25 T

```
namelist_vdiff.T
```

3.22.2.26 total_path

```
string namelist_vdiff.total_path = folder_path+folder_name
```

3.22.2.27 va

```
namelist_vdiff.va
```

3.22.2.28 X

```
namelist_vdiff.X
```

Initial value:

```
1 = grid.grid(Xmin, Xmax, 2**NX, xgridtype,  
2           s_center = box_center, width = 200.*cst.pc, smooth = 20.*cst.pc, dXmin = 1.*cst.pc)
```

3.22.2.29 xgridtype

```
string namelist_vdiff.xgridtype = "cartesian"
```

3.22.2.30 Xi

```
namelist_vdiff.Xi
```

3.22.2.31 Xmax

```
int namelist_vdiff.Xmax = 2000.*cst.pc
```

3.22.2.32 Xmin

```
int namelist_vdiff.Xmin = 0.*cst.pc
```

3.23 Output_functions Namespace Reference

3.24 pcr_ip_2D Namespace Reference

Functions

- def [readDataXE](#) (file_name, NX, NE)
- def [readAxis](#) (file_name)
- def [getData](#) (var, file_number)
- def [getTimeID](#) (time_test, delta_t, t_ini, kind="linear")

Variables

- `int t_ini = 0.*cst.kyr`
- `int t_max = 200.*cst.kyr`
- `float delta_t = 0.1*cst.kyr`
- `int x_center = 1000.`
- `float time_test = 5.9*cst.kyr`
- `def out_id = getTimeID(time_test, delta_t, t_ini)`
- `X`
- `E`
- `Pcr`
- `lp`
- `EV`
- `XV`
- `sparse`
- `False`
- `indexing`
- `cmap = plt.get_cmap('jet')`
- `XV_log = XV/cst.pc`
- `EV_log = np.log10(EV/cst.GeV)`
- `ax0`
- `ax1`
- `ncols`
- `sharey`
- `figsize`
- `im0 = ax0.pcolormesh(XV_log-x_center, EV_log, np.log10(Pcr), cmap=cmap)`
- `ax`
- `cax`
- `label`
- `im1 = ax1.pcolormesh(XV_log-x_center, EV_log, np.log10(lp), cmap=cmap)`
- `fontsize`

3.24.1 Function Documentation

3.24.1.1 `getData()`

```
def pcr_ip_2D.getData (
    var,
    file_number )
```

3.24.1.2 `getTimeID()`

```
def pcr_ip_2D.getTimeID (
    time_test,
    delta_t,
    t_ini,
    kind = "linear" )
```

3.24.1.3 readAxis()

```
def pcr_ip_2D.readAxis (  
    file_name )
```

3.24.1.4 readDataXE()

```
def pcr_ip_2D.readDataXE (  
    file_name,  
    NX,  
    NE )
```

3.24.2 Variable Documentation

3.24.2.1 ax

```
pcr_ip_2D.ax
```

3.24.2.2 ax0

```
pcr_ip_2D.ax0
```

3.24.2.3 ax1

```
pcr_ip_2D.ax1
```

3.24.2.4 cax

```
pcr_ip_2D.cax
```

3.24.2.5 cmap

```
pcr_ip_2D.cmap = plt.get_cmap('jet')
```


3.24.2.6 delta_t

```
float pcr_ip_2D.delta_t = 0.1*cst.kyr
```

3.24.2.7 E

```
pcr_ip_2D.E
```

3.24.2.8 EV

```
pcr_ip_2D.EV
```

3.24.2.9 EV_log

```
pcr_ip_2D.EV_log = np.log10(EV/cst.GeV)
```

3.24.2.10 False

```
pcr_ip_2D.False
```

3.24.2.11 figsize

```
pcr_ip_2D.figsize
```

3.24.2.12 fontsize

```
pcr_ip_2D.fontsize
```

3.24.2.13 im0

```
pcr_ip_2D.im0 = ax0.pcolormesh(XV_log-x_center, EV_log, np.log10(Pcr), cmap=cmap)
```

3.24.2.14 im1

```
pcr_ip_2D.im1 = ax1.pcolormesh(XV_log-x_center, EV_log, np.log10(Ip), cmap=cmap)
```

3.24.2.15 indexing

```
pcr_ip_2D.indexing
```

3.24.2.16 Ip

```
pcr_ip_2D.Ip
```

3.24.2.17 label

```
pcr_ip_2D.label
```

3.24.2.18 ncols

```
pcr_ip_2D.ncols
```

3.24.2.19 out_id

```
def pcr_ip_2D.out_id = getTimeID(time_test, delta_t, t_ini)
```

3.24.2.20 Pcr

```
pcr_ip_2D.Pcr
```

3.24.2.21 sharey

```
pcr_ip_2D.sharey
```

3.24.2.22 sparse

pcr_ip_2D.sparse

3.24.2.23 t_ini

```
int pcr_ip_2D.t_ini = 0.*cst.kyr
```

3.24.2.24 t_max

```
int pcr_ip_2D.t_max = 200.*cst.kyr
```

3.24.2.25 time_test

```
float pcr_ip_2D.time_test = 5.9*cst.kyr
```

3.24.2.26 X

pcr_ip_2D.X

3.24.2.27 x_center

```
int pcr_ip_2D.x_center = 1000.
```

3.24.2.28 XV

pcr_ip_2D.XV

3.24.2.29 XV_log

```
pcr_ip_2D.XV_log = XV/cst.pc
```

3.25 PDE_solvers Namespace Reference

Functions

- def `TDMA` (a, b, c, d)
- def `A` (x)
- def `B` (x)
- def `C` (x)
- def `Q` (x)
- def `T` (x)
- def `finiteDiffSolver` (dt, Tmax, u)
- def `SimpleImplicitSolver` (dt, Tmax, u)
- def `CC70Solver` (dt, Tmax, u)

Variables

- float `pc` = 3.086e18
Tri Diagonal Matrix Algorithm(a.k.a Thomas algorithm) solver def TDMA solver(a, b, c, d): ''' TDMA solver, a b c d can be NumPy array type or Python list type.
- float `yr` = 365.25*86400
- int `kyr` = 1e3*yr
- int `M` = 2048
- int `Xmin` = 0.
- int `Xmax` = 1000.*pc
- `X` = np.linspace(`Xmin`, `Xmax`, `M`+1)
- `u` = np.zeros(`M`+1)
plt.semilogy(X/pc, u, c="blue") plt.plot(X/pc, u/U, c="red") plt.axhline(1.) print ("ratio = ",sum(u)/sum(U))
- `u0` = `u`[1]
- `uM` = `u`[`M`-1]
- `u_ini` = `u`.copy()
- `u_end` = `u`
- int `Tmax` = 10.*kyr
- def `u_0` = `CC70Solver`(0.1*kyr, Tmax, u)
- def `u_1` = `CC70Solver`(0.5*kyr, Tmax, u)
- def `u_2` = `CC70Solver`(5.*kyr, Tmax, u)
- `figsize`
- `c`

3.25.1 Function Documentation

3.25.1.1 A()

```
def PDE_solvers.A (
    x )
```

3.25.1.2 B()

```
def PDE_solvers.B (  
    x )
```

3.25.1.3 C()

```
def PDE_solvers.C (  
    x )
```

3.25.1.4 CC70Solver()

```
def PDE_solvers.CC70Solver (  
    dt,  
    Tmax,  
    u )
```

3.25.1.5 finiteDiffSolver()

```
def PDE_solvers.finiteDiffSolver (  
    dt,  
    Tmax,  
    u )
```

3.25.1.6 Q()

```
def PDE_solvers.Q (  
    x )
```

3.25.1.7 SimpleImplicitSolver()

```
def PDE_solvers.SimpleImplicitSolver (  
    dt,  
    Tmax,  
    u )
```

3.25.1.8 T()

```
def PDE_solvers.T (
    x )
```

3.25.1.9 TDMA()

```
def PDE_solvers.TDMA (
    a,
    b,
    c,
    d )
```

3.25.2 Variable Documentation

3.25.2.1 c

```
PDE_solvers.c
```

3.25.2.2 figsize

```
PDE_solvers.figsize
```

3.25.2.3 kyr

```
int PDE_solvers.kyr = 1e3*yr
```

3.25.2.4 M

```
int PDE_solvers.M = 2048
```

3.25.2.5 pc

```
float PDE_solvers.pc = 3.086e18
```

Tri Diagonal Matrix Algorithm(a.k.a Thomas algorithm) solver def TDMA solver(a, b, c, d): ''' TDMA solver, a b c d can be NumPy array type or Python list type.

refer to http://en.wikipedia.org/wiki/Tridiagonal_matrix_algorithm and to [http://www.cfd-online.com/Wiki/Tridiagonal_matrix_algorithm_-_TDMA_\(Thomas_algorithm\)](http://www.cfd-online.com/Wiki/Tridiagonal_matrix_algorithm_-_TDMA_(Thomas_algorithm)) ''' nf = len(d) # number of equations ac, bc, cc, dc = map(np.array, (a, b, c, d)) # copy arrays for it in range(1, nf): mc = ac[it-1]/bc[it-1] bc[it] = bc[it] - mc*cc[it-1] dc[it] = dc[it] - mc*dc[it-1]

```
xc = bc xc[-1] = dc[-1]/bc[-1]
```

```
for il in range(nf-2, -1, -1): xc[il] = (dc[il]-cc[il]*xc[il+1])/bc[il]
```

```
return xc
```

3.25.2.6 Tmax

```
int PDE_solvers.Tmax = 10.*kyr
```

3.25.2.7 u

```
PDE_solvers.u = np.zeros(M+1)
```

```
plt.semilogy(X/pc, u, c="blue") plt.plot(X/pc, u/U, c="red") plt.axhline(1.) print ("ratio = ",sum(u)/sum(U))
```

3.25.2.8 u0

```
PDE_solvers.u0 = u[1]
```

3.25.2.9 u_0

```
PDE_solvers.u_0 = CC70Solver(0.1*kyr, Tmax, u)
```

3.25.2.10 u_1

```
PDE_solvers.u_1 = CC70Solver(0.5*kyr, Tmax, u)
```

3.25.2.11 u_2

```
PDE_solvers.u_2 = CC70Solver(5.*kyr, Tmax, u)
```

3.25.2.12 u_end

```
PDE_solvers.u_end = u
```

3.25.2.13 u_ini

```
PDE_solvers.u_ini = u.copy()
```

3.25.2.14 uM

```
PDE_solvers.uM = u[M-1]
```

3.25.2.15 X

```
PDE_solvers.X = np.linspace(Xmin, Xmax, M+1)
```

3.25.2.16 Xmax

```
int PDE_solvers.Xmax = 1000.*pc
```

3.25.2.17 Xmin

```
int PDE_solvers.Xmin = 0.
```

3.25.2.18 yr

```
float PDE_solvers.yr = 365.25*86400
```


3.26 phases_collection Namespace Reference

Functions

- def `ism_phase` (Temp, Bfiel, nion, ntot, mion, mneutral)

Variables

- def `HII` = `ism_phase`(8000, 10.e-6, 99.9, 100., 0.93*cst.mHII+0.07*cst.mHeII, 0.93*cst.mHI+0.07*cst.mHeI)
- def `WIM` = `ism_phase`(8000, 5.00001e-6, 0.315, 0.35, cst.mHII, 0.93*cst.mHI+0.07*cst.mHeI)
- def `WNM` = `ism_phase`(8000, 5.00001e-6, 7e-3, 0.35, cst.mHII, 0.93*cst.mHI+0.07*cst.mHeI)
- def `CNM` = `ism_phase`(50, 6.00001e-6, 2.3e-2, 30.0, cst.mCII, 0.93*cst.mHI+0.07*cst.mHeI)
- def `DiM` = `ism_phase`(50, 6.00001e-6, 3.0e-2, 300, cst.mCII, 0.93*(0.5*cst.mHI + 0.5*cst.mH2) + 0.07*cst.mHeI↵mHeI)
- def `DeM` = `ism_phase`(30, 26.0001e-6, 3.0e-2, 3000, cst.mHCOII, 0.93*cst.mH2 + 0.07*cst.mHeI)
- def `DeC` = `ism_phase`(20, 59.0001e-6, 1.0e-2, 1e4, cst.mHCOII, 0.93*cst.mH2 + 0.07*cst.mHeI)

3.26.1 Function Documentation

3.26.1.1 `ism_phase()`

```
def phases_collection.ism_phase (
    Temp,
    Bfiel,
    nion,
    ntot,
    mion,
    mneutral )
```

3.26.2 Variable Documentation

3.26.2.1 `CNM`

```
def phases_collection.CNM = ism_phase( 50, 6.00001e-6, 2.3e-2, 30.0, cst.mCII, 0.93*cst.mH↵I+0.07*cst.mHeI)
```

3.26.2.2 `DeC`

```
def phases_collection.DeC = ism_phase( 20, 59.0001e-6, 1.0e-2, 1e4, cst.mHCOII, 0.93*cst.mH2 + 0.07*cst.mHeI)
```

3.26.2.3 DeM

```
def phases_collection.DeM = ism_phase( 30, 26.0001e-6, 3.0e-2, 3000, cst.mHCOII, 0.93*cst.mH2
+ 0.07*cst.mHeI)
```

3.26.2.4 DiM

```
def phases_collection.DiM = ism_phase( 50, 6.00001e-6, 3.0e-2, 300, cst.mCII, 0.93*(0.5*cst.mH
HI + 0.5*cst.mH2) + 0.07*cst.mHeI)
```

3.26.2.5 HII

```
def phases_collection.HII = ism_phase(8000, 10.e-6, 99.9, 100., 0.93*cst.mHII+0.07*cst.mHeII,
0.93*cst.mHI+0.07*cst.mHeI)
```

3.26.2.6 WIM

```
def phases_collection.WIM = ism_phase(8000, 5.00001e-6, 0.315, 0.35, cst.mHII, 0.93*cst.mH
I+0.07*cst.mHeI)
```

3.26.2.7 WNM

```
def phases_collection.WNM = ism_phase(8000, 5.00001e-6, 7e-3, 0.35, cst.mHII, 0.93*cst.mHI+0.
07*cst.mHeI)
```

3.27 physical_models Namespace Reference

Functions

- def [collision_rate](#) (specie1, specie2, phase)
- def [cr_escape_time_model](#) (option, model, Ecr, props)
- def [cr_escape_radius_model](#) (option, model, time, props)

3.27.1 Function Documentation

3.27.1.1 collision_rate()

```
def physical_models.collision_rate (
    specie1,
    specie2,
    phase )
```

3.27.1.2 cr_escape_radius_model()

```
def physical_models.cr_escape_radius_model (
    option,
    model,
    time,
    props )
```

3.27.1.3 cr_escape_time_model()

```
def physical_models.cr_escape_time_model (
    option,
    model,
    Ecr,
    props )
```

3.28 setup Namespace Reference

Variables

- `X` = nml.X
- `E` = nml.E
- `nx` = nml.NX
- `END : INITIAL ISM CONDITIONS #.`
- `ne` = nml.NE
- `x_center` = nml.box_center
- `x_center_index` = int(`x_center`/(`X`[1] - `X`[0]))
- `ism_values` = nml.ism_values
- `B` = ism_values.get("B")
- `nn` = ism_values.get("nn")
- `ni` = ism_values.get("ni")
- `mn` = ism_values.get("mn")
- `mi` = ism_values.get("mi")
- `T` = ism_values.get("T")
- `Xi` = ism_values.get("X")
- `va` = ism_values.get("VA")
- `g_in` = ism_values.get("gamma_in")
- `g_lz` = ism_values.get("gamma_lz")
- `d00` = np.zeros(len(`X`))

- INITIAL ISM CONDITIONS #.*
- `D` = `np.zeros((len(E), len(X)))`
 - `Db` = `np.zeros((len(E), len(X)))`
 - `Ip` = `np.zeros((len(E), len(X)))`
 - `Im` = `np.zeros((len(E), len(X)))`
 - `VA` = `np.zeros((len(E), len(X)))`
 - `gamma_in` = `np.zeros((len(E), len(X)))`
 - `gamma_lazarian` = `np.zeros((len(E), len(X)))`
 - `gamma_nlld` = `np.zeros((len(E), len(X)))`
 - `gamma_tot` = `np.zeros((len(E), len(X)))`
 - `Pcr` = `np.zeros((len(E), len(X)))`
 - `Pe` = `np.zeros((len(E), len(X)))`
 - dictionary `medium_props`
 - `mass`
 - `kmin`
 - `q`
 - `I`
 - `in_damping` = `dp.indamping_alfven(xi, E[e], ism_values)`
 - `variable`
 - `path`
 - dictionary `variables`
 - `ext`

3.28.1 Variable Documentation

3.28.1.1 B

```
setup.B = ism_values.get("B")
```

3.28.1.2 D

```
setup.D = np.zeros((len(E), len(X)))
```

3.28.1.3 d00

```
setup.d00 = np.zeros(len(X))
```

INITIAL ISM CONDITIONS #.

You can modify this part #

3.28.1.4 Db

```
setup.Db = np.zeros((len(E), len(X)))
```

3.28.1.5 E

```
setup.E = nml.E
```

3.28.1.6 ext

```
setup.ext
```

3.28.1.7 g_in

```
setup.g_in = ism_values.get("gamma_in")
```

3.28.1.8 g_lz

```
setup.g_lz = ism_values.get("gamma_lz")
```

3.28.1.9 gamma_in

```
setup.gamma_in = np.zeros((len(E), len(X)))
```

3.28.1.10 gamma_lazarian

```
setup.gamma_lazarian = np.zeros((len(E), len(X)))
```

3.28.1.11 gamma_nlld

```
setup.gamma_nlld = np.zeros((len(E), len(X)))
```

3.28.1.12 gamma_tot

```
setup.gamma_tot = np.zeros((len(E), len(X)))
```

3.28.1.13 I

```
setup.I
```

3.28.1.14 Im

```
setup.Im = np.zeros((len(E), len(X)))
```

3.28.1.15 in_damping

```
setup.in_damping = dp.indamping_alfven(xi , E[e], ism_values)
```

3.28.1.16 Ip

```
setup.Ip = np.zeros((len(E), len(X)))
```

3.28.1.17 ism_values

```
setup.ism_values = nml.ism_values
```

3.28.1.18 kmin

```
setup.kmin
```

3.28.1.19 mass

```
setup.mass
```

3.28.1.20 medium_props

dictionary setup.medium_props

Initial value:

```
1 = {"B" : B[xi],
2     "mn" : mn[xi],
3     "nn" : nn[xi],
4     "mi" : mi[xi],
5     "ni" : ni[xi],
6     "X" : Xi[xi],
7     "T" : T[xi]}
```

3.28.1.21 mi

```
setup.mi = ism_values.get("mi")
```

3.28.1.22 mn

```
setup.mn = ism_values.get("mn")
```

3.28.1.23 ne

```
setup.ne = nml.NE
```

3.28.1.24 ni

```
setup.ni = ism_values.get("ni")
```

3.28.1.25 nn

```
setup.nn = ism_values.get("nn")
```

3.28.1.26 nx

```
setup.nx = nml.NX
```

```
END : INITIAL ISM CONDITIONS #
```

```
WRITE THE INITAL CONDITIONS #
```

3.28.1.27 path

```
setup.path
```

3.28.1.28 Pcr

```
setup.Pcr = np.zeros((len(E), len(X)))
```

3.28.1.29 Pe

```
setup.Pe = np.zeros((len(E), len(X)))
```

3.28.1.30 q

```
setup.q
```

3.28.1.31 T

```
setup.T = ism_values.get("T")
```

3.28.1.32 va

```
setup.va = ism_values.get("VA")
```


3.28.1.33 VA

```
setup.VA = np.zeros((len(E), len(X)))
```

3.28.1.34 variable

```
setup.variable
```

3.28.1.35 variables

```
setup.variables
```

Initial value:

```
1 = {"NX"      : nx,  
2     "NE"      : ne,  
3     "ni"      : ni[x_center_index],  
4     "X"       : Xi[x_center_index],  
5     "mn"      : mn[x_center_index],  
6     "T"       : T[x_center_index],  
7     "center"  : x_center,  
8     "center_index": x_center_index,  
9     "B"       : B[x_center_index]}
```

3.28.1.36 X

```
setup.X = nml.X
```

3.28.1.37 x_center

```
setup.x_center = nml.box_center
```

3.28.1.38 x_center_index

```
setup.x_center_index = int(x_center/(X[1] - X[0]))
```

3.28.1.39 Xi

```
setup.Xi = ism_values.get("X")
```

3.29 setup_adv Namespace Reference

Functions

- def [door](#) ([X](#), X1, X2, V)

Variables

- [X](#) = nml.X
- [E](#) = nml.E
- [nx](#) = nml.NX
- *END : INITIAL ISM CONDITIONS #.*
- [ne](#) = nml.NE
- [x_center](#) = nml.box_center
- [x_center_index](#) = int([x_center](#)/([X](#)[1] - [X](#)[0]))
- [ism_values](#) = nml.ism_values
- [B](#) = ism_values.get("B")
- [nn](#) = ism_values.get("nn")
- [ni](#) = ism_values.get("ni")
- [mn](#) = ism_values.get("mn")
- [mi](#) = ism_values.get("mi")
- [T](#) = ism_values.get("T")
- [Xi](#) = ism_values.get("X")
- [va](#) = ism_values.get("VA")
- [d00](#) = np.zeros(len([X](#)))
- *INITIAL ISM CONDITIONS #.*
- [D](#) = np.zeros((len([E](#)), len([X](#))))
- [Db](#) = np.zeros((len([E](#)), len([X](#))))
- [lp](#) = np.zeros((len([E](#)), len([X](#))))
- [lm](#) = np.zeros((len([E](#)), len([X](#))))
- [VA](#) = np.zeros((len([E](#)), len([X](#))))
- [gamma_in](#) = np.zeros((len([E](#)), len([X](#))))
- [gamma_lazarian](#) = np.zeros((len([E](#)), len([X](#))))
- [gamma_nlld](#) = np.zeros((len([E](#)), len([X](#))))
- [gamma_tot](#) = np.zeros((len([E](#)), len([X](#))))
- [Pcr](#) = np.zeros((len([E](#)), len([X](#))))
- [Pe](#) = np.zeros((len([E](#)), len([X](#))))
- [variable](#)
- [path](#)
- dictionary [variables](#)
- [ext](#)

3.29.1 Function Documentation

3.29.1.1 door()

```
def setup_adv.door (
    X,
    X1,
    X2,
    V )
```

3.29.2 Variable Documentation

3.29.2.1 B

```
setup_adv.B = ism_values.get("B")
```

3.29.2.2 D

```
setup_adv.D = np.zeros((len(E), len(X)))
```

3.29.2.3 d00

```
setup_adv.d00 = np.zeros(len(X))
```

INITIAL ISM CONDITIONS #.

You can modify this part #

3.29.2.4 Db

```
setup_adv.Db = np.zeros((len(E), len(X)))
```

3.29.2.5 E

```
setup_adv.E = nml.E
```

3.29.2.6 ext

```
setup_adv.ext
```

3.29.2.7 gamma_in

```
setup_adv.gamma_in = np.zeros((len(E), len(X)))
```

3.29.2.8 gamma_lazarian

```
setup_adv.gamma_lazarian = np.zeros((len(E), len(X)))
```

3.29.2.9 gamma_nlld

```
setup_adv.gamma_nlld = np.zeros((len(E), len(X)))
```

3.29.2.10 gamma_tot

```
setup_adv.gamma_tot = np.zeros((len(E), len(X)))
```

3.29.2.11 Im

```
setup_adv.Im = np.zeros((len(E), len(X)))
```

3.29.2.12 Ip

```
setup_adv.Ip = np.zeros((len(E), len(X)))
```

3.29.2.13 ism_values

```
setup_adv.ism_values = nml.ism_values
```

3.29.2.14 mi

```
setup_adv.mi = ism_values.get("mi")
```

3.29.2.15 mn

```
setup_adv.mn = ism_values.get("mn")
```

3.29.2.16 ne

```
setup_adv.ne = nml.NE
```

3.29.2.17 ni

```
setup_adv.ni = ism_values.get("ni")
```

3.29.2.18 nn

```
setup_adv.nn = ism_values.get("nn")
```

3.29.2.19 nx

```
setup_adv.nx = nml.NX
```

END : INITIAL ISM CONDITIONS #.

WRITE THE INITAL CONDITIONS #

3.29.2.20 path

```
setup_adv.path
```

3.29.2.21 Pcr

```
setup_adv.Pcr = np.zeros((len(E), len(X)))
```

3.29.2.22 Pe

```
setup_adv.Pe = np.zeros((len(E), len(X)))
```

3.29.2.23 T

```
setup_adv.T = ism_values.get("T")
```

3.29.2.24 va

```
setup_adv.va = ism_values.get("VA")
```

3.29.2.25 VA

```
setup_adv.VA = np.zeros((len(E), len(X)))
```

3.29.2.26 variable

```
setup_adv.variable
```

3.29.2.27 variables

```
setup_adv.variables
```

Initial value:

```
1 = {"NX"      : nx,
2     "NE"      : ne,
3     "ni"      : ni[x_center_index],
4     "X"       : Xi[x_center_index],
5     "mn"      : mn[x_center_index],
6     "T"       : T[x_center_index],
7     "center"  : x_center,
8     "center_index": x_center_index,
9     "B"       : B[x_center_index]}
```

3.29.2.28 X

```
setup_adv.X = nml.X
```

3.29.2.29 x_center

```
setup_adv.x_center = nml.box_center
```

3.29.2.30 x_center_index

```
setup_adv.x_center_index = int(x_center/(X[1] - X[0]))
```

3.29.2.31 Xi

```
setup_adv.Xi = ism_values.get("X")
```

3.30 setup_adv Namespace Reference

Functions

- def [door](#) (X, X1, X2, V)
- def [spec](#) (E, q)

Variables

- [X](#) = nml.X
- [E](#) = nml.E
- [nx](#) = nml.NX
END : INITIAL ISM CONDITIONS #.
- [ne](#) = nml.NE
- [x_center](#) = nml.box_center
- [x_center_index](#) = int([x_center](#)/(X[1] - X[0]))
- [ism_values](#) = nml.ism_values
- [B](#) = ism_values.get("B")
- [nn](#) = ism_values.get("nn")
- [ni](#) = ism_values.get("ni")
- [mn](#) = ism_values.get("mn")
- [mi](#) = ism_values.get("mi")
- [T](#) = ism_values.get("T")
- [Xi](#) = ism_values.get("X")
- [va](#) = ism_values.get("VA")

- `d00 = np.zeros(len(X))`
INITIAL ISM CONDITIONS #.
- `D = np.zeros((len(E), len(X)))`
- `Db = np.zeros((len(E), len(X)))`
- `lp = np.zeros((len(E), len(X)))`
- `lm = np.zeros((len(E), len(X)))`
- `VA = np.zeros((len(E), len(X)))`
- `gamma_in = np.zeros((len(E), len(X)))`
- `gamma_lazarian = np.zeros((len(E), len(X)))`
- `gamma_nlld = np.zeros((len(E), len(X)))`
- `gamma_tot = np.zeros((len(E), len(X)))`
- `Pcr = np.zeros((len(E), len(X)))`
- `Pe = np.zeros((len(E), len(X)))`
- `variable`
- `path`
- dictionary `variables`
- `ext`

3.30.1 Function Documentation

3.30.1.1 `door()`

```
def setup_adve.door (
    X,
    X1,
    X2,
    V )
```

3.30.1.2 `spec()`

```
def setup_adve.spec (
    E,
    q )
```

3.30.2 Variable Documentation

3.30.2.1 `B`

```
setup_adve.B = ism_values.get("B")
```


3.30.2.2 D

```
setup_adve.D = np.zeros((len(E), len(X)))
```

3.30.2.3 d00

```
setup_adve.d00 = np.zeros(len(X))
```

INITIAL ISM CONDITIONS #.

You can modify this part #

3.30.2.4 Db

```
setup_adve.Db = np.zeros((len(E), len(X)))
```

3.30.2.5 E

```
setup_adve.E = nml.E
```

3.30.2.6 ext

```
setup_adve.ext
```

3.30.2.7 gamma_in

```
setup_adve.gamma_in = np.zeros((len(E), len(X)))
```

3.30.2.8 gamma_lazarian

```
setup_adve.gamma_lazarian = np.zeros((len(E), len(X)))
```

3.30.2.9 gamma_nlld

```
setup_adve.gamma_nlld = np.zeros((len(E), len(X)))
```

3.30.2.10 gamma_tot

```
setup_adve.gamma_tot = np.zeros((len(E), len(X)))
```

3.30.2.11 lm

```
setup_adve.lm = np.zeros((len(E), len(X)))
```

3.30.2.12 lp

```
setup_adve.lp = np.zeros((len(E), len(X)))
```

3.30.2.13 ism_values

```
setup_adve.ism_values = nml.ism_values
```

3.30.2.14 mi

```
setup_adve.mi = ism_values.get("mi")
```

3.30.2.15 mn

```
setup_adve.mn = ism_values.get("mn")
```

3.30.2.16 ne

```
setup_adve.ne = nml.NE
```

3.30.2.17 ni

```
setup_adve.ni = ism_values.get("ni")
```

3.30.2.18 nn

```
setup_adve.nn = ism_values.get("nn")
```

3.30.2.19 nx

```
setup_adve.nx = nml.NX
```

END : INITIAL ISM CONDITIONS #.

WRITE THE INITAL CONDITIONS #

3.30.2.20 path

```
setup_adve.path
```

3.30.2.21 Pcr

```
setup_adve.Pcr = np.zeros((len(E), len(X)))
```

3.30.2.22 Pe

```
setup_adve.Pe = np.zeros((len(E), len(X)))
```

3.30.2.23 T

```
setup_adve.T = ism_values.get("T")
```

3.30.2.24 va

```
setup_adve.va = ism_values.get("VA")
```

3.30.2.25 VA

```
setup_adve.VA = np.zeros((len(E), len(X)))
```

3.30.2.26 variable

```
setup_adve.variable
```

3.30.2.27 variables

```
setup_adve.variables
```

Initial value:

```
1 = {"NX"      : nx,
2    "NE"      : ne,
3    "ni"      : ni[x_center_index],
4    "X"       : Xi[x_center_index],
5    "mn"      : mn[x_center_index],
6    "T"       : T[x_center_index],
7    "center"  : x_center,
8    "center_index": x_center_index,
9    "B"       : B[x_center_index]}
```

3.30.2.28 X

```
setup_adve.X = nml.X
```

3.30.2.29 x_center

```
setup_adve.x_center = nml.box_center
```

3.30.2.30 x_center_index

```
setup_adve.x_center_index = int(x_center/(X[1] - X[0]))
```

3.30.2.31 Xi

```
setup_adve.Xi = ism_values.get("X")
```

3.31 setup_advst Namespace Reference

Functions

- def `door` (X, X1, X2, V)
- def `spec` (E, q)

Variables

- `X` = nml.X
- `E` = nml.E
- `nx` = nml.NX
- *END : INITIAL ISM CONDITIONS #.*
- `ne` = nml.NE
- `x_center` = nml.box_center
- `x_center_index` = int(x_center/(X[1] - X[0]))
- `ism_values` = nml.ism_values
- `B` = ism_values.get("B")
- `nn` = ism_values.get("nn")
- `ni` = ism_values.get("ni")
- `mn` = ism_values.get("mn")
- `mi` = ism_values.get("mi")
- `T` = ism_values.get("T")
- `Xi` = ism_values.get("X")
- `va` = ism_values.get("VA")
- `d00` = np.zeros(len(X))
- *INITIAL ISM CONDITIONS #.*
- `D` = np.zeros((len(E), len(X)))
- `Db` = np.zeros((len(E), len(X)))
- `lp` = np.zeros((len(E), len(X)))
- `lm` = np.zeros((len(E), len(X)))
- `VA` = np.zeros((len(E), len(X)))
- `gamma_in` = np.zeros((len(E), len(X)))
- `gamma_lazarian` = np.zeros((len(E), len(X)))
- `gamma_nlld` = np.zeros((len(E), len(X)))
- `gamma_tot` = np.zeros((len(E), len(X)))
- `Pcr` = np.zeros((len(E), len(X)))
- `Pe` = np.zeros((len(E), len(X)))
- `variable`
- `path`
- dictionary `variables`
- `ext`

3.31.1 Function Documentation

3.31.1.1 door()

```
def setup_advst.door (
    X,
    X1,
    X2,
    V )
```

3.31.1.2 spec()

```
def setup_advst.spec (
    E,
    q )
```

3.31.2 Variable Documentation

3.31.2.1 B

```
setup_advst.B = ism_values.get("B")
```

3.31.2.2 D

```
setup_advst.D = np.zeros((len(E), len(X)))
```

3.31.2.3 d00

```
setup_advst.d00 = np.zeros(len(X))
```

INITIAL ISM CONDITIONS #.

You can modify this part #

3.31.2.4 Db

```
setup_advst.Db = np.zeros((len(E), len(X)))
```

3.31.2.5 E

```
setup_advst.E = nml.E
```

3.31.2.6 ext

```
setup_advst.ext
```

3.31.2.7 gamma_in

```
setup_advst.gamma_in = np.zeros((len(E), len(X)))
```

3.31.2.8 gamma_lazarian

```
setup_advst.gamma_lazarian = np.zeros((len(E), len(X)))
```

3.31.2.9 gamma_nlld

```
setup_advst.gamma_nlld = np.zeros((len(E), len(X)))
```

3.31.2.10 gamma_tot

```
setup_advst.gamma_tot = np.zeros((len(E), len(X)))
```

3.31.2.11 Im

```
setup_advst.Im = np.zeros((len(E), len(X)))
```

3.31.2.12 lp

```
setup_advst.Ip = np.zeros((len(E), len(X)))
```

3.31.2.13 ism_values

```
setup_advst.ism_values = nml.ism_values
```

3.31.2.14 mi

```
setup_advst.mi = ism_values.get("mi")
```

3.31.2.15 mn

```
setup_advst.mn = ism_values.get("mn")
```

3.31.2.16 ne

```
setup_advst.ne = nml.NE
```

3.31.2.17 ni

```
setup_advst.ni = ism_values.get("ni")
```

3.31.2.18 nn

```
setup_advst.nn = ism_values.get("nn")
```


3.31.2.19 nx

```
setup_advst.nx = nml.NX
```

END : INITIAL ISM CONDITIONS #.

