

GLISSANDO 3

Generated by Doxygen 1.8.6

Wed Jun 27 2018 10:27:50

Contents

1	Main Page	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Class Documentation	9
5.1	bin_t Struct Reference	9
5.1.1	Detailed Description	9
5.1.2	Member Data Documentation	9
5.1.2.1	bin	9
5.2	collision Class Reference	9
5.2.1	Detailed Description	10
5.2.2	Constructor & Destructor Documentation	10
5.2.2.1	collision	10
5.2.2.2	collision	11
5.2.2.3	collision	11
5.2.3	Member Function Documentation	11
5.2.3.1	gen_RDS	11
5.2.3.2	operator=	11
5.2.3.3	shift_cmx_w_c	11
5.2.3.4	shift_cmy_w_c	11
5.2.3.5	writerds	11
5.2.4	Member Data Documentation	12
5.2.4.1	nbin	12
5.2.4.2	nhotspot	12
5.2.4.3	nwA	12
5.2.4.4	nwAB	12

5.2.4.5	nwB	12
5.2.4.6	nzw	12
5.2.4.7	rd2	12
5.2.4.8	rpa	12
5.2.4.9	specA	12
5.2.4.10	specB	12
5.2.4.11	wc	12
5.2.4.12	wwA	13
5.2.4.13	wwAB	13
5.2.4.14	wwB	13
5.2.4.15	xcmA	13
5.2.4.16	xcmB	13
5.2.4.17	ycmA	13
5.2.4.18	ycmB	13
5.3	counter Class Reference	13
5.3.1	Detailed Description	14
5.3.2	Constructor & Destructor Documentation	14
5.3.2.1	counter	14
5.3.2.2	~counter	14
5.3.3	Member Function Documentation	14
5.3.3.1	add	14
5.3.3.2	get	14
5.3.3.3	getN	14
5.3.3.4	mean	14
5.3.3.5	reset	14
5.3.4	Member Data Documentation	15
5.3.4.1	count	15
5.3.4.2	value	15
5.4	counter2 Class Reference	15
5.4.1	Detailed Description	16
5.4.2	Constructor & Destructor Documentation	16
5.4.2.1	counter2	16
5.4.2.2	~counter2	16
5.4.3	Member Function Documentation	16
5.4.3.1	add	16
5.4.3.2	get	16
5.4.3.3	get2	16
5.4.3.4	getN	16
5.4.3.5	mean	16
5.4.3.6	reset	16

5.4.3.7	var	16
5.4.3.8	vara	17
5.4.4	Member Data Documentation	17
5.4.4.1	count	17
5.4.4.2	value	17
5.4.4.3	value2	17
5.5	counter_2D Class Reference	17
5.5.1	Detailed Description	18
5.5.2	Constructor & Destructor Documentation	18
5.5.2.1	counter_2D	18
5.5.2.2	~counter_2D	18
5.5.3	Member Function Documentation	18
5.5.3.1	add	18
5.5.3.2	corr	19
5.5.3.3	cov	19
5.5.3.4	get_x	19
5.5.3.5	get_x2	19
5.5.3.6	get_xy	19
5.5.3.7	get_y	19
5.5.3.8	get_y2	19
5.5.3.9	getN	19
5.5.3.10	mean_x	19
5.5.3.11	mean_y	19
5.5.3.12	reset	19
5.5.3.13	var_x	20
5.5.3.14	var_y	20
5.5.4	Member Data Documentation	20
5.5.4.1	count	20
5.5.4.2	valuex	20
5.5.4.3	valuex2	20
5.5.4.4	valuexy	20
5.5.4.5	valuey	20
5.5.4.6	valuey2	20
5.6	distr Class Reference	20
5.6.1	Detailed Description	24
5.6.2	Constructor & Destructor Documentation	24
5.6.2.1	distr	24
5.6.2.2	distr	24
5.6.2.3	distr	24
5.6.3	Member Function Documentation	24

5.6.3.1	cmx	24
5.6.3.2	cmx_w	24
5.6.3.3	cmx_w	24
5.6.3.4	cmy	24
5.6.3.5	cmy_w	25
5.6.3.6	cmy_w	25
5.6.3.7	cmz	25
5.6.3.8	cmz_w	25
5.6.3.9	eps	25
5.6.3.10	eps	25
5.6.3.11	eps	25
5.6.3.12	eps	26
5.6.3.13	eps2p_sq	26
5.6.3.14	eps_s	26
5.6.3.15	eps_s	26
5.6.3.16	eps_s	26
5.6.3.17	eps_s	27
5.6.3.18	fill_polar	27
5.6.3.19	fill_polar	27
5.6.3.20	fill_polar_s	27
5.6.3.21	fill_polar_s	27
5.6.3.22	fill_xy	28
5.6.3.23	fill_xy	28
5.6.3.24	mA	28
5.6.3.25	mAnz	28
5.6.3.26	mB	28
5.6.3.27	mBnz	28
5.6.3.28	msrad	29
5.6.3.29	msrad_t	29
5.6.3.30	msrad_t_w	29
5.6.3.31	msrad_w	29
5.6.3.32	msx	29
5.6.3.33	msy	29
5.6.3.34	mxy	29
5.6.3.35	operator=	29
5.6.3.36	phrot	29
5.6.3.37	phrot	29
5.6.3.38	qx	30
5.6.3.39	qx	30
5.6.3.40	qy	30

5.6.3.41	qy	30
5.6.3.42	rotate	31
5.6.3.43	rotate_polar	32
5.6.3.44	shift_cmx	32
5.6.3.45	shift_cmx_w	32
5.6.3.46	shift_cmx_w	32
5.6.3.47	shift_cmy	32
5.6.3.48	shift_cmy_w	32
5.6.3.49	shift_cmy_w	32
5.6.3.50	shift_cmz	32
5.6.3.51	shift_cmz_w	32
5.6.3.52	shift_x	33
5.6.3.53	shift_y	34
5.6.3.54	shift_z	34
5.6.3.55	size	34
5.6.3.56	sum_w	34
5.6.3.57	sum_w	34
5.6.3.58	surf	34
5.6.3.59	tilt	34
5.6.3.60	uncluster	34
5.6.4	Member Data Documentation	35
5.6.4.1	c	35
5.6.4.2	msr	35
5.6.4.3	msrt	35
5.6.4.4	n	35
5.6.4.5	sumw	35
5.6.4.6	w	35
5.6.4.7	x	35
5.6.4.8	xcm	35
5.6.4.9	y	35
5.6.4.10	ycm	35
5.6.4.11	ye	35
5.6.4.12	ys	36
5.6.4.13	z	36
5.6.4.14	zcm	36
5.7	nucleus Class Reference	36
5.7.1	Detailed Description	37
5.7.2	Constructor & Destructor Documentation	38
5.7.2.1	nucleus	38
5.7.2.2	nucleus	38

5.7.3	Member Function Documentation	38
5.7.3.1	dist2	38
5.7.3.2	good_all	38
5.7.3.3	good_down	38
5.7.3.4	good_pair	38
5.7.3.5	operator=	39
5.7.3.6	set_alpha_cluster	39
5.7.3.7	set_berillium_7_cluster	39
5.7.3.8	set_berillium_8_cluster	39
5.7.3.9	set_berillium_9_cluster	39
5.7.3.10	set_carbon_cluster	39
5.7.3.11	set_deuteron	40
5.7.3.12	set_file	40
5.7.3.13	set_file_uncor	40
5.7.3.14	set_oxygen_cluster	40
5.7.3.15	set_oxygen_cluster_square	41
5.7.3.16	set_parton_A	41
5.7.3.17	set_parton_B	41
5.7.3.18	set_proton	41
5.7.3.19	set_random_A	41
5.7.3.20	set_random_A	41
5.7.3.21	set_random_A_def	42
5.7.3.22	set_random_A_def	42
5.7.3.23	set_random_A_hos	42
5.7.3.24	set_random_A_hos	42
5.7.3.25	set_random_B	42
5.7.3.26	set_random_B	42
5.7.3.27	set_random_B_def	42
5.7.3.28	set_random_B_def	42
5.7.3.29	set_random_B_hos	43
5.7.3.30	set_random_B_hos	43
5.7.3.31	set_tritium	43
5.7.4	Member Data Documentation	43
5.7.4.1	cth	43
5.7.4.2	g	43
5.7.4.3	phi	43
5.7.4.4	r	43
5.7.4.5	sth	43
5.8	SOURCE Struct Reference	43
5.8.1	Detailed Description	44

5.8.2	Member Data Documentation	44
5.8.2.1	KK	44
5.8.2.2	W	44
5.8.2.3	X	44
5.8.2.4	Y	44
5.9	tr_his_c Class Reference	44
5.9.1	Detailed Description	47
5.9.2	Member Function Documentation	47
5.9.2.1	fill	47
5.9.2.2	fill_res	47
5.9.2.3	fill_tr	47
5.9.2.4	gen	47
5.9.2.5	init	47
5.9.2.6	write	47
5.9.2.7	write_d	48
5.9.2.8	write_r	48
5.9.2.9	write_w	48
5.9.2.10	write_wpro	48
5.9.3	Member Data Documentation	48
5.9.3.1	full_event	48
5.9.3.2	nbinar	48
5.9.3.3	nbinar2	48
5.9.3.4	neps	48
5.9.3.5	neps2	48
5.9.3.6	neps2b	48
5.9.3.7	neps4	48
5.9.3.8	nepsb	48
5.9.3.9	nepsp	49
5.9.3.10	nepsp2	49
5.9.3.11	nepsp22	49
5.9.3.12	nepsp2b	49
5.9.3.13	nepsp4	49
5.9.3.14	nepspb	49
5.9.3.15	nsize	49
5.9.3.16	nsize2	49
5.9.3.17	ntarg	49
5.9.3.18	ntarg2	49
5.9.3.19	nuni	49
5.9.3.20	nunib	49
5.9.3.21	nuniRDS	50

5.9.3.22	nunp	50
5.9.3.23	nw2b	50
5.9.3.24	nwb	50
5.9.3.25	nwei	50
5.9.3.26	nwei2	50
5.9.3.27	nx	50
5.9.3.28	nx2	50
5.9.3.29	ny	50
5.9.3.30	ny2	50
5.9.3.31	param	50
5.9.3.32	phys	50
5.9.3.33	radA	51
5.9.3.34	radA2	51
5.9.3.35	radA3	51
5.9.3.36	radB	51
5.9.3.37	rcostheta_nuclA	51
5.9.3.38	rcostheta_nuclB	51
5.9.3.39	rrel_u	51
5.9.3.40	rrelA	51
5.9.3.41	rrelB	51
5.9.3.42	tree	51
5.9.3.43	weih	51
5.9.3.44	weih_bin	51
5.9.3.45	wpro	51
5.9.3.46	xyhist	52
5.9.3.47	xyhist_core	52
5.9.3.48	xyhist_mantle	52
5.9.3.49	xyhist_nuclA	52
5.9.3.50	xyhist_nuclB	52
5.9.3.51	xyhistr	52
6	File Documentation	53
6.1	build/include/collision.h File Reference	53
6.1.1	Detailed Description	53
6.2	build/include/counter.h File Reference	53
6.2.1	Detailed Description	53
6.3	build/include/distrib.h File Reference	53
6.3.1	Detailed Description	54
6.4	build/include/functions.h File Reference	54
6.4.1	Detailed Description	61

6.4.2	Function Documentation	61
6.4.2.1	disp	61
6.4.2.2	dist	62
6.4.2.3	echopar	62
6.4.2.4	epilog	62
6.4.2.5	fg	62
6.4.2.6	fpm	62
6.4.2.7	fqscale	62
6.4.2.8	fsNN	62
6.4.2.9	fsQQ	63
6.4.2.10	gamgen	63
6.4.2.11	header	63
6.4.2.12	helper	63
6.4.2.13	los	63
6.4.2.14	los_rap_A	63
6.4.2.15	los_rap_B	63
6.4.2.16	los_rap_bin	64
6.4.2.17	negbin	64
6.4.2.18	readpar	64
6.4.2.19	reset_counters	64
6.4.2.20	rlos_abf	64
6.4.2.21	rlos_alpha	64
6.4.2.22	rlos_hole	64
6.4.2.23	rlos_hult	64
6.4.2.24	rlos_parton	65
6.4.2.25	rlosA	65
6.4.2.26	rlosA_def	65
6.4.2.27	rlosA_hos	65
6.4.2.28	rlosB	65
6.4.2.29	rlosB_def	65
6.4.2.30	rlosB_hos	65
6.4.2.31	time_start	66
6.4.2.32	time_stop	66
6.4.3	Variable Documentation	66
6.4.3.1	ALPHA	66
6.4.3.2	angles	66
6.4.3.3	ARANK	66
6.4.3.4	AWSA	66
6.4.3.5	AWSB	66
6.4.3.6	b	66