WRITE THE INITAL CONDITIONS #

3.31.2.20 path

```
setup_advst.path
```

3.31.2.21 Pcr

```
setup_advst.Pcr = np.zeros((len(E), len(X)))
```

3.31.2.22 Pe

```
setup_advst.Pe = np.zeros((len(E), len(X)))
```

3.31.2.23 T

```
setup_advst.T = ism_values.get("T")
```

3.31.2.24 va

```
setup_advst.va = ism_values.get("VA")
```

3.31.2.25 VA

```
setup_advst.VA = np.zeros((len(E), len(X)))
```

3.31.2.26 variable

```
setup_advst.variable
```

3.31.2.27 variables

```
setup_advst.variables
```

Initial value:

```
1 = {"NX"      : nx,
2    "NE"      : ne,
3    "ni"      : ni[x_center_index],
4    "X"       : Xi[x_center_index],
5    "mn"      : mn[x_center_index],
6    "T"       : T[x_center_index],
7    "center"  : x_center,
8    "center_index": x_center_index,
9    "B"       : B[x_center_index]}
```

3.31.2.28 X

```
setup_advst.X = nml.X
```

3.31.2.29 x_center

```
setup_advst.x_center = nml.box_center
```

3.31.2.30 x_center_index

```
setup_advst.x_center_index = int(x\_center/(X[1] - X[0]))
```

3.31.2.31 Xi

```
setup_advst.Xi = ism_values.get("X")
```

3.32 setup_diff Namespace Reference

Functions

- def `door` (`X`, `X1`, `X2`, `V`)

Variables

- `X` = `nml.X`
- `E` = `nml.E`
- `nx` = `nml.NX`
- *END : INITIAL ISM CONDITIONS #.*
- `ne` = `nml.NE`
- `x_center` = `nml.box_center`
- `x_center_index` = `int(x_center/(X[1] - X[0]))`
- `ism_values` = `nml.ism_values`
- `B` = `ism_values.get("B")`
- `nn` = `ism_values.get("nn")`
- `ni` = `ism_values.get("ni")`
- `mn` = `ism_values.get("mn")`
- `mi` = `ism_values.get("mi")`
- `T` = `ism_values.get("T")`
- `Xi` = `ism_values.get("X")`
- `va` = `ism_values.get("VA")`
- `d00` = `np.zeros(len(X))`
- *INITIAL ISM CONDITIONS #.*
- `D` = `np.zeros((len(E), len(X)))`
- `Db` = `np.zeros((len(E), len(X)))`
- `lp` = `np.zeros((len(E), len(X)))`
- `lm` = `np.zeros((len(E), len(X)))`
- `VA` = `np.zeros((len(E), len(X)))`
- `gamma_in` = `np.zeros((len(E), len(X)))`
- `gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `gamma_nlld` = `np.zeros((len(E), len(X)))`
- `gamma_tot` = `np.zeros((len(E), len(X)))`
- `Pcr` = `np.zeros((len(E), len(X)))`
- `Pe` = `np.zeros((len(E), len(X)))`
- `variable`
- `path`
- dictionary `variables`
- `ext`

3.32.1 Function Documentation

3.32.1.1 `door()`

```
def setup_diff.door (
    X,
    X1,
    X2,
    V )
```

3.32.2 Variable Documentation

3.32.2.1 B

```
setup_diff.B = ism_values.get("B")
```

3.32.2.2 D

```
setup_diff.D = np.zeros((len(E), len(X)))
```

3.32.2.3 d00

```
setup_diff.d00 = np.zeros(len(X))
```

INITIAL ISM CONDITIONS #.

You can modify this part #

3.32.2.4 Db

```
setup_diff.Db = np.zeros((len(E), len(X)))
```

3.32.2.5 E

```
setup_diff.E = nml.E
```

3.32.2.6 ext

```
setup_diff.ext
```

3.32.2.7 gamma_in

```
setup_diff.gamma_in = np.zeros((len(E), len(X)))
```

3.32.2.8 gamma_lazarian

```
setup_diff.gamma_lazarian = np.zeros((len(E), len(X)))
```

3.32.2.9 gamma_nlld

```
setup_diff.gamma_nlld = np.zeros((len(E), len(X)))
```

3.32.2.10 gamma_tot

```
setup_diff.gamma_tot = np.zeros((len(E), len(X)))
```

3.32.2.11 lm

```
setup_diff.lm = np.zeros((len(E), len(X)))
```

3.32.2.12 lp

```
setup_diff.lp = np.zeros((len(E), len(X)))
```

3.32.2.13 ism_values

```
setup_diff.ism_values = nml.ism_values
```

3.32.2.14 mi

```
setup_diff.mi = ism_values.get("mi")
```

3.32.2.15 mn

```
setup_diff.mn = ism_values.get("mn")
```

3.32.2.16 ne

```
setup_diff.ne = nml.NE
```

3.32.2.17 ni

```
setup_diff.ni = ism_values.get("ni")
```

3.32.2.18 nn

```
setup_diff.nn = ism_values.get("nn")
```

3.32.2.19 nx

```
setup_diff.nx = nml.NX
```

END : INITIAL ISM CONDITIONS #.

WRITE THE INITAL CONDITIONS #

3.32.2.20 path

```
setup_diff.path
```

3.32.2.21 Pcr

```
setup_diff.Pcr = np.zeros((len(E), len(X)))
```

3.32.2.22 Pe

```
setup_diff.Pe = np.zeros((len(E), len(X)))
```

3.32.2.23 T

```
setup_diff.T = ism_values.get("T")
```

3.32.2.24 va

```
setup_diff.va = ism_values.get("VA")
```

3.32.2.25 VA

```
setup_diff.VA = np.zeros((len(E), len(X)))
```

3.32.2.26 variable

```
setup_diff.variable
```

3.32.2.27 variables

```
setup_diff.variables
```

Initial value:

```
1 = {"NX"      : nx,  
2    "NE"      : ne,  
3    "ni"      : ni[x_center_index],  
4    "X"       : Xi[x_center_index],  
5    "mn"      : mn[x_center_index],  
6    "T"       : T[x_center_index],  
7    "center"  : x_center,  
8    "center_index": x_center_index,  
9    "B"       : B[x_center_index]}
```

3.32.2.28 X

```
setup_diff.X = nml.X
```

3.32.2.29 `x_center`

```
setup_diff.x_center = nml.box_center
```

3.32.2.30 `x_center_index`

```
setup_diff.x_center_index = int(x_center/(X[1] - X[0]))
```

3.32.2.31 `Xi`

```
setup_diff.Xi = ism_values.get("X")
```

3.33 `setup_uniform` Namespace Reference

Variables

- `X` = `nml.X`
- `E` = `nml.E`
- `nx` = `nml.NX`
- *END : INITIAL ISM CONDITIONS #.*
- `ne` = `nml.NE`
- `x_center` = `nml.box_center`
- `x_center_index` = `int(x_center/(X[1] - X[0]))`
- `ism_values` = `nml.ism_values`
- `B` = `ism_values.get("B")`
- `nn` = `ism_values.get("nn")`
- `ni` = `ism_values.get("ni")`
- `mn` = `ism_values.get("mn")`
- `mi` = `ism_values.get("mi")`
- `T` = `ism_values.get("T")`
- `Xi` = `ism_values.get("X")`
- `va` = `ism_values.get("VA")`
- `g_in` = `ism_values.get("gamma_in")`
- `g_lz` = `ism_values.get("gamma_lz")`
- `d00` = `np.zeros(len(X))`
- *INITIAL ISM CONDITIONS #.*
- `D` = `np.zeros((len(E), len(X)))`
- `Db` = `np.zeros((len(E), len(X)))`
- `lp` = `np.zeros((len(E), len(X)))`
- `lm` = `np.zeros((len(E), len(X)))`
- `VA` = `np.zeros((len(E), len(X)))`
- `gamma_in` = `np.zeros((len(E), len(X)))`
- `gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `gamma_nld` = `np.zeros((len(E), len(X)))`
- `gamma_tot` = `np.zeros((len(E), len(X)))`

- `Pcr` = `np.zeros((len(E), len(X)))`
- `Pe` = `np.zeros((len(E), len(X)))`
- dictionary `medium_props`
- `mass`
- `kmin`
- `q`
- `l`
- `in_damping` = `dp.indamping_alfven(xi, E[e], ism_values)`
- `variable`
- `path`
- dictionary `variables`
- `ext`

3.33.1 Variable Documentation

3.33.1.1 B

```
setup_uniform.B = ism_values.get("B")
```

3.33.1.2 D

```
setup_uniform.D = np.zeros((len(E), len(X)))
```

3.33.1.3 d00

```
setup_uniform.d00 = np.zeros(len(X))
```

INITIAL ISM CONDITIONS #.

You can modify this part #

3.33.1.4 Db

```
setup_uniform.Db = np.zeros((len(E), len(X)))
```

3.33.1.5 E

```
setup_uniform.E = nml.E
```

3.33.1.6 ext

```
setup_uniform.ext
```

3.33.1.7 g_in

```
setup_uniform.g_in = ism_values.get("gamma_in")
```

3.33.1.8 g_lz

```
setup_uniform.g_lz = ism_values.get("gamma_lz")
```

3.33.1.9 gamma_in

```
setup_uniform.gamma_in = np.zeros((len(E), len(X)))
```

3.33.1.10 gamma_lazarian

```
setup_uniform.gamma_lazarian = np.zeros((len(E), len(X)))
```

3.33.1.11 gamma_nlld

```
setup_uniform.gamma_nlld = np.zeros((len(E), len(X)))
```

3.33.1.12 gamma_tot

```
setup_uniform.gamma_tot = np.zeros((len(E), len(X)))
```

3.33.1.13 I

```
setup_uniform.I
```

3.33.1.14 Im

```
setup_uniform.Im = np.zeros((len(E), len(X)))
```

3.33.1.15 in_damping

```
setup_uniform.in_damping = dp.indamping_alfven(xi , E[e], ism_values)
```

3.33.1.16 Ip

```
setup_uniform.Ip = np.zeros((len(E), len(X)))
```

3.33.1.17 ism_values

```
setup_uniform.ism_values = nml.ism_values
```

3.33.1.18 kmin

```
setup_uniform.kmin
```

3.33.1.19 mass

```
setup_uniform.mass
```

3.33.1.20 medium_props

```
dictionary setup_uniform.medium_props
```

Initial value:

```
1 = {"B" : B[xi],
2     "mn" : mn[xi],
3     "nn" : nn[xi],
4     "mi" : mi[xi],
5     "ni" : ni[xi],
6     "X" : Xi[xi],
7     "T" : T[xi]}
```

3.33.1.21 mi

```
setup_uniform.mi = ism_values.get("mi")
```

3.33.1.22 mn

```
setup_uniform.mn = ism_values.get("mn")
```

3.33.1.23 ne

```
setup_uniform.ne = nml.NE
```

3.33.1.24 ni

```
setup_uniform.ni = ism_values.get("ni")
```

3.33.1.25 nn

```
setup_uniform.nn = ism_values.get("nn")
```

3.33.1.26 nx

```
setup_uniform.nx = nml.NX
```

END : INITIAL ISM CONDITIONS #.

WRITE THE INITAL CONDITIONS #

3.33.1.27 path

```
setup_uniform.path
```

3.33.1.28 Pcr

```
setup_uniform.Pcr = np.zeros((len(E), len(X)))
```

3.33.1.29 Pe

```
setup_uniform.Pe = np.zeros((len(E), len(X)))
```

3.33.1.30 q

```
setup_uniform.q
```

3.33.1.31 T

```
setup_uniform.T = ism_values.get("T")
```

3.33.1.32 va

```
setup_uniform.va = ism_values.get("VA")
```

3.33.1.33 VA

```
setup_uniform.VA = np.zeros((len(E), len(X)))
```

3.33.1.34 variable

```
setup_uniform.variable
```

3.33.1.35 variables

setup_uniform.variables

Initial value:

```

1 = {"NX"      : nx,
2    "NE"      : ne,
3    "ni"      : ni[x_center_index],
4    "X"       : Xi[x_center_index],
5    "mn"      : mn[x_center_index],
6    "T"       : T[x_center_index],
7    "center"  : x_center,
8    "center_index": x_center_index,
9    "B"       : B[x_center_index]}
```

3.33.1.36 X

setup_uniform.X = nml.X

3.33.1.37 x_center

setup_uniform.x_center = nml.box_center

3.33.1.38 x_center_index

setup_uniform.x_center_index = int([x_center](#)/(X[1] - X[0]))

3.33.1.39 Xi

setup_uniform.Xi = ism_values.get("X")

3.34 setup_vdiff Namespace Reference

Functions

- def [door](#) (X, X1, X2, V)
- def [f](#) (X, X1=900.*[cst.pc](#), X2=1100.*[cst.pc](#), sig=20.*[cst.pc](#), R=1e-3)

Variables

- `X` = `nml.X`
- `E` = `nml.E`
- `nx` = `nml.NX`
- `END : INITIAL ISM CONDITIONS #.`
- `ne` = `nml.NE`
- `x_center` = `nml.box_center`
- `x_center_index` = `int(x_center/(X[1] - X[0]))`
- `ism_values` = `nml.ism_values`
- `B` = `ism_values.get("B")`
- `nn` = `ism_values.get("nn")`
- `ni` = `ism_values.get("ni")`
- `mn` = `ism_values.get("mn")`
- `mi` = `ism_values.get("mi")`
- `T` = `ism_values.get("T")`
- `Xi` = `ism_values.get("X")`
- `va` = `ism_values.get("VA")`
- `d00` = `np.zeros(len(X))`
- `INITIAL ISM CONDITIONS #.`
- `D` = `np.zeros((len(E), len(X)))`
- `Db` = `np.zeros((len(E), len(X)))`
- `lp` = `np.zeros((len(E), len(X)))`
- `lm` = `np.zeros((len(E), len(X)))`
- `VA` = `np.zeros((len(E), len(X)))`
- `gamma_in` = `np.zeros((len(E), len(X)))`
- `gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `gamma_nlld` = `np.zeros((len(E), len(X)))`
- `gamma_tot` = `np.zeros((len(E), len(X)))`
- `Pcr` = `np.zeros((len(E), len(X)))`
- `Pe` = `np.zeros((len(E), len(X)))`
- `variable`
- `path`
- dictionary `variables`
- `ext`

3.34.1 Function Documentation

3.34.1.1 `door()`

```
def setup_vdiff.door (
    X,
    X1,
    X2,
    V )
```

3.34.1.2 f()

```
def setup_vdiff.f (
    X,
    X1 = 900.*cst.pc,
    X2 = 1100.*cst.pc,
    sig = 20.*cst.pc,
    R = 1e-3 )
```

3.34.2 Variable Documentation

3.34.2.1 B

```
setup_vdiff.B = ism_values.get("B")
```

3.34.2.2 D

```
setup_vdiff.D = np.zeros((len(E), len(X)))
```

3.34.2.3 d00

```
setup_vdiff.d00 = np.zeros(len(X))
```

INITIAL ISM CONDITIONS #.

You can modify this part #

3.34.2.4 Db

```
setup_vdiff.Db = np.zeros((len(E), len(X)))
```

3.34.2.5 E

```
setup_vdiff.E = nml.E
```


3.34.2.6 ext

```
setup_vdiff.ext
```

3.34.2.7 gamma_in

```
setup_vdiff.gamma_in = np.zeros((len(E), len(X)))
```

3.34.2.8 gamma_lazarian

```
setup_vdiff.gamma_lazarian = np.zeros((len(E), len(X)))
```

3.34.2.9 gamma_nlld

```
setup_vdiff.gamma_nlld = np.zeros((len(E), len(X)))
```

3.34.2.10 gamma_tot

```
setup_vdiff.gamma_tot = np.zeros((len(E), len(X)))
```

3.34.2.11 Im

```
setup_vdiff.Im = np.zeros((len(E), len(X)))
```

3.34.2.12 Ip

```
setup_vdiff.Ip = np.zeros((len(E), len(X)))
```

3.34.2.13 ism_values

```
setup_vdiff.ism_values = nml.ism_values
```

3.34.2.14 mi

```
setup_vdiff.mi = ism_values.get("mi")
```

3.34.2.15 mn

```
setup_vdiff.mn = ism_values.get("mn")
```

3.34.2.16 ne

```
setup_vdiff.ne = nml.NE
```

3.34.2.17 ni

```
setup_vdiff.ni = ism_values.get("ni")
```

3.34.2.18 nn

```
setup_vdiff.nn = ism_values.get("nn")
```

3.34.2.19 nx

```
setup_vdiff.nx = nml.NX
```

END : INITIAL ISM CONDITIONS #.

WRITE THE INITAL CONDITIONS #

3.34.2.20 path

```
setup_vdiff.path
```

3.34.2.21 Pcr

```
setup_vdiff.Pcr = np.zeros((len(E), len(X)))
```

3.34.2.22 Pe

```
setup_vdiff.Pe = np.zeros((len(E), len(X)))
```

3.34.2.23 T

```
setup_vdiff.T = ism_values.get("T")
```

3.34.2.24 va

```
setup_vdiff.va = ism_values.get("VA")
```

3.34.2.25 VA

```
setup_vdiff.VA = np.zeros((len(E), len(X)))
```

3.34.2.26 variable

```
setup_vdiff.variable
```

3.34.2.27 variables

```
setup_vdiff.variables
```

Initial value:

```
1 = {"NX"      : nx,  
2     "NE"      : ne,  
3     "ni"      : ni[x_center_index],  
4     "X"       : Xi[x_center_index],  
5     "mn"      : mn[x_center_index],  
6     "T"       : T[x_center_index],  
7     "center"   : x_center,  
8     "center_index": x_center_index,  
9     "B"       : B[x_center_index]}
```

3.34.2.28 X

```
setup_vdiff.X = nml.X
```

3.34.2.29 x_center

```
setup_vdiff.x_center = nml.box_center
```

3.34.2.30 x_center_index

```
setup_vdiff.x_center_index = int(x_center/(X[1] - X[0]))
```

3.34.2.31 Xi

```
setup_vdiff.Xi = ism_values.get("X")
```

3.35 show_data Namespace Reference**Functions**

- def [colorFader](#) (c1, c2, mix=0)
- def [colorArray](#) (c1, c2, n=10)
- def [readDataXE](#) (file_name, NX, NE)
DATA READING ROUTINES #.
- def [readInfo](#) (file_number)
- def [readAxis](#) (file_name)
- def [getData](#) (var, file_number)
- def [show](#) (variable, time, position, energy, xlim=None, [elim](#)=None, [vlim](#)=None, [source_center](#)=0, [fig_↔](#)
[save](#)=False, info=False)

Variables

- [elim](#)
SHOW DATA #.
- [vlim](#)
- [source_center](#)
- [fig_save](#)

3.35.1 Function Documentation

3.35.1.1 colorArray()

```
def show_data.colorArray (
    c1,
    c2,
    n = 10 )
```

3.35.1.2 colorFader()

```
def show_data.colorFader (
    c1,
    c2,
    mix = 0 )
```

3.35.1.3 getData()

```
def show_data.getData (
    var,
    file_number )
```

3.35.1.4 readAxis()

```
def show_data.readAxis (
    file_name )
```

3.35.1.5 readDataXE()

```
def show_data.readDataXE (
    file_name,
    NX,
    NE )
```

DATA READING ROUTINES #.

!!! DO NOT MODIFY !!! #

3.35.1.6 readInfo()

```
def show_data.readInfo (
    file_number )
```

3.35.1.7 show()

```
def show_data.show (
    variable,
    time,
    position,
    energy,
    xlim = None,
    elim = None,
    vlim = None,
    source_center = 0,
    fig_save = False,
    info = False )
```

3.35.2 Variable Documentation

3.35.2.1 elim

`show_data.elim`

SHOW DATA #.

3.35.2.2 fig_save

`show_data.fig_save`

3.35.2.3 source_center

`show_data.source_center`

3.35.2.4 vlim

`show_data.vlim`

3.36 ShowInjectionEvolution Namespace Reference

Functions

- def [readDataXE](#) (file_name, NX, NE)
- def [readAxis](#) (file_name)

Variables

- [index](#) = np.logspace(1, np.log10(900), 10)
- [figsize](#)
- def [X](#) = [readAxis](#)("../data_ini/X.dat")
- def [E](#) = [readAxis](#)("../data_ini/E.dat")
- string [loc_id](#) = ""
- def [data](#) = [readDataXE](#)("../data_out/Pcr_"+loc_id+".dat", 2**11, 2**5)

3.36.1 Function Documentation

3.36.1.1 [readAxis\(\)](#)

```
def ShowInjectionEvolution.readAxis (  
    file_name )
```

3.36.1.2 [readDataXE\(\)](#)

```
def ShowInjectionEvolution.readDataXE (  
    file_name,  
    NX,  
    NE )
```

3.36.2 Variable Documentation

3.36.2.1 [data](#)

```
def ShowInjectionEvolution.data = readDataXE("../data_out/Pcr_"+loc_id+".dat", 2**11, 2**5)
```

3.36.2.2 E

```
def ShowInjectionEvolution.E = readAxis("../data_ini/E.dat")
```

3.36.2.3 figsize

```
ShowInjectionEvolution.figsize
```

3.36.2.4 index

```
ShowInjectionEvolution.index = np.logspace(1, np.log10(900), 10)
```

3.36.2.5 loc_id

```
string ShowInjectionEvolution.loc_id = ""
```

3.36.2.6 X

```
def ShowInjectionEvolution.X = readAxis("../data_ini/X.dat")
```

3.37 SNR_evolution Namespace Reference

Functions

- def [InverseTrigonalMatrix](#) (T)
FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- def [ProductMatrix](#) (A, B)
- def [InterpolatingSpline](#) (X, Y)
- def [Rsh](#) (nt)

Variables

- list `marker` = ['X', 'o', 's', 'v']
- int `size_x` = 6
- int `size_y` = 4
- int `sub_x` = 1
- int `sub_y` = 2
- `fig` = plt.figure(figsize=(`size_x*sub_x`,`size_y*sub_y`))
- `gs` = gridspec.GridSpec(ncols= `sub_x`, nrows = `sub_y`, figure = `fig`)
- `wspace`
- `hspace`
- `ax0` = fig.add_subplot(`gs`[0])
- `pc`
- `c`
- `label`
- `lw`
- `loc`
- `ncol`
- `bbox_to_anchor`
- `ax1` = fig.add_subplot(`gs`[1])
- `kms`

3.37.1 Function Documentation

3.37.1.1 InterpolatingSpline()

```
def SNR_evolution.InterpolatingSpline (
    X,
    Y )
```

3.37.1.2 InverseTrigonalMatrix()

```
def SNR_evolution.InverseTrigonalMatrix (
    T )
```

FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.

TriDiagonal matrix inversion function

3.37.1.3 ProductMatrix()

```
def SNR_evolution.ProductMatrix (
    A,
    B )
```

3.37.1.4 Rsh()

```
def SNR_evolution.Rsh (
    nt )
```

3.37.2 Variable Documentation

3.37.2.1 ax0

```
SNR_evolution.ax0 = fig.add_subplot(gs[0])
```

3.37.2.2 ax1

```
SNR_evolution.ax1 = fig.add_subplot(gs[1])
```

3.37.2.3 bbox_to_anchor

```
SNR_evolution.bbox_to_anchor
```

3.37.2.4 c

```
SNR_evolution.c
```

3.37.2.5 fig

```
SNR_evolution.fig = plt.figure(figsize=(size_x*sub_x, size_y*sub_y))
```

3.37.2.6 gs

```
SNR_evolution.gs = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig )
```

3.37.2.7 hspace

SNR_evolution.hspace

3.37.2.8 kms

SNR_evolution.kms

3.37.2.9 label

SNR_evolution.label

3.37.2.10 loc

SNR_evolution.loc

3.37.2.11 lw

SNR_evolution.lw

3.37.2.12 marker

SNR_evolution.marker = ['X', 'o', 's', 'v']

3.37.2.13 ncol

SNR_evolution.ncol

3.37.2.14 pc

SNR_evolution.pc

3.37.2.15 size_x

```
int SNR_evolution.size_x = 6
```

3.37.2.16 size_y

```
int SNR_evolution.size_y = 4
```

3.37.2.17 sub_x

```
int SNR_evolution.sub_x = 1
```

3.37.2.18 sub_y

```
int SNR_evolution.sub_y = 2
```

3.37.2.19 wspace

```
SNR_evolution.wspace
```

3.38 Split_solvers Namespace Reference

Functions

- def [TDMA](#) (a, b, c, d)
- def [generalized_diffusion](#) (u, X, D, dt, theta=0.5)

Variables

- int [NX](#) = 1000
- int [Xmin](#) = -1
- int [Xmax](#) = 1
- [X](#) = np.linspace([Xmin](#), [Xmax](#), [NX](#)+1)
- int [D](#) = np.ones(len([X](#)))*1e-2
- int [u](#) = np.ones(len([X](#)))*0.
- [c](#)
- int [t_ini](#) = 0.
- int [dt](#) = 1e-1
- int [t_max](#) = 1.
- int [t](#) = [t_ini](#)
- int [u_new_0](#) = [u](#)
- int [u_old](#) = [u_new_0](#)
- int [u_new_1](#) = [u](#)

3.38.1 Function Documentation

3.38.1.1 generalized_diffusion()

```
def Split_solvers.generalized_diffusion (
    u,
    X,
    D,
    dt,
    theta = 0.5 )
```

3.38.1.2 TDMA()

```
def Split_solvers.TDMA (
    a,
    b,
    c,
    d )
```

3.38.2 Variable Documentation

3.38.2.1 c

```
Split_solvers.c
```

3.38.2.2 D

```
int Split_solvers.D = np.ones(len(X))*1e-2
```

3.38.2.3 dt

```
int Split_solvers.dt = 1e-1
```

3.38.2.4 NX

```
int Split_solvers.NX = 1000
```

3.38.2.5 t

```
int Split_solvers.t = t\_ini
```

3.38.2.6 t_ini

```
int Split_solvers.t_ini = 0.
```

3.38.2.7 t_max

```
int Split_solvers.t_max = 1.
```

3.38.2.8 u

```
Split_solvers.u = np.ones(len(X))*0.
```

3.38.2.9 u_new_0

```
def Split_solvers.u_new_0 = u
```

3.38.2.10 u_new_1

```
def Split_solvers.u_new_1 = u
```

3.38.2.11 u_old

```
int Split_solvers.u_old = u\_new\_0
```

3.38.2.12 X

```
Split_solvers.X = np.linspace(Xmin, Xmax, NX+1)
```

3.38.2.13 Xmax

```
int Split_solvers.Xmax = 1
```

3.38.2.14 Xmin

```
int Split_solvers.Xmin = -1
```

3.39 test_tesc Namespace Reference

Functions

- def `tesc` (E)
- def `gauss` (t, sig, mu)
- def `sig` (t)

Variables

- float `GeV` = 0.00160218
- int `kyr` = 1e3*24*60*60*365.25
- float `rho_0` = 0.35
- float `e` = 4.8032e-10
- float `c` = 2.998e10
- float `xhi_cr` = 0.1
- float `xhi_0` = 2.026
- float `beta` = 0.2
- int `Esn` = 1e51
- float `Emin` = 0.1*GeV
- `E` = np.logspace(np.log10(10.*GeV), np.log10(1e3*GeV), 100)
- `t` = np.linspace(0.01*kyr, 1e3*kyr, 10000)
- `Qcr` = np.empty((len(E), len(t)))
- `figsize`
- `label`

3.39.1 Function Documentation

3.39.1.1 gauss()

```
def test_tesc.gauss (
    t,
    sig,
    mu )
```

3.39.1.2 sig()

```
def test_tesc.sig (
    t )
```

3.39.1.3 tesc()

```
def test_tesc.tesc (
    E )
```

3.39.2 Variable Documentation

3.39.2.1 beta

```
float test_tesc.beta = 0.2
```

3.39.2.2 c

```
float test_tesc.c = 2.998e10
```

3.39.2.3 e

```
float test_tesc.e = 4.8032e-10
```


3.39.2.4 E

```
test_tesc.E = np.logspace(np.log10(10.*GeV), np.log10(1e3*GeV), 100)
```

3.39.2.5 Emin

```
float test_tesc.Emin = 0.1*GeV
```

3.39.2.6 Esn

```
int test_tesc.Esn = 1e51
```

3.39.2.7 figsize

```
test_tesc.figsize
```

3.39.2.8 GeV

```
float test_tesc.GeV = 0.00160218
```

3.39.2.9 kyr

```
int test_tesc.kyr = 1e3*24*60*60*365.25
```

3.39.2.10 label

```
test_tesc.label
```

3.39.2.11 Qcr

```
test_tesc.Qcr = np.empty((len(E), len(t)))
```

3.39.2.12 rho_0

```
float test_tesc.rho_0 = 0.35
```

3.39.2.13 t

```
test_tesc.t = np.linspace(0.01*kyr, 1e3*kyr, 10000)
```

3.39.2.14 xhi_0

```
float test_tesc.xhi_0 = 2.026
```

3.39.2.15 xhi_cr

```
float test_tesc.xhi_cr = 0.1
```

Chapter 4

File Documentation

4.1 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/data_analysis/pcr_ip_2D.py File Reference ↩↪

Namespaces

- [pcr_ip_2D](#)

Functions

- def [pcr_ip_2D.readDataXE](#) (file_name, NX, NE)
- def [pcr_ip_2D.readAxis](#) (file_name)
- def [pcr_ip_2D.getData](#) (var, file_number)
- def [pcr_ip_2D.getTimeID](#) (time_test, delta_t, t_ini, kind="linear")

Variables

- int [pcr_ip_2D.t_ini](#) = 0.*[cst.kyr](#)
- int [pcr_ip_2D.t_max](#) = 200.*[cst.kyr](#)
- float [pcr_ip_2D.delta_t](#) = 0.1*[cst.kyr](#)
- int [pcr_ip_2D.x_center](#) = 1000.
- float [pcr_ip_2D.time_test](#) = 5.9*[cst.kyr](#)
- def [pcr_ip_2D.out_id](#) = [getTimeID](#)(time_test, delta_t, t_ini)
- [pcr_ip_2D.X](#)
- [pcr_ip_2D.E](#)
- [pcr_ip_2D.Pcr](#)
- [pcr_ip_2D.Ip](#)
- [pcr_ip_2D.EV](#)
- [pcr_ip_2D.XV](#)
- [pcr_ip_2D.sparse](#)
- [pcr_ip_2D.False](#)
- [pcr_ip_2D.indexing](#)
- [pcr_ip_2D.cmap](#) = [plt.get_cmap](#)('jet')
- [pcr_ip_2D.XV_log](#) = [XV/cst.pc](#)
- [pcr_ip_2D.EV_log](#) = [np.log10](#)([EV/cst.GeV](#))
- [pcr_ip_2D.ax0](#)

- [pcr_ip_2D.ax1](#)
- [pcr_ip_2D.ncols](#)
- [pcr_ip_2D.sharey](#)
- [pcr_ip_2D.figsize](#)
- [pcr_ip_2D.im0](#) = `ax0.pcolormesh(XV_log-x_center, EV_log, np.log10(Pcr), cmap=cmap)`
- [pcr_ip_2D.ax](#)
- [pcr_ip_2D.cax](#)
- [pcr_ip_2D.label](#)
- [pcr_ip_2D.im1](#) = `ax1.pcolormesh(XV_log-x_center, EV_log, np.log10(lp), cmap=cmap)`
- [pcr_ip_2D.fontSize](#)

4.2 [/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/data_analysis/show_data.py](#) File Reference

Namespaces

- [show_data](#)

Functions

- `def show_data.colorFader (c1, c2, mix=0)`
- `def show_data.colorArray (c1, c2, n=10)`
- `def show_data.readDataXE (file_name, NX, NE)`
DATA READING ROUTINES #.
- `def show_data.readInfo (file_number)`
- `def show_data.readAxis (file_name)`
- `def show_data.getData (var, file_number)`
- `def show_data.show (variable, time, position, energy, xlim=None, elim=None, vlim=None, source_center=0, fig_save=False, info=False)`

Variables

- [show_data.elim](#)
SHOW DATA #.
- [show_data.vlim](#)
- [show_data.source_center](#)
- [show_data.fig_save](#)

4.3 [/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/namelist.py](#) File Reference

Namespaces

- [namelist](#)

- def [namelist.getVA](#) (E, phase)
- def [namelist.getDamping](#) (E, phase)

Variables

- string [namelist.folder_name](#) = "Test_standard_full"
OUTPUT FOLDER CREATOR #.
- string [namelist.folder_path](#) = "../WorkFolder/"
- string [namelist.total_path](#) = folder_path+folder_name
- int [namelist.NX](#) = 10
GRID PARAMETERS #.
- int [namelist.NE](#) = 7
- int [namelist.Xmin](#) = 0.*[cst.pc](#)
- int [namelist.Xmax](#) = 2000.*[cst.pc](#)
- string [namelist.xgridtype](#) = "cartesian"
- float [namelist.Emin](#) = 0.99*[cst.GeV](#)
- float [namelist.Emax](#) = 50.01*[cst.TeV](#)
- string [namelist.egridtype](#) = "logspace"
- int [namelist.box_center](#) = 1000.*[cst.pc](#)
- [namelist.X](#) = grid.grid(Xmin, Xmax, 2**NX, xgridtype, s_center = box_center)
- [namelist.E](#) = grid.grid([Emin](#), [Emax](#), 2**NE, egridtype)
- bool [namelist.in_damping](#) = True
OTHER TERMS #.
- bool [namelist.lz_damping](#) = True
- bool [namelist.nlld_damping](#) = True
- int [namelist.Pcr_1GeV](#) = 1*[cst.eV](#)
- int [namelist.Pe_1GeV](#) = 1e-2*[cst.eV](#)
- string [namelist.bdiff_model](#) = "ISM_independant"
- list [namelist.phases](#) = []
ISM STRUCTURE #.
- list [namelist.smooth_width_transition](#) = [10.*[cst.pc](#), 3.*[cst.pc](#), 3.*[cst.pc](#), 10.*[cst.pc](#), 10.*[cst.pc](#), 3.*[cst.pc](#), 3.*[cst.pc](#), 10.*[cst.pc](#)]
- [namelist.T](#)
- [namelist.B](#)
- [namelist.ni](#)
- [namelist.nn](#)
- [namelist.nt](#)
- [namelist.Xi](#)
- [namelist.mi](#)
- [namelist.mn](#)
- [namelist.va](#)
- [namelist.gamma_in](#)
- [namelist.gamma_lz](#)
- [namelist.ism_values](#) = dict([T](#)=[T](#), [B](#)=[B](#), [ni](#)=[ni](#), [nn](#)=[nn](#), [nt](#)=[nt](#), [X](#)=[Xi](#), [mi](#)=[mi](#), [mn](#)=[mn](#), [VA](#)=[va](#), [gamma_in](#) = [gamma_in](#), [gamma_lz](#) = [gamma_lz](#))

4.4 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/fiducial_↵ _tests/phases_energy_dependance/namelist_uniform.py File Reference

Namespaces

- [namelist_uniform](#)

Functions

- def `namelist_uniform.getVA` (E, phase)
- def `namelist_uniform.getDamping` (E, phase)

Variables

- string `namelist_uniform.folder_name` = "WNM-CNM-DiM_all_dependant"
OUTPUT FOLDER CREATOR #.
- string `namelist_uniform.folder_path` = "../.../WorkFolder/Fiducial_tests_for_thesis_2/"
- string `namelist_uniform.total_path` = folder_path+folder_name
- int `namelist_uniform.NX` = 12
GRID PARAMETERS #.
- int `namelist_uniform.NE` = 8
- int `namelist_uniform.Xmin` = 0.*cst.pc
- int `namelist_uniform.Xmax` = 2000.*cst.pc
- string `namelist_uniform.xgridtype` = "cartesian"
- float `namelist_uniform.Emin` = 0.99*cst.GeV
- float `namelist_uniform.Emax` = 50.01*cst.TeV
- string `namelist_uniform.egridtype` = "logspace"
- int `namelist_uniform.box_center` = 1000.*cst.pc
- `namelist_uniform.X` = grid.grid(Xmin, Xmax, 2**NX, xgridtype, s_center = box_center)
- `namelist_uniform.E` = grid.grid(Emin, Emax, 2**NE, egridtype)
- bool `namelist_uniform.in_damping` = True
OTHER TERMS #.
- bool `namelist_uniform.lz_damping` = True
- bool `namelist_uniform.nlld_damping` = True
- int `namelist_uniform.Pcr_1GeV` = 1*cst.eV
- int `namelist_uniform.Pe_1GeV` = 1e-2*cst.eV
- string `namelist_uniform.bdiff_model` = "ISM_dependant"
- list `namelist_uniform.phases` = []
ISM STRUCTURE #.
- list `namelist_uniform.smooth_width_transition` = [10.*cst.pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc, 10.*cst.pc, 3.*cst.pc, 3.*cst.pc, 10.*cst.pc]
- `namelist_uniform.T`
- `namelist_uniform.B`
- `namelist_uniform.ni`
- `namelist_uniform.nn`
- `namelist_uniform.nt`
- `namelist_uniform.Xi`
- `namelist_uniform.mi`
- `namelist_uniform.mn`
- `namelist_uniform.va`
- `namelist_uniform.gamma_in`
- `namelist_uniform.gamma_lz`
- `namelist_uniform.ism_values` = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va, gamma_in = gamma_in, gamma_lz = gamma_lz)

4.5 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/fiducial_tests/phases_energy_dependance/setup_uniform.py File Reference

Namespaces

- `setup_uniform`

Variables

- `setup_uniform.X` = `nml.X`
- `setup_uniform.E` = `nml.E`
- `setup_uniform.nx` = `nml.NX`
- *END : INITIAL ISM CONDITIONS #.*
- `setup_uniform.ne` = `nml.NE`
- `setup_uniform.x_center` = `nml.box_center`
- `setup_uniform.x_center_index` = `int(x_center/(X[1] - X[0]))`
- `setup_uniform.ism_values` = `nml.ism_values`
- `setup_uniform.B` = `ism_values.get("B")`
- `setup_uniform.nn` = `ism_values.get("nn")`
- `setup_uniform.ni` = `ism_values.get("ni")`
- `setup_uniform.mn` = `ism_values.get("mn")`
- `setup_uniform.mi` = `ism_values.get("mi")`
- `setup_uniform.T` = `ism_values.get("T")`
- `setup_uniform.Xi` = `ism_values.get("X")`
- `setup_uniform.va` = `ism_values.get("VA")`
- `setup_uniform.g_in` = `ism_values.get("gamma_in")`
- `setup_uniform.g_lz` = `ism_values.get("gamma_lz")`
- `setup_uniform.d00` = `np.zeros(len(X))`
- *INITIAL ISM CONDITIONS #.*
- `setup_uniform.D` = `np.zeros((len(E), len(X)))`
- `setup_uniform.Db` = `np.zeros((len(E), len(X)))`
- `setup_uniform.lp` = `np.zeros((len(E), len(X)))`
- `setup_uniform.lm` = `np.zeros((len(E), len(X)))`
- `setup_uniform.VA` = `np.zeros((len(E), len(X)))`
- `setup_uniform.gamma_in` = `np.zeros((len(E), len(X)))`
- `setup_uniform.gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `setup_uniform.gamma_nlld` = `np.zeros((len(E), len(X)))`
- `setup_uniform.gamma_tot` = `np.zeros((len(E), len(X)))`
- `setup_uniform.Pcr` = `np.zeros((len(E), len(X)))`
- `setup_uniform.Pe` = `np.zeros((len(E), len(X)))`
- dictionary `setup_uniform.medium_props`
- `setup_uniform.mass`
- `setup_uniform.kmin`
- `setup_uniform.q`
- `setup_uniform.l`
- `setup_uniform.in_damping` = `dp.indamping_alfven(xi, E[e], ism_values)`
- `setup_uniform.variable`
- `setup_uniform.path`
- dictionary `setup_uniform.variables`
- `setup_uniform.ext`

4.6 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_adv.py File Reference

Namespaces

- `namelist_adv`

Functions

- def [namelist_adv.getVA](#) (E, phase)

Variables

- string [namelist_adv.folder_name](#) = "advc_z_X12"
OUTPUT FOLDER CREATOR #.
- string [namelist_adv.folder_path](#) = "../WorkFolder/Fiducial_tests_for_thesis/"
- string [namelist_adv.total_path](#) = folder_path+folder_name
- int [namelist_adv.NX](#) = 12
GRID PARAMETERS #.
- int [namelist_adv.NE](#) = 4
- int [namelist_adv.Xmin](#) = 0.*cst.pc
- int [namelist_adv.Xmax](#) = 2000.*cst.pc
- string [namelist_adv.xgridtype](#) = "cartesian"
- int [namelist_adv.Emin](#) = 10.*cst.GeV
- int [namelist_adv.Emax](#) = 10.*cst.TeV
- string [namelist_adv.egridtype](#) = "logspace"
- int [namelist_adv.box_center](#) = 1000.*cst.pc
- [namelist_adv.X](#)
- [namelist_adv.E](#) = grid.grid([Emin](#), [Emax](#), 2**[NE](#), [egridtype](#))
- bool [namelist_adv.in_damping](#) = True
OTHER TERMS #.
- bool [namelist_adv.lz_damping](#) = True
- bool [namelist_adv.nlld_damping](#) = True
- int [namelist_adv.Pcr_1GeV](#) = 1*cst.eV
- int [namelist_adv.Pe_1GeV](#) = 1*cst.eV
- string [namelist_adv.bdiff_model](#) = "ISM_independant"
- list [namelist_adv.phases](#) = []
ISM STRUCTURE #.
- int [namelist_adv.smooth_width_transition](#) = 10.*cst.pc
- [namelist_adv.T](#)
- [namelist_adv.B](#)
- [namelist_adv.ni](#)
- [namelist_adv.nn](#)
- [namelist_adv.nt](#)
- [namelist_adv.Xi](#)
- [namelist_adv.mi](#)
- [namelist_adv.mn](#)
- [namelist_adv.va](#)
- [namelist_adv.ism_values](#) = dict([T](#)=[T](#), [B](#)=[B](#), [ni](#)=[ni](#), [nn](#)=[nn](#), [nt](#)=[nt](#), [X](#)=[Xi](#), [mi](#)=[mi](#), [mn](#)=[mn](#), [VA](#)=[va](#))

4.7 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_↵ _adve.py File Reference

Namespaces

- [namelist_adve](#)

Functions

- def `namelist_adve.getVA` (E, phase)

Variables

- string `namelist_adve.folder_name` = "adv_e_X6"
OUTPUT FOLDER CREATOR #.
- string `namelist_adve.folder_path` = "../WorkFolder/Fiducial_tests_for_thesis/"
- string `namelist_adve.total_path` = folder_path+folder_name
- int `namelist_adve.NX` = 6
GRID PARAMETERS #.
- int `namelist_adve.NE` = 6
- int `namelist_adve.Xmin` = 0.*cst.pc
- int `namelist_adve.Xmax` = 2000.*cst.pc
- string `namelist_adve.xgridtype` = "cartesian"
- int `namelist_adve.Emin` = 10.*cst.GeV
- int `namelist_adve.Emax` = 10.*cst.TeV
- string `namelist_adve.egridtype` = "logspace"
- int `namelist_adve.box_center` = 1000.*cst.pc
- `namelist_adve.X`
- `namelist_adve.E` = grid.grid(Emin, Emax, 2**NE, egridtype)
- bool `namelist_adve.in_damping` = True
OTHER TERMS #.
- bool `namelist_adve.lz_damping` = True
- bool `namelist_adve.nlld_damping` = True
- int `namelist_adve.Pcr_1GeV` = 1*cst.eV
- int `namelist_adve.Pe_1GeV` = 1*cst.eV
- string `namelist_adve.bdiff_model` = "ISM_independant"
- list `namelist_adve.phases` = []
ISM STRUCTURE #.
- int `namelist_adve.smooth_width_transition` = 10.*cst.pc
- `namelist_adve.T`
- `namelist_adve.B`
- `namelist_adve.ni`
- `namelist_adve.nn`
- `namelist_adve.nt`
- `namelist_adve.Xi`
- `namelist_adve.mi`
- `namelist_adve.mn`
- `namelist_adve.va`
- `namelist_adve.ism_values` = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)

4.8 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_advst.py File Reference

Namespaces

- `namelist_advst`

Functions

- def `namelist_advst.getVA` (E, phase)

Variables

- string `namelist_advst.folder_name` = "adv_sss_E6"
OUTPUT FOLDER CREATOR #.
- string `namelist_advst.folder_path` = "../WorkFolder/Fiducial_tests_for_thesis/"
- string `namelist_advst.total_path` = folder_path+folder_name
- int `namelist_advst.NX` = 4
GRID PARAMETERS #.
- int `namelist_advst.NE` = 6
- int `namelist_advst.Xmin` = 0.*cst.pc
- int `namelist_advst.Xmax` = 2000.*cst.pc
- string `namelist_advst.xgridtype` = "cartesian"
- int `namelist_advst.Emin` = 10.*cst.GeV
- int `namelist_advst.Emax` = 10.*cst.TeV
- string `namelist_advst.egridtype` = "logspace"
- int `namelist_advst.box_center` = 1000.*cst.pc
- `namelist_advst.X`
- `namelist_advst.E` = grid.grid(Emin, Emax, 2**NE, egridtype)
- bool `namelist_advst.in_damping` = True
OTHER TERMS #.
- bool `namelist_advst.lz_damping` = True
- bool `namelist_advst.nlld_damping` = True
- int `namelist_advst.Pcr_1GeV` = 1*cst.eV
- int `namelist_advst.Pe_1GeV` = 1*cst.eV
- string `namelist_advst.bdiff_model` = "ISM_independant"
- list `namelist_advst.phases` = []
ISM STRUCTURE #.
- int `namelist_advst.smooth_width_transition` = 10.*cst.pc
- `namelist_advst.T`
- `namelist_advst.B`
- `namelist_advst.ni`
- `namelist_advst.nn`
- `namelist_advst.nt`
- `namelist_advst.Xi`
- `namelist_advst.mi`
- `namelist_advst.mn`
- `namelist_advst.va`
- `namelist_advst.ism_values` = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)

4.9 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/namelist_diff.py File Reference

Namespaces

- `namelist_diff`

Functions

- def [namelist_diff.getVA](#) (E, phase)

Variables

- string [namelist_diff.folder_name](#) = "diff_z_X12"
OUTPUT FOLDER CREATOR #.
- string [namelist_diff.folder_path](#) = ".././WorkFolder/Fiducial_tests_for_thesis/"
- string [namelist_diff.total_path](#) = folder_path+folder_name
- int [namelist_diff.NX](#) = 12
GRID PARAMETERS #.
- int [namelist_diff.NE](#) = 4
- int [namelist_diff.Xmin](#) = 0.*cst.pc
- int [namelist_diff.Xmax](#) = 2000.*cst.pc
- string [namelist_diff.xgridtype](#) = "cartesian"
- int [namelist_diff.Emin](#) = 10.*cst.GeV
- int [namelist_diff.Emax](#) = 10.*cst.TeV
- string [namelist_diff.egridtype](#) = "logspace"
- int [namelist_diff.box_center](#) = 1000.*cst.pc
- [namelist_diff.X](#)
- [namelist_diff.E](#) = grid.grid(Emin, Emax, 2**NE, egridtype)
- bool [namelist_diff.in_damping](#) = True
OTHER TERMS #.
- bool [namelist_diff.lz_damping](#) = True
- bool [namelist_diff.nlld_damping](#) = True
- int [namelist_diff.Pcr_1GeV](#) = 1*cst.eV
- int [namelist_diff.Pe_1GeV](#) = 1*cst.eV
- string [namelist_diff.bdiff_model](#) = "ISM_independant"
- list [namelist_diff.phases](#) = []
ISM STRUCTURE #.
- int [namelist_diff.smooth_width_transition](#) = 10.*cst.pc
- [namelist_diff.T](#)
- [namelist_diff.B](#)
- [namelist_diff.ni](#)
- [namelist_diff.nn](#)
- [namelist_diff.nt](#)
- [namelist_diff.Xi](#)
- [namelist_diff.mi](#)
- [namelist_diff.mn](#)
- [namelist_diff.va](#)
- [namelist_diff.ism_values](#) = dict([T](#)=[T](#), [B](#)=[B](#), [ni](#)=[ni](#), [nn](#)=[nn](#), [nt](#)=[nt](#), [X](#)=[Xi](#), [mi](#)=[mi](#), [mn](#)=[mn](#), [VA](#)=[va](#))

4.10 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_↵ setups/namelist_vdiff.py File Reference

Namespaces

- [namelist_vdiff](#)

Functions

- def [namelist_vdiff.getVA](#) (E, phase)

Variables

- string [namelist_vdiff.folder_name](#) = "vdiff_z_X14"
OUTPUT FOLDER CREATOR #.
- string [namelist_vdiff.folder_path](#) = ".././WorkFolder/Fiducial_tests_for_thesis/"
- string [namelist_vdiff.total_path](#) = folder_path+folder_name
- int [namelist_vdiff.NX](#) = 14
GRID PARAMETERS #.
- int [namelist_vdiff.NE](#) = 4
- int [namelist_vdiff.Xmin](#) = 0.*cst.pc
- int [namelist_vdiff.Xmax](#) = 2000.*cst.pc
- string [namelist_vdiff.xgridtype](#) = "cartesian"
- int [namelist_vdiff.Emin](#) = 10.*cst.GeV
- int [namelist_vdiff.Emax](#) = 10.*cst.TeV
- string [namelist_vdiff.egridtype](#) = "logspace"
- int [namelist_vdiff.box_center](#) = 1000.*cst.pc
- [namelist_vdiff.X](#)
- [namelist_vdiff.E](#) = grid.grid(Emin, Emax, 2**NE, egridtype)
- bool [namelist_vdiff.in_damping](#) = True
OTHER TERMS #.
- bool [namelist_vdiff.lz_damping](#) = True
- bool [namelist_vdiff.nlld_damping](#) = True
- int [namelist_vdiff.Pcr_1GeV](#) = 1*cst.eV
- int [namelist_vdiff.Pe_1GeV](#) = 1*cst.eV
- string [namelist_vdiff.bdiff_model](#) = "ISM_independant"
- list [namelist_vdiff.phases](#) = []
ISM STRUCTURE #.
- int [namelist_vdiff.smooth_width_transition](#) = 10.*cst.pc
- [namelist_vdiff.T](#)
- [namelist_vdiff.B](#)
- [namelist_vdiff.ni](#)
- [namelist_vdiff.nn](#)
- [namelist_vdiff.nt](#)
- [namelist_vdiff.Xi](#)
- [namelist_vdiff.mi](#)
- [namelist_vdiff.mn](#)
- [namelist_vdiff.va](#)
- [namelist_vdiff.ism_values](#) = dict(T=T, B=B, ni=ni, nn=nn, nt=nt, X=Xi, mi=mi, mn=mn, VA=va)

4.11 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_↵ setups/setup_adv.py File Reference

Namespaces

- [setup_adv](#)

Functions

- def `setup_adv.door` (X, X1, X2, V)

Variables

- `setup_adv.X` = nml.X
- `setup_adv.E` = nml.E
- `setup_adv.nx` = nml.NX
- *END : INITIAL ISM CONDITIONS #.*
- `setup_adv.ne` = nml.NE
- `setup_adv.x_center` = nml.box_center
- `setup_adv.x_center_index` = int(`x_center`/(`X[1]` - `X[0]`))
- `setup_adv.ism_values` = nml.ism_values
- `setup_adv.B` = `ism_values.get("B")`
- `setup_adv.nn` = `ism_values.get("nn")`
- `setup_adv.ni` = `ism_values.get("ni")`
- `setup_adv.mn` = `ism_values.get("mn")`
- `setup_adv.mi` = `ism_values.get("mi")`
- `setup_adv.T` = `ism_values.get("T")`
- `setup_adv.Xi` = `ism_values.get("X")`
- `setup_adv.va` = `ism_values.get("VA")`
- `setup_adv.d00` = `np.zeros(len(X))`
- *INITIAL ISM CONDITIONS #.*
- `setup_adv.D` = `np.zeros((len(E), len(X)))`
- `setup_adv.Db` = `np.zeros((len(E), len(X)))`
- `setup_adv.lp` = `np.zeros((len(E), len(X)))`
- `setup_adv.lm` = `np.zeros((len(E), len(X)))`
- `setup_adv.VA` = `np.zeros((len(E), len(X)))`
- `setup_adv.gamma_in` = `np.zeros((len(E), len(X)))`
- `setup_adv.gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `setup_adv.gamma_nlld` = `np.zeros((len(E), len(X)))`
- `setup_adv.gamma_tot` = `np.zeros((len(E), len(X)))`
- `setup_adv.Pcr` = `np.zeros((len(E), len(X)))`
- `setup_adv.Pe` = `np.zeros((len(E), len(X)))`
- `setup_adv.variable`
- `setup_adv.path`
- dictionary `setup_adv.variables`
- `setup_adv.ext`

4.12 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_↵ setups/setup_adve.py File Reference

Namespaces

- `setup_adve`

Functions

- def `setup_adve.door` (X, X1, X2, V)
- def `setup_adve.spec` (E, q)

Variables

- `setup_adve.X` = `nml.X`
- `setup_adve.E` = `nml.E`
- `setup_adve.nx` = `nml.NX`
- *END : INITIAL ISM CONDITIONS #.*
- `setup_adve.ne` = `nml.NE`
- `setup_adve.x_center` = `nml.box_center`
- `setup_adve.x_center_index` = `int(x_center/(X[1] - X[0]))`
- `setup_adve.ism_values` = `nml.ism_values`
- `setup_adve.B` = `ism_values.get("B")`
- `setup_adve.nn` = `ism_values.get("nn")`
- `setup_adve.ni` = `ism_values.get("ni")`
- `setup_adve.mn` = `ism_values.get("mn")`
- `setup_adve.mi` = `ism_values.get("mi")`
- `setup_adve.T` = `ism_values.get("T")`
- `setup_adve.Xi` = `ism_values.get("X")`
- `setup_adve.va` = `ism_values.get("VA")`
- `setup_adve.d00` = `np.zeros(len(X))`
- *INITIAL ISM CONDITIONS #.*
- `setup_adve.D` = `np.zeros((len(E), len(X)))`
- `setup_adve.Db` = `np.zeros((len(E), len(X)))`
- `setup_adve.lp` = `np.zeros((len(E), len(X)))`
- `setup_adve.lm` = `np.zeros((len(E), len(X)))`
- `setup_adve.VA` = `np.zeros((len(E), len(X)))`
- `setup_adve.gamma_in` = `np.zeros((len(E), len(X)))`
- `setup_adve.gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `setup_adve.gamma_nlld` = `np.zeros((len(E), len(X)))`
- `setup_adve.gamma_tot` = `np.zeros((len(E), len(X)))`
- `setup_adve.Pcr` = `np.zeros((len(E), len(X)))`
- `setup_adve.Pe` = `np.zeros((len(E), len(X)))`
- `setup_adve.variable`
- `setup_adve.path`
- dictionary `setup_adve.variables`
- `setup_adve.ext`

4.13 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_↵ setups/setup_advst.py File Reference

Namespaces

- `setup_advst`

Functions

- def `setup_advst.door` (X, X1, X2, V)
- def `setup_advst.spec` (E, q)

Variables

- `setup_advst.X` = `nml.X`
- `setup_advst.E` = `nml.E`
- `setup_advst.nx` = `nml.NX`
- `END : INITIAL ISM CONDITIONS #.`
- `setup_advst.ne` = `nml.NE`
- `setup_advst.x_center` = `nml.box_center`
- `setup_advst.x_center_index` = `int(x_center/(X[1] - X[0]))`
- `setup_advst.ism_values` = `nml.ism_values`
- `setup_advst.B` = `ism_values.get("B")`
- `setup_advst.nn` = `ism_values.get("nn")`
- `setup_advst.ni` = `ism_values.get("ni")`
- `setup_advst.mn` = `ism_values.get("mn")`
- `setup_advst.mi` = `ism_values.get("mi")`
- `setup_advst.T` = `ism_values.get("T")`
- `setup_advst.Xi` = `ism_values.get("X")`
- `setup_advst.va` = `ism_values.get("VA")`
- `setup_advst.d00` = `np.zeros(len(X))`
- `INITIAL ISM CONDITIONS #.`
- `setup_advst.D` = `np.zeros((len(E), len(X)))`
- `setup_advst.Db` = `np.zeros((len(E), len(X)))`
- `setup_advst.lp` = `np.zeros((len(E), len(X)))`
- `setup_advst.lm` = `np.zeros((len(E), len(X)))`
- `setup_advst.VA` = `np.zeros((len(E), len(X)))`
- `setup_advst.gamma_in` = `np.zeros((len(E), len(X)))`
- `setup_advst.gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `setup_advst.gamma_nld` = `np.zeros((len(E), len(X)))`
- `setup_advst.gamma_tot` = `np.zeros((len(E), len(X)))`
- `setup_advst.Pcr` = `np.zeros((len(E), len(X)))`
- `setup_advst.Pe` = `np.zeros((len(E), len(X)))`
- `setup_advst.variable`
- `setup_advst.path`
- dictionary `setup_advst.variables`
- `setup_advst.ext`

4.14 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_setups/setup_diff.py File Reference ↩

Namespaces

- `setup_diff`

Functions

- def `setup_diff.door` (`X`, `X1`, `X2`, `V`)

Variables

- `setup_diff.X` = `nml.X`
- `setup_diff.E` = `nml.E`
- `setup_diff.nx` = `nml.NX`
- *END : INITIAL ISM CONDITIONS #.*
- `setup_diff.ne` = `nml.NE`
- `setup_diff.x_center` = `nml.box_center`
- `setup_diff.x_center_index` = `int(x_center/(X[1] - X[0]))`
- `setup_diff.ism_values` = `nml.ism_values`
- `setup_diff.B` = `ism_values.get("B")`
- `setup_diff.nn` = `ism_values.get("nn")`
- `setup_diff.ni` = `ism_values.get("ni")`
- `setup_diff.mn` = `ism_values.get("mn")`
- `setup_diff.mi` = `ism_values.get("mi")`
- `setup_diff.T` = `ism_values.get("T")`
- `setup_diff.Xi` = `ism_values.get("X")`
- `setup_diff.va` = `ism_values.get("VA")`
- `setup_diff.d00` = `np.zeros(len(X))`
- *INITIAL ISM CONDITIONS #.*
- `setup_diff.D` = `np.zeros((len(E), len(X)))`
- `setup_diff.Db` = `np.zeros((len(E), len(X)))`
- `setup_diff.lp` = `np.zeros((len(E), len(X)))`
- `setup_diff.lm` = `np.zeros((len(E), len(X)))`
- `setup_diff.VA` = `np.zeros((len(E), len(X)))`
- `setup_diff.gamma_in` = `np.zeros((len(E), len(X)))`
- `setup_diff.gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `setup_diff.gamma_nlld` = `np.zeros((len(E), len(X)))`
- `setup_diff.gamma_tot` = `np.zeros((len(E), len(X)))`
- `setup_diff.Pcr` = `np.zeros((len(E), len(X)))`
- `setup_diff.Pe` = `np.zeros((len(E), len(X)))`
- `setup_diff.variable`
- `setup_diff.path`
- dictionary `setup_diff.variables`
- `setup_diff.ext`

4.15 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/other_↵ setups/setup_vdiff.py File Reference

Namespaces

- `setup_vdiff`

Functions

- def `setup_vdiff.door` (X, X1, X2, V)
- def `setup_vdiff.f` (X, X1=900.*`cst.pc`, X2=1100.*`cst.pc`, sig=20.*`cst.pc`, R=1e-3)

Variables

- `setup_vdiff.X` = `nml.X`
- `setup_vdiff.E` = `nml.E`
- `setup_vdiff.nx` = `nml.NX`
- `END : INITIAL ISM CONDITIONS #.`
- `setup_vdiff.ne` = `nml.NE`
- `setup_vdiff.x_center` = `nml.box_center`
- `setup_vdiff.x_center_index` = `int(x_center/(X[1] - X[0]))`
- `setup_vdiff.ism_values` = `nml.ism_values`
- `setup_vdiff.B` = `ism_values.get("B")`
- `setup_vdiff.nn` = `ism_values.get("nn")`
- `setup_vdiff.ni` = `ism_values.get("ni")`
- `setup_vdiff.mn` = `ism_values.get("mn")`
- `setup_vdiff.mi` = `ism_values.get("mi")`
- `setup_vdiff.T` = `ism_values.get("T")`
- `setup_vdiff.Xi` = `ism_values.get("X")`
- `setup_vdiff.va` = `ism_values.get("VA")`
- `setup_vdiff.d00` = `np.zeros(len(X))`
- `INITIAL ISM CONDITIONS #.`
- `setup_vdiff.D` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.Db` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.lp` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.lm` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.VA` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.gamma_in` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.gamma_nlld` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.gamma_tot` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.Pcr` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.Pe` = `np.zeros((len(E), len(X)))`
- `setup_vdiff.variable`
- `setup_vdiff.path`
- dictionary `setup_vdiff.variables`
- `setup_vdiff.ext`

4.16 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/setup.py File Reference

Namespaces

- `setup`

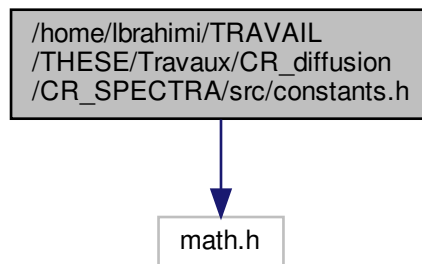
Variables

- `setup.X` = `nml.X`
- `setup.E` = `nml.E`
- `setup.nx` = `nml.NX`
- *END : INITIAL ISM CONDITIONS #.*
- `setup.ne` = `nml.NE`
- `setup.x_center` = `nml.box_center`
- `setup.x_center_index` = `int(x_center/(X[1] - X[0]))`
- `setup.ism_values` = `nml.ism_values`
- `setup.B` = `ism_values.get("B")`
- `setup.nn` = `ism_values.get("nn")`
- `setup.ni` = `ism_values.get("ni")`
- `setup.mn` = `ism_values.get("mn")`
- `setup.mi` = `ism_values.get("mi")`
- `setup.T` = `ism_values.get("T")`
- `setup.Xi` = `ism_values.get("X")`
- `setup.va` = `ism_values.get("VA")`
- `setup.g_in` = `ism_values.get("gamma_in")`
- `setup.g_lz` = `ism_values.get("gamma_lz")`
- `setup.d00` = `np.zeros(len(X))`
- *INITIAL ISM CONDITIONS #.*
- `setup.D` = `np.zeros((len(E), len(X)))`
- `setup.Db` = `np.zeros((len(E), len(X)))`
- `setup.lp` = `np.zeros((len(E), len(X)))`
- `setup.lm` = `np.zeros((len(E), len(X)))`
- `setup.VA` = `np.zeros((len(E), len(X)))`
- `setup.gamma_in` = `np.zeros((len(E), len(X)))`
- `setup.gamma_lazarian` = `np.zeros((len(E), len(X)))`
- `setup.gamma_nlld` = `np.zeros((len(E), len(X)))`
- `setup.gamma_tot` = `np.zeros((len(E), len(X)))`
- `setup.Pcr` = `np.zeros((len(E), len(X)))`
- `setup.Pe` = `np.zeros((len(E), len(X)))`
- dictionary `setup.medium_props`
- `setup.mass`
- `setup.kmin`
- `setup.q`
- `setup.l`
- `setup.in_damping` = `dp.indamping_alfven(xi, E[e], ism_values)`
- `setup.variable`
- `setup.path`
- dictionary `setup.variables`
- `setup.ext`

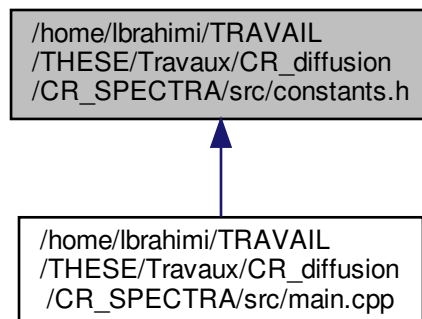
4.17 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/constants.h File Reference

```
#include <math.h>
```

Include dependency graph for constants.h:



This graph shows which files directly or indirectly include this file:



Variables

- const double `pi` = 3.1472
Inclusion of basic mathematic functions.
- const double `yr` = 3.154e+7
- const double `kyr` = 1e3*yr
1 yr in [s]
- const double `km` = 1e5
1 kyr in [s]
- const double `pc` = 3.086e18

- 1 km in [cm]*
- const double **kpc** = 1.e3***pc**
- 1 pc in [cm]*
- const double **GeV** = 0.00160218
- 1 kpc in [cm]*
- const double **TeV** = 1e3*GeV
- 1 GeV in [erg]*
- const double **eV** = **GeV***1e-9
- 1 TeV in [erg]*
- const double **MeV** = 1e-3*GeV
- 1 eV in [erg]*
- const double **mp** = 1.6726219e-24
- 1 MeV in [erg]*
- const double **mn** = 1.6749286e-24
- Proton mass [g].*
- const double **me** = 9.1095e-28
- Neutron mass [g].*
- const double **mH** = **mp**
- Electron mass [g].*
- const double **mHII** = **mp**
- HI mass [g].*
- const double **mHel** = 2***mp** + 2***mn**
- HII mass [g].*
- const double **mCII** = 6***mp** + 6***mn**
- Hel mass [g].*
- const double **mH2** = 2***mp**
- CII mass [g].*
- const double **e** = 4.80326e-10
- H2 mass [g].*
- const double **c** = 29979245800.
- e charge in [statC]*
- const double **kbol** = 1.3807e-16
- light celerity in [cm/s]*
- const double **kms** = 1e5
- Boltzmann constant in CGS.*
- const double **sig_T** = 6.65e-25
- 1 km/s in cm/s*
- const int **solver_PcrAdvection** = 1
- [cm²] Thomson cross section (see Schlickeiser (2002) p.80)*
- const int **solver_PcrDiffusion** = 1
- Advective term of the CR Pressure (the classical one -> V_A*grad ...)*
- const int **solver_PcrAdvection2** = 1
- Diffusive term of the CR Pressure.*
- const int **solver_PcrAdvectionE** = 1
- Explicit Advection solver for Pcr by the energy derivative of Alfvén velocity.*
- const int **solver_PcrSource1** = 1
- Explicit Advection solver for Pcr in energy cdVAdX.*
- const int **solver_PcrSource2** = 1
- Source term effect due to the dependance of the Alfvén velocity to the space.*
- const int **solver_PcrPerpDiff** = 0
- Source term effect due to the CR injection from the source in the system.*

- const int `solver_PeAdvection` = 1
Perpendicular diffusion of CRs after the coherence length.
- const int `solver_PeDiffusion` = 1
Advective term of the e- Pressure (the classical one -> $V_A \cdot \text{grad} \dots$)
- const int `solver_PeAdvection2` = 1
Diffusive term of the e- Pressure.
- const int `solver_PeAdvectionE` = 1
Explicit Advection solver for e- by the energy derivative of Alfvén velocity.
- const int `solver_PeAdvectionE1` = 1
Explicit Advection solver for e- in energy $cdVAdX$.
- const int `solver_PeAdvectionE2` = 1
Synchrotron radiations of e- (looses of energy) - Advection term.
- const int `solver_PeSource1` = 1
Synchrotron radiations of e- (looses of energy) - source term.
- const int `solver_PeSource2` = 1
Source term effect due to the dependance of the Alfvén velocity to the space (for e-)
- const int `solver_PePerpDiff` = 0
Source term effect due to the e- injection from the source in the system.
- const int `solver_lpAdvection` = 1
Perpendicular diffusion of e- after the coherence length.
- const int `solver_lmAdvection` = 1
Advective term of the foward waves.
- const int `solver_lpSource1` = 1
Advective term of the backward waves.
- const int `solver_lmSource1` = 1
Source term effect applied on foward waves due to the dependance of the Alfvén velocity to the space.
- const int `solver_lpDampGrowth` = 1
Source term effect applied on backward waves due to the dependance of the Alfvén velocity to the space.
- const int `solver_lmDampGrowth` = 1
Source term effect due to production of self-turbulence - damping applied on foward waves.
- const int `solver_Dilution` = 0
Source term effect due to production of self-turbulence - damping applied on backward waves.
- const int `set_background` = 1
- const double `step_implicit` = 20.*yr
If need to perform some test without background conditions (1 : On, 0 : off)
- const int `source_terms_exact` = 1
- const int `nproc` = 1
The way to solve the source terms : 1 -> Exact solutions, 0 -> 1st order numerical solution.
- const int `output_freq` = 0
Number of processors for the run (! Still experimental)
- const double `t_data_out_min` = 0.*kyr
Model of output frequency (0 : n output between t_{start} and t_{end} , 1 : 1 output each n timestep) $n = \text{number_out_data}$.
- const double `t_data_out_max` = 500.*kyr
Instant of the first output data.
- const int `number_out_data` = 100
*200.*kyr; Instant of the last output data*
- const int `time_distrib_of_data` = 1
Total number of output data.
- const double `log_first_data` = 3e-1*kyr
- const int `delta_log_output` = 100
Time value of the first output in the logscaled output method.