6.4.3.7	BETA2A	66
6.4.3.8	BETA2B	66
6.4.3.9	BETA4A	66
6.4.3.10	BETA4B	67
6.4.3.11	BMAX	67
6.4.3.12	BMIN	67
6.4.3.13	BTOT	67
6.4.3.14	CD	67
6.4.3.15	d	67
6.4.3.16	DBIN	67
6.4.3.17	dbin	67
6.4.3.18	DOBIN	67
6.4.3.19	DS	67
6.4.3.20	DW	67
6.4.3.21	ECM	67
6.4.3.22	ep	68
6.4.3.23	ep1	68
6.4.3.24	ep1s	68
6.4.3.25	ep22	68
6.4.3.26	ep3	68
6.4.3.27	ep32	68
6.4.3.28	ep3s	68
6.4.3.29	ep4	68
6.4.3.30	ep42	68
6.4.3.31	ep4s	68
6.4.3.32	ep5	68
6.4.3.33	ep5s	68
6.4.3.34	ep6	68
6.4.3.35	ep6s	68
6.4.3.36	epA2	69
6.4.3.37	epA3	69
6.4.3.38	epA4	69
6.4.3.39	epart	69
6.4.3.40	epart1	69
6.4.3.41	epart3	69
6.4.3.42	epart4	69
6.4.3.43	epart5	69
6.4.3.44	epart6	69
6.4.3.45	eps	69
6.4.3.46	es	69

6.4.3.47	es3	69
6.4.3.48	es3s	69
6.4.3.49	es4	69
6.4.3.50	es4s	70
6.4.3.51	es5	70
6.4.3.52	es5s	70
6.4.3.53	es6	70
6.4.3.54	es6s	70
6.4.3.55	ess	70
6.4.3.56	ETA0	70
6.4.3.57	ETACC	70
6.4.3.58	ETAM	70
6.4.3.59	evall	70
6.4.3.60	EVENTS	70
6.4.3.61	FBIN	70
6.4.3.62	FBRAP	71
6.4.3.63	FILES	71
6.4.3.64	FULL	71
6.4.3.65	GA	71
6.4.3.66	GAMA	71
6.4.3.67	ISEED	71
6.4.3.68	ISEED1	71
6.4.3.69	kk	71
6.4.3.70	MAXYRAP	71
6.4.3.71	MODEL	71
6.4.3.72	NBIN	71
6.4.3.73	nbinary	71
6.4.3.74	NCS	72
6.4.3.75	nhot	72
6.4.3.76	NNWP	72
6.4.3.77	NUMA	72
6.4.3.78	NUMB	72
6.4.3.79	NUMRAP	72
6.4.3.80	nweight	72
6.4.3.81	nwounded	72
6.4.3.82	OMEGA	72
6.4.3.83	PARTONS	72
6.4.3.84	phirot	72
6.4.3.85	phirot3	72
6.4.3.86	phirot4	72

6.4.3.87	phirot5	73
6.4.3.88	phirot6	73
6.4.3.89	phirot_minus	73
6.4.3.90	phirot_plus	73
6.4.3.91	PI	73
6.4.3.92	PP	73
6.4.3.93	ppp	73
6.4.3.94	QSCALE	73
6.4.3.95	RAPRANGE	73
6.4.3.96	RAPSHIFT	73
6.4.3.97	rbin	73
6.4.3.98	RCHA	73
6.4.3.99	RCHB	74
6.4.3.100	RCHP	74
6.4.3.101	RDS0	74
6.4.3.102	RDS1	74
6.4.3.103	RET	74
6.4.3.104	rhotspot	74
6.4.3.105	RO	74
6.4.3.106	roo	74
6.4.3.107	ROTA_PHI	74
6.4.3.108	ROTA_THETA	74
6.4.3.109	ROTB_PHI	74
6.4.3.110	ROTB_THETA	74
6.4.3.111	rpa	75
6.4.3.112	rwA	75
6.4.3.113	rwAB	75
6.4.3.114	rwB	75
6.4.3.115	RWSA	75
6.4.3.116	RWSB	75
6.4.3.117	SBIN	75
6.4.3.118	SCALEA	75
6.4.3.119	SCALEB	75
6.4.3.120	SHIFT	75
6.4.3.121	SIGETA	75
6.4.3.122	SIGMAA	75
6.4.3.123	SIGMAB	76
6.4.3.124	SIGMABISA	76
6.4.3.125	SIGMABISB	76
6.4.3.126	sirad	76

6.4.3.127	sitot	76
6.4.3.128	sizeav	76
6.4.3.129	SNN	76
6.4.3.130	surfA	76
6.4.3.131	tiltA	76
6.4.3.132	Ubin	76
6.4.3.133	Uw	76
6.4.3.134	Vbin	76
6.4.3.135	Vw	77
6.4.3.136	W0	77
6.4.3.137	W1	77
6.4.3.138	WFA	77
6.4.3.139	WFB	77
6.4.3.140	wfq	77
6.4.3.141	wfqA	77
6.4.3.142	wfqB	77
6.4.3.143	WMIN	77
6.4.3.144	xepp	77
6.4.3.145	xsepp	77
6.4.3.146	xx	77
6.4.3.147	YB	78
6.4.3.148	yy	78
6.5	build/src/glissando3.cxx File Reference	78
6.5.1	Detailed Description	78
6.5.2	Macro Definition Documentation	79
6.5.2.1	_VER_	79
6.5.3	Function Documentation	79
6.5.3.1	main	79
6.5.4	Variable Documentation	80
6.5.4.1	raa	80
6.5.4.2	ver	80
6.6	macro/angles.C File Reference	80
6.6.1	Detailed Description	80
6.6.2	Function Documentation	80
6.6.2.1	angles	81
6.7	macro/cen_exa.C File Reference	81
6.7.1	Function Documentation	81
6.7.1.1	cen_exa	81
6.8	macro/cen_exa_RDS.C File Reference	81
6.8.1	Function Documentation	81

6.8.1.1	cen_exa_RDS	81
6.9	macro/demo_3/cen_exa_RDS.C File Reference	82
6.9.1	Function Documentation	82
6.9.1.1	cen_exa_RDS	82
6.10	macro/cen_exa_RDS_moj.C File Reference	82
6.10.1	Function Documentation	82
6.10.1.1	cen_exa_RDS_moj	82
6.11	macro/cen_ratio.C File Reference	83
6.11.1	Function Documentation	83
6.11.1.1	cen_ratio	83
6.12	macro/cenpPb.C File Reference	83
6.12.1	Detailed Description	83
6.12.2	Function Documentation	83
6.12.2.1	cenpPb	83
6.13	macro/centrality.C File Reference	84
6.13.1	Function Documentation	84
6.13.1.1	centrality	84
6.14	macro/demo_2/centrality.C File Reference	84
6.14.1	Function Documentation	84
6.14.1.1	centrality	84
6.15	macro/centrality2.C File Reference	85
6.15.1	Function Documentation	85
6.15.1.1	centrality2	85
6.16	macro/demo_2/centrality2.C File Reference	85
6.16.1	Function Documentation	85
6.16.1.1	centrality2	85
6.17	macro/centrality2l.C File Reference	86
6.17.1	Function Documentation	86
6.17.1.1	centrality2l	86
6.18	macro/centrality3.C File Reference	86
6.18.1	Detailed Description	86
6.18.2	Function Documentation	86
6.18.2.1	centrality3	86
6.19	macro/col_chain_8_q.C File Reference	87
6.19.1	Function Documentation	87
6.19.1.1	col_chain_8_q	87
6.19.1.2	f2	87
6.19.1.3	f3	87
6.19.1.4	krot	87
6.19.1.5	los	87

6.19.2	Variable Documentation	87
6.19.2.1	raa	87
6.20	macro/core_mantle.C File Reference	87
6.20.1	Function Documentation	88
6.20.1.1	core_mantle	88
6.21	macro/demo_2/core_mantle.C File Reference	89
6.21.1	Function Documentation	89
6.21.1.1	core_mantle	89
6.22	macro/corr.C File Reference	89
6.22.1	Function Documentation	89
6.22.1.1	corr	89
6.23	macro/demo_2/corr.C File Reference	90
6.23.1	Function Documentation	90
6.23.1.1	corr	90
6.24	macro/demo_2/density.C File Reference	90
6.24.1	Function Documentation	90
6.24.1.1	density	90
6.25	macro/density.C File Reference	90
6.25.1	Function Documentation	91
6.25.1.1	density	91
6.26	macro/demo_2/dxdy.C File Reference	91
6.26.1	Function Documentation	91
6.26.1.1	dxdy	91
6.27	macro/dxdy.C File Reference	91
6.27.1	Function Documentation	92
6.27.1.1	dxdy	92
6.28	macro/demo_2/epsilon.C File Reference	92
6.28.1	Function Documentation	92
6.28.1.1	epsilon	92
6.29	macro/epsilon.C File Reference	92
6.29.1	Function Documentation	92
6.29.1.1	epsilon	93
6.30	macro/demo_2/epsilon_b.C File Reference	93
6.30.1	Function Documentation	93
6.30.1.1	epsilon_b	93
6.31	macro/epsilon_b.C File Reference	93
6.31.1	Function Documentation	93
6.31.1.1	epsilon_b	94
6.32	macro/demo_2/epsilon_c.C File Reference	94
6.32.1	Function Documentation	94

6.32.1.1	epsilon_c	94
6.33	macro/epsilon_c.C File Reference	94
6.33.1	Function Documentation	94
6.33.1.1	epsilon_c	94
6.34	macro/demo_2/fitr.C File Reference	95
6.34.1	Function Documentation	95
6.34.1.1	fitr	95
6.34.1.2	saxon	95
6.35	macro/fitr.C File Reference	95
6.35.1	Function Documentation	96
6.35.1.1	fitr	96
6.35.1.2	saxon	96
6.36	macro/demo_2/fourier.C File Reference	96
6.36.1	Function Documentation	96
6.36.1.1	fourier	96
6.37	macro/fourier.C File Reference	96
6.37.1	Function Documentation	97
6.37.1.1	fourier	97
6.38	macro/demo_2/info.C File Reference	97
6.38.1	Function Documentation	97
6.38.1.1	info	97
6.39	macro/info.C File Reference	97
6.39.1	Function Documentation	97
6.39.1.1	info	97
6.40	macro/demo_2/label.C File Reference	98
6.40.1	Function Documentation	98
6.40.1.1	label	98
6.40.1.2	label_fit	98
6.41	macro/demo_3/label.C File Reference	98
6.41.1	Function Documentation	98
6.41.1.1	label	98
6.41.1.2	label_fit	98
6.42	macro/label.C File Reference	99
6.42.1	Function Documentation	99
6.42.1.1	label	99
6.42.1.2	label_fit	99
6.43	macro/demo_2/mult.C File Reference	99
6.43.1	Function Documentation	99
6.43.1.1	mult	99
6.44	macro/mult.C File Reference	100

6.44.1	Function Documentation	100
6.44.1.1	mult	100
6.45	macro/demo_2/overlay.C File Reference	100
6.45.1	Function Documentation	100
6.45.1.1	overlay	100
6.46	macro/overlay.C File Reference	100
6.46.1	Function Documentation	101
6.46.1.1	overlay	101
6.47	macro/demo_2/profile2.C File Reference	101
6.47.1	Function Documentation	101
6.47.1.1	profile2	101
6.48	macro/profile2.C File Reference	101
6.48.1	Function Documentation	102
6.48.1.1	profile2	102
6.49	macro/demo_2/profile2_deformation_63Cu.C File Reference	102
6.49.1	Function Documentation	102
6.49.1.1	profile2_deformation_63Cu	102
6.50	macro/profile2_deformation_63Cu.C File Reference	102
6.50.1	Function Documentation	102
6.50.1.1	profile2_deformation_63Cu	102
6.51	macro/demo_2/profile2_deformation_Au.C File Reference	103
6.51.1	Function Documentation	103
6.51.1.1	profile2_deformation_Au	103
6.52	macro/profile2_deformation_Au.C File Reference	103
6.52.1	Function Documentation	103
6.52.1.1	profile2_deformation_Au	103
6.53	macro/demo_2/profile2_deformation_U.C File Reference	104
6.53.1	Function Documentation	104
6.53.1.1	profile2_deformation_U	104
6.54	macro/profile2_deformation_U.C File Reference	104
6.54.1	Function Documentation	104
6.54.1.1	profile2_deformation_U	104
6.55	macro/demo_2/size.C File Reference	105
6.55.1	Function Documentation	105
6.55.1.1	size	105
6.56	macro/size.C File Reference	105
6.56.1	Function Documentation	105
6.56.1.1	size	105
6.57	macro/demo_2/wounding_profile.C File Reference	105
6.57.1	Function Documentation	106

6.57.1.1	wounding_profile	106
6.58	macro/wounding_profile.C File Reference	106
6.58.1	Function Documentation	106
6.58.1.1	wounding_profile	106
6.59	macro/demo_3/cent.C File Reference	106
6.59.1	Detailed Description	106
6.59.2	Function Documentation	107
6.59.2.1	cent	107
6.60	macro/demo_3/inel_prof.C File Reference	107
6.60.1	Function Documentation	107
6.60.1.1	inel_prof	107
6.61	macro/demo_3/rad_dis.C File Reference	107
6.61.1	Detailed Description	107
6.61.2	Function Documentation	107
6.61.2.1	rad_dis	107
6.62	macro/eps_fluct.C File Reference	109
6.62.1	Detailed Description	109
6.62.2	Function Documentation	109
6.62.2.1	eps_fluct	109
6.63	macro/fourier_sm_eps_chain.C File Reference	109
6.63.1	Function Documentation	109
6.63.1.1	fourier_sm_eps_chain	109
6.64	macro/fourier_sm_n2_chain.C File Reference	109
6.64.1	Function Documentation	110
6.64.1.1	fourier_sm_n2_chain	110
6.65	macro/fourier_sm_n4_chain.C File Reference	110
6.65.1	Function Documentation	110
6.65.1.1	fourier_sm_n4_chain	110
6.66	macro/g_005.C File Reference	110
6.66.1	Macro Definition Documentation	110
6.66.1.1	PI	110
6.66.2	Function Documentation	110
6.66.2.1	g_005	110
6.66.2.2	lin	110
6.67	macro/g_005_Q.C File Reference	111
6.67.1	Macro Definition Documentation	111
6.67.1.1	PI	111
6.67.2	Function Documentation	111
6.67.2.1	g_005_Q	111
6.67.2.2	lin	111

6.68	macro/g_005t.C File Reference	111
6.68.1	Macro Definition Documentation	111
6.68.1.1	PI	111
6.68.2	Function Documentation	111
6.68.2.1	g_005	111
6.68.2.2	lin	111
6.69	macro/g_2030.C File Reference	112
6.69.1	Macro Definition Documentation	112
6.69.1.1	PI	112
6.69.2	Function Documentation	112
6.69.2.1	g_2030	112
6.69.2.2	lin	112
6.70	macro/g_2030_Q.C File Reference	112
6.70.1	Macro Definition Documentation	112
6.70.1.1	PI	112
6.70.2	Function Documentation	112
6.70.2.1	g_2030_Q	112
6.70.2.2	lin	112
6.71	macro/g_2030t.C File Reference	113
6.71.1	Macro Definition Documentation	113
6.71.1.1	PI	113
6.71.2	Function Documentation	113
6.71.2.1	g_2030	113
6.71.2.2	lin	113
6.72	macro/g_5060.C File Reference	113
6.72.1	Macro Definition Documentation	113
6.72.1.1	PI	113
6.72.2	Function Documentation	113
6.72.2.1	g_5060	113
6.72.2.2	lin	113
6.73	macro/g_5060_Q.C File Reference	114
6.73.1	Macro Definition Documentation	114
6.73.1.1	PI	114
6.73.2	Function Documentation	114
6.73.2.1	g_5060_Q	114
6.73.2.2	lin	114
6.74	macro/g_5060t.C File Reference	114
6.74.1	Macro Definition Documentation	114
6.74.1.1	PI	114
6.74.2	Function Documentation	114

6.74.2.1	g_5060	114
6.74.2.2	lin	114
6.75	macro/g_pp_n.C File Reference	115
6.75.1	Macro Definition Documentation	115
6.75.1.1	PI	115
6.75.2	Function Documentation	115
6.75.2.1	g_pp_n	115
6.75.2.2	lin	115
6.76	macro/g_pp_q (Case Conflict).C File Reference	115
6.76.1	Macro Definition Documentation	115
6.76.1.1	PI	115
6.76.2	Function Documentation	115
6.76.2.1	g_pp_Q	115
6.76.2.2	lin	115
6.77	macro/g_pp_q.C File Reference	116
6.77.1	Macro Definition Documentation	116
6.77.1.1	PI	116
6.77.2	Function Documentation	116
6.77.2.1	g_pp_q	116
6.77.2.2	lin	116
6.77.2.3	los	116
6.77.3	Variable Documentation	116
6.77.3.1	raa	116
6.78	macro/g_pp_Qt.C File Reference	116
6.78.1	Macro Definition Documentation	117
6.78.1.1	PI	117
6.78.2	Function Documentation	117
6.78.2.1	g_pp_Qt	117
6.78.2.2	lin	117
6.79	macro/g_pPb_03_Q.C File Reference	117
6.79.1	Macro Definition Documentation	117
6.79.1.1	PI	117
6.79.2	Function Documentation	117
6.79.2.1	g_pPb_03_Q	117
6.79.2.2	lin	117
6.80	macro/g_pPb_n21.C File Reference	117
6.80.1	Macro Definition Documentation	118
6.80.1.1	PI	118
6.80.2	Function Documentation	118
6.80.2.1	g_pPb_n21	118

6.80.2.2	lin	118
6.81	macro/g_pPb_n21_Q.C File Reference	118
6.81.1	Macro Definition Documentation	118
6.81.1.1	PI	118
6.81.2	Function Documentation	118
6.81.2.1	g_pPb_n21_Q	118
6.81.2.2	lin	118
6.82	macro/g_pPb_n25.C File Reference	118
6.82.1	Macro Definition Documentation	119
6.82.1.1	PI	119
6.82.2	Function Documentation	119
6.82.2.1	g_pPb_n25	119
6.82.2.2	lin	119
6.83	macro/g_Q_2030.C File Reference	119
6.83.1	Macro Definition Documentation	119
6.83.1.1	PI	119
6.83.2	Function Documentation	119
6.83.2.1	g_Q_2030	119
6.83.2.2	lin	119
6.84	macro/hydro.C File Reference	119
6.84.1	Detailed Description	120
6.84.2	Function Documentation	120
6.84.2.1	hydro	120
6.85	macro/npa_dist.C File Reference	120
6.85.1	Function Documentation	120
6.85.1.1	npa_dist	120
6.86	macro/nq.C File Reference	120
6.86.1	Function Documentation	120
6.86.1.1	nq	120
6.87	macro/nw_dist.C File Reference	120
6.87.1	Function Documentation	121
6.87.1.1	nw_dist	121
6.88	macro/tilted.C File Reference	121
6.88.1	Detailed Description	121
6.88.2	Function Documentation	121
6.88.2.1	tilted	121
6.89	macro/w_prof_norm.C File Reference	121
6.89.1	Function Documentation	121
6.89.1.1	nn	121
6.89.1.2	w_prof_norm	122

6.90 macro/w_prof_norm_wb.C File Reference	123
6.90.1 Function Documentation	123
6.90.1.1 nn	123
6.90.1.2 nn_1	123
6.90.1.3 w_prof_norm_wb	123
6.90.2 Variable Documentation	123
6.90.2.1 gama	123
6.90.2.2 omega	123
6.90.2.3 siexp	123
6.91 macro/w_prof_norm_wb0.C File Reference	123
6.91.1 Function Documentation	124
6.91.1.1 nn	124
6.91.1.2 nn_1	124
6.91.1.3 w_prof_norm_wb0	124
6.91.2 Variable Documentation	124
6.91.2.1 gama	124
6.91.2.2 omega	124
Index	125

Chapter 1

Main Page

GLISSANDO 3 - GLauber Initial State Simulation AND mOre... ver. 3.102, 30 June 2018

Homepage: <http://www.ujk.edu.pl/homepages/mryb/GLISSANDO/index.html>

Authors:

- Wojciech Broniowski (Wojciech.Broniowski@ifj.edu.pl)
- Maciej Rybczynski (Maciej.Rybczynski@ujk.edu.pl)
- Grzegorz Stefanek (Grzegorz.Stefanek@ujk.edu.pl)
- Piotr Bozek (Piotr.Bozek@fis.agh.edu.pl)

For ver. 3, see arXiv: (* fill arXiv ref. *)

For ver. 2, see Computer Physics Communications 185 (2014) 1759, arXiv:1310.5475 [nucl-th]

For ver. 1, see Computer Physics Communications 180(2009)69, arXiv:0710.5731 [nucl-th]

GLISSANDO is a Glauber Monte-Carlo generator for early-stages of relativistic heavy-ion collisions, written in c++ and interfaced to ROOT.

A reference manual, generated by doxygen, is supplied at the Homepage.

The code can be freely used and redistributed. However, if you decide to make modifications, the authors would appreciate notification for the record. In any publication or display of results obtained using GLISSANDO, please, include a reference to our papers

(* fill arXiv ref. *), Computer Physics Communications 180 (2009) 69, arXiv:0710.5731 [nucl-th], and Computer Physics Communications 185 (2014) 1759, arXiv:1310.5475 [nucl-th]

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

bin_t	9
counter	13
counter2	15
counter_2D	17
distr	20
collision	9
nucleus	36
SOURCE	43
tr_his_c	44

Chapter 3

Class Index

3.1 Class List

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

bin_t	Structure for storing observables in subsequent rapidity bins	9
collision	Collision class	9
counter	Simplest counting class	13
counter2	Counting class with variance	15
counter_2D	2-dimensional counting class	17
distr	Distribution of sources in space	20
nucleus	Nucleus class	36
SOURCE	Structure for output of the full event - transverse coordinates, weight, number of the event . . .	43
tr_his_c	Class storing the trees and histograms	44

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

build/include/collision.h	53
build/include/counter.h	53
build/include/distrib.h	53
build/include/functions.h	54
build/src/glissando3.cxx	78
macro/angles.C	80
macro/cen_exa.C	81
macro/cen_exa_RDS.C	81
macro/cen_exa_RDS_moj.C	82
macro/cen_ratio.C	83
macro/cenpPb.C	83
macro/centrality.C	84
macro/centrality2.C	85
macro/centrality2l.C	86
macro/centrality3.C	86
macro/col_chain_8_q.C	87
macro/core_mantle.C	87
macro/corr.C	89
macro/density.C	90
macro/dxdy.C	91
macro/eps_fluct.C	109
macro/epsilon.C	92
macro/epsilon_b.C	93
macro/epsilon_c.C	94
macro/fitr.C	95
macro/fourier.C	96
macro/fourier_sm_eps_chain.C	109
macro/fourier_sm_n2_chain.C	109
macro/fourier_sm_n4_chain.C	110
macro/g_005.C	110
macro/g_005_Q.C	111
macro/g_005t.C	111
macro/g_2030.C	112
macro/g_2030_Q.C	112
macro/g_2030t.C	113
macro/g_5060.C	113
macro/g_5060_Q.C	114
macro/g_5060t.C	114

macro/g_pp_n.C	115
macro/g_pp_q (Case Conflict).C	115
macro/g_pp_q.C	116
macro/g_pp_Qt.C	116
macro/g_pPb_03_Q.C	117
macro/g_pPb_n21.C	117
macro/g_pPb_n21_Q.C	118
macro/g_pPb_n25.C	118
macro/g_Q_2030.C	119
macro/hydro.C	119
macro/info.C	97
macro/label.C	99
macro/mult.C	100
macro/npa_dist.C	120
macro/nq.C	120
macro/nw_dist.C	120
macro/overlay.C	100
macro/profile2.C	101
macro/profile2_deformation_63Cu.C	102
macro/profile2_deformation_Au.C	103
macro/profile2_deformation_U.C	104
macro/size.C	105
macro/tilted.C	121
macro/w_prof_norm.C	121
macro/w_prof_norm_wb.C	123
macro/w_prof_norm_wb0.C	123
macro/wounding_profile.C	106
macro/demo_2/centrality.C	84
macro/demo_2/centrality2.C	85
macro/demo_2/core_mantle.C	89
macro/demo_2/corr.C	90
macro/demo_2/density.C	90
macro/demo_2/dxdy.C	91
macro/demo_2/epsilon.C	92
macro/demo_2/epsilon_b.C	93
macro/demo_2/epsilon_c.C	94
macro/demo_2/fitr.C	95
macro/demo_2/fourier.C	96
macro/demo_2/info.C	97
macro/demo_2/label.C	98
macro/demo_2/mult.C	99
macro/demo_2/overlay.C	100
macro/demo_2/profile2.C	101
macro/demo_2/profile2_deformation_63Cu.C	102
macro/demo_2/profile2_deformation_Au.C	103
macro/demo_2/profile2_deformation_U.C	104
macro/demo_2/size.C	105
macro/demo_2/wounding_profile.C	105
macro/demo_3/cen_exa_RDS.C	82
macro/demo_3/cent.C	106
macro/demo_3/inel_prof.C	107
macro/demo_3/label.C	98
macro/demo_3/rad_dis.C	107