- const double Tmax = 500.1*kyr
Number of time-step between two LogOutput.
- const int verbose = 0
*200.1*kyr; Define the limit time of your simulation*
- const double ttau_sat = - log(0.1)/0.1
0 : False, 1 : True -> Show extra informations during the simulation
- const double Esn = 1
- const double Mej = 1
in units of 1e51 erg : total energy released by SNR
- const double xi_n = 1
Msun : total mass released by SNR in sun mass units.
- const double phi_c = 1
For solar abundances.
- const double bbeta = 2
Actual thermal cond. / the Sptitzer (1962) value.
- const double C06 = 1.
- const double xhi_cr = 0.1
- const double xhi_0 = 2.026
Efficiency of CRs acceleration.
- const double gam = 2.2
- const int injection_cutoff = 0
CRs injection energy power law index.
- const double inj_exp_alpha = 1
- const double Emin = 0.1*GeV
*Factor of the exponential cutoff in the CRs injection spectra ($dN(E)/dE \sim \exp(-inj_exp_alpha * E/E_Max)$)*
- const double delta = 4.
Minimum accelereed CRs during the Sedov phase.
- const int injection_shape_time = 0
From Celli et al. (2019) - see Brahimi et al. (2020)
- const double t_start_injection = 1e-6*kyr
0 : Time Dirac CRs, 1 : Time Gaussian CRs
- const double t_end_injection = 2
Time start CRs injection function.
- const double injection_function_width = 10.
[in tesc[E] units] Time end CRs injection function (number of tesc)
- const int injection_function_norm = 1000
- const double r_snr_thickness = 100
Constant in order to easily and rapidly normalize the injection function.
- const double electron_injection_rate = 1e-2
= R_SNR(t)/by the value you chose, it allows to smooth the injection shape of CRs
- const int tesc_model = 1
Corresponds to the energy injected in the electron spectrum compared to the energy injected in the proton spectrum.
- const int oh_model = 3
- const double eta_gfree = 1
- const double eta_acc = 10
- const double Eknee = 1e15*eV
- const double alpha = 2.6
- const double coherence_length = 100*pc
- const double sigma_coherence = 10*pc
Coherence length of the magnetic field.
- const double isotropy = 1

4.17.1 Variable Documentation

4.17.1.1 alpha

```
const double alpha = 2.6
```

4.17.1.2 bbeta

```
const double bbeta = 2
```

Actual thermal cond. / the Sptitzer (1962) value.

4.17.1.3 c

```
const double c = 29979245800.
```

e charge in [statC]

4.17.1.4 C06

```
const double C06 = 1.
```

4.17.1.5 coherence_length

```
const double coherence_length = 100*pc
```

4.17.1.6 delta

```
const double delta = 4.
```

Minimum accelerated CRs during the Sedov phase.

4.17.1.7 `delta_log_output`

```
const int delta_log_output = 100
```

Time value of the first output in the logscaled output method.

4.17.1.8 `e`

```
const double e = 4.80326e-10
```

H2 mass [g].

4.17.1.9 `Eknee`

```
const double Eknee = 1e15*eV
```

4.17.1.10 `electron_injection_rate`

```
const double electron_injection_rate = 1e-2
```

= $R_{\text{SNR}}(t)$ /by the value you chose, it allows to smooth the injection shape of CRs

4.17.1.11 `Emin`

```
const double Emin = 0.1*GeV
```

Factor of the exponential cutoff in the CRs injection spectra ($dN(E)/dE \sim \exp(-\text{inj_exp_alpha} * E/E_{\text{Max}})$)

4.17.1.12 `Esn`

```
const double Esn = 1
```

Growth waves saturation rate

- $\log(0.1)/0.1$; Has the form $-\log(a)/b$ where b : characteristic max value, a : suppression factor after b , $t\tau_{\text{sat}} = 0 \rightarrow$ Linear growth This term is still experimental

4.17.1.13 eta_acc

```
const double eta_acc = 10
```

4.17.1.14 eta_gfree

```
const double eta_gfree = 1
```

4.17.1.15 eV

```
const double eV = GeV*1e-9
```

1 TeV in [erg]

4.17.1.16 gam

```
const double gam = 2.2
```

4.17.1.17 GeV

```
const double GeV = 0.00160218
```

1 kpc in [cm]

4.17.1.18 inj_exp_alpha

```
const double inj_exp_alpha = 1
```

Cut-off kind of the CRs injection spectra (0 : Abrupt no smooth, 1 : Exponential_cutoff see. inj_exp_alpha) Note : This model is only applicable to the proton spectrum. For electrons, the injection spectrum will be automatically cut by the synchrotron radiations

4.17.1.19 injection_cutoff

```
const int injection_cutoff = 0
```

CRs injection energy power law index.

4.17.1.20 injection_function_norm

```
const int injection_function_norm = 1000
```

Corresponds approximately to the width of the escape time divided (take care : too high value may affect the calculation and create noise) Max value -> value such as the width of $F_{inj} \gg dt$, max recommended value = 10 for NX,NE = 10, 7

4.17.1.21 injection_function_width

```
const double injection_function_width = 10.
```

[in tesc[E] units] Time end CRs injection function (number of tesc)

4.17.1.22 injection_shape_time

```
const int injection_shape_time = 0
```

From Celli et al. (2019) - see Brahimi et al. (2020)

4.17.1.23 isotropy

```
const double isotropy = 1
```

Width of the transition from the part close to the source where the diffusion coefficient is anitropic and the part far from the source where the diffusion is isotropic

4.17.1.24 kbolz

```
const double kbolz = 1.3807e-16
```

light celerity in [cm/s]

4.17.1.25 km

```
const double km = 1e5
```

1 kyr in [s]

4.17.1.26 kms

```
const double kms = 1e5
```

Boltzmann constant in CGS.

4.17.1.27 kpc

```
const double kpc = 1.e3*pc
```

1 pc in [cm]

4.17.1.28 kyr

```
const double kyr = 1e3*yr
```

1 yr in [s]

4.17.1.29 log_first_data

```
const double log_first_data = 3e-1*kyr
```

Time distribution of output data (0 : linspace, 1 : log10-space 2 : Custom output times, see the function [specific↵](#)
[OutputData\(\)](#) in the file : [out.h](#))

4.17.1.30 mCII

```
const double mCII = 6*mp + 6*mn
```

HeI mass [g].

4.17.1.31 me

```
const double me = 9.1095e-28
```

Neutron mass [g].

4.17.1.32 Mej

```
const double Mej = 1
```

in units of $1e51$ erg : total energy released by SNR

4.17.1.33 MeV

```
const double MeV = 1e-3*GeV
```

1 eV in [erg]

4.17.1.34 mH2

```
const double mH2 = 2*mp
```

CII mass [g].

4.17.1.35 mHeI

```
const double mHeI = 2*mp + 2*mn
```

HII mass [g].

4.17.1.36 mHI

```
const double mHI = mp
```

Electron mass [g].

4.17.1.37 mHII

```
const double mHII = mp
```

HI mass [g].

4.17.1.38 mn

```
const double mn = 1.6749286e-24
```

Proton mass [g].

4.17.1.39 mp

```
const double mp = 1.6726219e-24
```

1 MeV in [erg]

4.17.1.40 nproc

```
const int nproc = 1
```

The way to solve the source terms : 1 -> Exact solutions, 0 -> 1st order numerical solution.

4.17.1.41 number_out_data

```
const int number_out_data = 100
```

200.*kyr; Instant of the last output data

4.17.1.42 oh_model

```
const int oh_model = 3
```

CR escape time model (1 : All CRs escape at the beginning of the radiative phase, 2 : If $v_{sh} < 110$ km/s, all CRs escape, 0 : No radiative escape model)

4.17.1.43 output_freq

```
const int output_freq = 0
```

Number of processors for the run (! Still experimental)

4.17.1.44 pc

```
const double pc = 3.086e18
```

1 km in [cm]

4.17.1.45 phi_c

```
const double phi_c = 1
```

For solar abundances.

4.17.1.46 pi

```
const double pi = 3.1472
```

Inclusion of basic mathematic functions.

4.17.1.47 r_snr_thickness

```
const double r_snr_thickness = 100
```

Constant in order to easily and rapidly normalize the injection function.

4.17.1.48 set_background

```
const int set_background = 1
```

Time dilution term according to the SNR shock evolution in the flux tube approx. ie. $R_{sh}^2(t_0) P(t_0) = R_{sh}^2(t_1) P(t_1)$ if conserved energy Still experimental !!!

4.17.1.49 sig_T

```
const double sig_T = 6.65e-25
```

1 km/s in cm/s

4.17.1.50 sigma_coherence

```
const double sigma_coherence = 10*pc
```

Coherence length of the magnetic field.

4.17.1.51 solver_Dilution

```
const int solver_Dilution = 0
```

Source term effect due to production of self-turbulence - damping applied on backward waves.

4.17.1.52 solver_ImAdvection

```
const int solver_ImAdvection = 1
```

Advective term of the forward waves.

4.17.1.53 solver_ImDampGrowth

```
const int solver_ImDampGrowth = 1
```

Source term effect due to production of self-turbulence - damping applied on forward waves.

4.17.1.54 solver_ImSource1

```
const int solver_ImSource1 = 1
```

Source term effect applied on forward waves due to the dependence of the Alfvén velocity to the space.

4.17.1.55 solver_IpAdvection

```
const int solver_IpAdvection = 1
```

Perpendicular diffusion of e- after the coherence length.

4.17.1.56 solver_IpDampGrowth

```
const int solver_IpDampGrowth = 1
```

Source term effect applied on backward waves due to the dependance of the Alfvén velocity to the space.

4.17.1.57 solver_IpSource1

```
const int solver_IpSource1 = 1
```

Advective term of the backward waves.

4.17.1.58 solver_PcrAdvection

```
const int solver_PcrAdvection = 1
```

[cm²] Thomson cross section (see Schlickeiser (2002) p.80)

4.17.1.59 solver_PcrAdvection2

```
const int solver_PcrAdvection2 = 1
```

Diffusive term of the CR Pressure.

4.17.1.60 solver_PcrAdvectionE

```
const int solver_PcrAdvectionE = 1
```

Explicit Advection solver for Pcr by the energy derivative of Alfvén velocity.

4.17.1.61 solver_PcrDiffusion

```
const int solver_PcrDiffusion = 1
```

Advective term of the CR Pressure (the classical one -> $V_A \cdot \text{grad} \dots$)

4.17.1.62 solver_PcrPerpDiff

```
const int solver_PcrPerpDiff = 0
```

Source term effect due to the CR injection from the source in the system.

4.17.1.63 solver_PcrSource1

```
const int solver_PcrSource1 = 1
```

Explicit Advection solver for Pcr in energy cdVAdX.

4.17.1.64 solver_PcrSource2

```
const int solver_PcrSource2 = 1
```

Source term effect due to the dependance of the Alfvén velocity to the space.

4.17.1.65 solver_PeAdvection

```
const int solver_PeAdvection = 1
```

Perpendicular diffusion of CRs after the coherence length.

4.17.1.66 solver_PeAdvection2

```
const int solver_PeAdvection2 = 1
```

Diffusive term of the e- Pressure.

4.17.1.67 solver_PeAdvectionE

```
const int solver_PeAdvectionE = 1
```

Explicit Advection solver for e- by the energy derivative of Alfvén velocity.

4.17.1.68 solver_PeAdvectionE1

```
const int solver_PeAdvectionE1 = 1
```

Explicit Advection solver for e- in energy cdVAdX.

4.17.1.69 solver_PeAdvectionE2

```
const int solver_PeAdvectionE2 = 1
```

Synchrotron radiations of e- (looses of energy) - Advection term.

4.17.1.70 solver_PeDiffusion

```
const int solver_PeDiffusion = 1
```

Advective term of the e- Pressure (the classical one -> $V_A \cdot \text{grad} \dots$)

4.17.1.71 solver_PePerpDiff

```
const int solver_PePerpDiff = 0
```

Source term effect due to the e- injection from the source in the system.

4.17.1.72 solver_PeSource1

```
const int solver_PeSource1 = 1
```

Synchrotron radiations of e- (looses of energy) - source term.

4.17.1.73 solver_PeSource2

```
const int solver_PeSource2 = 1
```

Source term effect due to the dependance of the Alfvén velocity to the space (for e-)

4.17.1.74 source_terms_exact

```
const int source_terms_exact = 1
```

Time step of the simulation if only implicit solvers are used and maximum time step value. This timestep has to be > than $10 \times \min(\text{tesc}(E))$ Example : If $\min(\text{tesc}) \sim 500$ yrs, then $dt < 500/10 = 50$ yrs

4.17.1.75 step_implicit

```
const double step_implicit = 20.*yr
```

If need to perform some test without background conditions (1 : On, 0 : off)

4.17.1.76 t_data_out_max

```
const double t_data_out_max = 500.*kyr
```

Instant of the first output data.

4.17.1.77 t_data_out_min

```
const double t_data_out_min = 0.*kyr
```

Model of output frequency (0 : n output between t_{start} and t_{end} , 1 : 1 output each n timestep) $n = \text{number_out_data}$.

4.17.1.78 t_end_injection

```
const double t_end_injection = 2
```

Time start CRs injection function.

4.17.1.79 t_start_injection

```
const double t_start_injection = 1e-6*kyr
```

0 : Time Dirac CRs, 1 : Time Gaussian CRs

4.17.1.80 tesc_model

```
const int tesc_model = 1
```

Corresponds to the energy injected in the electron spectrum compared to the energy injected in the proton spectrum.

4.17.1.81 TeV

```
const double TeV = 1e3*GeV
```

1 GeV in [erg]

4.17.1.82 time_distrib_of_data

```
const int time_distrib_of_data = 1
```

Total number of output data.

4.17.1.83 Tmax

```
const double Tmax = 500.1*kyr
```

Number of time-step between two LogOutput.

4.17.1.84 ttau_sat

```
const double ttau_sat = - log(0.1)/0.1
```

0 : False, 1 : True -> Show extra informations during the simulation

4.17.1.85 verbose

```
const int verbose = 0
```

200.1*kyr; Define the limit time of your simulation

4.17.1.86 xhi_0

```
const double xhi_0 = 2.026
```

Efficiency of CRs acceleration.

4.17.1.87 xhi_cr

```
const double xhi_cr = 0.1
```

4.17.1.88 xi_n

```
const double xi_n = 1
```

Msun : total mass released by SNR in sun mass units.

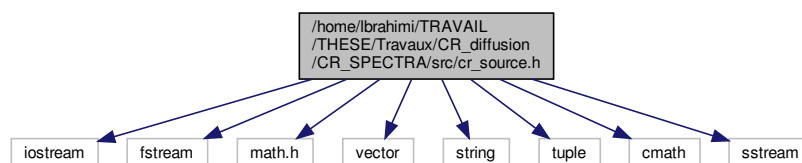
4.17.1.89 yr

```
const double yr = 3.154e+7
```

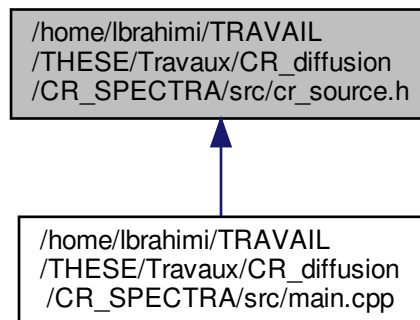
4.18 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/cr_source.h File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
```

Include dependency graph for cr_source.h:



This graph shows which files directly or indirectly include this file:



Functions

- double [GetTSed](#) ()
- double [RSNR](#) (double time)
- double [u_sh](#) (double time)
Shock velocity in time.
- double [GetEM](#) ()
Get the EMAX value for the tesc calculation.
- double [tesc](#) (double E)
- double [B_sat](#) (double time)
- double [GetEM_e](#) ()
- double [tesc_e](#) (double E)
- double [Resc](#) (double E)
This function return the value of the escape shock radius at a given energy of protons.
- double [sigm](#) (double E, double ttesc)
This function defines the variance of the time injection function Finj of CRs.
- double [Finj](#) (double t, double dt, double E, double ttesc)
CRs time injection function.
- double [theta](#) (double z, double t, double Rsnr)
This function define the spatial shape of the CRs distribution at a given time.
- double [dNdE](#) (double E)
This function defines the injection spectrum of both electrons of protons escaping from the SNR.
- double [ff](#) (double E)
This function defines the distribution function of both electrons and protons escaping from the SNR.
- double [Pcr_ini](#) (double E)
This function defines the initial pressure distribution of both electrons and protons escaping from the SNR.

Variables

- `std::string parameters = "./parameters.dat"`
- `std::string snx = search(parameters, "NX")`
- `int nx = stoi(snx)`
- `std::string sne = search(parameters, "NE")`
- `int ne = stoi(sne)`
- `std::string sni = search(parameters, "ni")`
- `double ni = stod(sni)`
- `std::string sX = search(parameters, "X")`
- `double Xi = stod(sX)`
- `double nt = ni/Xi`
- `std::string smn = search(parameters, "mn")`
- `double m_neutral = stod(smn)`
- `std::string sT = search(parameters, "T")`
- `double T = stod(sT)`
- `std::string scenter = search(parameters, "center")`
- `double x_center = stod(scenter)`
- `std::string scenter_index = search(parameters, "center_index")`
- `int x_center_index = stod(scenter_index)`
- `std::string sBcenter = search(parameters, "B")`
- `double Bcenter = stod(sBcenter)`

4.18.1 Function Documentation

4.18.1.1 B_sat()

```
double B_sat (
    double time )
```

4.18.1.2 dNdE()

```
double dNdE (
    double E )
```

This function defines the injection spectrum of both electrons of protons escaping from the SNR.

4.18.1.3 ff()

```
double ff (
    double E )
```

This function defines the distribution function of both electrons and protons escaping from the SNR.

4.18.1.4 Finj()

```
double Finj (
    double t,
    double dt,
    double E,
    double ttesc )
```

CRs time injection function.

4.18.1.5 GetEM()

```
double GetEM ( )
```

Get the EMAX value for the tesc calculation.

4.18.1.6 GetEM_e()

```
double GetEM_e ( )
```

This function allows to calculate the maximum energy provided to the electrons escaping from the SNR according to the values in the `constants.h` file.

4.18.1.7 GetTSed()

```
double GetTSed ( )
```

This function return the value of `t_sedov` according to the SNR properties specified in the [constants.h](#) file

4.18.1.8 Pcr_ini()

```
double Pcr_ini (
    double E )
```

This function defines the initial pressure distribution of both electrons and protons escaping from the SNR.

4.18.1.9 Resc()

```
double Resc (
    double E )
```

This function return the value of the escape shock radius at a given energy of protons.

4.18.1.10 RSNR()

```
double RSNR (
    double time )
```

This function provides the value of the shock radius as a function of the time according to the SNR expansion model from Cioffi et al. 1998 and Truelove & McKee 19... The SNR shock has different propagation stages which are separated by characteristic times defined as below :

$$\begin{aligned}
 t_{\text{ini,kyr}} &= 10^{-4} \\
 t_{\text{free,kyr}} &= 0.3 E_{\text{SN}}^{-1/2} M_{\text{ej}} n_t^{-1/3} \\
 t_{\text{PDS,yr}} &= e^{-1} \times 3.61 \times 10^4 E_{\text{SN}}^{3/14} / (\xi_n^{5/14} n_t^{4/7}) \\
 t_{\text{MCS,yr}} &= \min \left[61 V_{\text{ej},8}^3 / (\xi_n^{9/14} * n_t^{3/7} * E_{\text{SN}}^{3/14}), 476 t_{\text{PDS,yr}} / (\xi_n \phi_c)^{9/14} \right] \\
 t_{\text{merge,yr}} &= 153 \left(E_{\text{SN}}^{1/14} n_t^{1/7} \xi_n^{3/14} / (\beta C_{06}) \right)^{10/7} t_{\text{PDS,yr}} \\
 t_{\text{max}} &= \max(t_{\text{MCS}}, t_{\text{merge}})
 \end{aligned}$$

where t_{ini} corresponds to the initial time of the numerical computation of the spline, t_{free} corresponds to the transition time between the free expansion phase and the Sedov-Taylor phase, t_{PDS} corresponds to the transition time between the Sedov-Taylor phase and the Pressure Driven Snowplow phase, t_{MCS} corresponds to the transition time between the Pressure Driven Snowplow phase and the Momentum Conserving Snowplow phase and finally t_{merge} means the time at which the shock pressure becomes of the order of the ISM pressure. In some ISM phases, the shock can merge before entering in the Momentum Conserving Snowplow phase explaining t_{max} . The SNR shock evolves with time according the following radii

$$R_{\text{ini}} = R_{\text{free}}(t_{\text{ini}}/t_{\text{free}})$$

Parameters

<i>time</i>	Time in [s].
-------------	--------------

See also

[RSNR\(\)](#), [InterpolatingSpline\(\)](#)

4.18.1.11 sigm()

```
double sigm (
    double E,
    double ttesc )
```

This function defines the variance of the time injection function Finj of CRs.

4.18.1.12 tesc()

```
double tesc (
    double E )
```

Protons escape time function according the model of Celli et al. (2019) but also alternatives models of CR escape during radiative stages

4.18.1.13 tesc_e()

```
double tesc_e (
    double E )
```

This function calculate the escape time of the electrons as a function of the energy according both model of escape from Celli et al. (2019) and alternative models of escape in radiative phases

4.18.1.14 theta()

```
double theta (
    double z,
    double t,
    double Rsnr )
```

This function define the spatial shape of the CRs distribution at a given time.

4.18.1.15 u_sh()

```
double u_sh (
    double time )
```

Shock velocity in time.

4.18.2 Variable Documentation**4.18.2.1 Bcenter**

```
double Bcenter = stod(sBcenter)
```

4.18.2.2 m_neutral

```
double m_neutral = stod(smn)
```

4.18.2.3 ne

```
int ne = stoi(sne)
```

4.18.2.4 ni

```
double ni = stod(sni)
```

4.18.2.5 nt

```
double nt = ni/Xi
```

4.18.2.6 nx

```
int nx = stoi(snx)
```

4.18.2.7 parameters

```
std::string parameters = "./parameters.dat"
```

4.18.2.8 sBcenter

```
std::string sBcenter = search(parameters, "B")
```

4.18.2.9 scenter

```
std::string scenter = search(parameters, "center")
```

4.18.2.10 scenter_index

```
std::string scenter_index = search(parameters, "center_index")
```

4.18.2.11 smn

```
std::string smn = search(parameters, "mn")
```

4.18.2.12 sne

```
std::string sne = search(parameters, "NE")
```

4.18.2.13 sni

```
std::string sni = search(parameters, "ni")
```

4.18.2.14 snx

```
std::string snx = search(parameters, "NX")
```

4.18.2.15 sT

```
std::string sT = search(parameters, "T")
```

4.18.2.16 sX

```
std::string sX = search(parameters, "X")
```

4.18.2.17 T

```
double T = stod(sT)
```

4.18.2.18 x_center

```
double x_center = stod(scenter)
```

4.18.2.19 x_center_index

```
int x_center_index = stod(scenter_index)
```

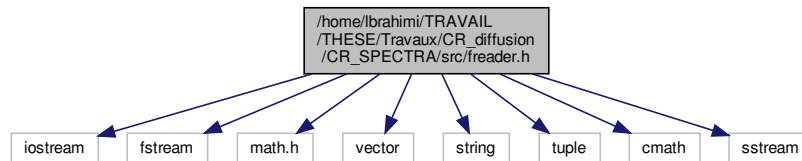
4.18.2.20 Xi

```
double Xi = stod(sX)
```

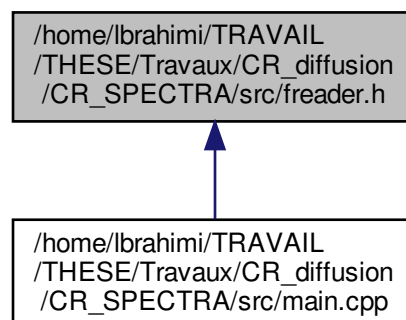
4.19 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/freader.h File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
```

Include dependency graph for freader.h:



This graph shows which files directly or indirectly include this file:



Functions

- `std::string` [search](#) (string `file_name`, string variable)
- `vector< double >` [readAxis](#) (std::string `filename`, int `vec_size`)

4.19.1 Function Documentation

4.19.1.1 `readAxis()`

```
vector<double> readAxis (
    std::string filename,
    int vec_size )
```

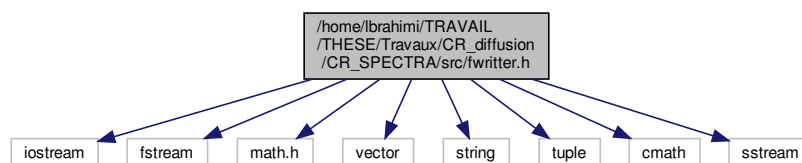
4.19.1.2 `search()`

```
std::string search (
    string file_name,
    string variable )
```

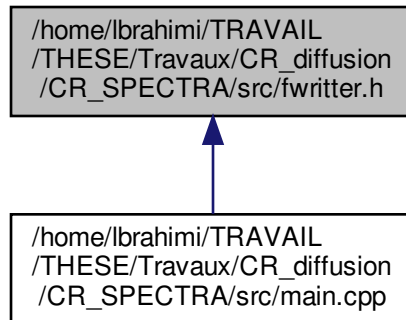
4.20 `/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/fwritter.h` File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
```

Include dependency graph for `fwritter.h`:



This graph shows which files directly or indirectly include this file:



Functions

- int [writeXE](#) (std::string filename, int index, vector< vector< double > > data, double NX, double NE)
- int [writeInfo](#) (std::string filename, int index, vector< double > info, vector< std::string > s_info)

4.20.1 Function Documentation

4.20.1.1 writeInfo()

```
int writeInfo (
    std::string filename,
    int index,
    vector< double > info,
    vector< std::string > s_info )
```

4.20.1.2 writeXE()

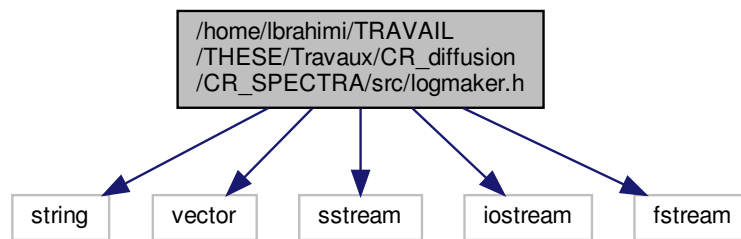
```
int writeXE (
    std::string filename,
    int index,
    vector< vector< double > > data,
    double NX,
    double NE )
```

4.21 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/logmaker.h

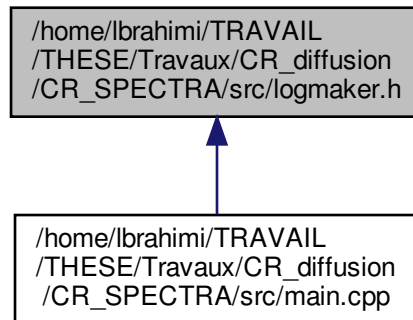
File Reference

```
#include <string>
#include <vector>
#include <sstream>
#include <iostream>
#include <fstream>
```

Include dependency graph for logmaker.h:



This graph shows which files directly or indirectly include this file:



Functions

- void `showLog_0` (double time, double Tmax, int nstep, int time_id, double duration, double dt)

4.21.1 Function Documentation

4.21.1.1 showLog_0()

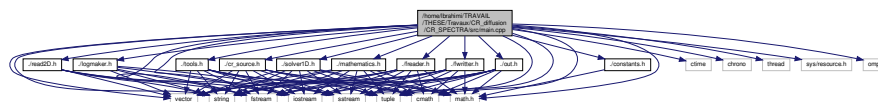
```
void showLog_0 (
    double time,
    double Tmax,
    int nstep,
    int time_id,
    double duration,
    double dt )
```

4.22 /home/Ibrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/main.cpp

File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
#include <ctime>
#include <chrono>
#include <thread>
#include <sys/resource.h>
#include <omp.h>
#include "../mathematics.h"
#include "../constants.h"
#include "../read2D.h"
#include "../freader.h"
#include "../fwriter.h"
#include "../out.h"
#include "../logmaker.h"
#include "../tools.h"
#include "../cr_source.h"
#include "../solver1D.h"
```

Include dependency graph for main.cpp:



Functions

- `int main ()`

4.22.1 Function Documentation

4.22.1.1 main()

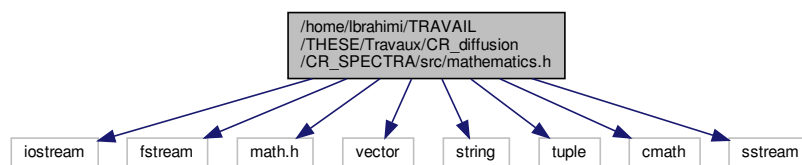
```
int main ( )
```

4.23 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/mathematics.h

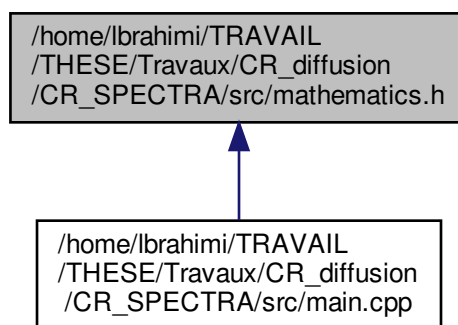
File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
```

Include dependency graph for mathematics.h:



This graph shows which files directly or indirectly include this file:



Functions

- `vector< double > TDMA (vector< double > a, vector< double > b, vector< double > c, vector< double > d)`

- void [InverseTrigonalMatrix](#) (vector< vector< double > > &T)
- void [ProductMatrix](#) (vector< vector< double > > A, vector< vector< double > > B, vector< vector< double > > &C)
- double [InterpolatingSpline](#) (vector< double > X, vector< double > Y, double x)
- double [f1](#) (double x, vector< double > cst)
- double [df1dx](#) (double x, vector< double > cst)
- double [f2](#) (double x, vector< double > cst)
- double [df2dx](#) (double x, vector< double > cst)
- double [NewtonRaphson](#) (double f(double, vector< double >), double df(double, vector< double >), double x0, double eps, vector< double > cst)
- double [GetMax](#) (vector< double > V)
- void [transpose](#) (vector< vector< double > > &b, vector< vector< double > > &c)
- double [minmod](#) (double a, double b)

4.23.1 Function Documentation

4.23.1.1 df1dx()

```
double df1dx (  
    double x,  
    vector< double > cst )
```

4.23.1.2 df2dx()

```
double df2dx (  
    double x,  
    vector< double > cst )
```

4.23.1.3 f1()

```
double f1 (  
    double x,  
    vector< double > cst )
```

4.23.1.4 f2()

```
double f2 (  
    double x,  
    vector< double > cst )
```

4.23.1.5 GetMax()

```
double GetMax (
    vector< double > V )
```

4.23.1.6 InterpolatingSpline()

```
double InterpolatingSpline (
    vector< double > X,
    vector< double > Y,
    double x )
```

4.23.1.7 InverseTrigonalMatrix()

```
void InverseTrigonalMatrix (
    vector< vector< double > > & T )
```

4.23.1.8 minmod()

```
double minmod (
    double a,
    double b )
```

4.23.1.9 NewtonRaphson()

```
double NewtonRaphson (
    double fdouble, vector< double >,
    double dfdouble, vector< double >,
    double x0,
    double eps,
    vector< double > cst )
```

4.23.1.10 ProductMatrix()

```
void ProductMatrix (
    vector< vector< double > > A,
    vector< vector< double > > B,
    vector< vector< double > > & C )
```

4.23.1.11 TDMA()

```
vector<double> TDMA (
    vector< double > a,
    vector< double > b,
    vector< double > c,
    vector< double > d )
```

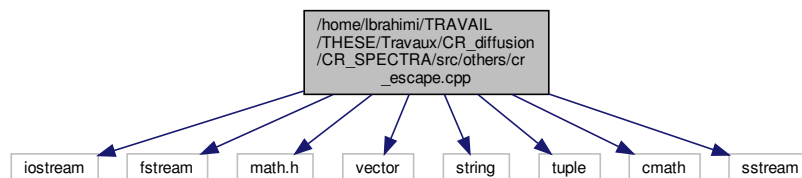
4.23.1.12 transpose()

```
void transpose (
    vector< vector< double > > & b,
    vector< vector< double > > & c )
```

4.24 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/cr_escape.cpp File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
```

Include dependency graph for cr_escape.cpp:



Functions

- void [InverseTrigonalMatrix](#) (vector< vector< double > > &T)
- void [ProductMatrix](#) (vector< vector< double > > A, vector< vector< double > > B, vector< vector< double > > &C)
- double [InterpolatingSpline](#) (vector< double > X, vector< double > Y, double x)
- double [f1](#) (double x, vector< double > cst)
- double [df1dx](#) (double x, vector< double > cst)
- double [f2](#) (double x, vector< double > cst)
- double [df2dx](#) (double x, vector< double > cst)
- double [NewtonRaphson](#) (double f(double, vector< double >), double df(double, vector< double >), double x0, double eps, vector< double > cst)
- double [GetMax](#) (vector< double > V)
- double [GetTesc](#) (double E, double [delta](#), vector< double > cst)
- int [main](#) ()

4.24.1 Function Documentation

4.24.1.1 df1dx()

```
double df1dx (  
    double x,  
    vector< double > cst )
```

4.24.1.2 df2dx()

```
double df2dx (  
    double x,  
    vector< double > cst )
```

4.24.1.3 f1()

```
double f1 (  
    double x,  
    vector< double > cst )
```

4.24.1.4 f2()

```
double f2 (  
    double x,  
    vector< double > cst )
```

4.24.1.5 GetMax()

```
double GetMax (  
    vector< double > V )
```

4.24.1.6 GetTesc()

```
double GetTesc (  
    double E,  
    double delta,  
    vector< double > cst )
```

4.24.1.7 InterpolatingSpline()

```
double InterpolatingSpline (
    vector< double > X,
    vector< double > Y,
    double x )
```

4.24.1.8 InverseTrigonalMatrix()

```
void InverseTrigonalMatrix (
    vector< vector< double > > & T )
```

4.24.1.9 main()

```
int main ( )
```

4.24.1.10 NewtonRaphson()

```
double NewtonRaphson (
    double fdouble, vector< double >,
    double dfdouble, vector< double >,
    double x0,
    double eps,
    vector< double > cst )
```

4.24.1.11 ProductMatrix()

```
void ProductMatrix (
    vector< vector< double > > A,
    vector< vector< double > > B,
    vector< vector< double > > & C )
```

4.25 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/PDE_solvers.py File Reference ↩

Namespaces

- [PDE_solvers](#)

Functions

- def [PDE_solvers.TDMA](#) (a, b, c, d)
- def [PDE_solvers.A](#) (x)
- def [PDE_solvers.B](#) (x)
- def [PDE_solvers.C](#) (x)
- def [PDE_solvers.Q](#) (x)
- def [PDE_solvers.T](#) (x)
- def [PDE_solvers.finiteDiffSolver](#) (dt, Tmax, u)
- def [PDE_solvers.SimpleImplicitSolver](#) (dt, Tmax, u)
- def [PDE_solvers.CC70Solver](#) (dt, Tmax, u)

Variables

- float [PDE_solvers.pc](#) = 3.086e18
Tri Diagonal Matrix Algorithm(a.k.a Thomas algorithm) solver def TDMA solver(a, b, c, d): "" TDMA solver, a b c d can be NumPy array type or Python list type.
- float [PDE_solvers.yr](#) = 365.25*86400
- int [PDE_solvers.kyr](#) = 1e3*yr
- int [PDE_solvers.M](#) = 2048
- int [PDE_solvers.Xmin](#) = 0.
- int [PDE_solvers.Xmax](#) = 1000.*[pc](#)
- [PDE_solvers.X](#) = np.linspace(Xmin, Xmax, M+1)
- [PDE_solvers.u](#) = np.zeros(M+1)
plt.semilogy(X/pc, u, c="blue") plt.plot(X/pc, u/U, c="red") plt.axhline(1.) print ("ratio = ",sum(u)/sum(U))
- [PDE_solvers.u0](#) = u[1]
- [PDE_solvers.uM](#) = u[M-1]
- [PDE_solvers.u_ini](#) = u.copy()
- [PDE_solvers.u_end](#) = u
- int [PDE_solvers.Tmax](#) = 10.*[kyr](#)
- def [PDE_solvers.u_0](#) = CC70Solver(0.1*[kyr](#), Tmax, u)
- def [PDE_solvers.u_1](#) = CC70Solver(0.5*[kyr](#), Tmax, u)
- def [PDE_solvers.u_2](#) = CC70Solver(5.*[kyr](#), Tmax, u)
- [PDE_solvers.figsize](#)
- [PDE_solvers.c](#)

4.26 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/others/ ↵ Split_solvers.py File Reference

Namespaces

- [Split_solvers](#)

Functions

- def [Split_solvers.TDMA](#) (a, b, c, d)
- def [Split_solvers.generalized_diffusion](#) (u, X, D, dt, theta=0.5)

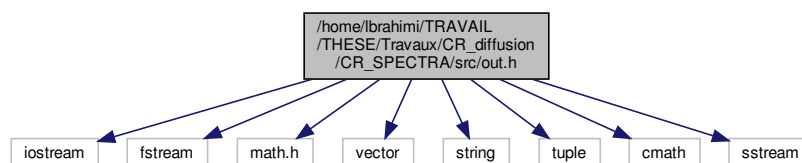
Variables

- `int Split_solvers.NX = 1000`
- `int Split_solvers.Xmin = -1`
- `int Split_solvers.Xmax = 1`
- `Split_solvers.X = np.linspace(Xmin, Xmax, NX+1)`
- `int Split_solvers.D = np.ones(len(X))*1e-2`
- `int Split_solvers.u = np.ones(len(X))*0.`
- `Split_solvers.c`
- `int Split_solvers.t_ini = 0.`
- `int Split_solvers.dt = 1e-1`
- `int Split_solvers.t_max = 1.`
- `int Split_solvers.t = t_ini`
- `int Split_solvers.u_new_0 = u`
- `int Split_solvers.u_old = u_new_0`
- `int Split_solvers.u_new_1 = u`

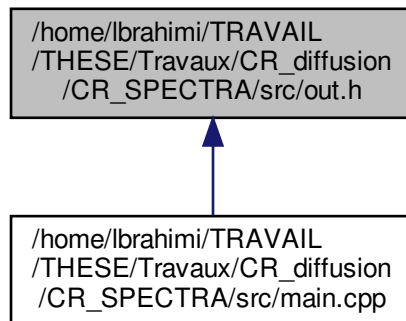
4.27 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/out.h File Reference

```
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
```

Include dependency graph for out.h:



This graph shows which files directly or indirectly include this file:



Functions

- `vector< double > specificOutputData ()`
- `vector< double > outputData (double tmin, double tmax, int nvalues, string modulation, double eps)`
Output data function for a regular and large amount of output data.
- `vector< double > getOutput ()`
Main output file function. This one will be used in the main file.
- `int getLogOutput ()`
Main output log file function. This one will be used in the main file.
- `double setTmax ()`
Define the limit time of your simulation.