Chapter 5

Class Documentation

5.1 bin_t Struct Reference

structure for storing observables in subsequent rapidity bins

```
#include <functions.h>
```

Public Attributes

- Float_t [bin](#) [20]

5.1.1 Detailed Description

structure for storing observables in subsequent rapidity bins

5.1.2 Member Data Documentation

5.1.2.1 Float_t bin_t::bin[20]

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

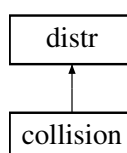
- build/include/[functions.h](#)

5.2 collision Class Reference

collision class

```
#include <collision.h>
```

Inheritance diagram for collision:



Public Member Functions

- [collision](#) (Int_t nnA, Int_t nnB)
constructor
- [collision](#) ()
default constructor
- [collision](#) (const [collision](#) &w1)
copying constructor
- [collision](#) & [operator=](#) (const [collision](#) &w1)
substitution overloading
- void [gen_RDS](#) (const [nucleus](#) &nA, const [nucleus](#) &nB, Float_t d2, Float_t dbin2, Float_t mb)
destructor
- void [shift_cmx_w_c](#) ()
translate to the cm reference frame in the x direction with weights, including the spectator coordinate
- void [shift_cmy_w_c](#) ()
- void [writerds](#) (ofstream &eveout)
output of the distribution of points to an external file

Public Attributes

- Float_t [rd2](#)
- Int_t [wc](#)
- Int_t * [wwA](#)
- Int_t * [wwB](#)
- Int_t * [wwAB](#)
- Int_t [nwA](#)
- Int_t [nwB](#)
- Int_t [nwAB](#)
- Int_t [nzw](#)
- Int_t [nbin](#)
- Int_t [nhotspot](#)
- Int_t [specA](#)
- Int_t [specB](#)
- Float_t [xcmA](#)
- Float_t [xcmB](#)
- Float_t [ycmA](#)
- Float_t [ycmB](#)
- Float_t [rpa](#)

5.2.1 Detailed Description

collision class

Class performing the collision of two nuclei.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 collision::collision (Int_t nnA, Int_t nnB) [inline]

constructor

< maximum number of wounded objects + binary

Parameters

<i>nnA</i>	number of objects in nucleus A
<i>nnB</i>	number of objects in nucleus B

5.2.2.2 collision::collision () [inline]

default constructor

The default constructor assumes 208Pb-208Pb collisions with nucleons.

5.2.2.3 collision::collision (const collision & w1) [inline]

copying constructor

5.2.3 Member Function Documentation

5.2.3.1 void collision::gen_RDS (const nucleus & nA, const nucleus & nB, Float_t d2, Float_t dbin2, Float_t mb) [inline]

destructor

collision between two nuclei

gen_RDS performs the collision of two nuclei, generating the wounded nucleon and the binary collisions, as well

their RDS (relative deposited strength, or weight). It is the core function of the code, implementing the specific model mechanism of the collision.

Parameters

<i>nA</i>	nucleus A, nA>0, nA=1 - proton, nA=2 - deuteron, nA>2 - other nuclei
<i>nB</i>	nucleus B, nB>0, nB=1 - proton, nB=2 - deuteron, nB>2 - other nuclei
<i>d2</i>	wounding distance squared
<i>dbin2</i>	binary-collision distance squared
<i>mb</i>	ratio of the wounding to binary cross sections (for hotspots)

5.2.3.2 collision& collision::operator= (const collision & w1) [inline]

substitution overloading

5.2.3.3 void collision::shift_cmx_w_c () [inline]

translate to the cm reference frame in the x direction with weights, including the spectator coordinates

5.2.3.4 void collision::shift_cmy_w_c () [inline]

5.2.3.5 void collision::writetds (ofstream & eveout) [inline]

output of the distribution of points to an external file

Parameters

<i>eveout</i>	external file for the full event data
---------------	---------------------------------------

5.2.4 Member Data Documentation

5.2.4.1 `Int_t collision::nbin`

number of binary collisions in the event

5.2.4.2 `Int_t collision::nhotspot`

number of hot spots

5.2.4.3 `Int_t collision::nwA`

number of wounded (collided at least once) objects in nucleus A

5.2.4.4 `Int_t collision::nwAB`

total number of wounded objects ($nwA+nwB$) generated in the event

5.2.4.5 `Int_t collision::nwB`

number of wounded (collided at least once) objects in nucleus B

5.2.4.6 `Int_t collision::nzw`

number of sources with non-zero weight

5.2.4.7 `Float_t collision::rd2`

square of the distance between the colliding objects (nucleons or partons)

5.2.4.8 `Float_t collision::rpa`

weight of the source (RDS)

5.2.4.9 `Int_t collision::specA`

number of spectators in A

5.2.4.10 `Int_t collision::specB`

number of spectators in B

5.2.4.11 `Int_t collision::wc`

counter of the sources

5.2.4.12 `Int_t* collision::wwA`

`wwA[i]` is the number of collisions of the *i*-th object from nucleus A with the objects from nucleus B

5.2.4.13 `Int_t* collision::wwAB`5.2.4.14 `Int_t * collision::wwB`

`wwB[i]` is the number of collisions of the *i*-th object from nucleus A with the objects from nucleus A

5.2.4.15 `Float_t collision::xcmA`

x cm cordinate of the spectators from the A nucleus

5.2.4.16 `Float_t collision::xcmB`

x cm cordinate of the spectators from the B nucleus

5.2.4.17 `Float_t collision::ycmA`

y cm cordinate of the spectators from the A nucleus

5.2.4.18 `Float_t collision::ycmB`

y cm cordinate of the spectators from the B nucleus

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

- [build/include/collision.h](#)

5.3 counter Class Reference

simplest counting class

```
#include <counter.h>
```

Public Member Functions

- [counter](#) ()
constructor
- [~counter](#) ()
destructor
- void [reset](#) ()
reset the counter
- void [add](#) (Double_t s)
add entry to the counter
- Int_t [getN](#) ()
get the number of entries
- Double_t [get](#) ()
get the sum of values
- Double_t [mean](#) ()
get the mean value

Private Attributes

- `Int_t count`
number of entries
- `Double_t value`
sum of values counted

5.3.1 Detailed Description

simplest counting class

Class counting the mean.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 `counter::counter( )` `[inline]`

constructor

5.3.2.2 `counter::~~counter( )` `[inline]`

destructor

5.3.3 Member Function Documentation

5.3.3.1 `void counter::add( Double_t s )` `[inline]`

add entry to the counter

Parameters

<code>s</code>	value added
----------------	-------------

5.3.3.2 `Double_t counter::get( )` `[inline]`

get the sum of values

5.3.3.3 `Int_t counter::getN( )` `[inline]`

get the number of entries

5.3.3.4 `Double_t counter::mean( )` `[inline]`

get the mean value

5.3.3.5 `void counter::reset( )` `[inline]`

reset the counter

5.3.4 Member Data Documentation

5.3.4.1 `Int_t counter::count` `[private]`

number of entries

5.3.4.2 `Double_t counter::value` `[private]`

sum of values counted

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

- build/include/[counter.h](#)

5.4 counter2 Class Reference

counting class with variance

```
#include <counter.h>
```

Public Member Functions

- [counter2](#) ()
constructor
- [~counter2](#) ()
destructor
- void [reset](#) ()
reset the counter
- void [add](#) (Double_t s)
add entry
- Int_t [getN](#) ()
get the number of entries
- Double_t [get](#) ()
get the sum of values
- Double_t [get2](#) ()
get the sum of squares of values
- Double_t [mean](#) ()
get the mean value
- Double_t [var](#) ()
get the variance
- Double_t [vara](#) ()
get the variance multiplied with (N-1)/N

Private Attributes

- Int_t [count](#)
number of entries
- Double_t [value](#)
sum of values
- Double_t [value2](#)
sum of squares of values

5.4.1 Detailed Description

counting class with variance

Class counting the mean and variance.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 `counter2::counter2( )` [inline]

constructor

5.4.2.2 `counter2::~~counter2( )` [inline]

destructor

5.4.3 Member Function Documentation

5.4.3.1 `void counter2::add( Double_t s )` [inline]

add entry

Parameters

s	value added
---	-------------

5.4.3.2 `Double_t counter2::get( )` [inline]

get the sum of values

5.4.3.3 `Double_t counter2::get2( )` [inline]

get the sum of squares of values

5.4.3.4 `Int_t counter2::getN( )` [inline]

get the number of entries

5.4.3.5 `Double_t counter2::mean( )` [inline]

get the mean value

5.4.3.6 `void counter2::reset( )` [inline]

reset the counter

5.4.3.7 `Double_t counter2::var( )` [inline]

get the variance

5.4.3.8 Double_t counter2::vara () [inline]

get the variance multiplied with (N-1)/N

5.4.4 Member Data Documentation

5.4.4.1 Int_t counter2::count [private]

number of entries

5.4.4.2 Double_t counter2::value [private]

sum of values

5.4.4.3 Double_t counter2::value2 [private]

sum of squares of values

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

- [build/include/counter.h](#)

5.5 counter_2D Class Reference

2-dimensional counting class

```
#include <counter.h>
```

Public Member Functions

- [counter_2D](#) ()
constructor
- [~counter_2D](#) ()
destructor
- void [reset](#) ()
reset the counter
- void [add](#) (Double_t sx, Double_t sy)
add entry
- Int_t [getN](#) ()
get the number of entries
- Double_t [get_x](#) ()
get the sum of x values
- Double_t [get_y](#) ()
get the sum of y values
- Double_t [get_x2](#) ()
get the sum of x²
- Double_t [get_y2](#) ()
get the sum of y²
- Double_t [get_xy](#) ()
get the sum of xy
- Double_t [mean_x](#) ()

- get the mean x value*
- Double_t `mean_y` ()
get the mean y value
- Double_t `var_x` ()
get the variance of x
- Double_t `var_y` ()
get the variance of y
- Double_t `cov` ()
get the covariance
- Double_t `corr` ()
get the Pearson correlation coefficient

Private Attributes

- Int_t `count`
number of entries
- Double_t `valuex`
sum of x values
- Double_t `valuey`
sum of y values
- Double_t `valuex2`
sum of x^2
- Double_t `valuey2`
sum of y^2
- Double_t `valuexy`
*sum of $x*y$*