4.27.1 Function Documentation

4.27.1.1 getLogOutput()

```
int getLogOutput ( )
```

Main output log file function. This one will be used in the main file.

4.27.1.2 getOutput()

```
vector<double> getOutput ( )
```

Main output file function. This one will be used in the main file.

4.27.1.3 outputData()

```
vector<double> outputData (
    double tmin,
    double tmax,
    int nvalues,
    string modulation,
    double eps )
```

Output data function for a regular and large amount of output data.

4.27.1.4 setTmax()

```
double setTmax ( )
```

Define the limit time of your simulation.

4.27.1.5 specificOutputData()

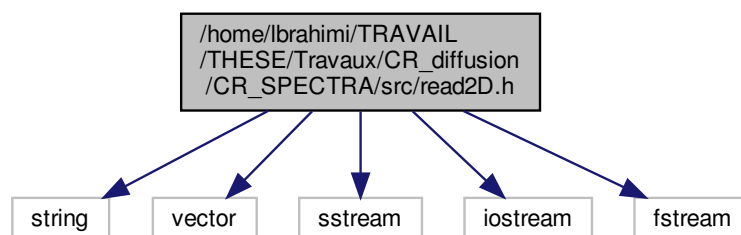
```
vector<double> specificOutputData ( )
```

This function allows you to specify your own data output time array You just need to fill the list loc_data, example:
double loc_data[] = {t1, t2, ..., tn}; where t_i is in seconds

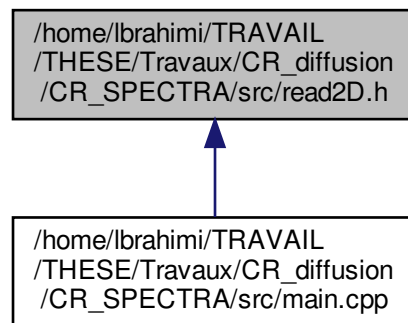
4.28 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/read2D.h File Reference

```
#include <string>
#include <vector>
#include <sstream>
#include <iostream>
#include <fstream>
```

Include dependency graph for read2D.h:



This graph shows which files directly or indirectly include this file:



Functions

- `vector< vector< double > > parse2DCsvFile (string inputFileName, int start)`

4.28.1 Function Documentation

4.28.1.1 `parse2DCsvFile()`

```
vector<vector<double> > parse2DCsvFile (
    string inputFileName,
    int start )
```

Reads csv file into table, exported as a vector of vector of doubles.

Parameters

<i>inputFileName</i>	input file name (full path).
----------------------	------------------------------

Returns

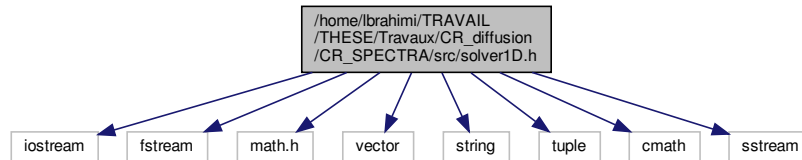
data as vector of vector of doubles.

4.29 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/solver1D.h File Reference

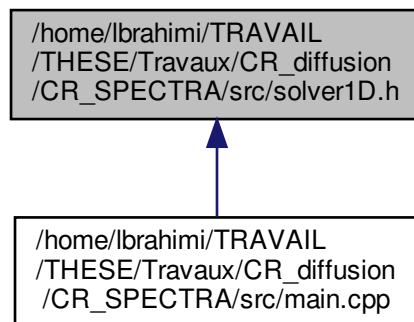
```
#include <iostream>
#include <fstream>
```

```
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>
```

Include dependency graph for solver1D.h:



This graph shows which files directly or indirectly include this file:



Functions

- void [thetaDiffusionSolver](#) (vector< vector< double > > &u, vector< vector< double > > &Pcr_new, double dt, vector< double > X, int NE, vector< vector< double > > Ip, vector< vector< double > > Im, vector< vector< double > > Db, vector< vector< double > > &Pcr_background)
- void [advectionSolverX](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, double dt, vector< double > X, int NE, vector< vector< double > > V, int sign, int border)
- void [advectionSolverE](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, double dt, vector< double > E, int NX, vector< vector< double > > V, vector< vector< double > > u_background)
- void [advectionSolverE1](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, double dt, vector< double > E, vector< double > BB, vector< double > EE, int NX, vector< vector< double > > u_background)
- void [advectionSolverE2](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, double dt, vector< double > E, int NX, vector< double > BB, vector< double > EE, vector< vector< double > > u_background)

- void [sourceSolver](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, double dt, vector< vector< double > > source, double factor)
- void [sourceGrowthDampRateSolver](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, vector< vector< double > > v_old, vector< vector< double > > source, vector< vector< double > > background, vector< double > X, double dt, vector< vector< double > > V, vector< double > B, int factor)
- void [CRsInjectionSourceSolver](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, double dt, vector< double > Pcr_ini, vector< double > Finj_temp, vector< double > vec_theta)
- void [dilute_solver](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, vector< vector< double > > u_background, double r_new, double r_old, vector< double > nn, vector< double > ni, vector< double > mn, vector< double > mi, vector< double > B, vector< double > T)
- void [perpendicular_diffusion_solver](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, vector< vector< double > > u_bc, vector< vector< double > > Db, vector< vector< double > > I0, vector< double > X, double dt, vector< double > r_esc)
- void [NotMove](#) (vector< vector< double > > u, vector< vector< double > > u_new)
- void [electron_source](#) (vector< vector< double > > &u_old, vector< vector< double > > &u_new, vector< double > E, double dt, int NE, int NX, vector< vector< double > > u_background)

4.29.1 Function Documentation

4.29.1.1 advectionSolverE()

```
void advectionSolverE (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    double dt,
    vector< double > E,
    int NX,
    vector< vector< double > > V,
    vector< vector< double > > u_background )
```

4.29.1.2 advectionSolverE1()

```
void advectionSolverE1 (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    double dt,
    vector< double > E,
    vector< double > BB,
    vector< double > EE,
    int NX,
    vector< vector< double > > u_background )
```

4.29.1.3 advectionSolverE2()

```
void advectionSolverE2 (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    double dt,
    vector< double > E,
    int NX,
    vector< double > BB,
    vector< double > EE,
    vector< vector< double > > u_background )
```

4.29.1.4 advectionSolverX()

```
void advectionSolverX (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    double dt,
    vector< double > X,
    int NE,
    vector< vector< double > > V,
    int sign,
    int border )
```

4.29.1.5 CRsInjectionSourceSolver()

```
void CRsInjectionSourceSolver (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    double dt,
    vector< double > Pcr_ini,
    vector< double > Finj_temp,
    vector< double > vec_theta )
```

4.29.1.6 dilute_solver()

```
void dilute_solver (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    vector< vector< double > > u_background,
    double r_new,
    double r_old,
    vector< double > nn,
    vector< double > ni,
    vector< double > mn,
    vector< double > mi,
    vector< double > B,
    vector< double > T )
```

4.29.1.7 electron_source()

```

void electron_source (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    vector< double > E,
    double dt,
    int NE,
    int NX,
    vector< vector< double > > u_background )

```

4.29.1.8 NotMove()

```

void NotMove (
    vector< vector< double > > u,
    vector< vector< double > > u_new )

```

4.29.1.9 perpendicular_diffusion_solver()

```

void perpendicular_diffusion_solver (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    vector< vector< double > > u_bc,
    vector< vector< double > > Db,
    vector< vector< double > > I0,
    vector< double > X,
    double dt,
    vector< double > r_esc )

```

4.29.1.10 sourceGrowthDampRateSolver()

```

void sourceGrowthDampRateSolver (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    vector< vector< double > > v_old,
    vector< vector< double > > source,
    vector< vector< double > > background,
    vector< double > X,
    double dt,
    vector< vector< double > > V,
    vector< double > B,
    int factor )

```


4.29.1.11 sourceSolver()

```

void sourceSolver (
    vector< vector< double > > & u_old,
    vector< vector< double > > & u_new,
    double dt,
    vector< vector< double > > source,
    double factor )

```

4.29.1.12 thetaDiffusionSolver()

```

void thetaDiffusionSolver (
    vector< vector< double > > & u,
    vector< vector< double > > & Pcr_new,
    double dt,
    vector< double > X,
    int NE,
    vector< vector< double > > Ip,
    vector< vector< double > > Im,
    vector< vector< double > > Db,
    vector< vector< double > > & Pcr_background )

```

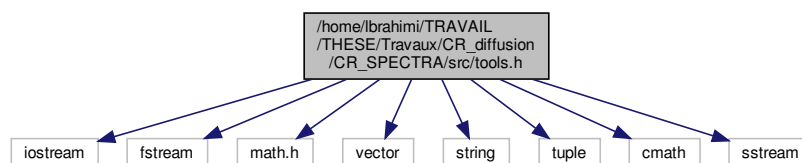
4.30 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/src/tools.h File Reference

```

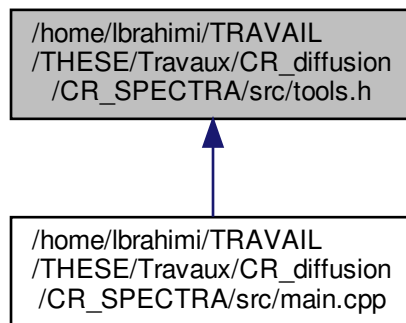
#include <iostream>
#include <fstream>
#include <math.h>
#include <vector>
#include <string>
#include <tuple>
#include <cmath>
#include <sstream>

```

Include dependency graph for tools.h:



This graph shows which files directly or indirectly include this file:



Functions

- double [maxElement1D](#) (vector< double > data)
- double [minElement1D](#) (vector< double > data)
- double [maxElement2D](#) (vector< vector< double > > data)
- double [minElement2D](#) (vector< vector< double > > data)
- double [absmaxElement2D](#) (vector< vector< double > > data)

4.30.1 Function Documentation

4.30.1.1 absmaxElement2D()

```
double absmaxElement2D (
    vector< vector< double > > data )
```

4.30.1.2 maxElement1D()

```
double maxElement1D (
    vector< double > data )
```

4.30.1.3 maxElement2D()

```
double maxElement2D (
    vector< vector< double > > data )
```

4.30.1.4 minElement1D()

```
double minElement1D (
    vector< double > data )
```

4.30.1.5 minElement2D()

```
double minElement2D (
    vector< vector< double > > data )
```

4.31 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/background_↵
_diffusion_coefficient.py File Reference

Namespaces

- [background_diffusion_coefficient](#)

Functions

- def [background_diffusion_coefficient.IonNeutral_Damping](#) (k, medium_props, nu_n=0, [theta](#)=0)
- def [background_diffusion_coefficient.Duu_Alfven_Slab_Linear_Undamped](#) (mu, E, medium_props, mass=[cst.mp](#), kmin=1e-20, q=1.5, l=1e-4)
- def [background_diffusion_coefficient.Kappa_zz](#) (E, medium_props, mass=[cst.mp](#), kmin=(50.*[cst.pc](#))*(-1), q=5./3, l=1e-4)

4.32 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_↵
_models/obsolete/background_diffusion_coefficient.py File Reference

Namespaces

- [background_diffusion_coefficient](#)

Functions

- def [background_diffusion_coefficient.J](#) (a, b, [c](#), D, q, x)

Variables

- `background_diffusion_coefficient.phase` = `ism.CNM`
- `background_diffusion_coefficient.B0` = `phase.get('B')`
- `background_diffusion_coefficient.mi` = `phase.get("mi")`
- `background_diffusion_coefficient.mn` = `phase.get("mn")`
- `background_diffusion_coefficient.ni` = `phase.get("ni")`
- `background_diffusion_coefficient.nn` = `phase.get("nn")`
- `background_diffusion_coefficient.T` = `phase.get("T")`
- `tuple background_diffusion_coefficient.chi` = `(mn*nn)/(mi*ni)`
- `background_diffusion_coefficient.VAi` = `B0/np.sqrt(4*np.pi*mi*ni)`
- `background_diffusion_coefficient.VA` = `B0/np.sqrt(4*np.pi*(mi*ni + mn*nn))`
- `int background_diffusion_coefficient.nu_in` = `2*nn*8.4e-9*(50/1e4)**0.4`
- `tuple background_diffusion_coefficient.nu_ni` = `chi**(-1.)*nu_in`
- `int background_diffusion_coefficient.k_min` = `1e-20`
- `int background_diffusion_coefficient.k_cm` = `1e-15`
- `int background_diffusion_coefficient.k_cp` = `1e-20`
- `int background_diffusion_coefficient.k_max` = `2*nu_ni/VA`
- `int background_diffusion_coefficient.E` = `10*cst.GeV`
- `background_diffusion_coefficient.m` = `cst.mp`
- `int background_diffusion_coefficient.gamma` = `1 + (E/(m*cst.c**2))`
- `background_diffusion_coefficient.v` = `cst.c*np.sqrt(1 - (1/(E/(m*cst.c**2) + 1))**2)`
- `int background_diffusion_coefficient.p` = `gamma*m*v`
- `background_diffusion_coefficient.Omega0` = `cst.e*B0/(m*cst.c)`
- `int background_diffusion_coefficient.Omega` = `Omega0/gamma`
- `background_diffusion_coefficient.mu` = `np.linspace(-0.99, 0.99, 100)`
- `background_diffusion_coefficient.d_uu` = `np.zeros(len(mu))`
- `int background_diffusion_coefficient.k_zz` = `0.`
- `background_diffusion_coefficient.dmu` = `mu[1] - mu[0]`
- `int background_diffusion_coefficient.ltot` = `1e-1`
- `float background_diffusion_coefficient.q` = `1.5`
- `tuple background_diffusion_coefficient.a` = `(v*mu[ij] - VA)**2`
- `int background_diffusion_coefficient.b` = `2*Omega*(v*mu[ij] - VA)`
- `int background_diffusion_coefficient.c` = `Omega**2 + (- nu_in/2)**2`
- `int background_diffusion_coefficient.D` = `b**2 - 4*a*c`
- `background_diffusion_coefficient.eps` = `VA/v`
- `int background_diffusion_coefficient.gs0` = `2*(q-1)*(B0**2/(8*np.pi))*ltot*k_cp**(q - 1)`
- `def background_diffusion_coefficient.fj_p` = `J(a, b, c, D, q, k_max) - J(a, b, c, D, q, k_cp)`
- `def background_diffusion_coefficient.fj_m` = `J(a, -b, c, D, q, k_max) - J(a, -b, c, D, q, k_cp)`
- `int background_diffusion_coefficient.l` = `-2*gs0*(1 - mu[ij]*eps)**2*(- nu_in/2.)*fj_p`

4.33 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/constants.py

File Reference

Namespaces

- `constants`

Variables

- float `constants.yr` = 3.154e+7
- int `constants.kyr` = 1e3*yr
- float `constants.pc` = 3.086e18
- int `constants.kpc` = 1.e3*pc
- float `constants.GeV` = 0.00160218
- int `constants.TeV` = 1e3*GeV
- float `constants.eV` = GeV*1e-9
- int `constants.MeV` = 1e-3*GeV
- float `constants.me` = 9.10938e-28
- float `constants.mp` = 1.6726219e-24
- float `constants.mn` = 1.6749286e-24
- float `constants.mHl` = mp
- float `constants.mHll` = mp
- int `constants.mHel` = 2*mp + 2*mn
- int `constants.mHell` = 2*mp + 2*mn
- int `constants.mCll` = 6*mp + 6*mn
- int `constants.mHCOll` = 8*mp + 8*mn + 6*mp + 6*mn + mp + mn
- int `constants.mH2` = 2*mp
- float `constants.e` = 4.80326e-10
- int `constants.c` = 29979245800.
- float `constants.kbolz` = 1.3807e-16
- int `constants.kms` = 1e5

4.34 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/d1_grid_generator.py File Reference

Namespaces

- `d1_grid_generator`

Functions

- def `d1_grid_generator.grid` (Smin, Smax, Ns, name, s_center=None, width=None, smooth=None, dXmin=0.01 *cst.pc)

4.35 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/damping.py File Reference

Namespaces

- `damping`

Functions

- def [damping.IN_damping_approx_2](#) (E, medium_props, [theta](#)=0)
- def [damping.IN_damping_approx_1](#) (E, medium_props, nu_n=0, [theta](#)=0)
- def [damping.IonNeutral_Damping](#) (E, medium_props, nu_n=0, [theta](#)=0)
- def [damping.indamping_alfven](#) (position_index, E, medium_props)
- def [damping.indamping_alfven_nopos](#) (E, medium_props)
- def [damping.damping_lazarian](#) (position_index, E, medium_props)
- def [damping.damping_lazarian_nopos](#) (E, medium_props)
- def [damping.non_linear_landau_damping](#) (T, lp, lm, mi, q, B0, Ecr)

4.36 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/Data_reader.py File Reference

Namespaces

- [Data_reader](#)

Functions

- def [Data_reader.readDataXE](#) (file_name, NX, NE)
- def [Data_reader.readAxis](#) (file_name)

Variables

- def [Data_reader.data](#) = readDataXE("../data_out/Pcr_0165.dat", 2**11, 2**5)
- def [Data_reader.X](#) = [readAxis](#)("../data_ini/X.dat")
- def [Data_reader.E](#) = [readAxis](#)("../data_ini/E.dat")
- [Data_reader.figsizes](#)

4.37 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/freader.py File Reference

Namespaces

- [freader](#)

Functions

- def [freader.search](#) (file_name, variable)

4.38 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/fwritter.py File Reference

Namespaces

- [fwritter](#)

Functions

- `def fwritter.search` (file_name, variable)
- `def fwritter.fileWrite` (file_name, variables={}, path='./', ext='.dat')
- `def fwritter.write1D` (file_name, nx=None, ne=None, variable=None, path=".")
- `def fwritter.write2D` (file_name, nx=None, ny=None, ne=None, variable=None, path=".")
- `def fwritter.write1Daxis` (file_name, variable=None, nx=None, path=".")

4.39 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/gaussian_subNormalization.py File Reference

Namespaces

- `gaussian_subNormalization`

Functions

- `def gaussian_subNormalization.gauss` (t, sig, mu)

Variables

- `float gaussian_subNormalization.tmin` = 0.01
- `float gaussian_subNormalization.tesc` = 25.23
- `int gaussian_subNormalization.tmax` = 2*tesc - tmin
- `int gaussian_subNormalization.sig` = 2
- `int gaussian_subNormalization.mu` = 25
- `int gaussian_subNormalization.Nc` = 1e6
- `gaussian_subNormalization.tc` = np.linspace(tmin, tmax, Nc)
- `int gaussian_subNormalization.Nv` = 100
- `gaussian_subNormalization.ta` = np.linspace(tmin, tmax, Nv)
- `int gaussian_subNormalization.r` = Nc/Nv
- `gaussian_subNormalization.gauss_c` = np.zeros(len(tc))
- `gaussian_subNormalization.gauss_a` = np.zeros(len(ta))
- `int gaussian_subNormalization.C_c` = 0.
- `int gaussian_subNormalization.C_a` = 0.
- `gaussian_subNormalization.color`
- `gaussian_subNormalization.c`

4.40 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/mathmethods.py File Reference

Namespaces

- `mathmethods`

Functions

- def [mathmethods.Cubic3](#) (a, b, c, d)
- def [mathmethods.findF](#) (a, b, c)
- def [mathmethods.findG](#) (a, b, c, d)
- def [mathmethods.findH](#) (g, f)
- def [mathmethods.cardano3](#) (a, b, c)
- def [mathmethods.histogram](#) (data, xi, xf, nbins, scale, normalization)
- def [mathmethods.g](#) (f, t)
- def [mathmethods.simpson_log](#) (f, a, b, N)
- def [mathmethods.glin](#) (f, t)
- def [mathmethods.simpson_lin](#) (f, a, b, N)
- def [mathmethods.g1](#) (x, xt, l)
- def [mathmethods.g2](#) (x, xt, l)
- def [mathmethods.f](#) (x, xt, l, v1, v2)
- def [mathmethods.shape](#) (X, Amp, Xmin=0., Xmax=1., sig_Xmin=0.1, sig_Xmax=0.2)
- def [mathmethods.multishape](#) (X, Amp, Xmin, Xmax, sig)
- def [mathmethods.SmoothPhaseTransition](#) (X, E, phases, smooth_width)

4.41 [/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/Output_](#) [_functions.py](#) File Reference

Namespaces

- [Output_functions](#)

4.42 [/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/phases_](#) [_collection.py](#) File Reference

Namespaces

- [phases_collection](#)

Functions

- def [phases_collection.ism_phase](#) (Temp, Bfiel, nion, ntot, mion, mneutral)

Variables

- def [phases_collection.HII](#) = [ism_phase](#)(8000, 10.e-6, 99.9, 100., 0.93*cst.mHII+0.07*cst.mHeII, 0.93*cst.mHI+0.07*cst.mHeI)
- def [phases_collection.WIM](#) = [ism_phase](#)(8000, 5.00001e-6, 0.315, 0.35, cst.mHII, 0.93*cst.mHI+0.07*cst.mHeI)
- def [phases_collection.WNM](#) = [ism_phase](#)(8000, 5.00001e-6, 7e-3, 0.35, cst.mHII, 0.93*cst.mHI+0.07*cst.mHeI)
- def [phases_collection.CNM](#) = [ism_phase](#)(50, 6.00001e-6, 2.3e-2, 30.0, cst.mCII, 0.93*cst.mHI+0.07*cst.mHeI)
- def [phases_collection.DiM](#) = [ism_phase](#)(50, 6.00001e-6, 3.0e-2, 300, cst.mCII, 0.93*(0.5*cst.mHI + 0.5*cst.mH2) + 0.07*cst.mHeI)
- def [phases_collection.DeM](#) = [ism_phase](#)(30, 26.0001e-6, 3.0e-2, 3000, cst.mHCOII, 0.93*cst.mH2 + 0.07*cst.mHeI)
- def [phases_collection.DeC](#) = [ism_phase](#)(20, 59.0001e-6, 1.0e-2, 1e4, cst.mHCOII, 0.93*cst.mH2 + 0.07*cst.mHeI)

4.43 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/physical_models.py File Reference ↩

Namespaces

- [physical_models](#)

Functions

- def [physical_models.collision_rate](#) (specie1, specie2, phase)
- def [physical_models.cr_escape_time_model](#) (option, model, Ecr, props)
- def [physical_models.cr_escape_radius_model](#) (option, model, time, props)

4.44 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/background_diffusion_coefficient_3.py File Reference ↩

Namespaces

- [background_diffusion_coefficient_3](#)

Functions

- def [background_diffusion_coefficient_3.IonNeutral_Damping](#) (k, medium_props, nu_n=0, theta=0)
- def [background_diffusion_coefficient_3.Duu_Alfven_Slab_Linear_Undamped](#) (mu, E, medium_props, mass=cst.mp, kmin=1e-20, q=1.5, l=1e-4)
- def [background_diffusion_coefficient_3.Kappa_zz](#) (E, medium_props, mass=cst.mp, kmin=1e-20, q=5./3, l=1e28)
- def [background_diffusion_coefficient_3.kappa_zz_BC](#) (E, d00, delta)

Variables

- float [background_diffusion_coefficient_3.Emin](#) = 0.1*cst.GeV
- int [background_diffusion_coefficient_3.Emax](#) = 100*cst.TeV
- [background_diffusion_coefficient_3.E](#) = np.logspace(np.log10(Emin), np.log10(Emax), 100)
- [background_diffusion_coefficient_3.K_zz_bc_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_bc_2](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_2](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_HII_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_WIM_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_WNM_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_CNM_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_DiM_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_DeM_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_DeC_1](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_HII_2](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_WIM_2](#) = np.zeros(len(E))
- [background_diffusion_coefficient_3.K_zz_WNM_2](#) = np.zeros(len(E))

- `background_diffusion_coefficient_3.K_zz_CNM_2 = np.zeros(len(E))`
- `background_diffusion_coefficient_3.K_zz_DiM_2 = np.zeros(len(E))`
- `background_diffusion_coefficient_3.K_zz_DeM_2 = np.zeros(len(E))`
- `background_diffusion_coefficient_3.K_zz_DeC_2 = np.zeros(len(E))`
- `background_diffusion_coefficient_3.HII`
- `background_diffusion_coefficient_3.mass`
- `background_diffusion_coefficient_3.mp`
- `background_diffusion_coefficient_3.kmin`
- `background_diffusion_coefficient_3.q`
- `background_diffusion_coefficient_3.l`
- `background_diffusion_coefficient_3.WIM`
- `background_diffusion_coefficient_3.WNM`
- `background_diffusion_coefficient_3.CNM`
- `background_diffusion_coefficient_3.DiM`
- `background_diffusion_coefficient_3.DeM`
- `background_diffusion_coefficient_3.DeC`
- `int background_diffusion_coefficient_3.size_x = 4`
- `int background_diffusion_coefficient_3.size_y = 3`
- `int background_diffusion_coefficient_3.sub_x = 2`
- `int background_diffusion_coefficient_3.sub_y = 2`
- `background_diffusion_coefficient_3.fig = plt.figure(figsize=(size_x*sub_x,size_y*sub_y))`
- `background_diffusion_coefficient_3.gs = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig)`
- `background_diffusion_coefficient_3.wspace`
- `background_diffusion_coefficient_3.hspace`
- `background_diffusion_coefficient_3.ax0 = fig.add_subplot(gs[0, 0])`
- `background_diffusion_coefficient_3.GeV`
- `background_diffusion_coefficient_3.c`
- `background_diffusion_coefficient_3.ls`
- `background_diffusion_coefficient_3.facecolor`
- `background_diffusion_coefficient_3.alpha`
- `background_diffusion_coefficient_3.hatch`
- `background_diffusion_coefficient_3.label`
- `background_diffusion_coefficient_3.loc`
- `background_diffusion_coefficient_3.ax1 = fig.add_subplot(gs[0, 1])`
- `background_diffusion_coefficient_3.ax2 = fig.add_subplot(gs[1, 0])`
- `background_diffusion_coefficient_3.ax3 = fig.add_subplot(gs[1, 1])`
- `list background_diffusion_coefficient_3.custom_lines`
- `background_diffusion_coefficient_3.handles`
- `background_diffusion_coefficient_3.bbox_to_anchor`
- `background_diffusion_coefficient_3.ncol`
- `int background_diffusion_coefficient_3.ymin = 1e27`
- `int background_diffusion_coefficient_3.ymax = 1e33`
- `int background_diffusion_coefficient_3.xmin = 1e-1`
- `int background_diffusion_coefficient_3.xmax = 1e5`

4.45 `/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/damping_models.py` File Reference ↩

Namespaces

- `damping_models`

- def [damping_models.damprate_to_damptime](#) (gamma)

Variables

- int [damping_models.NE](#) = 10
- float [damping_models.Emin](#) = 0.99*cst.GeV
- float [damping_models.Emax](#) = 100.01*cst.TeV
- string [damping_models.egridtype](#) = "logspace"
- [damping_models.E](#) = grid.grid(Emin, Emax, 2**NE, egridtype)
- list [damping_models.phases](#) = [ism.HII, ism.WIM, ism.WNM, ism.CNM, ism.DiM, ism.DeM, ism.DeC]
- list [damping_models.xlim](#) = [Emin/cst.GeV, Emax/cst.GeV]
- list [damping_models.xlims](#) = [xlim, xlim, xlim, xlim, xlim, xlim, xlim]
- list [damping_models.ylim](#) = [1e-14, 1e-3]
- list [damping_models.ylims](#) = [ylim, ylim, ylim, ylim, ylim, ylim, ylim]
- list [damping_models.Name](#) = ["HII", "WIM", "WNM", "CNM", "DiM", "DeM", "DeC"]
- int [damping_models.size_x](#) = 4
- int [damping_models.size_y](#) = 3
- int [damping_models.sub_x](#) = 2
- int [damping_models.sub_y](#) = 4
- [damping_models.fig](#) = plt.figure(figsize=(size_x*sub_x,size_y*sub_y))
- [damping_models.gs](#) = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig)
- [damping_models.wspace](#)
- [damping_models.hspace](#)
- list [damping_models.pos_1](#) = [0, 0, 1, 1, 2, 2, 3, 3]
- list [damping_models.pos_2](#) = [0, 1, 0, 1, 0, 1, 0, 1]
- [damping_models.wR_Alfven](#) = np.zeros(len(E))
- [damping_models.wl_Alfven](#) = np.zeros(len(E))
- [damping_models.wR_Alfven_o1](#) = np.zeros(len(E))
- [damping_models.wl_Alfven_o1](#) = np.zeros(len(E))
- [damping_models.wR_Alfven_o2](#) = np.zeros(len(E))
- [damping_models.wl_Alfven_o2](#) = np.zeros(len(E))
- [damping_models.Ep](#) = np.NaN
- [damping_models.Em](#) = np.NaN
- [damping_models.Gamma_lz](#) = np.zeros(len(E))
- [damping_models.Gamma_nlld_inf](#) = np.zeros(len(E))
- [damping_models.Gamma_nlld_sup](#) = np.zeros(len(E))
- [damping_models.in_damping](#) = dp.IonNeutral_Damping(E[e], phases[pi], nu_n = 0, theta = 0)
- [damping_models.lz_damping](#) = dp.damping_lazarian_nopos(E[e], phases[pi])
- [damping_models.lz_min](#) = lz_damping[1]
- int [damping_models.linf](#) = 1e-4
- int [damping_models.lsup](#) = 1e-1
- [damping_models.ax](#) = fig.add_subplot(gs[pos_1[pi], pos_2[pi]])
- [damping_models.GeV](#)
- [damping_models.c](#)
- [damping_models.ls](#)
- [damping_models.lw](#)
- [damping_models.alpha](#)
- [damping_models.color](#)
- [damping_models.facecolor](#)
- [damping_models.x](#)
- [damping_models.y](#)
- [damping_models.label](#)
- [damping_models.loc](#)
- [damping_models.bbox_to_anchor](#)
- [damping_models.bbox_inches](#)
- [damping_models.pad_inches](#)

4.46 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_↵ _models/electrons_emax_t.py File Reference

Namespaces

- [electrons_emax_t](#)