5.5.1 Detailed Description

2-dimensional counting class

Class counting the means, variances, and covariance for a 2D sample.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 `counter_2D::counter_2D ( )` [inline]

constructor

5.5.2.2 `counter_2D::~~counter_2D ( )` [inline]

destructor

5.5.3 Member Function Documentation

5.5.3.1 `void counter_2D::add ( Double_t sx, Double_t sy )` [inline]

add entry

Parameters

<code>sx</code>	x value added
<code>sy</code>	y value added

5.5.3.2 `Double_t counter_2D::corr ( ) [inline]`

get the Pearson correlation coefficient

5.5.3.3 `Double_t counter_2D::cov ( ) [inline]`

get the covariance

5.5.3.4 `Double_t counter_2D::get_x ( ) [inline]`

get the sum of x values

5.5.3.5 `Double_t counter_2D::get_x2 ( ) [inline]`

get the sum of x2

5.5.3.6 `Double_t counter_2D::get_xy ( ) [inline]`

get the sum of xy

5.5.3.7 `Double_t counter_2D::get_y ( ) [inline]`

get the sum of y values

5.5.3.8 `Double_t counter_2D::get_y2 ( ) [inline]`

get the sum of y2

5.5.3.9 `Int_t counter_2D::getN ( ) [inline]`

get the number of entries

5.5.3.10 `Double_t counter_2D::mean_x ( ) [inline]`

get the mean x value

5.5.3.11 `Double_t counter_2D::mean_y ( ) [inline]`

get the mean y value

5.5.3.12 `void counter_2D::reset ( ) [inline]`

reset the counter

5.5.3.13 `Double_t counter_2D::var_x( ) [inline]`

get the variance of x

5.5.3.14 `Double_t counter_2D::var_y( ) [inline]`

get the variance of y

5.5.4 Member Data Documentation

5.5.4.1 `Int_t counter_2D::count [private]`

number of entries

5.5.4.2 `Double_t counter_2D::valuex [private]`

sum of x values

5.5.4.3 `Double_t counter_2D::valuex2 [private]`

sum of x^2

5.5.4.4 `Double_t counter_2D::valuey [private]`

sum of $x*y$

5.5.4.5 `Double_t counter_2D::valuey [private]`

sum of y values

5.5.4.6 `Double_t counter_2D::valuey2 [private]`

sum of y^2

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

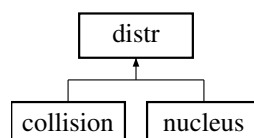
- [build/include/counter.h](#)

5.6 distr Class Reference

Distribution of sources in space.

```
#include <distrib.h>
```

Inheritance diagram for distr:



Public Member Functions

- [distr](#) (Int_t k)
constructor
- [distr](#) (void)
default constructor, 208 sources
- [distr](#) (const [distr](#) &w1)
copying constructor
- Float_t [sum_w](#) ()
destructor
- Float_t [sum_w](#) (Float_t et)
generate the sum of weights at pseudorapidity et with fluctuating strings of average triangles
- Float_t [cmx](#) ()
generate the center-of-mass x coordinate, no weights
- Float_t [cmy](#) ()
generate the center-of-mass y coordinate, no weights
- Float_t [cmz](#) ()
generate the center-of-mass z coordinate, no weights
- Float_t [cmx_w](#) (Float_t et)
generate the center-of-mass x coordinate with weights at pseudorapidity et
- Float_t [cmy_w](#) (Float_t et)
generate the center-of-mass y coordinate with weights at pseudorapidity et
- Float_t [cmx_w](#) ()
generate the center-of-mass x coordinate with weights
- Float_t [cmy_w](#) ()
generate the center-of-mass y coordinate with weights
- Float_t [cmz_w](#) ()
generate the center-of-mass z coordinate with weights
- void [shift_x](#) (Float_t xt)
translate in the x direction
- void [shift_y](#) (Float_t yt)
translate in the y direction
- void [shift_z](#) (Float_t zt)
translate in the z direction
- void [shift_cmx](#) ()
translate to the cm reference frame in the x direction, no weights
- void [shift_cmy](#) ()
translate to the cm reference frame in the y direction, no weights
- void [shift_cmz](#) ()
translate to the cm reference frame in the z direction, no weights
- void [shift_cmx_w](#) (Float_t et)
translate to the cm reference frame in the x direction with weights at rapidity et
- void [shift_cmy_w](#) (Float_t et)
translate to the cm reference frame in the y direction with weights at rapidity et
- void [shift_cmx_w](#) ()
translate to the cm reference frame in the x direction with weights
- void [shift_cmy_w](#) ()
translate to the cm reference frame in the y direction with weights
- void [shift_cmz_w](#) ()
translate to the cm reference frame in the z direction with weights
- Float_t [msx](#) ()

- mean squared x, no weights*
 • Float_t [msy](#) ()
- mean squared y, no weights*
 • Float_t [mxy](#) ()
- mean x*y, no weights*
 • Float_t [msrad](#) ()
- mean squared radius, no weights*
 • Float_t [msrad_t](#) ()
- mean squared transverse radius, no weights*
 • Float_t [msrad_w](#) ()
- mean squared radius with weights*
 • Float_t [msrad_t_w](#) ()
- mean squared transverse radius with weights*
 • Float_t [size](#) ()
- size - the weighted average of the distance from the origin (in the cm frame)*
 • Float_t [phrot](#) (Int_t m)
- rotation angle maximizing the m-th cosine Fourier moment with weight r^2*
 • Float_t [phrot](#) (Int_t m, Float_t k)
- rotation angle maximizing the m-th cosine Fourier moment with weight r^k*
 • void [rotate](#) (Float_t ph)
- rotate in the transverse plane by the specified angle*
 • void [rotate_polar](#) (Float_t costh)
- rotate in the ZX plane by the theta angle*
 • void [uncluster](#) ()
- uncluster*
 • Float_t [tilt](#) ()
- tilt angle of the triangle*
 • Float_t [surf](#) ()
- tilt size of the triangle*
 • Float_t [eps](#) (Int_t m, Int_t imin, Int_t imax)
- m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , limited charge range*
 • Float_t [eps](#) (Int_t m)
- m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , no limit on charge*
 • Float_t [eps_s](#) (Int_t m, Int_t imin, Int_t imax)
- m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , limited charge*
 • Float_t [eps_s](#) (Int_t m)
- m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , no limit on charge*
 • Float_t [eps](#) (Int_t m, Float_t k, Int_t imin, Int_t imax)
- m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , limited charge range*
 • Float_t [mA](#) (Float_t et)
- multiplicity from nucleus A at rapidity et*
 • Float_t [mB](#) (Float_t et)
- multiplicity from nucleus B at rapidity et*
 • Float_t [mAnz](#) (Float_t et)
- number of sources with nonzero weight from nucleus A at rapidity et*
 • Float_t [mBnz](#) (Float_t et)
- number of sources with nonzero weight from nucleus A at rapidity et*
 • Float_t [qx](#) (Int_t m, Float_t k, Float_t d)
- x-component of the Q vector in the transverse plane, weight r^k , no limit on charge, with smearing*
 • Float_t [qx](#) (Int_t m, Float_t k, Float_t d, Float_t et)
- x-component of the Q vector with fluctuating strings*

- `Float_t qy` (`Int_t m`, `Float_t k`, `Float_t d`)
y-component of the Q vector in the transverse plane, weight r^k , no limit on charge, with smearing
- `Float_t qy` (`Int_t m`, `Float_t k`, `Float_t d`, `Float_t et`)
y-component of the Q vector with fluctuating strings
- `Float_t eps` (`Int_t m`, `Float_t k`)
m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , no limit on charge
- `Float_t eps_s` (`Int_t m`, `Float_t k`, `Int_t imin`, `Int_t imax`)
m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , limited charge
- `Float_t eps_s` (`Int_t m`, `Float_t k`)
m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , no limit on charge
- `Float_t eps2p_sq` (`Int_t m`, `Float_t k`)
epsilon square from 2p cumulants
- `void fill_xy` (`TH2D *xyh`, `Float_t fac`, `Int_t imin`, `Int_t imax`)
fill the histogram of the transverse distribution in cartesian coordinates, limited charge
- `void fill_xy` (`TH2D *xyh`, `Float_t fac`)
fill the histogram of the transverse distribution in cartesian coordinates, no limit on charge
- `void fill_polar` (`TH2D *polh`, `Int_t m`, `Float_t fac`, `Int_t imin`, `Int_t imax`)
fill the histogram of the transverse distribution in polar coordinates, limited charge
- `void fill_polar_s` (`TH2D *polh`, `Int_t m`, `Float_t fac`, `Int_t imin`, `Int_t imax`)
fill the histogram of the transverse sine distribution in polar coordinates, limited charge
- `void fill_polar` (`TH2D *polh`, `Int_t m`, `Float_t fac`)
fill the histogram of the transverse distribution in polar coordinates, no limit on charge
- `void fill_polar_s` (`TH2D *polh`, `Int_t m`, `Float_t fac`)
fill the histogram of the transverse sine distribution in polar coordinates, no limit on charge
- `distr & operator=` (`const distr &w1`)
substitution overloading

Public Attributes

- `Int_t n`
number of sources (points)
- `Float_t * x`
x space coordinate
- `Float_t * y`
y space coordinate
- `Float_t * z`
z space coordinate
- `Float_t * ys`
string start-point
- `Float_t * ye`
string end-point
- `Int_t * c`
some integer property, here called "charge"
- `Float_t * w`
weight
- `Float_t xcm`
center-of-mass x coordinate of the distribution
- `Float_t ycm`
center-of-mass y coordinate of the distribution
- `Float_t zcm`

- *center-of-mass z coordinate of the distribution*
- Float_t [msr](#)
mean squared radius of the distribution
- Float_t [msrt](#)
mean squared transverse radius of the distribution
- Float_t [sumw](#)
sum of weights of the distribution

5.6.1 Detailed Description

Distribution of sources in space.

Class for storage and basic operations (translation, rotation) of a distribution of "sources" in space. A source is a point with real weight and some additional integer property, e.g., charge.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 `distr::distr( Int_t k )` `[inline]`

constructor

Parameters

<i>k</i>	number of sources, $k > 0$
----------	----------------------------

5.6.2.2 `distr::distr( void )` `[inline]`

default constructor, 208 sources

208 corresponds to the number on nucleons in the 208Pb nucleus

5.6.2.3 `distr::distr( const distr & w1 )` `[inline]`

copying constructor

5.6.3 Member Function Documentation

5.6.3.1 `Float_t distr::cmx ( )` `[inline]`

generate the center-of-mass x coordinate, no weights

5.6.3.2 `Float_t distr::cmx_w( Float_t et )` `[inline]`

generate the center-of-mass x coordinate with weights at pseudorapidity et

5.6.3.3 `Float_t distr::cmx_w ( )` `[inline]`

generate the center-of-mass x coordinate with weights

5.6.3.4 `Float_t distr::cmy ( )` `[inline]`

generate the center-of-mass y coordinate, no weights

5.6.3.5 `Float_t distr::cmy_w( Float_t et ) [inline]`

generate the center-of-mass y coordinate with weights at pseudorapidity et

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.6 `Float_t distr::cmy_w( ) [inline]`

generate the center-of-mass y coordinate with weights

5.6.3.7 `Float_t distr::cmz( ) [inline]`

generate the center-of-mass z coordinate, no weights

5.6.3.8 `Float_t distr::cmz_w( ) [inline]`

generate the center-of-mass z coordinate with weights

5.6.3.9 `Float_t distr::eps( Int_t m, Int_t imin, Int_t imax ) [inline]`

m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , limited charge range

Limited charge range may be used to get the core and mantle (corona) distributions.

Parameters

<i>m</i>	rank of the Fourier moment, m=0,2,3,4,5,...
<i>imin</i>	lowest charge for the sources included
<i>imax</i>	highest charge for the sources included

5.6.3.10 `Float_t distr::eps( Int_t m ) [inline]`

m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , no limit on charge

Most popular is the second moment, known as eccentricity. The third moment is relevant for the triangular flow.

Parameters

<i>m</i>	rank of the Fourier moment, m=0,2,3,4,5,...
----------	---

5.6.3.11 `Float_t distr::eps( Int_t m, Float_t k, Int_t imin, Int_t imax ) [inline]`

m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , limited charge range

Limited charge range may be used to get the core and mantle (corona) distributions.

Parameters

<i>m</i>	rank of the Fourier moment, m=0,2,3,4,5,...
<i>k</i>	power of transverse radius in the weight

<i>imin</i>	lowest charge for the sources included
<i>imax</i>	highest charge for the sources included

5.6.3.12 `Float_t distr::eps ( Int_t m, Float_t k ) [inline]`

m-th cosine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , no limit on charge

Most popular is the second moment, known as eccentricity. The third moment is relevant for the triangular flow.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight

5.6.3.13 `Float_t distr::eps2p_sq ( Int_t m, Float_t k ) [inline]`

epsilon square from 2p cumulants

epsilon square from 2p cumulants.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight

5.6.3.14 `Float_t distr::eps_s ( Int_t m, Int_t imin, Int_t imax ) [inline]`

m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , limited charge

Limited charge range may be used to get the core and mantle (corona) distributions.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>imin</i>	lowest charge for the sources included
<i>imax</i>	highest charge for the sources included

5.6.3.15 `Float_t distr::eps_s ( Int_t m ) [inline]`

m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^2 , no limit on charge

Parameters

<i>m</i>	rank of the Fourier moment, $m=2,3,4,5,\dots$
----------	---

5.6.3.16 `Float_t distr::eps_s ( Int_t m, Float_t k, Int_t imin, Int_t imax ) [inline]`

m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , limited charge

Limited charge range may be used to get the core and mantle (corona) distributions.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight
<i>imin</i>	lowest charge for the sources included
<i>imax</i>	highest charge for the sources included

5.6.3.17 `Float_t distr::eps_s ( Int_t m, Float_t k )` [inline]

m-th sine Fourier moment in the azimuthal angle in the transverse plane, weight r^k , no limit on charge

Parameters

<i>m</i>	rank of the Fourier moment, $m=2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight

5.6.3.18 `void distr::fill_polar ( TH2D * polh, Int_t m, Float_t fac, Int_t imin, Int_t imax )` [inline]

fill the histogram of the transverse distribution in polar coordinates, limited charge

Limited charge range may be used to get the core and mantle (corona) distributions.

Parameters

<i>polh</i>	2-dim polar histogram in the transverse plane
<i>m</i>	rank of the Fourier moment
<i>fac</i>	normalization factor
<i>imin</i>	lowest charge for the sources included
<i>imax</i>	highest charge for the sources included

5.6.3.19 `void distr::fill_polar ( TH2D * polh, Int_t m, Float_t fac )` [inline]

fill the histogram of the transverse distribution in polar coordinates, no limit on charge

Parameters

<i>polh</i>	2-dim polar histogram in the transverse plane
<i>m</i>	rank of the Fourier moment
<i>fac</i>	normalization factor

5.6.3.20 `void distr::fill_polar_s ( TH2D * polh, Int_t m, Float_t fac, Int_t imin, Int_t imax )` [inline]

fill the histogram of the transverse sine distribution in polar coordinates, limited charge

Limited charge range may be used to get the core and mantle (corona) distributions.

Parameters

<i>polh</i>	2-dim polar sine histogram in the transverse plane
<i>m</i>	rank of the sine Fourier moment
<i>fac</i>	normalization factor
<i>imin</i>	lowest charge for the sources included
<i>imax</i>	highest charge for the sources included

5.6.3.21 `void distr::fill_polar_s ( TH2D * polh, Int_t m, Float_t fac )` [inline]

fill the histogram of the transverse sine distribution in polar coordinates, no limit on charge

Parameters

<i>polh</i>	2-dim polar sine histogram in the transverse plane
<i>m</i>	rank of the sine Fourier moment
<i>fac</i>	normalization factor

5.6.3.22 `void distr::fill_xy ( TH2D * xyh, Float_t fac, Int_t imin, Int_t imax ) [inline]`

fill the histogram of the transverse distribution in cartesian coordinates, limited charge

Limited charge range may be used to get the core and mantle (corona) distributions.

Parameters

<i>xyh</i>	2-dim cartesian histogram in the transverse plane
<i>fac</i>	normalization factor
<i>imin</i>	lowest charge for the sources included
<i>imax</i>	highest charge for the sources included

5.6.3.23 `void distr::fill_xy ( TH2D * xyh, Float_t fac ) [inline]`

fill the histogram of the transverse distribution in cartesian coordinates, no limit on charge

Parameters

<i>xyh</i>	2-dim cartesian histogram in the transverse plane
<i>fac</i>	normalization factor

5.6.3.24 `Float_t distr::mA ( Float_t et ) [inline]`

multiplicity from nucleus A at rapidity et

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.25 `Float_t distr::mAnz ( Float_t et ) [inline]`

number of sources with nonzero weight from nucleus A at rapidity et

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.26 `Float_t distr::mB ( Float_t et ) [inline]`

multiplicity from nucleus B at rapidity et

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.27 `Float_t distr::mBnz ( Float_t et ) [inline]`

number of sources with nonzero weight from nucleus A at rapidity et

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.28 `Float_t distr::msrad ( ) [inline]`

mean squared radius, no weights

5.6.3.29 `Float_t distr::msrad_t ( ) [inline]`

mean squared transverse radius, no weights

5.6.3.30 `Float_t distr::msrad_t_w ( ) [inline]`

mean squared transverse radius with weights

5.6.3.31 `Float_t distr::msrad_w ( ) [inline]`

mean squared radius with weights

5.6.3.32 `Float_t distr::msx ( ) [inline]`

mean squared x, no weights

5.6.3.33 `Float_t distr::msy ( ) [inline]`

mean squared y, no weights

5.6.3.34 `Float_t distr::mxy ( ) [inline]`

mean x*y, no weights

5.6.3.35 `distr & distr::operator= ( const distr & w1 )`

substitution overloading

5.6.3.36 `Float_t distr::phrot ( Int_t m ) [inline]`

rotation angle maximizing the m-th cosine Fourier moment with weight r^2

for m=2 this is the angle for passing to the "participant" frame

Parameters

<i>m</i>	rank of the Fourier moment, m=0,2,3,4,5,...
----------	---

5.6.3.37 `Float_t distr::phrot ( Int_t m, Float_t k ) [inline]`

rotation angle maximizing the m-th cosine Fourier moment with weight r^k

for m=2 this is the angle for passing to the "participant" frame

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight

5.6.3.38 `Float_t distr::qx ( Int_t m, Float_t k, Float_t d )` `[inline]`

x-component of the Q vector in the transverse plane, weight r^k , no limit on charge, with smearing

The smearing formula is exact for $k=2$ and $k=4$, and approximate (expanded for small d) for $k=3$.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight
<i>d</i>	smearing parameter

5.6.3.39 `Float_t distr::qx ( Int_t m, Float_t k, Float_t d, Float_t et )` `[inline]`

x-component of the Q vector with fluctuating strings

The smearing formula is exact for $k=2$ and $k=4$, and approximate (expanded for small d) for $k=3$.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight
<i>d</i>	smearing parameter
<i>et</i>	pseudorapidity

5.6.3.40 `Float_t distr::qy ( Int_t m, Float_t k, Float_t d )` `[inline]`

y-component of the Q vector in the transverse plane, weight r^k , no limit on charge, with smearing

The smearing formula is exact for $k=2$ and $k=4$, and approximate (expanded for small d) for $k=3$.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight
<i>d</i>	smearing parameter

5.6.3.41 `Float_t distr::qy ( Int_t m, Float_t k, Float_t d, Float_t et )` `[inline]`

y-component of the Q vector with fluctuating strings

The smearing formula is exact for $k=2$ and $k=4$, and approximate (expanded for small d) for $k=3$.

Parameters

<i>m</i>	rank of the Fourier moment, $m=0,2,3,4,5,\dots$
<i>k</i>	power of transverse radius in the weight
<i>d</i>	smearing parameter
<i>et</i>	pseudorapidity

5.6.3.42 `void distr::rotate ( Float_t ph ) [inline]`

rotate in the transverse plane by the specified angle

Parameters

<i>ph</i>	rotation angle in the transverse plane
-----------	--

5.6.3.43 `void distr::rotate_polar ( Float_t costh ) [inline]`

rotate in the ZX plane by the theta angle

Parameters

<i>costh</i>	cosine of the rotation angle (theta) in the ZX plane
--------------	--

5.6.3.44 `void distr::shift_cmx ( ) [inline]`

translate to the cm reference frame in the x direction, no weights

5.6.3.45 `void distr::shift_cmx_w ( Float_t et ) [inline]`

translate to the cm reference frame in the x direction with weights at rapidity *et*

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.46 `void distr::shift_cmx_w ( ) [inline]`

translate to the cm reference frame in the x direction with weights

5.6.3.47 `void distr::shift_cmy ( ) [inline]`

translate to the cm reference frame in the y direction, no weights

5.6.3.48 `void distr::shift_cmy_w ( Float_t et ) [inline]`

translate to the cm reference frame in the y direction with weights at rapidity *et*

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.49 `void distr::shift_cmy_w ( ) [inline]`

translate to the cm reference frame in the y direction with weights

5.6.3.50 `void distr::shift_cmz ( ) [inline]`

translate to the cm reference frame in the z direction, no weights

5.6.3.51 `void distr::shift_cmz_w ( ) [inline]`

translate to the cm reference frame in the z direction with weights

5.6.3.52 `void distr::shift_x ( Float_t xt ) [inline]`

translate in the x direction

Parameters

<i>xt</i>	displacement in the x direction
-----------	---------------------------------

5.6.3.53 `void distr::shift_y ( Float_t yt ) [inline]`

translate in the y direction

Parameters

<i>yt</i>	displacement in the y direction
-----------	---------------------------------

5.6.3.54 `void distr::shift_z ( Float_t zt ) [inline]`

translate in the z direction

Parameters

<i>zt</i>	displacement in the z direction
-----------	---------------------------------

5.6.3.55 `Float_t distr::size ( ) [inline]`

size - the weighted average of the distance from the origin (in the cm frame)

5.6.3.56 `Float_t distr::sum_w ( ) [inline]`

destructor

generate sum of weights

5.6.3.57 `Float_t distr::sum_w ( Float_t et ) [inline]`

generate the sum of weights at pseudorapidity et with fluctuating strings of average triangles

Parameters

<i>et</i>	pseudorapidity
-----------	----------------

5.6.3.58 `Float_t distr::surf ( ) [inline]`

tilt size of the triangle

5.6.3.59 `Float_t distr::tilt ( ) [inline]`

tilt angle of the triangle

5.6.3.60 `void distr::uncluster ( ) [inline]`

uncluster

5.6.4 Member Data Documentation

5.6.4.1 `Int_t* distr::c`

some integer property, here called "charge"

Depending on the situation, c is the electric charge of the nucleon in the nucleus, number of collisions of the nucleon, etc.

5.6.4.2 `Float_t distr::msr`

mean squared radius of the distribution

5.6.4.3 `Float_t distr::msrt`

mean squared transverse radius of the distribution

5.6.4.4 `Int_t distr::n`

number of sources (points)

5.6.4.5 `Float_t distr::sumw`

sum of weights of the distribution

5.6.4.6 `Float_t* distr::w`

weight

Depending on the situation, the weight may describe the amount of the deposited energy, entropy, etc., in the source

5.6.4.7 `Float_t* distr::x`

x space coordinate

5.6.4.8 `Float_t distr::xcm`

center-of-mass x coordinate of the distribution

5.6.4.9 `Float_t* distr::y`

y space coordinate

5.6.4.10 `Float_t distr::ycm`

center-of-mass y coordinate of the distribution

5.6.4.11 `Float_t* distr::ye`

string end-point

5.6.4.12 `Float_t* distr::ys`

string start-point

5.6.4.13 `Float_t* distr::z`

z space coordinate

5.6.4.14 `Float_t distr::zcm`

center-of-mass z coordinate of the distribution

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

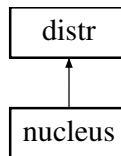
- [build/include/distrib.h](#)

5.7 nucleus Class Reference

nucleus class

```
#include <distrib.h>
```

Inheritance diagram for nucleus:



Public Member Functions

- `nucleus` (`Int_t k`)
constructor
- `nucleus` (`const nucleus &w1`)
copying constructor
- `nucleus & operator=` (`const nucleus &w1`)
substitution overloading
- `void set_parton_A` (`const nucleus &nucA, Int_t np`)
destructor
- `void set_parton_B` (`const nucleus &nucB, Int_t np`)
set randomly the distribution of partons around centres of nucleons inside nucleus B
- `void set_proton` ()
set the distribution for the proton
- `void set_deuteron` ()
set the distribution for the deuteron
- `void set_random_A` ()
set randomly the distribution of nucleons in nucleus A, use the `rlosA()` function, no correlations
- `void set_random_A_hos` ()
set randomly the distribution of nucleons in nucleus A, use the `rlosA_hos()` function, no correlations
- `void set_random_B` ()
set randomly the distribution of nucleons in nucleus B, use the `rlosB()` function, no correlations