Functions

- def [electrons_emax_t.InverseTrigonalMatrix](#) (T)
FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- def [electrons_emax_t.ProductMatrix](#) (A, B)
- def [electrons_emax_t.InterpolatingSpline](#) (X, Y)
- def [electrons_emax_t.Rsh](#) (nt)
- def [electrons_emax_t.B](#) (t, ts, tB, alpha_B, BISM, Bfree)
ELECTRON ESCAPE MODEL (from Ohira et al.
- def [electrons_emax_t.eta_g](#) (t, ts, tB, [alpha](#), alpha_B, eta_free)
- def [electrons_emax_t.Em_age](#) (t, ts, tB, alpha_B, [alpha](#), BISM, Bfree, [eta_acc](#), eta_free, Rs)
- def [electrons_emax_t.Em_cool](#) (t, ts, tB, alpha_B, [alpha](#), BISM, Bfree, Ems)
- def [electrons_emax_t.Em_esc](#) (t, ts, tB, alpha_B, [alpha](#), BISM, Bfree, [eta_acc](#), eta_free, eta_esc, Rs)
- def [electrons_emax_t.Emax_electrons](#) (time)
- def [electrons_emax_t.escape_time](#) (E, tSed, EM, [delta](#))

Variables

- float [electrons_emax_t.alpha](#) = 2.6
- int [electrons_emax_t.alpha_B](#) = 9./10
- int [electrons_emax_t.xhi_m](#) = 1
- float [electrons_emax_t.xhi_cr](#) = 0.1
- int [electrons_emax_t.E51](#) = 1
- int [electrons_emax_t.Mej](#) = 1
- int [electrons_emax_t.C06](#) = 1
- int [electrons_emax_t.beta](#) = 1
- int [electrons_emax_t.phi_c](#) = 1
- int [electrons_emax_t.vej8](#) = 10.*(E51/Mej)**(0.5)
- float [electrons_emax_t.nt](#) = 0.35
- float [electrons_emax_t.tsed](#) = 0.3*E51**(-0.5)*Mej*nt**(-1./3)*cst.kyr
FUNCTIONS IN ORDER TO MAKE OUR SNR EXPAND IN THE ISM #.
- def [electrons_emax_t.EM](#) = Emax_electrons(tsed)
- [electrons_emax_t.E](#) = np.logspace(np.log10(1*cst.GeV), np.log10(100*cst.TeV), num=100)
- [electrons_emax_t.tesc](#) = np.zeros(len(E))

4.47 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_↵ _models/EscapeModel_protons.py File Reference

Namespaces

- [EscapeModel_protons](#)

- def `EscapeModel_protons.InverseTrigonalMatrix` (T)
 FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- def `EscapeModel_protons.ProductMatrix` (A, B)
- def `EscapeModel_protons.InterpolatingSpline` (X, Y)
- def `EscapeModel_protons.f1` (x, const)
- def `EscapeModel_protons.df1dx` (x, const)
- def `EscapeModel_protons.f2` (x, const)
- def `EscapeModel_protons.df2dx` (x, const)
- def `EscapeModel_protons.NewtonRaphson` (f, df, x0, eps, const)
- def `EscapeModel_protons.Gettesc` (E, delta)

Variables

- float `EscapeModel_protons.nt` = 0.35
- int `EscapeModel_protons.xhi_m` = 1
- float `EscapeModel_protons.xhi_cr` = 0.1
- int `EscapeModel_protons.E51` = 1
- int `EscapeModel_protons.Mej` = 1
- int `EscapeModel_protons.C06` = 1
- int `EscapeModel_protons.beta` = 1
- int `EscapeModel_protons.phi_c` = 1
- int `EscapeModel_protons.vej8` = 10.*(E51/Mej)**(0.5)
- int `EscapeModel_protons.tini` = 1e-4*cst.kyr
 FUNCTIONS IN ORDER TO MAKE OUR SNR EXPAND IN THE ISM #.
- float `EscapeModel_protons.tfree` = 0.3*E51**(-0.5)*Mej*nt**(-1./3)*cst.kyr
- float `EscapeModel_protons.tPDS` = np.exp(-1.)*3.61e4*E51**(3./14)/(xhi_m**(5./14)*nt**(4./7))*cst.yr
- float `EscapeModel_protons.tMCS` = min(61*vej8**3/(xhi_m**(9./14)*nt**(3./7)*E51**(3./14)), 476./(xhi_m*phi_c)**(9./14))*tPDS
- int `EscapeModel_protons.tmerge` = 153.*(E51**(1./14)*nt**(1./7)*xhi_m**(3./14)/(beta*C06))**(10./7)*tPDS
- `EscapeModel_protons.tmax` = min(tMCS, tmerge)
- float `EscapeModel_protons.R_free` = 5.0*(E51/nt)**(1./5)*(1 - (0.05*Mej**(5./6))/(E51**0.5*nt**(1./3)*(tfree/cst.kyr))**(2./5)*(tfree/cst.kyr)**(2./5)*cst.pc
- float `EscapeModel_protons.R_ini` = R_free*(tini/tfree)**(1.)
- float `EscapeModel_protons.R_PDS` = 5.0*(E51/nt)**(1./5)*(1 - (0.05*Mej**(5./6))/(E51**0.5*nt**(1./3)*(tPDS/cst.kyr))**(2./5)*(tPDS/cst.kyr)**(2./5)*cst.pc
- float `EscapeModel_protons.R_MCS` = R_PDS*(tMCS/tPDS)**(3./10)
- float `EscapeModel_protons.R_merge` = R_MCS*(tmerge/tMCS)**(1./4)
- `EscapeModel_protons.t` = np.array([tini, tfree, tPDS, tMCS, tmerge])
- `EscapeModel_protons.R` = np.array([R_ini, R_free, R_PDS, R_MCS, R_merge])
- `EscapeModel_protons.logt` = np.empty(len(t))
- `EscapeModel_protons.logR` = np.empty(len(R))
- def `EscapeModel_protons.f_SNR` = `InterpolatingSpline`(logt, logR)
- `EscapeModel_protons.logt_new` = np.linspace(logt[0], logt[-1], 100)
- `EscapeModel_protons.logr_new` = np.empty(len(logt_new))
- `EscapeModel_protons.t_new` = np.empty(len(logt_new))
- `EscapeModel_protons.r_new` = np.empty(len(logr_new))
- `EscapeModel_protons.u_sh` = np.empty(len(r_new))
- float `EscapeModel_protons.gamma` = 2.2
- float `EscapeModel_protons.Emin` = 0.1*cst.GeV
- `EscapeModel_protons.Emax` = np.empty(len(t_new))
- int `EscapeModel_protons.eps` = 1e-4

- `int EscapeModel_protons.x0 = 10.*cst.GeV`
- `float EscapeModel_protons.a = Emin`
- `int EscapeModel_protons.b = beta`
- `tuple EscapeModel_protons.c = (beta/(1+beta))*cst.e*np.sqrt(4*np.pi*nt*cst.mp)/(10.*cst.c)*xhi_cr*u_↵
sh[iii]**2*r_new[iii]`
- `EscapeModel_protons.niter`
- `EscapeModel_protons.EMAX = max(Emax)`
- `int EscapeModel_protons.delta = 2.`
- `float EscapeModel_protons.tSed = tfree`
- `EscapeModel_protons.Ecr = np.logspace(np.log10(0.1*cst.GeV), np.log10(100.*cst.TeV), 100)`
- `EscapeModel_protons.tesc = np.empty(len(Ecr))`
- `EscapeModel_protons.figsize`
Model figure #.
- `EscapeModel_protons.pc`
- `EscapeModel_protons.marker`
- `EscapeModel_protons.lw`
- `EscapeModel_protons.label`
- `EscapeModel_protons.GeV`
- `EscapeModel_protons.kyr`

4.48 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_↵ _models/EscapeModel_protons_2.py File Reference

Namespaces

- `EscapeModel_protons_2`

Functions

- `def EscapeModel_protons_2.InverseTrigonalMatrix (T)`
FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- `def EscapeModel_protons_2.ProductMatrix (A, B)`
- `def EscapeModel_protons_2.InterpolatingSpline (X, Y)`
- `def EscapeModel_protons_2.f1 (x, const)`
- `def EscapeModel_protons_2.df1dx (x, const)`
- `def EscapeModel_protons_2.f2 (x, const)`
- `def EscapeModel_protons_2.df2dx (x, const)`
- `def EscapeModel_protons_2.NewtonRaphson (f, df, x0, eps, const)`
- `def EscapeModel_protons_2.getSNR (phase, size=100)`
- `def EscapeModel_protons_2.Gettesc (E, delta, tSed, EMAX, Emin=0.1 *cst.GeV)`
- `def EscapeModel_protons_2.getEmax (t_new, u_sh, r_new, phase, gamma=2.2, Emin=0.1 *cst.GeV,
eps=1e-4, x0=10.*cst.GeV)`

- `EscapeModel_protons_2.Ecr = np.logspace(np.log10(0.1*cst.GeV), np.log10(100.*cst.TeV), 1000)`
- `def EscapeModel_protons_2.SNR_HII = getSNR(ism.HII, size = 100)`
- `def EscapeModel_protons_2.emax_HII`
- `def EscapeModel_protons_2.SNR_WIM = getSNR(ism.WIM, size = 100)`
- `def EscapeModel_protons_2.emax_WIM`
- `def EscapeModel_protons_2.SNR_WNM = getSNR(ism.WNM, size = 100)`
- `def EscapeModel_protons_2.emax_WNM`
- `def EscapeModel_protons_2.SNR_CNM = getSNR(ism.CNM, size = 100)`
- `def EscapeModel_protons_2.emax_CNM`
- `def EscapeModel_protons_2.SNR_DiM = getSNR(ism.DiM, size = 100)`
- `def EscapeModel_protons_2.emax_DiM`
- `int EscapeModel_protons_2.delta = 2.`
- `EscapeModel_protons_2.tesc_HII = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_WIM = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_WNM = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_CNM = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_DiM = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_HII_3 = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_WIM_3 = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_WNM_3 = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_CNM_3 = np.empty(len(Ecr))`
- `EscapeModel_protons_2.tesc_DiM_3 = np.empty(len(Ecr))`
- `int EscapeModel_protons_2.size_x = 4`
- `float EscapeModel_protons_2.size_y = 3.5`
- `int EscapeModel_protons_2.sub_x = 2`
- `int EscapeModel_protons_2.sub_y = 1`
- `EscapeModel_protons_2.fig = plt.figure(figsize=(size_x*sub_x,size_y*sub_y))`
- `EscapeModel_protons_2.gs = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig)`
- `EscapeModel_protons_2.wspace`
- `EscapeModel_protons_2.hspace`
- `EscapeModel_protons_2.ax0 = fig.add_subplot(gs[0])`
- `EscapeModel_protons_2.GeV`
- `EscapeModel_protons_2.c`
- `EscapeModel_protons_2.label`
- `EscapeModel_protons_2.loc`
- `EscapeModel_protons_2.ncol`
- `EscapeModel_protons_2.bbox_to_anchor`
- `EscapeModel_protons_2.ax1 = fig.add_subplot(gs[1])`
- `EscapeModel_protons_2.kyr`
- `EscapeModel_protons_2.ls`
- `EscapeModel_protons_2.pad`

4.49 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_↵ _models/obsolete/background_diffusion_coefficient_2.py File Reference

Namespaces

- `background_diffusion_coefficient_2`

Functions

- def `background_diffusion_coefficient_2.IonNeutral_Damping` (k, medium_props, nu_n=0, theta=0)
- def `background_diffusion_coefficient_2.R` (k, medium_props, particles_props, mu, kind="+")
- def `background_diffusion_coefficient_2.g_s` (k, kmin, q, medium_props)

Variables

- `background_diffusion_coefficient_2.medium_props` = ism.WNM
- float `background_diffusion_coefficient_2.E` = 0.001*cst.GeV
- `background_diffusion_coefficient_2.m` = cst.mp
- int `background_diffusion_coefficient_2.gamma` = 1 + (E/(m*cst.c**2))
- `background_diffusion_coefficient_2.v` = cst.c*np.sqrt(1 - (1/(E/(m*cst.c**2) + 1))**2)
- int `background_diffusion_coefficient_2.p` = gamma*m*v
- `background_diffusion_coefficient_2.Omega0` = cst.e*medium_props.get("B")/(m*cst.c)
- int `background_diffusion_coefficient_2.Omega` = Omega0/gamma
- dictionary `background_diffusion_coefficient_2.particles_props` = {"v":v, "Omega":Omega}
- `background_diffusion_coefficient_2.mu` = np.linspace(-0.99, 0.99, 100)
- `background_diffusion_coefficient_2.k` = np.logspace(-20, -10, 100)
- `background_diffusion_coefficient_2.D_uu` = np.zeros(len(mu))
- int `background_diffusion_coefficient_2.kmin` = 1e-17
- float `background_diffusion_coefficient_2.q` = 1.5
- int `background_diffusion_coefficient_2.k_zz` = 0.
- float `background_diffusion_coefficient_2.dk` = 0.5*(k[ii+1] - k[ii-1])
- `background_diffusion_coefficient_2.B0` = medium_props.get("B")
- def `background_diffusion_coefficient_2.w` = IonNeutral_Damping(k[ii], medium_props)
- def `background_diffusion_coefficient_2.wtot` = w.get("wr") + 1j*w.get("wi")
- float `background_diffusion_coefficient_2.l` = dk*g_s(k[ii], kmin, q, medium_props)*(1 - (mu[jj]*wtot)/(k[ii]*v))**2

4.50 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/show_models/SNR_evolution.py File Reference ↩

Namespaces

- `SNR_evolution`

Functions

- def `SNR_evolution.InverseTrigonalMatrix` (T)
FUNCTIONS IN ORDER TO CREATE A GOOD SPLINE !!! #.
- def `SNR_evolution.ProductMatrix` (A, B)
- def `SNR_evolution.InterpolatingSpline` (X, Y)
- def `SNR_evolution.Rsh` (nt)

Variables

- list [SNR_evolution.marker](#) = ['X', 'o', 's', 'v']
- int [SNR_evolution.size_x](#) = 6
- int [SNR_evolution.size_y](#) = 4
- int [SNR_evolution.sub_x](#) = 1
- int [SNR_evolution.sub_y](#) = 2
- [SNR_evolution.fig](#) = plt.figure(figsize=(size_x*sub_x,size_y*sub_y))
- [SNR_evolution.gs](#) = gridspec.GridSpec(ncols= sub_x, nrows = sub_y, figure = fig)
- [SNR_evolution.wspace](#)
- [SNR_evolution.hspace](#)
- [SNR_evolution.ax0](#) = fig.add_subplot(gs[0])
- [SNR_evolution.pc](#)
- [SNR_evolution.c](#)
- [SNR_evolution.label](#)
- [SNR_evolution.lw](#)
- [SNR_evolution.loc](#)
- [SNR_evolution.ncol](#)
- [SNR_evolution.bbox_to_anchor](#)
- [SNR_evolution.ax1](#) = fig.add_subplot(gs[1])
- [SNR_evolution.kms](#)

4.51 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/ShowInjectionEvolution.py File Reference ↩

Namespaces

- [ShowInjectionEvolution](#)

Functions

- def [ShowInjectionEvolution.readDataXE](#) (file_name, NX, NE)
- def [ShowInjectionEvolution.readAxis](#) (file_name)

Variables

- [ShowInjectionEvolution.index](#) = np.logspace(1, np.log10(900), 10)
- [ShowInjectionEvolution.figsize](#)
- def [ShowInjectionEvolution.X](#) = [readAxis](#)("../data_ini/X.dat")
- def [ShowInjectionEvolution.E](#) = [readAxis](#)("../data_ini/E.dat")
- string [ShowInjectionEvolution.loc_id](#) = ""
- def [ShowInjectionEvolution.data](#) = [readDataXE](#)("../data_out/Pcr_"+loc_id+".dat", 2**11, 2**5)

4.52 /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_diffusion/CR_SPECTRA/tools/test_tesc.py File Reference ↩

Namespaces

- [test_tesc](#)

Functions

- def `test_tesc.tesc` (E)
- def `test_tesc.gauss` (t, sig, mu)
- def `test_tesc.sig` (t)

Variables

- float `test_tesc.GeV` = 0.00160218
- int `test_tesc.kyr` = 1e3*24*60*60*365.25
- float `test_tesc.rho_0` = 0.35
- float `test_tesc.e` = 4.8032e-10
- float `test_tesc.c` = 2.998e10
- float `test_tesc.xhi_cr` = 0.1
- float `test_tesc.xhi_0` = 2.026
- float `test_tesc.beta` = 0.2
- int `test_tesc.Esn` = 1e51
- float `test_tesc.Emin` = 0.1*GeV
- `test_tesc.E` = np.logspace(np.log10(10.*GeV), np.log10(1e3*GeV), 100)
- `test_tesc.t` = np.linspace(0.01*kyr, 1e3*kyr, 10000)
- `test_tesc.Qcr` = np.empty((len(E), len(t)))
- `test_tesc.figsizes`
- `test_tesc.label`

Index

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/data_analysis/pcr_↔
_ip_2D.py, 185

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/data_analysis/show_↔
_data.py, 186

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/namelist.py, 186

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/fiducial_↔
_tests/phases_energy_dependance/namelist_↔
_uniform.py, 187

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/fiducial_↔
_tests/phases_energy_dependance/setup_↔
uniform.py, 188

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/namelist_↔
_adv.py, 189

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/namelist_↔
_adve.py, 190

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/namelist_↔
_advst.py, 191

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/namelist_↔
_diff.py, 192

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/namelist_↔
_vdiff.py, 193

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/setup_↔
_adv.py, 194

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/setup_↔
_adve.py, 195

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/setup_↔
_advst.py, 196

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/setup_↔
_diff.py, 197

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/other_setups/setup_↔
_vdiff.py, 198

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/setup.py, 199

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/constants.h, 201

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/cr_source.h, 219

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/freader.h, 227

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/fwriter.h, 228

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/logmaker.h, 230

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/main.cpp, 231

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/mathematics.h, 232

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/others/PDE_↔
solvers.py, 237

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/others/Split_↔
solvers.py, 238

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/others/cr_↔
escape.cpp, 235

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/out.h, 239

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/read2D.h, 241

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/solver1D.h, 242

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/src/tools.h, 247

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/Data_reader_↔
py, 252

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/Output_↔
functions.py, 254

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/ShowInjection_↔
Evolution.py, 263

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/background_↔
diffusion_coefficient.py, 249

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/constants.py, 250

/home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/d1_grid_↔
generator.py, 251

- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/damping.py, 251
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/freader.py, 252
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/fwritter.py, 252
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/gaussian_↔
subNormalization.py, 253
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/mathmethods.↔
py, 253
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/phases_↔
collection.py, 254
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/physical_↔
models.py, 255
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/EscapeModel_protons.py, 258
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/EscapeModel_protons_2.py, 260
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/SNR_evolution.py, 262
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/background_diffusion_coefficient_↔
_3.py, 255
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/damping_models.py, 256
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/electrons_emax_t.py, 258
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/obsolete/background_diffusion_↔
coefficient.py, 249
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/show_↔
models/obsolete/background_diffusion_↔
coefficient_2.py, 261
- /home/lbrahimi/TRAVAIL/THESE/Travaux/CR_↔
diffusion/CR_SPECTRA/tools/test_tesc.py, 263
- A
 - PDE_solvers, 122
- a
 - background_diffusion_coefficient, 7
 - EscapeModel_protons, 51
- absmaxElement2D
 - tools.h, 248
- advectionSolverE1
 - solver1D.h, 244
- advectionSolverE2
 - solver1D.h, 244
- advectionSolverE
 - solver1D.h, 244
- advectionSolverX
 - solver1D.h, 245
- alpha
 - background_diffusion_coefficient_3, 19
 - constants.h, 205
 - damping_models, 35
 - electrons_emax_t, 46
- alpha_B
 - electrons_emax_t, 46
- ax
 - damping_models, 35
 - pcr_ip_2D, 118
- ax0
 - background_diffusion_coefficient_3, 19
 - EscapeModel_protons_2, 61
 - pcr_ip_2D, 118
 - SNR_evolution, 176
- ax1
 - background_diffusion_coefficient_3, 19
 - EscapeModel_protons_2, 61
 - pcr_ip_2D, 118
 - SNR_evolution, 176
- ax2
 - background_diffusion_coefficient_3, 20
- ax3
 - background_diffusion_coefficient_3, 20
- B
 - electrons_emax_t, 43
 - namelist, 76
 - namelist_adv, 83
 - namelist_adve, 88
 - namelist_advst, 94
 - namelist_diff, 100
 - namelist_uniform, 105
 - namelist_vdiff, 112
 - PDE_solvers, 122
 - setup, 130
 - setup_adv, 137
 - setup_adve, 142
 - setup_advst, 148
 - setup_diff, 154
 - setup_uniform, 159
 - setup_vdiff, 166
- b
 - background_diffusion_coefficient, 8
 - EscapeModel_protons, 51
- B0
 - background_diffusion_coefficient, 8
 - background_diffusion_coefficient_2, 14
- B_sat
 - cr_source.h, 221
- background_diffusion_coefficient, 5
 - a, 7
 - b, 8

- B0, [8](#)
- c, [8](#)
- chi, [8](#)
- D, [8](#)
- d_uu, [8](#)
- dmu, [8](#)
- Duu_Alfven_Slab_Linear_Undamped, [6](#)
- E, [8](#)
- eps, [9](#)
- fj_m, [9](#)
- fj_p, [9](#)
- gamma, [9](#)
- gs0, [9](#)
- I, [9](#)
- IonNeutral_Damping, [6](#)
- ltot, [9](#)
- J, [7](#)
- k_cm, [9](#)
- k_cp, [10](#)
- k_max, [10](#)
- k_min, [10](#)
- k_zz, [10](#)
- Kappa_zz, [7](#)
- m, [10](#)
- mi, [10](#)
- mn, [10](#)
- mu, [10](#)
- ni, [11](#)
- nn, [11](#)
- nu_in, [11](#)
- nu_ni, [11](#)
- Omega, [11](#)
- Omega0, [11](#)
- p, [11](#)
- phase, [11](#)
- q, [12](#)
- T, [12](#)
- v, [12](#)
- VAi, [12](#)
- VA, [12](#)
- background_diffusion_coefficient_2, [12](#)
- B0, [14](#)
- D_uu, [14](#)
- dk, [14](#)
- E, [14](#)
- g_s, [13](#)
- gamma, [14](#)
- I, [14](#)
- IonNeutral_Damping, [13](#)
- k, [15](#)
- k_zz, [15](#)
- kmin, [15](#)
- m, [15](#)
- medium_props, [15](#)
- mu, [15](#)
- Omega, [15](#)
- Omega0, [15](#)
- p, [16](#)
- particles_props, [16](#)
- q, [16](#)
- R, [13](#)
- v, [16](#)
- w, [16](#)
- wtot, [16](#)
- background_diffusion_coefficient_3, [16](#)
- alpha, [19](#)
- ax0, [19](#)
- ax1, [19](#)
- ax2, [20](#)
- ax3, [20](#)
- bbox_to_anchor, [20](#)
- c, [20](#)
- CNM, [20](#)
- custom_lines, [20](#)
- DeC, [20](#)
- DeM, [21](#)
- DiM, [21](#)
- Duu_Alfven_Slab_Linear_Undamped, [18](#)
- E, [21](#)
- Emax, [21](#)
- Emin, [21](#)
- facecolor, [21](#)
- fig, [21](#)
- GeV, [21](#)
- gs, [22](#)
- HII, [22](#)
- handles, [22](#)
- hatch, [22](#)
- hspace, [22](#)
- I, [22](#)
- IonNeutral_Damping, [18](#)
- K_zz_1, [22](#)
- K_zz_2, [23](#)
- K_zz_CNM_1, [23](#)
- K_zz_CNM_2, [23](#)
- K_zz_DeC_1, [23](#)
- K_zz_DeC_2, [23](#)
- K_zz_DeM_1, [23](#)
- K_zz_DeM_2, [24](#)
- K_zz_DiM_1, [24](#)
- K_zz_DiM_2, [24](#)
- K_zz_HII_1, [24](#)
- K_zz_HII_2, [24](#)
- K_zz_WIM_1, [24](#)
- K_zz_WIM_2, [24](#)
- K_zz_WNM_1, [24](#)
- K_zz_WNM_2, [25](#)
- K_zz_bc_1, [23](#)
- K_zz_bc_2, [23](#)
- Kappa_zz, [18](#)
- kappa_zz_BC, [19](#)
- kmin, [25](#)
- label, [25](#)
- loc, [25](#)
- ls, [25](#)
- mass, [25](#)

- mp, 25
- ncol, 25
- q, 26
- size_x, 26
- size_y, 26
- sub_x, 26
- sub_y, 26
- WIM, 26
- WNM, 26
- wspace, 26
- xmax, 27
- xmin, 27
- ymax, 27
- ymin, 27
- bbeta
 - constants.h, 205
- bbox_inches
 - damping_models, 35
- bbox_to_anchor
 - background_diffusion_coefficient_3, 20
 - damping_models, 35
 - EscapeModel_protons_2, 61
 - SNR_evolution, 176
- Bcenter
 - cr_source.h, 224
- bdiff_model
 - namelist, 76
 - namelist_adv, 83
 - namelist_adve, 88
 - namelist_advst, 94
 - namelist_diff, 100
 - namelist_uniform, 106
 - namelist_vdiff, 112
- beta
 - electrons_emax_t, 46
 - EscapeModel_protons, 51
 - test_tesc, 182
- box_center
 - namelist, 77
 - namelist_adv, 83
 - namelist_adve, 88
 - namelist_advst, 94
 - namelist_diff, 100
 - namelist_uniform, 106
 - namelist_vdiff, 112
- C
 - PDE_solvers, 123
- c
 - background_diffusion_coefficient, 8
 - background_diffusion_coefficient_3, 20
 - constants, 28
 - constants.h, 205
 - damping_models, 35
 - EscapeModel_protons, 51
 - EscapeModel_protons_2, 61
 - gaussian_subNormalization, 69
 - PDE_solvers, 124
 - SNR_evolution, 176
 - Split_solvers, 179
 - test_tesc, 182
- C06
 - constants.h, 205
 - electrons_emax_t, 46
 - EscapeModel_protons, 51
- C_a
 - gaussian_subNormalization, 69
- C_c
 - gaussian_subNormalization, 70
- CC70Solver
 - PDE_solvers, 123
- CNM
 - background_diffusion_coefficient_3, 20
 - phases_collection, 127
- CRsInjectionSourceSolver
 - solver1D.h, 245
- cardano3
 - mathmethods, 72
- cax
 - pcr_ip_2D, 118
- chi
 - background_diffusion_coefficient, 8
- cmap
 - pcr_ip_2D, 118
- coherence_length
 - constants.h, 205
- collision_rate
 - physical_models, 128
- color
 - damping_models, 35
 - gaussian_subNormalization, 70
- colorArray
 - show_data, 171
- colorFader
 - show_data, 171
- constants, 27
 - c, 28
 - e, 28
 - eV, 28
 - GeV, 28
 - kbolz, 28
 - kms, 28
 - kpc, 28
 - kyr, 28
 - mCII, 29
 - mH2, 29
 - mHCOII, 29
 - mHII, 30
 - mHell, 29
 - mHel, 29
 - mHI, 29
 - me, 29
 - MeV, 29
 - mn, 30
 - mp, 30
 - pc, 30
 - TeV, 30

- yr, [30](#)
- constants.h
 - alpha, [205](#)
 - bbeta, [205](#)
 - c, [205](#)
 - C06, [205](#)
 - coherence_length, [205](#)
 - delta, [205](#)
 - delta_log_output, [205](#)
 - e, [206](#)
 - Eknee, [206](#)
 - electron_injection_rate, [206](#)
 - Emin, [206](#)
 - Esn, [206](#)
 - eta_acc, [206](#)
 - eta_gfree, [207](#)
 - eV, [207](#)
 - gam, [207](#)
 - GeV, [207](#)
 - inj_exp_alpha, [207](#)
 - injection_cutoff, [207](#)
 - injection_function_norm, [207](#)
 - injection_function_width, [208](#)
 - injection_shape_time, [208](#)
 - isotropy, [208](#)
 - kbolz, [208](#)
 - km, [208](#)
 - kms, [208](#)
 - kpc, [209](#)
 - kyr, [209](#)
 - log_first_data, [209](#)
 - mCII, [209](#)
 - mH2, [210](#)
 - mHII, [210](#)
 - mHeI, [210](#)
 - mHI, [210](#)
 - me, [209](#)
 - Mej, [209](#)
 - MeV, [210](#)
 - mn, [210](#)
 - mp, [211](#)
 - nproc, [211](#)
 - number_out_data, [211](#)
 - oh_model, [211](#)
 - output_freq, [211](#)
 - pc, [211](#)
 - phi_c, [212](#)
 - pi, [212](#)
 - r_snr_thickness, [212](#)
 - set_background, [212](#)
 - sig_T, [212](#)
 - sigma_coherence, [212](#)
 - solver_Dilution, [213](#)
 - solver_ImAdvection, [213](#)
 - solver_ImDampGrowth, [213](#)
 - solver_ImSource1, [213](#)
 - solver_IpAdvection, [213](#)
 - solver_IpDampGrowth, [213](#)
 - solver_IpSource1, [214](#)
 - solver_PcrAdvection, [214](#)
 - solver_PcrAdvection2, [214](#)
 - solver_PcrAdvectionE, [214](#)
 - solver_PcrDiffusion, [214](#)
 - solver_PcrPerpDiff, [214](#)
 - solver_PcrSource1, [215](#)
 - solver_PcrSource2, [215](#)
 - solver_PeAdvection, [215](#)
 - solver_PeAdvection2, [215](#)
 - solver_PeAdvectionE1, [215](#)
 - solver_PeAdvectionE2, [216](#)
 - solver_PeAdvectionE, [215](#)
 - solver_PeDiffusion, [216](#)
 - solver_PePerpDiff, [216](#)
 - solver_PeSource1, [216](#)
 - solver_PeSource2, [216](#)
 - source_terms_exact, [216](#)
 - step_implicit, [217](#)
 - t_data_out_max, [217](#)
 - t_data_out_min, [217](#)
 - t_end_injection, [217](#)
 - t_start_injection, [217](#)
 - tesc_model, [217](#)
 - TeV, [218](#)
 - time_distrib_of_data, [218](#)
 - Tmax, [218](#)
 - ttau_sat, [218](#)
 - verbose, [218](#)
 - xhi_0, [218](#)
 - xhi_cr, [219](#)
 - xi_n, [219](#)
 - yr, [219](#)
- cr_escape.cpp
 - df1dx, [236](#)
 - df2dx, [236](#)
 - f1, [236](#)
 - f2, [236](#)
 - GetMax, [236](#)
 - GetTesc, [236](#)
 - InterpolatingSpline, [236](#)
 - InverseTrigonalMatrix, [237](#)
 - main, [237](#)
 - NewtonRaphson, [237](#)
 - ProductMatrix, [237](#)
- cr_escape_radius_model
 - physical_models, [129](#)
- cr_escape_time_model
 - physical_models, [129](#)
- cr_source.h
 - B_sat, [221](#)
 - Bcenter, [224](#)
 - dNdE, [221](#)
 - ff, [221](#)
 - Finj, [221](#)
 - GetEM_e, [222](#)
 - GetEM, [222](#)
 - GetTSed, [222](#)

- m_neutral, [224](#)
- ne, [224](#)
- ni, [225](#)
- nt, [225](#)
- nx, [225](#)
- parameters, [225](#)
- Pcr_ini, [222](#)
- RSNR, [222](#)
- Resc, [222](#)
- sBcenter, [225](#)
- scenter, [225](#)
- scenter_index, [225](#)
- sigm, [223](#)
- smn, [225](#)
- sne, [226](#)
- sni, [226](#)
- snx, [226](#)
- sT, [226](#)
- sX, [226](#)
- T, [226](#)
- tesc, [223](#)
- tesc_e, [224](#)
- theta, [224](#)
- u_sh, [224](#)
- x_center, [226](#)
- x_center_index, [226](#)
- Xi, [227](#)
- Cubic3
 - mathmethods, [72](#)
- custom_lines
 - background_diffusion_coefficient_3, [20](#)
- D
 - background_diffusion_coefficient, [8](#)
 - setup, [130](#)
 - setup_adv, [137](#)
 - setup_adve, [142](#)
 - setup_advst, [148](#)
 - setup_diff, [154](#)
 - setup_uniform, [159](#)
 - setup_vdiff, [166](#)
 - Split_solvers, [179](#)
- d00
 - setup, [130](#)
 - setup_adv, [137](#)
 - setup_adve, [143](#)
 - setup_advst, [148](#)
 - setup_diff, [154](#)
 - setup_uniform, [159](#)
 - setup_vdiff, [166](#)
- d1_grid_generator, [30](#)
- grid, [31](#)
- D_uu
 - background_diffusion_coefficient_2, [14](#)
- d_uu
 - background_diffusion_coefficient, [8](#)
- dNdE
 - cr_source.h, [221](#)
- damping, [31](#)
- damping_lazarian, [31](#)
- damping_lazarian_nopos, [31](#)
- IN_damping_approx_1, [31](#)
- IN_damping_approx_2, [32](#)
- indamping_alfven, [32](#)
- indamping_alfven_nopos, [32](#)
- IonNeutral_Damping, [33](#)
- non_linear_landau_damping, [33](#)
- damping_lazarian
 - damping, [31](#)
- damping_lazarian_nopos
 - damping, [31](#)
- damping_models, [33](#)
 - alpha, [35](#)
 - ax, [35](#)
 - bbox_inches, [35](#)
 - bbox_to_anchor, [35](#)
 - c, [35](#)
 - color, [35](#)
 - damprate_to_damptime, [34](#)
 - E, [35](#)
 - egridtype, [35](#)
 - Em, [36](#)
 - Emax, [36](#)
 - Emin, [36](#)
 - Ep, [36](#)
 - facecolor, [36](#)
 - fig, [36](#)
 - Gamma_lz, [36](#)
 - Gamma_nlld_inf, [36](#)
 - Gamma_nlld_sup, [37](#)
 - GeV, [37](#)
 - gs, [37](#)
 - hspace, [37](#)
 - linf, [37](#)
 - in_damping, [37](#)
 - lsup, [37](#)
 - label, [37](#)
 - loc, [38](#)
 - ls, [38](#)
 - lw, [38](#)
 - lz_damping, [38](#)
 - lz_min, [38](#)
 - Name, [38](#)
 - NE, [38](#)
 - pad_inches, [38](#)
 - phases, [39](#)
 - pos_1, [39](#)
 - pos_2, [39](#)
 - size_x, [39](#)
 - size_y, [39](#)
 - sub_x, [39](#)
 - sub_y, [39](#)
 - wl_Alfven, [39](#)
 - wl_Alfven_o1, [40](#)
 - wl_Alfven_o2, [40](#)
 - wR_Alfven, [40](#)
 - wR_Alfven_o1, [40](#)

- wR_Alfven_o2, [40](#)
- wspace, [40](#)
- x, [40](#)
- xlim, [40](#)
- xlims, [41](#)
- y, [41](#)
- ylim, [41](#)
- ylims, [41](#)
- damprate_to_damptime
 - damping_models, [34](#)
- data
 - Data_reader, [42](#)
 - ShowInjectionEvolution, [173](#)
- Data_reader, [41](#)
 - data, [42](#)
 - E, [42](#)
 - figsize, [42](#)
 - readAxis, [41](#)
 - readDataXE, [42](#)
 - X, [42](#)
- Db
 - setup, [130](#)
 - setup_adv, [137](#)
 - setup_adve, [143](#)
 - setup_advst, [148](#)
 - setup_diff, [154](#)
 - setup_uniform, [159](#)
 - setup_vdiff, [166](#)
- DeC
 - background_diffusion_coefficient_3, [20](#)
 - phases_collection, [127](#)
- delta
 - constants.h, [205](#)
 - EscapeModel_protons, [51](#)
 - EscapeModel_protons_2, [62](#)
- delta_log_output
 - constants.h, [205](#)
- delta_t
 - pcr_ip_2D, [118](#)
- DeM
 - background_diffusion_coefficient_3, [21](#)
 - phases_collection, [127](#)
- df1dx
 - cr_escape.cpp, [236](#)
 - EscapeModel_protons, [49](#)
 - EscapeModel_protons_2, [59](#)
 - mathematics.h, [233](#)
- df2dx
 - cr_escape.cpp, [236](#)
 - EscapeModel_protons, [49](#)
 - EscapeModel_protons_2, [59](#)
 - mathematics.h, [233](#)
- dilute_solver
 - solver1D.h, [245](#)
- DiM
 - background_diffusion_coefficient_3, [21](#)
 - phases_collection, [128](#)
- dk
 - background_diffusion_coefficient_2, [14](#)
- dmu
 - background_diffusion_coefficient, [8](#)
- door
 - setup_adv, [136](#)
 - setup_adve, [142](#)
 - setup_advst, [148](#)
 - setup_diff, [153](#)
 - setup_vdiff, [165](#)
- dt
 - Split_solvers, [179](#)
- Duu_Alfven_Slab_Linear_Undamped
 - background_diffusion_coefficient, [6](#)
 - background_diffusion_coefficient_3, [18](#)
- E
 - background_diffusion_coefficient, [8](#)
 - background_diffusion_coefficient_2, [14](#)
 - background_diffusion_coefficient_3, [21](#)
 - damping_models, [35](#)
 - Data_reader, [42](#)
 - electrons_emax_t, [46](#)
 - namelist, [77](#)
 - namelist_adv, [83](#)
 - namelist_adve, [89](#)
 - namelist_advst, [94](#)
 - namelist_diff, [100](#)
 - namelist_uniform, [106](#)
 - namelist_vdiff, [112](#)
 - pcr_ip_2D, [119](#)
 - setup, [131](#)
 - setup_adv, [137](#)
 - setup_adve, [143](#)
 - setup_advst, [149](#)
 - setup_diff, [154](#)
 - setup_uniform, [159](#)
 - setup_vdiff, [166](#)
 - ShowInjectionEvolution, [173](#)
 - test_tesc, [182](#)
- e
 - constants, [28](#)
 - constants.h, [206](#)
 - test_tesc, [182](#)
- E51
 - electrons_emax_t, [46](#)
 - EscapeModel_protons, [51](#)
- EMAX
 - EscapeModel_protons, [52](#)
- EV_log
 - pcr_ip_2D, [119](#)
- Ecr
 - EscapeModel_protons, [52](#)
 - EscapeModel_protons_2, [62](#)
- egridtype
 - damping_models, [35](#)
 - namelist, [77](#)
 - namelist_adv, [83](#)
 - namelist_adve, [89](#)
 - namelist_advst, [94](#)

- namelist_diff, 100
- namelist_uniform, 106
- namelist_vdiff, 112
- Eknee
 - constants.h, 206
- electron_injection_rate
 - constants.h, 206
- electron_source
 - solver1D.h, 245
- electrons_emax_t, 43
 - alpha, 46
 - alpha_B, 46
 - B, 43
 - beta, 46
 - C06, 46
 - E, 46
 - E51, 46
 - EM, 46
 - Em_age, 43
 - Em_cool, 44
 - Em_esc, 44
 - Emax_electrons, 44
 - escape_time, 44
 - eta_g, 45
 - InterpolatingSpline, 45
 - InverseTrigonalMatrix, 45
 - Mej, 46
 - nt, 47
 - phi_c, 47
 - ProductMatrix, 45
 - Rsh, 45
 - tesc, 47
 - tsed, 47
 - vej8, 47
 - xhi_cr, 47
 - xhi_m, 47
- elim
 - show_data, 172
- EM
 - electrons_emax_t, 46
- Em
 - damping_models, 36
- Em_age
 - electrons_emax_t, 43
- Em_cool
 - electrons_emax_t, 44
- Em_esc
 - electrons_emax_t, 44
- Emax
 - background_diffusion_coefficient_3, 21
 - damping_models, 36
 - EscapeModel_protons, 52
 - namelist, 77
 - namelist_adv, 83
 - namelist_adve, 89
 - namelist_advst, 94
 - namelist_diff, 100
 - namelist_uniform, 106
- namelist_vdiff, 112
- emax_CNM
 - EscapeModel_protons_2, 62
- emax_DiM
 - EscapeModel_protons_2, 62
- emax_HII
 - EscapeModel_protons_2, 62
- emax_WIM
 - EscapeModel_protons_2, 62
- emax_WNM
 - EscapeModel_protons_2, 63
- Emax_electrons
 - electrons_emax_t, 44
- Emin
 - background_diffusion_coefficient_3, 21
 - constants.h, 206
 - damping_models, 36
 - EscapeModel_protons, 52
 - namelist, 77
 - namelist_adv, 83
 - namelist_adve, 89
 - namelist_advst, 95
 - namelist_diff, 100
 - namelist_uniform, 106
 - namelist_vdiff, 112
 - test_tesc, 183
- Ep
 - damping_models, 36
- eps
 - background_diffusion_coefficient, 9
 - EscapeModel_protons, 52
- escape_time
 - electrons_emax_t, 44
- EscapeModel_protons, 48
 - a, 51
 - b, 51
 - beta, 51
 - c, 51
 - C06, 51
 - delta, 51
 - df1dx, 49
 - df2dx, 49
 - E51, 51
 - EMAX, 52
 - Ecr, 52
 - Emax, 52
 - Emin, 52
 - eps, 52
 - f1, 49
 - f2, 50
 - f_SNR, 52
 - figsize, 52
 - gamma, 53
 - Gettesc, 50
 - GeV, 53
 - InterpolatingSpline, 50
 - InverseTrigonalMatrix, 50
 - kyr, 53

- label, [53](#)
- logR, [53](#)
- logr_new, [53](#)
- logt, [53](#)
- logt_new, [54](#)
- lw, [54](#)
- marker, [54](#)
- Mej, [54](#)
- NewtonRaphson, [50](#)
- niter, [54](#)
- nt, [54](#)
- pc, [54](#)
- phi_c, [54](#)
- ProductMatrix, [50](#)
- R, [55](#)
- R_MCS, [55](#)
- R_PDS, [55](#)
- R_free, [55](#)
- R_ini, [55](#)
- R_merge, [55](#)
- r_new, [55](#)
- t, [56](#)
- t_new, [56](#)
- tMCS, [56](#)
- tPDS, [57](#)
- tSed, [57](#)
- tesc, [56](#)
- tfree, [56](#)
- tini, [56](#)
- tmax, [56](#)
- tmerge, [57](#)
- u_sh, [57](#)
- vej8, [57](#)
- x0, [57](#)
- xhi_cr, [57](#)
- xhi_m, [58](#)
- EscapeModel_protons_2, [58](#)
 - ax0, [61](#)
 - ax1, [61](#)
 - bbox_to_anchor, [61](#)
 - c, [61](#)
 - delta, [62](#)
 - df1dx, [59](#)
 - df2dx, [59](#)
 - Ecr, [62](#)
 - emax_CNM, [62](#)
 - emax_DiM, [62](#)
 - emax_HII, [62](#)
 - emax_WIM, [62](#)
 - emax_WNM, [63](#)
 - f1, [59](#)
 - f2, [59](#)
 - fig, [63](#)
 - getEmax, [60](#)
 - getSNR, [60](#)
 - Gettesc, [60](#)
 - GeV, [63](#)
 - gs, [63](#)
 - hspace, [63](#)
 - InterpolatingSpline, [60](#)
 - InverseTrigonalMatrix, [60](#)
 - kyr, [63](#)
 - label, [64](#)
 - loc, [64](#)
 - ls, [64](#)
 - ncol, [64](#)
 - NewtonRaphson, [61](#)
 - pad, [64](#)
 - ProductMatrix, [61](#)
 - SNR_CNM, [64](#)
 - SNR_DiM, [65](#)
 - SNR_HII, [65](#)
 - SNR_WIM, [65](#)
 - SNR_WNM, [65](#)
 - size_x, [64](#)
 - size_y, [64](#)
 - sub_x, [65](#)
 - sub_y, [65](#)
 - tesc_CNM_3, [65](#)
 - tesc_CNM, [65](#)
 - tesc_DiM_3, [66](#)
 - tesc_DiM, [66](#)
 - tesc_HII_3, [66](#)
 - tesc_HII, [66](#)
 - tesc_WIM_3, [66](#)
 - tesc_WIM, [66](#)
 - tesc_WNM_3, [66](#)
 - tesc_WNM, [66](#)
 - wspace, [67](#)
- Esn
 - constants.h, [206](#)
 - test_tesc, [183](#)
- eta_acc
 - constants.h, [206](#)
- eta_g
 - electrons_emax_t, [45](#)
- eta_gfree
 - constants.h, [207](#)
- EV
 - pcr_ip_2D, [119](#)
- eV
 - constants, [28](#)
 - constants.h, [207](#)
- ext
 - setup, [131](#)
 - setup_adv, [137](#)
 - setup_adve, [143](#)
 - setup_advst, [149](#)
 - setup_diff, [154](#)
 - setup_uniform, [159](#)
 - setup_vdiff, [166](#)
- f
 - mathmethods, [72](#)
 - setup_vdiff, [165](#)
- f1
 - cr_escape.cpp, [236](#)

- EscapeModel_protons, [49](#)
- EscapeModel_protons_2, [59](#)
- mathematics.h, [233](#)
- f2
 - cr_escape.cpp, [236](#)
 - EscapeModel_protons, [50](#)
 - EscapeModel_protons_2, [59](#)
 - mathematics.h, [233](#)
- f_SNR
 - EscapeModel_protons, [52](#)
- facecolor
 - background_diffusion_coefficient_3, [21](#)
 - damping_models, [36](#)
- False
 - pcr_ip_2D, [119](#)
- ff
 - cr_source.h, [221](#)
- fig
 - background_diffusion_coefficient_3, [21](#)
 - damping_models, [36](#)
 - EscapeModel_protons_2, [63](#)
 - SNR_evolution, [176](#)
- fig_save
 - show_data, [172](#)
- figsize
 - Data_reader, [42](#)
 - EscapeModel_protons, [52](#)
 - PDE_solvers, [124](#)
 - pcr_ip_2D, [119](#)
 - ShowInjectionEvolution, [174](#)
 - test_tesc, [183](#)
- fileWrite
 - fwriter, [67](#)
- findF
 - mathmethods, [72](#)
- findG
 - mathmethods, [73](#)
- findH
 - mathmethods, [73](#)
- finiteDiffSolver
 - PDE_solvers, [123](#)
- Finj
 - cr_source.h, [221](#)
- fj_m
 - background_diffusion_coefficient, [9](#)
- fj_p
 - background_diffusion_coefficient, [9](#)
- folder_name
 - namelist, [77](#)
 - namelist_adv, [83](#)
 - namelist_adve, [89](#)
 - namelist_advst, [95](#)
 - namelist_diff, [100](#)
 - namelist_uniform, [106](#)
 - namelist_vdiff, [112](#)
- folder_path
 - namelist, [77](#)
 - namelist_adv, [84](#)
 - namelist_adve, [89](#)
 - namelist_advst, [95](#)
 - namelist_diff, [101](#)
 - namelist_uniform, [107](#)
 - namelist_vdiff, [113](#)
- fontsize
 - pcr_ip_2D, [119](#)
- freader, [67](#)
 - search, [67](#)
- freader.h
 - readAxis, [228](#)
 - search, [228](#)
- fwriter, [67](#)
 - fileWrite, [67](#)
 - search, [68](#)
 - write1Daxis, [68](#)
 - write1D, [68](#)
 - write2D, [68](#)
- fwriter.h
 - writeInfo, [229](#)
 - writeXE, [229](#)
- g
 - mathmethods, [73](#)
- g1
 - mathmethods, [73](#)
- g2
 - mathmethods, [73](#)
- g_in
 - setup, [131](#)
 - setup_uniform, [160](#)
- g_lz
 - setup, [131](#)
 - setup_uniform, [160](#)
- g_s
 - background_diffusion_coefficient_2, [13](#)
- gam
 - constants.h, [207](#)
- gamma
 - background_diffusion_coefficient, [9](#)
 - background_diffusion_coefficient_2, [14](#)
 - EscapeModel_protons, [53](#)
- gamma_in
 - namelist, [78](#)
 - namelist_uniform, [107](#)
 - setup, [131](#)
 - setup_adv, [138](#)
 - setup_adve, [143](#)
 - setup_advst, [149](#)
 - setup_diff, [154](#)
 - setup_uniform, [160](#)
 - setup_vdiff, [167](#)
- gamma_lazarian
 - setup, [131](#)
 - setup_adv, [138](#)
 - setup_adve, [143](#)
 - setup_advst, [149](#)
 - setup_diff, [154](#)
 - setup_uniform, [160](#)

- setup_vdiff, 167
- Gamma_lz
 - damping_models, 36
- gamma_lz
 - namelist, 78
 - namelist_uniform, 107
- gamma_nlld
 - setup, 131
 - setup_adv, 138
 - setup_adve, 143
 - setup_advst, 149
 - setup_diff, 155
 - setup_uniform, 160
 - setup_vdiff, 167
- Gamma_nlld_inf
 - damping_models, 36
- Gamma_nlld_sup
 - damping_models, 37
- gamma_tot
 - setup, 131
 - setup_adv, 138
 - setup_adve, 144
 - setup_advst, 149
 - setup_diff, 155
 - setup_uniform, 160
 - setup_vdiff, 167
- gauss
 - gaussian_subNormalization, 69
 - test_tesc, 181
- gauss_a
 - gaussian_subNormalization, 70
- gauss_c
 - gaussian_subNormalization, 70
- gaussian_subNormalization, 69
 - c, 69
 - C_a, 69
 - C_c, 70
 - color, 70
 - gauss, 69
 - gauss_a, 70
 - gauss_c, 70
 - mu, 70
 - Nc, 70
 - Nv, 70
 - r, 70
 - sig, 71
 - ta, 71
 - tc, 71
 - tesc, 71
 - tmax, 71
 - tmin, 71
- generalized_diffusion
 - Split_solvers, 179
- getDamping
 - namelist, 76
 - namelist_uniform, 105
- getData
 - pcr_ip_2D, 117
- show_data, 171
- GetEM_e
 - cr_source.h, 222
- GetEM
 - cr_source.h, 222
- getEmax
 - EscapeModel_protons_2, 60
- getLogOutput
 - out.h, 240
- GetMax
 - cr_escape.cpp, 236
 - mathematics.h, 233
- getOutput
 - out.h, 240
- getSNR
 - EscapeModel_protons_2, 60
- GetTSed
 - cr_source.h, 222
- GetTesc
 - cr_escape.cpp, 236
- getTimeID
 - pcr_ip_2D, 117
- getVA
 - namelist, 76
 - namelist_adv, 82
 - namelist_adve, 88
 - namelist_advst, 94
 - namelist_diff, 99
 - namelist_uniform, 105
 - namelist_vdiff, 111
- Gettesc
 - EscapeModel_protons, 50
 - EscapeModel_protons_2, 60
- GeV
 - background_diffusion_coefficient_3, 21
 - constants, 28
 - constants.h, 207
 - damping_models, 37
 - EscapeModel_protons, 53
 - EscapeModel_protons_2, 63
 - test_tesc, 183
- glin
 - mathmethods, 73
- grid
 - d1_grid_generator, 31
- gs
 - background_diffusion_coefficient_3, 22
 - damping_models, 37
 - EscapeModel_protons_2, 63
 - SNR_evolution, 176
- gs0
 - background_diffusion_coefficient, 9
- HII
 - background_diffusion_coefficient_3, 22
 - phases_collection, 128
- handles
 - background_diffusion_coefficient_3, 22
- hatch

- background_diffusion_coefficient_3, [22](#)
- histogram
 - mathmethods, [74](#)
- hspace
 - background_diffusion_coefficient_3, [22](#)
 - damping_models, [37](#)
 - EscapeModel_protons_2, [63](#)
 - SNR_evolution, [176](#)
- I
 - background_diffusion_coefficient, [9](#)
 - background_diffusion_coefficient_2, [14](#)
 - background_diffusion_coefficient_3, [22](#)
 - setup, [132](#)
 - setup_uniform, [160](#)
- IN_damping_approx_1
 - damping, [31](#)
- IN_damping_approx_2
 - damping, [32](#)
- linf
 - damping_models, [37](#)
- Im
 - setup, [132](#)
 - setup_adv, [138](#)
 - setup_adve, [144](#)
 - setup_advst, [149](#)
 - setup_diff, [155](#)
 - setup_uniform, [160](#)
 - setup_vdiff, [167](#)
- im0
 - pcr_ip_2D, [119](#)
- im1
 - pcr_ip_2D, [119](#)
- in_damping
 - damping_models, [37](#)
 - namelist, [78](#)
 - namelist_adv, [84](#)
 - namelist_adve, [89](#)
 - namelist_advst, [95](#)
 - namelist_diff, [101](#)
 - namelist_uniform, [107](#)
 - namelist_vdiff, [113](#)
 - setup, [132](#)
 - setup_uniform, [161](#)
- indamping_alfven
 - damping, [32](#)
- indamping_alfven_nopos
 - damping, [32](#)
- index
 - ShowInjectionEvolution, [174](#)
- indexing
 - pcr_ip_2D, [120](#)
- inj_exp_alpha
 - constants.h, [207](#)
- injection_cutoff
 - constants.h, [207](#)
- injection_function_norm
 - constants.h, [207](#)
- injection_function_width
 - constants.h, [208](#)
- injection_shape_time
 - constants.h, [208](#)
- InterpolatingSpline
 - cr_escape.cpp, [236](#)
 - electrons_emax_t, [45](#)
 - EscapeModel_protons, [50](#)
 - EscapeModel_protons_2, [60](#)
 - mathematics.h, [234](#)
 - SNR_evolution, [175](#)
- InverseTrigonalMatrix
 - cr_escape.cpp, [237](#)
 - electrons_emax_t, [45](#)
 - EscapeModel_protons, [50](#)
 - EscapeModel_protons_2, [60](#)
 - mathematics.h, [234](#)
 - SNR_evolution, [175](#)
- IonNeutral_Damping
 - background_diffusion_coefficient, [6](#)
 - background_diffusion_coefficient_2, [13](#)
 - background_diffusion_coefficient_3, [18](#)
 - damping, [33](#)
- lp
 - pcr_ip_2D, [120](#)
 - setup, [132](#)
 - setup_adv, [138](#)
 - setup_adve, [144](#)
 - setup_advst, [149](#)
 - setup_diff, [155](#)
 - setup_uniform, [161](#)
 - setup_vdiff, [167](#)
- ism_phase
 - phases_collection, [127](#)
- ism_values
 - namelist, [78](#)
 - namelist_adv, [84](#)
 - namelist_adve, [90](#)
 - namelist_advst, [95](#)
 - namelist_diff, [101](#)
 - namelist_uniform, [107](#)
 - namelist_vdiff, [113](#)
 - setup, [132](#)
 - setup_adv, [138](#)
 - setup_adve, [144](#)
 - setup_advst, [150](#)
 - setup_diff, [155](#)
 - setup_uniform, [161](#)
 - setup_vdiff, [167](#)
- isotropy
 - constants.h, [208](#)
- lsup
 - damping_models, [37](#)
- ltot
 - background_diffusion_coefficient, [9](#)
- J
 - background_diffusion_coefficient, [7](#)
- k

- background_diffusion_coefficient_2, [15](#)
- k_cm
 - background_diffusion_coefficient, [9](#)
- k_cp
 - background_diffusion_coefficient, [10](#)
- k_max
 - background_diffusion_coefficient, [10](#)
- k_min
 - background_diffusion_coefficient, [10](#)
- k_zz
 - background_diffusion_coefficient, [10](#)
 - background_diffusion_coefficient_2, [15](#)
- K_zz_1
 - background_diffusion_coefficient_3, [22](#)
- K_zz_2
 - background_diffusion_coefficient_3, [23](#)
- K_zz_CNM_1
 - background_diffusion_coefficient_3, [23](#)
- K_zz_CNM_2
 - background_diffusion_coefficient_3, [23](#)
- K_zz_DeC_1
 - background_diffusion_coefficient_3, [23](#)
- K_zz_DeC_2
 - background_diffusion_coefficient_3, [23](#)
- K_zz_DeM_1
 - background_diffusion_coefficient_3, [23](#)
- K_zz_DeM_2
 - background_diffusion_coefficient_3, [24](#)
- K_zz_DiM_1
 - background_diffusion_coefficient_3, [24](#)
- K_zz_DiM_2
 - background_diffusion_coefficient_3, [24](#)
- K_zz_HII_1
 - background_diffusion_coefficient_3, [24](#)
- K_zz_HII_2
 - background_diffusion_coefficient_3, [24](#)
- K_zz_WIM_1
 - background_diffusion_coefficient_3, [24](#)
- K_zz_WIM_2
 - background_diffusion_coefficient_3, [24](#)
- K_zz_WNM_1
 - background_diffusion_coefficient_3, [24](#)
- K_zz_WNM_2
 - background_diffusion_coefficient_3, [25](#)
- K_zz_bc_1
 - background_diffusion_coefficient_3, [23](#)
- K_zz_bc_2
 - background_diffusion_coefficient_3, [23](#)
- Kappa_zz
 - background_diffusion_coefficient, [7](#)
 - background_diffusion_coefficient_3, [18](#)
- kappa_zz_BC
 - background_diffusion_coefficient_3, [19](#)
- kbolz
 - constants, [28](#)
 - constants.h, [208](#)
- km
 - constants.h, [208](#)
- kmin
 - background_diffusion_coefficient_2, [15](#)
 - background_diffusion_coefficient_3, [25](#)
 - setup, [132](#)
 - setup_uniform, [161](#)
- kms
 - constants, [28](#)
 - constants.h, [208](#)
 - SNR_evolution, [177](#)
- kpc
 - constants, [28](#)
 - constants.h, [209](#)
- kyr
 - constants, [28](#)
 - constants.h, [209](#)
 - EscapeModel_protons, [53](#)
 - EscapeModel_protons_2, [63](#)
 - PDE_solvers, [124](#)
 - test_tesc, [183](#)
- label
 - background_diffusion_coefficient_3, [25](#)
 - damping_models, [37](#)
 - EscapeModel_protons, [53](#)
 - EscapeModel_protons_2, [64](#)
 - pcr_ip_2D, [120](#)
 - SNR_evolution, [177](#)
 - test_tesc, [183](#)
- loc
 - background_diffusion_coefficient_3, [25](#)
 - damping_models, [38](#)
 - EscapeModel_protons_2, [64](#)
 - SNR_evolution, [177](#)
- loc_id
 - ShowInjectionEvolution, [174](#)
- log_first_data
 - constants.h, [209](#)
- logmaker.h
 - showLog_0, [230](#)
- logR
 - EscapeModel_protons, [53](#)
- logr_new
 - EscapeModel_protons, [53](#)
- logt
 - EscapeModel_protons, [53](#)
- logt_new
 - EscapeModel_protons, [54](#)
- ls
 - background_diffusion_coefficient_3, [25](#)
 - damping_models, [38](#)
 - EscapeModel_protons_2, [64](#)
- lw
 - damping_models, [38](#)
 - EscapeModel_protons, [54](#)
 - SNR_evolution, [177](#)
- lz_damping
 - damping_models, [38](#)
 - namelist, [78](#)
 - namelist_adv, [84](#)

- namelist_adve, 90
 - namelist_advst, 95
 - namelist_diff, 101
 - namelist_uniform, 107
 - namelist_vdiff, 113
- lz_min
 - damping_models, 38
- M
 - PDE_solvers, 124
- m
 - background_diffusion_coefficient, 10
 - background_diffusion_coefficient_2, 15
- m_neutral
 - cr_source.h, 224
- mCII
 - constants, 29
 - constants.h, 209
- mH2
 - constants, 29
 - constants.h, 210
- mHCOII
 - constants, 29
- mHII
 - constants, 30
 - constants.h, 210
- mHell
 - constants, 29
- mHel
 - constants, 29
 - constants.h, 210
- mHI
 - constants, 29
 - constants.h, 210
- main
 - cr_escape.cpp, 237
 - main.cpp, 231
- main.cpp
 - main, 231
- marker
 - EscapeModel_protons, 54
 - SNR_evolution, 177
- mass
 - background_diffusion_coefficient_3, 25
 - setup, 132
 - setup_uniform, 161
- mathematics.h
 - df1dx, 233
 - df2dx, 233
 - f1, 233
 - f2, 233
 - GetMax, 233
 - InterpolatingSpline, 234
 - InverseTrigonalMatrix, 234
 - minmod, 234
 - NewtonRaphson, 234
 - ProductMatrix, 234
 - TDMA, 234
 - transpose, 235
- mathmethods, 72
 - cardano3, 72
 - Cubic3, 72
 - f, 72
 - findF, 72
 - findG, 73
 - findH, 73
 - g, 73
 - g1, 73
 - g2, 73
 - glin, 73
 - histogram, 74
 - multishape, 74
 - shape, 74
 - simpson_lin, 74
 - simpson_log, 74
 - SmoothPhaseTransition, 75
- maxElement1D
 - tools.h, 248
- maxElement2D
 - tools.h, 248
- me
 - constants, 29
 - constants.h, 209
- medium_props
 - background_diffusion_coefficient_2, 15
 - setup, 132
 - setup_uniform, 161
- Mej
 - constants.h, 209
 - electrons_emax_t, 46
 - EscapeModel_protons, 54
- MeV
 - constants, 29
 - constants.h, 210
- mi
 - background_diffusion_coefficient, 10
 - namelist, 78
 - namelist_adv, 84
 - namelist_adve, 90
 - namelist_advst, 95
 - namelist_diff, 101
 - namelist_uniform, 107
 - namelist_vdiff, 113
 - setup, 133
 - setup_adv, 138
 - setup_adve, 144
 - setup_advst, 150
 - setup_diff, 155
 - setup_uniform, 161
 - setup_vdiff, 167
- minElement1D
 - tools.h, 248
- minElement2D
 - tools.h, 249
- minmod
 - mathematics.h, 234
- mn

- background_diffusion_coefficient, 10
- constants, 30
- constants.h, 210
- namelist, 78
- namelist_adv, 84
- namelist_adve, 90
- namelist_advst, 96
- namelist_diff, 101
- namelist_uniform, 108
- namelist_vdiff, 113
- setup, 133
- setup_adv, 139
- setup_adve, 144
- setup_advst, 150
- setup_diff, 155
- setup_uniform, 162
- setup_vdiff, 168
- mp
 - background_diffusion_coefficient_3, 25
 - constants, 30
 - constants.h, 211
- mu
 - background_diffusion_coefficient, 10
 - background_diffusion_coefficient_2, 15
 - gaussian_subNormalization, 70
- multishape
 - mathmethods, 74
- Name
 - damping_models, 38
- namelist, 75
 - B, 76
 - bdiff_model, 76
 - box_center, 77
 - E, 77
 - egridtype, 77
 - Emax, 77
 - Emin, 77
 - folder_name, 77
 - folder_path, 77
 - gamma_in, 78
 - gamma_lz, 78
 - getDamping, 76
 - getVA, 76
 - in_damping, 78
 - ism_values, 78
 - lz_damping, 78
 - mi, 78
 - mn, 78
 - NE, 79
 - ni, 79
 - nll_damping, 79
 - nn, 79
 - nt, 79
 - NX, 79
 - Pcr_1GeV, 79
 - Pe_1GeV, 80
 - phases, 80
 - smooth_width_transition, 80
 - T, 80
 - total_path, 80
 - va, 80
 - X, 80
 - xgridtype, 81
 - Xi, 81
 - Xmax, 81
 - Xmin, 81
- namelist_adv, 81
 - B, 83
 - bdiff_model, 83
 - box_center, 83
 - E, 83
 - egridtype, 83
 - Emax, 83
 - Emin, 83
 - folder_name, 83
 - folder_path, 84
 - getVA, 82
 - in_damping, 84
 - ism_values, 84
 - lz_damping, 84
 - mi, 84
 - mn, 84
 - NE, 84
 - ni, 85
 - nll_damping, 85
 - nn, 85
 - nt, 85
 - NX, 85
 - Pcr_1GeV, 85
 - Pe_1GeV, 85
 - phases, 86
 - smooth_width_transition, 86
 - T, 86
 - total_path, 86
 - va, 86
 - X, 86
 - xgridtype, 86
 - Xi, 87
 - Xmax, 87
 - Xmin, 87
- namelist_adve, 87
 - B, 88
 - bdiff_model, 88
 - box_center, 88
 - E, 89
 - egridtype, 89
 - Emax, 89
 - Emin, 89
 - folder_name, 89
 - folder_path, 89
 - getVA, 88
 - in_damping, 89
 - ism_values, 90
 - lz_damping, 90
 - mi, 90
 - mn, 90

- NE, 90
- ni, 90
- nll_damping, 90
- nn, 91
- nt, 91
- NX, 91
- Pcr_1GeV, 91
- Pe_1GeV, 91
- phases, 91
- smooth_width_transition, 91
- T, 92
- total_path, 92
- va, 92
- X, 92
- xgridtype, 92
- Xi, 92
- Xmax, 92
- Xmin, 93
- namelist_advst, 93
 - B, 94
 - bdiff_model, 94
 - box_center, 94
 - E, 94
 - egridtype, 94
 - Emax, 94
 - Emin, 95
 - folder_name, 95
 - folder_path, 95
 - getVA, 94
 - in_damping, 95
 - ism_values, 95
 - lz_damping, 95
 - mi, 95
 - mn, 96
 - NE, 96
 - ni, 96
 - nll_damping, 96
 - nn, 96
 - nt, 96
 - NX, 96
 - Pcr_1GeV, 97
 - Pe_1GeV, 97
 - phases, 97
 - smooth_width_transition, 97
 - T, 97
 - total_path, 97
 - va, 97
 - X, 98
 - xgridtype, 98
 - Xi, 98
 - Xmax, 98
 - Xmin, 98
- namelist_diff, 98
 - B, 100
 - bdiff_model, 100
 - box_center, 100
 - E, 100
 - egridtype, 100
 - Emax, 100
 - Emin, 100
 - folder_name, 100
 - folder_path, 101
 - getVA, 99
 - in_damping, 101
 - ism_values, 101
 - lz_damping, 101
 - mi, 101
 - mn, 101
 - NE, 101
 - ni, 102
 - nll_damping, 102
 - nn, 102
 - nt, 102
 - NX, 102
 - Pcr_1GeV, 102
 - Pe_1GeV, 102
 - phases, 103
 - smooth_width_transition, 103
 - T, 103
 - total_path, 103
 - va, 103
 - X, 103
 - xgridtype, 103
 - Xi, 104
 - Xmax, 104
 - Xmin, 104
- namelist_uniform, 104
 - B, 105
 - bdiff_model, 106
 - box_center, 106
 - E, 106
 - egridtype, 106
 - Emax, 106
 - Emin, 106
 - folder_name, 106
 - folder_path, 107
 - gamma_in, 107
 - gamma_lz, 107
 - getDamping, 105
 - getVA, 105
 - in_damping, 107
 - ism_values, 107
 - lz_damping, 107
 - mi, 107
 - mn, 108
 - NE, 108
 - ni, 108
 - nll_damping, 108
 - nn, 108
 - nt, 108
 - NX, 108
 - Pcr_1GeV, 109
 - Pe_1GeV, 109
 - phases, 109
 - smooth_width_transition, 109
 - T, 109

- total_path, [109](#)
- va, [109](#)
- X, [110](#)
- xgridtype, [110](#)
- Xi, [110](#)
- Xmax, [110](#)
- Xmin, [110](#)
- namelist_vdiff, [110](#)
 - B, [112](#)
 - bdiff_model, [112](#)
 - box_center, [112](#)
 - E, [112](#)
 - egridtype, [112](#)
 - Emax, [112](#)
 - Emin, [112](#)
 - folder_name, [112](#)
 - folder_path, [113](#)
 - getVA, [111](#)
 - in_damping, [113](#)
 - ism_values, [113](#)
 - lz_damping, [113](#)
 - mi, [113](#)
 - mn, [113](#)
 - NE, [113](#)
 - ni, [114](#)
 - nlld_damping, [114](#)
 - nn, [114](#)
 - nt, [114](#)
 - NX, [114](#)
 - Pcr_1GeV, [114](#)
 - Pe_1GeV, [114](#)
 - phases, [115](#)
 - smooth_width_transition, [115](#)
 - T, [115](#)
 - total_path, [115](#)
 - va, [115](#)
 - X, [115](#)
 - xgridtype, [115](#)
 - Xi, [116](#)
 - Xmax, [116](#)
 - Xmin, [116](#)
- Nc
 - gaussian_subNormalization, [70](#)
- ncol
 - background_diffusion_coefficient_3, [25](#)
 - EscapeModel_protons_2, [64](#)
 - SNR_evolution, [177](#)
- ncols
 - pcr_ip_2D, [120](#)
- NE
 - damping_models, [38](#)
 - namelist, [79](#)
 - namelist_adv, [84](#)
 - namelist_adve, [90](#)
 - namelist_advst, [96](#)
 - namelist_diff, [101](#)
 - namelist_uniform, [108](#)
 - namelist_vdiff, [113](#)
- ne
 - cr_source.h, [224](#)
 - setup, [133](#)
 - setup_adv, [139](#)
 - setup_adve, [144](#)
 - setup_advst, [150](#)
 - setup_diff, [155](#)
 - setup_uniform, [162](#)
 - setup_vdiff, [168](#)
- NewtonRaphson
 - cr_escape.cpp, [237](#)
 - EscapeModel_protons, [50](#)
 - EscapeModel_protons_2, [61](#)
 - mathematics.h, [234](#)
- ni
 - background_diffusion_coefficient, [11](#)
 - cr_source.h, [225](#)
 - namelist, [79](#)
 - namelist_adv, [85](#)
 - namelist_adve, [90](#)
 - namelist_advst, [96](#)
 - namelist_diff, [102](#)
 - namelist_uniform, [108](#)
 - namelist_vdiff, [114](#)
 - setup, [133](#)
 - setup_adv, [139](#)
 - setup_adve, [144](#)
 - setup_advst, [150](#)
 - setup_diff, [156](#)
 - setup_uniform, [162](#)
 - setup_vdiff, [168](#)
- niter
 - EscapeModel_protons, [54](#)
- nlld_damping
 - namelist, [79](#)
 - namelist_adv, [85](#)
 - namelist_adve, [90](#)
 - namelist_advst, [96](#)
 - namelist_diff, [102](#)
 - namelist_uniform, [108](#)
 - namelist_vdiff, [114](#)
- nn
 - background_diffusion_coefficient, [11](#)
 - namelist, [79](#)
 - namelist_adv, [85](#)
 - namelist_adve, [91](#)
 - namelist_advst, [96](#)
 - namelist_diff, [102](#)
 - namelist_uniform, [108](#)
 - namelist_vdiff, [114](#)
 - setup, [133](#)
 - setup_adv, [139](#)
 - setup_adve, [145](#)
 - setup_advst, [150](#)
 - setup_diff, [156](#)
 - setup_uniform, [162](#)
 - setup_vdiff, [168](#)
- non_linear_landau_damping