- void [set_random_B_hos](#) ()
set randomly the distribution of nucleons in nucleus B, use the [rlosB_hos\(\)](#) function, no correlations
- void [set_random_A_def](#) ()
set randomly the distribution of nucleons in nucleus A with deformation, no correlations
- void [set_random_B_def](#) ()
set randomly the distribution of nucleons in nucleus B with deformation, no correlations
- void [set_random_A_def](#) (Float_t d)
- void [set_random_B_def](#) (Float_t d)
- void [set_tritium](#) (Float_t scale)
- void [set_alpha_cluster](#) (Float_t d, Float_t sigma, Float_t bp, Float_t dp)
- void [set_berillium_7_cluster](#) (Float_t scale, Float_t d, Float_t sigma, Float_t sigmabis)
- void [set_berillium_8_cluster](#) (Float_t scale, Float_t d, Float_t sigma)
- void [set_berillium_9_cluster](#) (Float_t scale, Float_t d, Float_t sigma, Float_t sigmabis)
- void [set_carbon_cluster](#) (Float_t scale, Float_t d, Float_t sigma)
- void [set_oxygen_cluster](#) (Float_t scale, Float_t d, Float_t sigma)
- void [set_oxygen_cluster_square](#) (Float_t scale, Float_t d, Float_t sigma)
- void [set_random_A](#) (Float_t d)
set randomly the distribution of nucleons in the nucleus A with expulsion, use the [rlosA\(\)](#) function
- void [set_random_A_hos](#) (Float_t d)
similar as the above function but for harmonic oscillator shell model, use use the [rlosA_hos\(\)](#) function.
- void [set_random_B](#) (Float_t d)
set randomly the distribution of nucleons in the nucleus B with expulsion, use the [rlosB\(\)](#) function
- void [set_random_B_hos](#) (Float_t d)
similar as above function but for harmonic oscillator shell model, use use the [rlosB_hos\(\)](#) function.
- void [set_file](#) (Float_t *px, Float_t *py, Float_t *pz, Int_t sn, Int_t nu)
set the distribution of nucleons in the nucleus from the tables generated earlier
- void [set_file_uncor](#) (Float_t *px, Float_t *py, Float_t *pz, Int_t sn, Int_t nu)
set the distribution of nucleons in the nucleus from the tables generated earlier, killing correlations
- Float_t [dist2](#) (Int_t j1, Int_t j2)
distance between two nucleons
- bool [good_pair](#) (Int_t j1, Int_t j2, Float_t d)
the pair of nucleons is "good" when the distance between the nucleons is larger than the expulsion distance d
- bool [good_down](#) (Int_t j, Float_t d)
nucleon j is "good" when the distance to all nucleons of index i with $i < j$ is larger than the expulsion distance d
- bool [good_all](#) (Float_t d)
configuration of nucleons in the nucleus is "good" when the distances between all nucleons are larger than the expulsion distance d

Private Attributes

- bool [g](#)
- Float_t [cth](#)
- Float_t [sth](#)
- Float_t [phi](#)
- Float_t [r](#)

Additional Inherited Members

5.7.1 Detailed Description

nucleus class

Class to store distributions of nucleons (or partons) in nuclei.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 `nucleus::nucleus ( Int_t k ) [inline]`

constructor

Parameters

<i>k</i>	number of nucleons (or partons) in the nucleus (the mass number of the nucleus), $k > 0$, $k=1$ - proton, $k=2$ - deuteron, $k > 2$ - other nucleus
----------	--

5.7.2.2 `nucleus::nucleus ( const nucleus & w1 ) [inline]`

copying constructor

5.7.3 Member Function Documentation

5.7.3.1 `Float_t nucleus::dist2 ( Int_t j1, Int_t j2 ) [inline]`

distance between two nucleons

Parameters

<i>j1</i>	index of nucleon 1
<i>j2</i>	index of nucleon 2

5.7.3.2 `bool nucleus::good_all ( Float_t d ) [inline]`

configuration of nucleons in the nucleus is "good" when the distances between all nucleons are larger than the expulsion distance d

Parameters

<i>d</i>	expulsion distance
----------	--------------------

5.7.3.3 `bool nucleus::good_down ( Int_t j, Float_t d ) [inline]`

nucleon j is "good" when the distance to all nucleons of index i with $i < j$ is larger than the expulsion distance d

Parameters

<i>j</i>	index of the nucleon
<i>d</i>	expulsion distance

5.7.3.4 `bool nucleus::good_pair ( Int_t j1, Int_t j2, Float_t d ) [inline]`

the pair of nucleons is "good" when the distance between the nucleons is larger than the expulsion distance d

Parameters

<i>j1</i>	index of nucleon 1
<i>j2</i>	index of nucleon 2

<i>d</i>	expulsion distance - nucleons cannot be closer to each other than d
----------	---

5.7.3.5 **nucleus& nucleus::operator= (const nucleus & w1)** [inline]

substitution overloading

5.7.3.6 **void nucleus::set_alpha_cluster (Float_t d, Float_t sigma, Float_t bp, Float_t dp)** [inline]

set alpha cluster

Parameters

<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
<i>sigma</i>	not used
<i>bp</i>	not used
<i>dp</i>	not used

5.7.3.7 **void nucleus::set_berillium_7_cluster (Float_t scale, Float_t d, Float_t sigma, Float_t sigmabis)** [inline]

set clustered 7Be (dumbbell of alpha and 3He)

Parameters

<i>scale</i>	scale parameter for the size of the nucleus
<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
<i>sigma</i>	standard deviation of x,y,z coordinates of nucleons in the alpha cluster
<i>sigmabis</i>	standard deviation of x,y,z coordinates of nucleons in the 3He cluster

5.7.3.8 **void nucleus::set_berillium_8_cluster (Float_t scale, Float_t d, Float_t sigma)** [inline]

set clustered 8Be (dumbbell)

Parameters

<i>scale</i>	scale parameter for the size of the nucleus
<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
<i>sigma</i>	standard deviation of x,y,z coordinates of nucleons in the cluster

5.7.3.9 **void nucleus::set_berillium_9_cluster (Float_t scale, Float_t d, Float_t sigma, Float_t sigmabis)** [inline]

set clustered 9Be (dumbbell + extra neutron)

Parameters

<i>scale</i>	scale parameter for the size of the nucleus
<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
<i>sigma</i>	standard deviation of x,y,z coordinates of nucleons in the cluster
<i>sigmabis</i>	standard deviation of x,y,z coordinates of nucleon no. 9

5.7.3.10 **void nucleus::set_carbon_cluster (Float_t scale, Float_t d, Float_t sigma)** [inline]

set clustered 12C (triangle)

Parameters

<i>scale</i>	scale parameter for the size of the nucleus
<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
<i>sigma</i>	standard deviation of x,y,z coordinates of nucleons in the cluster

5.7.3.11 void nucleus::set_deuteron () [inline]

set the distribution for the deuteron

Use the Hulthen distribution.

5.7.3.12 void nucleus::set_file (Float_t* px, Float_t* py, Float_t* pz, Int_t sn, Int_t nu) [inline]

set the distribution of nucleons in the nucleus from the tables generated earlier

The nucleon position (x,y,z) is taken from the tables read earlier. The pointers x, y, z are originally positioned at px, py, pz at a place corresponding to the beginning of a randomly selected nucleus. The tables with nuclear distributions have to be prepared externally, see, e.g., <http://www.phys.psu.edu/~malvioli/eventgenerator/> for distributions involving correlations.

Parameters

<i>px</i>	x coordinate of distributions read from files
<i>py</i>	y coordinate of distributions read from files
<i>pz</i>	z coordinate of distributions read from files
<i>sn</i>	number of entries in the file (should be the mass number time the number of stored nuclei)
<i>nu</i>	mass number of the nucleus

5.7.3.13 void nucleus::set_file_uncor (Float_t* px, Float_t* py, Float_t* pz, Int_t sn, Int_t nu) [inline]

set the distribution of nucleons in the nucleus from the tables generated earlier, killing correlations

The nucleon position (x,y,z) is taken from the tables read earlier. The pointers x, y, z are positioned at px, py, pz at a place corresponding to a completely randomly selected nucleon. This procedure kills any correlations and is (probably) equivalent to the mixing technique. The tables with nuclear distributions have to be prepared externally.

Parameters

<i>px</i>	x coordinate of distributions read from files
<i>py</i>	y coordinate of distributions read from files
<i>pz</i>	z coordinate of distributions read from files
<i>sn</i>	number of entries in the file (should be the mass number time the number of stored nuclei)
<i>nu</i>	mass number of the nucleus

5.7.3.14 void nucleus::set_oxygen_cluster (Float_t scale, Float_t d, Float_t sigma) [inline]

set clustered 16O (tetrahedron)

Parameters

<i>scale</i>	scale parameter for the size of the nucleus
<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d

<i>sigma</i>	standard deviations of x,y,z coordinates of nucleons in the cluster
--------------	---

5.7.3.15 `void nucleus::set_oxygen_cluster_square ( Float_t scale, Float_t d, Float_t sigma )` `[inline]`

set clustered 16O (square)

Parameters

<i>scale</i>	scale parameter for the size of the nucleus
<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
<i>sigma</i>	standard deviation of x,y,z coordinates of nucleons in the cluster

5.7.3.16 `void nucleus::set_parton_A ( const nucleus & nucA, Int_t np )` `[inline]`

destructor

set randomly the distribution of np partons around centre of nucleon A

Parameters

<i>nucA</i>	nucleus A
<i>np</i>	number of partons in nucleon

5.7.3.17 `void nucleus::set_parton_B ( const nucleus & nucB, Int_t np )` `[inline]`

set randomly the distribution of partons around centres of nucleons inside nucleus B

Parameters

<i>nucB</i>	nucleus B
<i>np</i>	number of partons in nucleon

5.7.3.18 `void nucleus::set_proton ( )` `[inline]`

set the distribution for the proton

The proton is just placed at the origin

5.7.3.19 `void nucleus::set_random_A ( )` `[inline]`

set randomly the distribution of nucleons in nucleus A, use the [rlosA\(\)](#) function, no correlations

5.7.3.20 `void nucleus::set_random_A ( Float_t d )` `[inline]`

set randomly the distribution of nucleons in the nucleus A with expulsion, use the [rlosA\(\)](#) function

The nucleon positions are subsequently generated according to the spherically-symmetric Woods-Saxon distribution. If the nucleon happens to be generated closer than the expulsion distance d to any of the previously generated nucleons, it is generated anew, until it is "good". Since this leads to some swelling, the original distribution must be a bit narrower to cancel neutralize this effect (see our original paper for a detailed discussion). The expulsion simulates in a simple manner the nuclear repulsion and generates the hard-core two-body correlations.

Parameters

d	expulsion distance, nucleons cannot be closer to each other than d
-----	--

5.7.3.21 `void nucleus::set_random_A_def ( ) [inline]`

set randomly the distribution of nucleons in nucleus A with deformation, no correlations

5.7.3.22 `void nucleus::set_random_A_def ( Float_t  $d$  ) [inline]`

Parameters

d	expulsion distance, nucleons cannot be closer to each other than d
-----	--

5.7.3.23 `void nucleus::set_random_A_hos ( ) [inline]`

set randomly the distribution of nucleons in nucleus A, use the [rlosA_hos\(\)](#) function, no correlations

5.7.3.24 `void nucleus::set_random_A_hos ( Float_t  $d$  ) [inline]`

similar as the above function but for harmonic oscillator shell model, use use the [rlosA_hos\(\)](#) function.

Parameters

d	expulsion distance, nucleons cannot be closer to each other than d
-----	--

5.7.3.25 `void nucleus::set_random_B ( ) [inline]`

set randomly the distribution of nucleons in nucleus B, use the [rlosB\(\)](#) function, no correlations

5.7.3.26 `void nucleus::set_random_B ( Float_t  $d$  ) [inline]`

set randomly the distribution of nucleons in the nucleus B with expulsion, use the [rlosB\(\)](#) function

Same as `set_random_A` for the case of the nucleus B, which in general is different from A

Parameters

d	expulsion distance, nucleons cannot be closer to each other than d
-----	--

5.7.3.27 `void nucleus::set_random_B_def ( ) [inline]`

set randomly the distribution of nucleons in nucleus B with deformation, no correlations

5.7.3.28 `void nucleus::set_random_B_def ( Float_t  $d$  ) [inline]`

Parameters

--	--

<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
----------	--

5.7.3.29 `void nucleus::set_random_B_hos ( ) [inline]`

set randomly the distribution of nucleons in nucleus B, use the [rlosB_hos\(\)](#) function, no correlations

5.7.3.30 `void nucleus::set_random_B_hos ( Float_t d ) [inline]`

similar as above function but for harmonic oscillator shell model, use use the [rlosB_hos\(\)](#) function.