- damping, [33](#)
- NotMove
 - solver1D.h, [246](#)
- nproc
 - constants.h, [211](#)
- nt
 - cr_source.h, [225](#)
 - electrons_emax_t, [47](#)
 - EscapeModel_protons, [54](#)
 - namelist, [79](#)
 - namelist_adv, [85](#)
 - namelist_adve, [91](#)
 - namelist_advst, [96](#)
 - namelist_diff, [102](#)
 - namelist_uniform, [108](#)
 - namelist_vdiff, [114](#)
- nu_in
 - background_diffusion_coefficient, [11](#)
- nu_ni
 - background_diffusion_coefficient, [11](#)
- number_out_data
 - constants.h, [211](#)
- Nv
 - gaussian_subNormalization, [70](#)
- NX
 - namelist, [79](#)
 - namelist_adv, [85](#)
 - namelist_adve, [91](#)
 - namelist_advst, [96](#)
 - namelist_diff, [102](#)
 - namelist_uniform, [108](#)
 - namelist_vdiff, [114](#)
 - Split_solvers, [179](#)
- nx
 - cr_source.h, [225](#)
 - setup, [133](#)
 - setup_adv, [139](#)
 - setup_adve, [145](#)
 - setup_advst, [150](#)
 - setup_diff, [156](#)
 - setup_uniform, [162](#)
 - setup_vdiff, [168](#)
- oh_model
 - constants.h, [211](#)
- Omega
 - background_diffusion_coefficient, [11](#)
 - background_diffusion_coefficient_2, [15](#)
- Omega0
 - background_diffusion_coefficient, [11](#)
 - background_diffusion_coefficient_2, [15](#)
- out.h
 - getLogOutput, [240](#)
 - getOutput, [240](#)
 - outputData, [240](#)
 - setTmax, [241](#)
 - specificOutputData, [241](#)
- out_id
 - pcr_ip_2D, [120](#)
- output_freq
 - constants.h, [211](#)
- Output_functions, [116](#)
- outputData
 - out.h, [240](#)
- P
 - background_diffusion_coefficient, [11](#)
 - background_diffusion_coefficient_2, [16](#)
- PDE_solvers, [122](#)
 - A, [122](#)
 - B, [122](#)
 - C, [123](#)
 - c, [124](#)
 - CC70Solver, [123](#)
 - figsize, [124](#)
 - finiteDiffSolver, [123](#)
 - kyr, [124](#)
 - M, [124](#)
 - pc, [124](#)
 - Q, [123](#)
 - SimpleImplicitSolver, [123](#)
 - T, [123](#)
 - TDMA, [124](#)
 - Tmax, [125](#)
 - u, [125](#)
 - u0, [125](#)
 - u_0, [125](#)
 - u_1, [125](#)
 - u_2, [125](#)
 - u_end, [126](#)
 - u_ini, [126](#)
 - uM, [126](#)
 - X, [126](#)
 - Xmax, [126](#)
 - Xmin, [126](#)
 - yr, [126](#)
- pad
 - EscapeModel_protons_2, [64](#)
- pad_inches
 - damping_models, [38](#)
- parameters
 - cr_source.h, [225](#)
- parse2DCsvFile
 - read2D.h, [242](#)
- particles_props
 - background_diffusion_coefficient_2, [16](#)
- path
 - setup, [134](#)
 - setup_adv, [139](#)
 - setup_adve, [145](#)
 - setup_advst, [151](#)
 - setup_diff, [156](#)
 - setup_uniform, [162](#)
 - setup_vdiff, [168](#)
- pc
 - constants, [30](#)
 - constants.h, [211](#)
 - EscapeModel_protons, [54](#)

- PDE_solvers, [124](#)
- SNR_evolution, [177](#)
- Pcr
 - pcr_ip_2D, [120](#)
 - setup, [134](#)
 - setup_adv, [139](#)
 - setup_adve, [145](#)
 - setup_advst, [151](#)
 - setup_diff, [156](#)
 - setup_uniform, [162](#)
 - setup_vdiff, [168](#)
- Pcr_1GeV
 - namelist, [79](#)
 - namelist_adv, [85](#)
 - namelist_adve, [91](#)
 - namelist_advst, [97](#)
 - namelist_diff, [102](#)
 - namelist_uniform, [109](#)
 - namelist_vdiff, [114](#)
- Pcr_ini
 - cr_source.h, [222](#)
- pcr_ip_2D, [116](#)
 - ax, [118](#)
 - ax0, [118](#)
 - ax1, [118](#)
 - cax, [118](#)
 - cmap, [118](#)
 - delta_t, [118](#)
 - E, [119](#)
 - EV_log, [119](#)
 - EV, [119](#)
 - False, [119](#)
 - figsize, [119](#)
 - fontsize, [119](#)
 - getData, [117](#)
 - getTimeID, [117](#)
 - im0, [119](#)
 - im1, [119](#)
 - indexing, [120](#)
 - lp, [120](#)
 - label, [120](#)
 - ncols, [120](#)
 - out_id, [120](#)
 - Pcr, [120](#)
 - readAxis, [117](#)
 - readDataXE, [118](#)
 - sharey, [120](#)
 - sparse, [120](#)
 - t_ini, [121](#)
 - t_max, [121](#)
 - time_test, [121](#)
 - X, [121](#)
 - x_center, [121](#)
 - XV_log, [121](#)
 - XV, [121](#)
- Pe
 - setup, [134](#)
 - setup_adv, [140](#)
 - setup_adve, [145](#)
 - setup_advst, [151](#)
 - setup_diff, [156](#)
 - setup_uniform, [163](#)
 - setup_vdiff, [169](#)
- Pe_1GeV
 - namelist, [80](#)
 - namelist_adv, [85](#)
 - namelist_adve, [91](#)
 - namelist_advst, [97](#)
 - namelist_diff, [102](#)
 - namelist_uniform, [109](#)
 - namelist_vdiff, [114](#)
- perpendicular_diffusion_solver
 - solver1D.h, [246](#)
- phase
 - background_diffusion_coefficient, [11](#)
- phases
 - damping_models, [39](#)
 - namelist, [80](#)
 - namelist_adv, [86](#)
 - namelist_adve, [91](#)
 - namelist_advst, [97](#)
 - namelist_diff, [103](#)
 - namelist_uniform, [109](#)
 - namelist_vdiff, [115](#)
- phases_collection, [127](#)
 - CNM, [127](#)
 - DeC, [127](#)
 - DeM, [127](#)
 - DiM, [128](#)
 - HII, [128](#)
 - ism_phase, [127](#)
 - WIM, [128](#)
 - WNM, [128](#)
- phi_c
 - constants.h, [212](#)
 - electrons_emax_t, [47](#)
 - EscapeModel_protons, [54](#)
- physical_models, [128](#)
 - collision_rate, [128](#)
 - cr_escape_radius_model, [129](#)
 - cr_escape_time_model, [129](#)
- pi
 - constants.h, [212](#)
- pos_1
 - damping_models, [39](#)
- pos_2
 - damping_models, [39](#)
- ProductMatrix
 - cr_escape.cpp, [237](#)
 - electrons_emax_t, [45](#)
 - EscapeModel_protons, [50](#)
 - EscapeModel_protons_2, [61](#)
 - mathematics.h, [234](#)
 - SNR_evolution, [175](#)
- Q
 - PDE_solvers, [123](#)

- q
 - background_diffusion_coefficient, [12](#)
 - background_diffusion_coefficient_2, [16](#)
 - background_diffusion_coefficient_3, [26](#)
 - setup, [134](#)
 - setup_uniform, [163](#)
- Qcr
 - test_tesc, [183](#)
- R
 - background_diffusion_coefficient_2, [13](#)
 - EscapeModel_protons, [55](#)
- r
 - gaussian_subNormalization, [70](#)
- R_MCS
 - EscapeModel_protons, [55](#)
- R_PDS
 - EscapeModel_protons, [55](#)
- R_free
 - EscapeModel_protons, [55](#)
- R_ini
 - EscapeModel_protons, [55](#)
- R_merge
 - EscapeModel_protons, [55](#)
- r_new
 - EscapeModel_protons, [55](#)
- r_snr_thickness
 - constants.h, [212](#)
- RSNR
 - cr_source.h, [222](#)
- read2D.h
 - parse2DCsvFile, [242](#)
- readAxis
 - Data_reader, [41](#)
 - freader.h, [228](#)
 - pcr_ip_2D, [117](#)
 - show_data, [171](#)
 - ShowInjectionEvolution, [173](#)
- readDataXE
 - Data_reader, [42](#)
 - pcr_ip_2D, [118](#)
 - show_data, [171](#)
 - ShowInjectionEvolution, [173](#)
- readInfo
 - show_data, [171](#)
- Resc
 - cr_source.h, [222](#)
- rho_0
 - test_tesc, [183](#)
- Rsh
 - electrons_emax_t, [45](#)
 - SNR_evolution, [175](#)
- sBcenter
 - cr_source.h, [225](#)
- SNR_CNМ
 - EscapeModel_protons_2, [64](#)
- SNR_DiM
 - EscapeModel_protons_2, [65](#)
- SNR_HIИ
 - EscapeModel_protons_2, [65](#)
- SNR_WIM
 - EscapeModel_protons_2, [65](#)
- SNR_WNM
 - EscapeModel_protons_2, [65](#)
- SNR_evolution, [174](#)
 - ax0, [176](#)
 - ax1, [176](#)
 - bbox_to_anchor, [176](#)
 - c, [176](#)
 - fig, [176](#)
 - gs, [176](#)
 - hspace, [176](#)
 - InterpolatingSpline, [175](#)
 - InverseTrigonalMatrix, [175](#)
 - kms, [177](#)
 - label, [177](#)
 - loc, [177](#)
 - lw, [177](#)
 - marker, [177](#)
 - ncol, [177](#)
 - pc, [177](#)
 - ProductMatrix, [175](#)
 - Rsh, [175](#)
 - size_x, [177](#)
 - size_y, [178](#)
 - sub_x, [178](#)
 - sub_y, [178](#)
 - wspace, [178](#)
- scenter
 - cr_source.h, [225](#)
- scenter_index
 - cr_source.h, [225](#)
- search
 - freader, [67](#)
 - freader.h, [228](#)
 - fwriter, [68](#)
- set_background
 - constants.h, [212](#)
- setTmax
 - out.h, [241](#)
- setup, [129](#)
 - B, [130](#)
 - D, [130](#)
 - d00, [130](#)
 - Db, [130](#)
 - E, [131](#)
 - ext, [131](#)
 - g_in, [131](#)
 - g_lz, [131](#)
 - gamma_in, [131](#)
 - gamma_lazarian, [131](#)
 - gamma_nlld, [131](#)
 - gamma_tot, [131](#)
 - l, [132](#)
 - lm, [132](#)
 - in_damping, [132](#)

- lp, [132](#)
- ism_values, [132](#)
- kmin, [132](#)
- mass, [132](#)
- medium_props, [132](#)
- mi, [133](#)
- mn, [133](#)
- ne, [133](#)
- ni, [133](#)
- nn, [133](#)
- nx, [133](#)
- path, [134](#)
- Pcr, [134](#)
- Pe, [134](#)
- q, [134](#)
- T, [134](#)
- VA, [134](#)
- va, [134](#)
- variable, [135](#)
- variables, [135](#)
- X, [135](#)
- x_center, [135](#)
- x_center_index, [135](#)
- Xi, [135](#)
- setup_adv, [136](#)
 - B, [137](#)
 - D, [137](#)
 - d00, [137](#)
 - Db, [137](#)
 - door, [136](#)
 - E, [137](#)
 - ext, [137](#)
 - gamma_in, [138](#)
 - gamma_lazarian, [138](#)
 - gamma_nlld, [138](#)
 - gamma_tot, [138](#)
 - Im, [138](#)
 - lp, [138](#)
 - ism_values, [138](#)
 - mi, [138](#)
 - mn, [139](#)
 - ne, [139](#)
 - ni, [139](#)
 - nn, [139](#)
 - nx, [139](#)
 - path, [139](#)
 - Pcr, [139](#)
 - Pe, [140](#)
 - T, [140](#)
 - VA, [140](#)
 - va, [140](#)
 - variable, [140](#)
 - variables, [140](#)
 - X, [140](#)
 - x_center, [141](#)
 - x_center_index, [141](#)
 - Xi, [141](#)
- setup_adve, [141](#)
- B, [142](#)
- D, [142](#)
- d00, [143](#)
- Db, [143](#)
- door, [142](#)
- E, [143](#)
- ext, [143](#)
- gamma_in, [143](#)
- gamma_lazarian, [143](#)
- gamma_nlld, [143](#)
- gamma_tot, [144](#)
- Im, [144](#)
- lp, [144](#)
- ism_values, [144](#)
- mi, [144](#)
- mn, [144](#)
- ne, [144](#)
- ni, [144](#)
- nn, [145](#)
- nx, [145](#)
- path, [145](#)
- Pcr, [145](#)
- Pe, [145](#)
- spec, [142](#)
- T, [145](#)
- VA, [146](#)
- va, [145](#)
- variable, [146](#)
- variables, [146](#)
- X, [146](#)
- x_center, [146](#)
- x_center_index, [146](#)
- Xi, [147](#)
- setup_advst, [147](#)
 - B, [148](#)
 - D, [148](#)
 - d00, [148](#)
 - Db, [148](#)
 - door, [148](#)
 - E, [149](#)
 - ext, [149](#)
 - gamma_in, [149](#)
 - gamma_lazarian, [149](#)
 - gamma_nlld, [149](#)
 - gamma_tot, [149](#)
 - Im, [149](#)
 - lp, [149](#)
 - ism_values, [150](#)
 - mi, [150](#)
 - mn, [150](#)
 - ne, [150](#)
 - ni, [150](#)
 - nn, [150](#)
 - nx, [150](#)
 - path, [151](#)
 - Pcr, [151](#)
 - Pe, [151](#)
 - spec, [148](#)

- T, [151](#)
- VA, [151](#)
- va, [151](#)
- variable, [151](#)
- variables, [152](#)
- X, [152](#)
- x_center, [152](#)
- x_center_index, [152](#)
- Xi, [152](#)
- setup_diff, [153](#)
 - B, [154](#)
 - D, [154](#)
 - d00, [154](#)
 - Db, [154](#)
 - door, [153](#)
 - E, [154](#)
 - ext, [154](#)
 - gamma_in, [154](#)
 - gamma_lazarian, [154](#)
 - gamma_nlld, [155](#)
 - gamma_tot, [155](#)
 - Im, [155](#)
 - Ip, [155](#)
 - ism_values, [155](#)
 - mi, [155](#)
 - mn, [155](#)
 - ne, [155](#)
 - ni, [156](#)
 - nn, [156](#)
 - nx, [156](#)
 - path, [156](#)
 - Pcr, [156](#)
 - Pe, [156](#)
 - T, [156](#)
 - VA, [157](#)
 - va, [157](#)
 - variable, [157](#)
 - variables, [157](#)
 - X, [157](#)
 - x_center, [157](#)
 - x_center_index, [158](#)
 - Xi, [158](#)
- setup_uniform, [158](#)
 - B, [159](#)
 - D, [159](#)
 - d00, [159](#)
 - Db, [159](#)
 - E, [159](#)
 - ext, [159](#)
 - g_in, [160](#)
 - g_lz, [160](#)
 - gamma_in, [160](#)
 - gamma_lazarian, [160](#)
 - gamma_nlld, [160](#)
 - gamma_tot, [160](#)
 - I, [160](#)
 - Im, [160](#)
 - in_damping, [161](#)
 - Ip, [161](#)
 - ism_values, [161](#)
 - kmin, [161](#)
 - mass, [161](#)
 - medium_props, [161](#)
 - mi, [161](#)
 - mn, [162](#)
 - ne, [162](#)
 - ni, [162](#)
 - nn, [162](#)
 - nx, [162](#)
 - path, [162](#)
 - Pcr, [162](#)
 - Pe, [163](#)
 - q, [163](#)
 - T, [163](#)
 - VA, [163](#)
 - va, [163](#)
 - variable, [163](#)
 - variables, [163](#)
 - X, [164](#)
 - x_center, [164](#)
 - x_center_index, [164](#)
 - Xi, [164](#)
- setup_vdiff, [164](#)
 - B, [166](#)
 - D, [166](#)
 - d00, [166](#)
 - Db, [166](#)
 - door, [165](#)
 - E, [166](#)
 - ext, [166](#)
 - f, [165](#)
 - gamma_in, [167](#)
 - gamma_lazarian, [167](#)
 - gamma_nlld, [167](#)
 - gamma_tot, [167](#)
 - Im, [167](#)
 - Ip, [167](#)
 - ism_values, [167](#)
 - mi, [167](#)
 - mn, [168](#)
 - ne, [168](#)
 - ni, [168](#)
 - nn, [168](#)
 - nx, [168](#)
 - path, [168](#)
 - Pcr, [168](#)
 - Pe, [169](#)
 - T, [169](#)
 - VA, [169](#)
 - va, [169](#)
 - variable, [169](#)
 - variables, [169](#)
 - X, [169](#)
 - x_center, [170](#)
 - x_center_index, [170](#)
 - Xi, [170](#)

- shape
 - mathmethods, [74](#)
- sharey
 - pcr_ip_2D, [120](#)
- show
 - show_data, [172](#)
- show_data, [170](#)
 - colorArray, [171](#)
 - colorFader, [171](#)
 - elim, [172](#)
 - fig_save, [172](#)
 - getData, [171](#)
 - readAxis, [171](#)
 - readDataXE, [171](#)
 - readInfo, [171](#)
 - show, [172](#)
 - source_center, [172](#)
 - vlim, [172](#)
- ShowInjectionEvolution, [173](#)
 - data, [173](#)
 - E, [173](#)
 - figsize, [174](#)
 - index, [174](#)
 - loc_id, [174](#)
 - readAxis, [173](#)
 - readDataXE, [173](#)
 - X, [174](#)
- showLog_0
 - logmaker.h, [230](#)
- sig
 - gaussian_subNormalization, [71](#)
 - test_tesc, [182](#)
- sig_T
 - constants.h, [212](#)
- sigm
 - cr_source.h, [223](#)
- sigma_coherence
 - constants.h, [212](#)
- SimpleImplicitSolver
 - PDE_solvers, [123](#)
- simpson_lin
 - mathmethods, [74](#)
- simpson_log
 - mathmethods, [74](#)
- size_x
 - background_diffusion_coefficient_3, [26](#)
 - damping_models, [39](#)
 - EscapeModel_protons_2, [64](#)
 - SNR_evolution, [177](#)
- size_y
 - background_diffusion_coefficient_3, [26](#)
 - damping_models, [39](#)
 - EscapeModel_protons_2, [64](#)
 - SNR_evolution, [178](#)
- smn
 - cr_source.h, [225](#)
- smooth_width_transition
 - namelist, [80](#)
- namelist_adv, [86](#)
- namelist_adve, [91](#)
- namelist_advst, [97](#)
- namelist_diff, [103](#)
- namelist_uniform, [109](#)
- namelist_vdiff, [115](#)
- SmoothPhaseTransition
 - mathmethods, [75](#)
- sne
 - cr_source.h, [226](#)
- sni
 - cr_source.h, [226](#)
- snx
 - cr_source.h, [226](#)
- solver1D.h
 - advectionSolverE1, [244](#)
 - advectionSolverE2, [244](#)
 - advectionSolverE, [244](#)
 - advectionSolverX, [245](#)
 - CRsInjectionSourceSolver, [245](#)
 - dilute_solver, [245](#)
 - electron_source, [245](#)
 - NotMove, [246](#)
 - perpendicular_diffusion_solver, [246](#)
 - sourceGrowthDampRateSolver, [246](#)
 - sourceSolver, [246](#)
 - thetaDiffusionSolver, [247](#)
- solver_Dilution
 - constants.h, [213](#)
- solver_ImAdvection
 - constants.h, [213](#)
- solver_ImDampGrowth
 - constants.h, [213](#)
- solver_ImSource1
 - constants.h, [213](#)
- solver_IpAdvection
 - constants.h, [213](#)
- solver_IpDampGrowth
 - constants.h, [213](#)
- solver_IpSource1
 - constants.h, [214](#)
- solver_PcrAdvection
 - constants.h, [214](#)
- solver_PcrAdvection2
 - constants.h, [214](#)
- solver_PcrAdvectionE
 - constants.h, [214](#)
- solver_PcrDiffusion
 - constants.h, [214](#)
- solver_PcrPerpDiff
 - constants.h, [214](#)
- solver_PcrSource1
 - constants.h, [215](#)
- solver_PcrSource2
 - constants.h, [215](#)
- solver_PeAdvection
 - constants.h, [215](#)
- solver_PeAdvection2

- constants.h, 215
- solver_PeAdvectionE1
 - constants.h, 215
- solver_PeAdvectionE2
 - constants.h, 216
- solver_PeAdvectionE
 - constants.h, 215
- solver_PeDiffusion
 - constants.h, 216
- solver_PePerpDiff
 - constants.h, 216
- solver_PeSource1
 - constants.h, 216
- solver_PeSource2
 - constants.h, 216
- source_center
 - show_data, 172
- source_terms_exact
 - constants.h, 216
- sourceGrowthDampRateSolver
 - solver1D.h, 246
- sourceSolver
 - solver1D.h, 246
- sparse
 - pcr_ip_2D, 120
- spec
 - setup_adve, 142
 - setup_advst, 148
- specificOutputData
 - out.h, 241
- Split_solvers, 178
 - c, 179
 - D, 179
 - dt, 179
 - generalized_diffusion, 179
 - NX, 179
 - t, 180
 - t_ini, 180
 - t_max, 180
 - TDMA, 179
 - u, 180
 - u_new_0, 180
 - u_new_1, 180
 - u_old, 180
 - X, 180
 - Xmax, 181
 - Xmin, 181
- sT
 - cr_source.h, 226
- step_implicit
 - constants.h, 217
- sub_x
 - background_diffusion_coefficient_3, 26
 - damping_models, 39
 - EscapeModel_protons_2, 65
 - SNR_evolution, 178
- sub_y
 - background_diffusion_coefficient_3, 26
- damping_models, 39
- EscapeModel_protons_2, 65
- SNR_evolution, 178
- sX
 - cr_source.h, 226
- T
 - background_diffusion_coefficient, 12
 - cr_source.h, 226
 - namelist, 80
 - namelist_adv, 86
 - namelist_adve, 92
 - namelist_advst, 97
 - namelist_diff, 103
 - namelist_uniform, 109
 - namelist_vdiff, 115
 - PDE_solvers, 123
 - setup, 134
 - setup_adv, 140
 - setup_adve, 145
 - setup_advst, 151
 - setup_diff, 156
 - setup_uniform, 163
 - setup_vdiff, 169
- t
 - EscapeModel_protons, 56
 - Split_solvers, 180
 - test_tesc, 184
- t_data_out_max
 - constants.h, 217
- t_data_out_min
 - constants.h, 217
- t_end_injection
 - constants.h, 217
- t_ini
 - pcr_ip_2D, 121
 - Split_solvers, 180
- t_max
 - pcr_ip_2D, 121
 - Split_solvers, 180
- t_new
 - EscapeModel_protons, 56
- t_start_injection
 - constants.h, 217
- TDMA
 - mathematics.h, 234
 - PDE_solvers, 124
 - Split_solvers, 179
- tMCS
 - EscapeModel_protons, 56
- tPDS
 - EscapeModel_protons, 57
- tSed
 - EscapeModel_protons, 57
- ta
 - gaussian_subNormalization, 71
- tc
 - gaussian_subNormalization, 71
- tesc

- cr_source.h, [223](#)
 - electrons_emax_t, [47](#)
 - EscapeModel_protons, [56](#)
 - gaussian_subNormalization, [71](#)
 - test_tesc, [182](#)
- tesc_CNM_3
 - EscapeModel_protons_2, [65](#)
- tesc_CNM
 - EscapeModel_protons_2, [65](#)
- tesc_DiM_3
 - EscapeModel_protons_2, [66](#)
- tesc_DiM
 - EscapeModel_protons_2, [66](#)
- tesc_HII_3
 - EscapeModel_protons_2, [66](#)
- tesc_HII
 - EscapeModel_protons_2, [66](#)
- tesc_WIM_3
 - EscapeModel_protons_2, [66](#)
- tesc_WIM
 - EscapeModel_protons_2, [66](#)
- tesc_WNM_3
 - EscapeModel_protons_2, [66](#)
- tesc_WNM
 - EscapeModel_protons_2, [66](#)
- tesc_e
 - cr_source.h, [224](#)
- tesc_model
 - constants.h, [217](#)
- test_tesc, [181](#)
 - beta, [182](#)
 - c, [182](#)
 - E, [182](#)
 - e, [182](#)
 - Emin, [183](#)
 - Esn, [183](#)
 - figsize, [183](#)
 - gauss, [181](#)
 - GeV, [183](#)
 - kyr, [183](#)
 - label, [183](#)
 - Qcr, [183](#)
 - rho_0, [183](#)
 - sig, [182](#)
 - t, [184](#)
 - tesc, [182](#)
 - xhi_0, [184](#)
 - xhi_cr, [184](#)
- TeV
 - constants, [30](#)
 - constants.h, [218](#)
- tfree
 - EscapeModel_protons, [56](#)
- theta
 - cr_source.h, [224](#)
- thetaDiffusionSolver
 - solver1D.h, [247](#)
- time_distrib_of_data
 - constants.h, [218](#)
- time_test
 - pcr_ip_2D, [121](#)
- tini
 - EscapeModel_protons, [56](#)
- Tmax
 - constants.h, [218](#)
 - PDE_solvers, [125](#)
- tmax
 - EscapeModel_protons, [56](#)
 - gaussian_subNormalization, [71](#)
- tmerge
 - EscapeModel_protons, [57](#)
- tmin
 - gaussian_subNormalization, [71](#)
- tools.h
 - absmaxElement2D, [248](#)
 - maxElement1D, [248](#)
 - maxElement2D, [248](#)
 - minElement1D, [248](#)
 - minElement2D, [249](#)
- total_path
 - namelist, [80](#)
 - namelist_adv, [86](#)
 - namelist_adve, [92](#)
 - namelist_advst, [97](#)
 - namelist_diff, [103](#)
 - namelist_uniform, [109](#)
 - namelist_vdiff, [115](#)
- transpose
 - mathematics.h, [235](#)
- tseed
 - electrons_emax_t, [47](#)
- ttau_sat
 - constants.h, [218](#)
- u
 - PDE_solvers, [125](#)
 - Split_solvers, [180](#)
- u0
 - PDE_solvers, [125](#)
- u_0
 - PDE_solvers, [125](#)
- u_1
 - PDE_solvers, [125](#)
- u_2
 - PDE_solvers, [125](#)
- u_end
 - PDE_solvers, [126](#)
- u_ini
 - PDE_solvers, [126](#)
- u_new_0
 - Split_solvers, [180](#)
- u_new_1
 - Split_solvers, [180](#)
- u_old
 - Split_solvers, [180](#)
- u_sh
 - cr_source.h, [224](#)

- EscapeModel_protons, 57
- uM
 - PDE_solvers, 126
- v
 - background_diffusion_coefficient, 12
 - background_diffusion_coefficient_2, 16
- VAi
 - background_diffusion_coefficient, 12
- VA
 - background_diffusion_coefficient, 12
 - setup, 134
 - setup_adv, 140
 - setup_adve, 146
 - setup_advst, 151
 - setup_diff, 157
 - setup_uniform, 163
 - setup_vdiff, 169
- va
 - namelist, 80
 - namelist_adv, 86
 - namelist_adve, 92
 - namelist_advst, 97
 - namelist_diff, 103
 - namelist_uniform, 109
 - namelist_vdiff, 115
 - setup, 134
 - setup_adv, 140
 - setup_adve, 145
 - setup_advst, 151
 - setup_diff, 157
 - setup_uniform, 163
 - setup_vdiff, 169
- variable
 - setup, 135
 - setup_adv, 140
 - setup_adve, 146
 - setup_advst, 151
 - setup_diff, 157
 - setup_uniform, 163
 - setup_vdiff, 169
- variables
 - setup, 135
 - setup_adv, 140
 - setup_adve, 146
 - setup_advst, 152
 - setup_diff, 157
 - setup_uniform, 163
 - setup_vdiff, 169
- vej8
 - electrons_emax_t, 47
 - EscapeModel_protons, 57
- verbose
 - constants.h, 218
- vlim
 - show_data, 172
- w
 - background_diffusion_coefficient_2, 16
- wl_Alfven
 - damping_models, 39
- wl_Alfven_o1
 - damping_models, 40
- wl_Alfven_o2
 - damping_models, 40
- WIM
 - background_diffusion_coefficient_3, 26
 - phases_collection, 128
- WNM
 - background_diffusion_coefficient_3, 26
 - phases_collection, 128
- wR_Alfven
 - damping_models, 40
- wR_Alfven_o1
 - damping_models, 40
- wR_Alfven_o2
 - damping_models, 40
- write1Daxis
 - fwritter, 68
- write1D
 - fwritter, 68
- write2D
 - fwritter, 68
- writelnInfo
 - fwritter.h, 229
- writeXE
 - fwritter.h, 229
- wspace
 - background_diffusion_coefficient_3, 26
 - damping_models, 40
 - EscapeModel_protons_2, 67
 - SNR_evolution, 178
- wtot
 - background_diffusion_coefficient_2, 16
- X
 - Data_reader, 42
 - namelist, 80
 - namelist_adv, 86
 - namelist_adve, 92
 - namelist_advst, 98
 - namelist_diff, 103
 - namelist_uniform, 110
 - namelist_vdiff, 115
 - PDE_solvers, 126
 - pcr_ip_2D, 121
 - setup, 135
 - setup_adv, 140
 - setup_adve, 146
 - setup_advst, 152
 - setup_diff, 157
 - setup_uniform, 164
 - setup_vdiff, 169
 - ShowInjectionEvolution, 174
 - Split_solvers, 180
- x
 - damping_models, 40
- x0

- EscapeModel_protons, [57](#)
- x_center
 - cr_source.h, [226](#)
 - pcr_ip_2D, [121](#)
 - setup, [135](#)
 - setup_adv, [141](#)
 - setup_adve, [146](#)
 - setup_advst, [152](#)
 - setup_diff, [157](#)
 - setup_uniform, [164](#)
 - setup_vdiff, [170](#)
- x_center_index
 - cr_source.h, [226](#)
 - setup, [135](#)
 - setup_adv, [141](#)
 - setup_adve, [146](#)
 - setup_advst, [152](#)
 - setup_diff, [158](#)
 - setup_uniform, [164](#)
 - setup_vdiff, [170](#)
- XV_log
 - pcr_ip_2D, [121](#)
- xgridtype
 - namelist, [81](#)
 - namelist_adv, [86](#)
 - namelist_adve, [92](#)
 - namelist_advst, [98](#)
 - namelist_diff, [103](#)
 - namelist_uniform, [110](#)
 - namelist_vdiff, [115](#)
- xhi_0
 - constants.h, [218](#)
 - test_tesc, [184](#)
- xhi_cr
 - constants.h, [219](#)
 - electrons_emax_t, [47](#)
 - EscapeModel_protons, [57](#)
 - test_tesc, [184](#)
- xhi_m
 - electrons_emax_t, [47](#)
 - EscapeModel_protons, [58](#)
- Xi
 - cr_source.h, [227](#)
 - namelist, [81](#)
 - namelist_adv, [87](#)
 - namelist_adve, [92](#)
 - namelist_advst, [98](#)
 - namelist_diff, [104](#)
 - namelist_uniform, [110](#)
 - namelist_vdiff, [116](#)
 - setup, [135](#)
 - setup_adv, [141](#)
 - setup_adve, [147](#)
 - setup_advst, [152](#)
 - setup_diff, [158](#)
 - setup_uniform, [164](#)
 - setup_vdiff, [170](#)
- xi_n
 - constants.h, [219](#)
- xlim
 - damping_models, [40](#)
- xlims
 - damping_models, [41](#)
- Xmax
 - namelist, [81](#)
 - namelist_adv, [87](#)
 - namelist_adve, [92](#)
 - namelist_advst, [98](#)
 - namelist_diff, [104](#)
 - namelist_uniform, [110](#)
 - namelist_vdiff, [116](#)
 - PDE_solvers, [126](#)
 - Split_solvers, [181](#)
- xmax
 - background_diffusion_coefficient_3, [27](#)
- Xmin
 - namelist, [81](#)
 - namelist_adv, [87](#)
 - namelist_adve, [93](#)
 - namelist_advst, [98](#)
 - namelist_diff, [104](#)
 - namelist_uniform, [110](#)
 - namelist_vdiff, [116](#)
 - PDE_solvers, [126](#)
 - Split_solvers, [181](#)
- xmin
 - background_diffusion_coefficient_3, [27](#)
- XV
 - pcr_ip_2D, [121](#)
- y
 - damping_models, [41](#)
- ylim
 - damping_models, [41](#)
- ylims
 - damping_models, [41](#)
- ymax
 - background_diffusion_coefficient_3, [27](#)
- ymin
 - background_diffusion_coefficient_3, [27](#)
- yr
 - constants, [30](#)
 - constants.h, [219](#)
 - PDE_solvers, [126](#)