Parameters

<i>d</i>	expulsion distance, nucleons cannot be closer to each other than d
----------	--

5.7.3.31 `void nucleus::set_tritium ( Float_t scale ) [inline]`

set triangular tritium

Parameters

<i>scale</i>	scale of the triangle
--------------	-----------------------

5.7.4 Member Data Documentation

5.7.4.1 `Float_t nucleus::cth [private]`

5.7.4.2 `bool nucleus::g [private]`

5.7.4.3 `Float_t nucleus::phi [private]`

5.7.4.4 `Float_t nucleus::r [private]`

5.7.4.5 `Float_t nucleus::sth [private]`

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

- [build/include/distrib.h](#)

5.8 SOURCE Struct Reference

structure for output of the full event - transverse coordinates, weight, number of the event

```
#include <functions.h>
```

Public Attributes

- `Float_t X`
x coordinate
- `Float_t Y`
y coordinate
- `Float_t W`

- weight*
 • `UInt_t KK`
number of the event

5.8.1 Detailed Description

structure for output of the full event - transverse coordinates, weight, number of the event

5.8.2 Member Data Documentation

5.8.2.1 `UInt_t SOURCE::KK`

number of the event

5.8.2.2 `Float_t SOURCE::W`

weight

5.8.2.3 `Float_t SOURCE::X`

x coordinate

5.8.2.4 `Float_t SOURCE::Y`

y coordinate

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

- `build/include/functions.h`

5.9 `tr_his_c` Class Reference

class storing the trees and histograms

```
#include <functions.h>
```

Public Member Functions

- void `init` ()
initialize the histograms
- void `fill` ()
fill trees param and phys
- void `fill_tr` ()
fill the main tree
- void `fill_res` ()
calculate eccentricity, size, etc. and their fluctuations vs. number of wounded objects or b
- void `write` ()
write out trees param, phys, full_event, and the main tree
- void `write_r` ()
write out the radial density distribution an the pair distance distribution in the nucleus

- void `write_w` ()
write out the overlaid distributions
- void `write_wpro` ()
write out the wounding profile
- void `write_d` ()
write out the histograms with the 2-dim distributions and the radial distributions of the Fourier components of the source profiles
- void `gen` ()
generate histograms of eccentricities and their variance, etc., vs. the number of wounded objects or b

Public Attributes

- TTree * `param`
parameters
- TTree * `phys`
A+B cross section and other physical results.
- TTree * `full_event`
full info on the event (positions and RDS of the sources)
- TTree * `tree`
basic physical results
- TH2D * `xyhist`
cartesian fixed-axes distribution
- TH2D * `xyhist_nuclA`
cartesian fixed-axes distribution of density in nucleus A
- TH2D * `xyhist_nuclB`
cartesian fixed-axes distribution of density in nucleus B
- TH2D * `rcostheta_nuclA`
($r, \cos(\theta)$) distribution of density in nucleus A
- TH2D * `rcostheta_nuclB`
($r, \cos(\theta)$) distribution of density in nucleus B
- TH2D * `xyhist_mantle`
cartesian fixed-axes mantle distribution
- TH2D * `xyhist_core`
cartesian fixed-axes core distribution
- TH2D * `xyhistr`
cartesian variable-axes distribution
- TH1D * `nx`
center-of-mass x coordinate of the source distribution vs. N_w
- TH1D * `nx2`
square of cm x coordinate, then its variance, vs. N_w
- TH1D * `ny`
center-of-mass y coordinate of the source distribution vs. N_w
- TH1D * `ny2`
square of cm y coordinate, then its variance, vs. N_w
- TH1D * `nsiz`
size vs. N_w
- TH1D * `nsiz2`
square of size, then its variance/size², vs. N_w
- TH1D * `neps`
fixed-axes eccentricity vs. N_w
- TH1D * `neps2`

- square of fixed-axes eccentricity, then its variance, vs. N_w*

 - TH1D * [neps4](#)
- fixed-axes fourth moment vs. N_w*

 - TH1D * [nepsp](#)
- variable-axes eccentricity vs. N_w*

 - TH1D * [nepsp2](#)
- square of variable-axes eccentricity, then its variance, vs. N_w*

 - TH1D * [nepsp22](#)
- fourth power of variable-axes eccentricity vs. N_w*

 - TH1D * [nepsp4](#)
- variable-axes fourth moment vs. N_w*

 - TH1D * [nuni](#)
- frequency of N_w , i.e. histogram of unity vs. N_w*

 - TH1D * [nuniRDS](#)
- frequency of RDS, i.e. histogram of unity vs. RDS*

 - TH1D * [nepsb](#)
- fixed-axes eccentricity vs. b*

 - TH1D * [nepsp2b](#)
- square of fixed-axes eccentricity, then its variance, vs. b*

 - TH1D * [nepspb](#)
- variable-axes eccentricity vs. b*

 - TH1D * [nepsp2b](#)
- square of variable-axes eccentricity, then its variance, vs. b*

 - TH1D * [nunib](#)
- frequency of b , i.e. histogram of unity vs. b*

 - TH1D * [nwb](#)
- number of wounded objects in nucleus B vs. total number of wounded nucleons*

 - TH1D * [nw2b](#)
- square of the number of wounded objects in nucleus B , then its variance, vs. total number of wounded nucleons*

 - TH1D * [nwei](#)
- RDS vs. number of wounded objects in nucleus A .*

 - TH1D * [nwei2](#)
- square of RDS, then its variance, vs. number of wounded objects in nucleus A*

 - TH1D * [ntarg](#)
- number of wounded objects in nucleus B vs. number of wounded objects in nucleus A*

 - TH1D * [ntarg2](#)
- square of the number of wounded objects in nucleus B , then its variance, vs. number of wounded objects in nucleus A*

 - TH1D * [nbinar](#)
- number of binary collisions vs. number of wounded objects in nucleus A*

 - TH1D * [nbinar2](#)
- square of the number of binary collisions, then its variance, vs. number of wounded objects in nucleus A*

 - TH1D * [nunp](#)
- frequency of the number of wounded objects in nucleus A*

 - TH1D * [radA](#)
- one-body radial distribution in the nucleus A*

 - TH1D * [radB](#)
- one-body radial distribution in the nucleus B*

 - TH1D * [rrelA](#)
- distance between the pair of objects in the nucleus A*

 - TH1D * [rrelB](#)
- distance between the pair of objects in the nucleus B*

- TH1D * [rrel_u](#)
uncorrelated distance between the pair of objects in the nucleus (one nucleon from A, the other one from B)
- TH1D * [weih](#)
the distribution overlaid on the wounded objects
- TH1D * [weih_bin](#)
the distribution overlaid over binary collisions
- TH1D * [wpro](#)
the wounding profile
- TH2D * [radA2](#)
- TH3D * [radA3](#)

5.9.1 Detailed Description

class storing the trees and histograms

Class for storage of ROOT structures (trees, histograms) used for later off-line analysis within ROOT or other codes

5.9.2 Member Function Documentation

5.9.2.1 void tr_his_c::fill () [inline]

fill trees param and phys

5.9.2.2 void tr_his_c::fill_res () [inline]

calculate eccentricity, size, etc. and their fluctuations vs. number of wounded objects or b

5.9.2.3 void tr_his_c::fill_tr () [inline]

fill the main tree

5.9.2.4 void tr_his_c::gen () [inline]

generate histograms of eccentricities and their variance, etc., vs. the number of wounded objects or b

5.9.2.5 void tr_his_c::init () [inline]

initialize the histograms

< tree storing parameters

< tree storing some physical quantities

< tree storing full info on the events

< tree storing basic information on events

5.9.2.6 void tr_his_c::write () [inline]

write out trees param, phys, full_event, and the main tree

5.9.2.7 void tr_his_c::write_d () [inline]

write out the histograms with the 2-dim distributions and the radial distributions of the Fourier components of the source profiles

5.9.2.8 void tr_his_c::write_r () [inline]

write out the radial density distribution and the pair distance distribution in the nucleus

5.9.2.9 void tr_his_c::write_w () [inline]

write out the overlaid distributions

5.9.2.10 void tr_his_c::write_wpro () [inline]

write out the wounding profile

5.9.3 Member Data Documentation**5.9.3.1 TTree * tr_his_c::full_event**

full info on the event (positions and RDS of the sources)

5.9.3.2 TH1D * tr_his_c::nbinar

number of binary collisions vs. number of wounded objects in nucleus A

5.9.3.3 TH1D * tr_his_c::nbinar2

square of the number of binary collisions, then its variance, vs. number of wounded objects in nucleus A

5.9.3.4 TH1D * tr_his_c::neps

fixed-axes eccentricity vs. Nw

5.9.3.5 TH1D * tr_his_c::neps2

square of fixed-axes eccentricity, then its variance, vs. Nw

5.9.3.6 TH1D * tr_his_c::neps2b

square of fixed-axes eccentricity, then its variance, vs. b

5.9.3.7 TH1D * tr_his_c::neps4

fixed-axes fourth moment vs. Nw

5.9.3.8 TH1D * tr_his_c::nepsb

fixed-axes eccentricity vs. b

5.9.3.9 TH1D * tr_his_c::nepsp

variable-axes eccentricity vs. Nw

5.9.3.10 TH1D * tr_his_c::nepsp2

square of variable-axes eccentricity, then its variance, vs. Nw

5.9.3.11 TH1D * tr_his_c::nepsp22

fourth power of variable-axes eccentricity vs. Nw

5.9.3.12 TH1D * tr_his_c::nepsp2b

square of variable-axes eccentricity, then its variance, vs. b

5.9.3.13 TH1D * tr_his_c::nepsp4

variable-axes fourth moment vs. Nw

5.9.3.14 TH1D * tr_his_c::nepspb

variable-axes eccentricity vs. b

5.9.3.15 TH1D * tr_his_c::nsize

size vs. Nw

5.9.3.16 TH1D * tr_his_c::nsize2

square of size, then its variance/size², vs. Nw

5.9.3.17 TH1D * tr_his_c::ntarg

number of wounded objects in nucle B vs. number of wounded objects in nucleus A

5.9.3.18 TH1D * tr_his_c::ntarg2

square of the number of wounded objects in nucle B, then its variance, vs. number of wounded objects in nucleus A

5.9.3.19 TH1D * tr_his_c::nuni

frequency of Nw, i.e. histogram of unity vs. Nw

5.9.3.20 TH1D * tr_his_c::nunib

frequency of b, i.e. histogram of unity vs. b

5.9.3.21 TH1D * tr_his_c::nuniRDS

frequency of RDS, i.e. histogram of unity vs. RDS

5.9.3.22 TH1D * tr_his_c::nunp

frequency of the number of wounded objects in nucleus A

5.9.3.23 TH1D * tr_his_c::nw2b

square of the number of wounded objects in nucleus B, then its variance, vs. total number of wounded nucleons

5.9.3.24 TH1D * tr_his_c::nwb

number of wounded objects in nucleus B vs. total number of wounded nucleons

5.9.3.25 TH1D* tr_his_c::nwei

RDS vs. number of wounded objects in nucleus A.

5.9.3.26 TH1D * tr_his_c::nwei2

square of RDS, then its variance, vs. number of wounded objects in nucleus A

5.9.3.27 TH1D* tr_his_c::nx

center-of-mass x coordinate of the source distribution vs. Nw

5.9.3.28 TH1D * tr_his_c::nx2

square of cm x coordinate, then its variance, vs. Nw

5.9.3.29 TH1D * tr_his_c::ny

center-of-mass y coordinate of the source distribution vs. Nw

5.9.3.30 TH1D * tr_his_c::ny2

square of cm y coordinate, then its variance, vs. Nw

5.9.3.31 TTree* tr_his_c::param

parameters

5.9.3.32 TTree * tr_his_c::phys

A+B cross section and other physical results.

5.9.3.33 TH1D* tr_his_c::radA

one-body radial distribution in the nucleus A

5.9.3.34 TH2D* tr_his_c::radA2**5.9.3.35 TH3D* tr_his_c::radA3****5.9.3.36 TH1D * tr_his_c::radB**

one-body radial distribution in the nucleus B

5.9.3.37 TH2D * tr_his_c::rcostheta_nuclA

($r, \cos(\theta)$) distribution of density in nucleus A

5.9.3.38 TH2D * tr_his_c::rcostheta_nuclB

($r, \cos(\theta)$) distribution of density in nucleus B

5.9.3.39 TH1D * tr_his_c::rrel_u

uncorrelated distance between the pair of objects in the nucleus (one nucleon from A, the other one from B)

5.9.3.40 TH1D * tr_his_c::rrelA

distance between the pair of objects in the nucleus A

5.9.3.41 TH1D * tr_his_c::rrelB

distance between the pair of objects in the nucleus B

5.9.3.42 TTree * tr_his_c::tree

basic physical results

5.9.3.43 TH1D * tr_his_c::weih

the distribution overlaid on the wounded objects

5.9.3.44 TH1D * tr_his_c::weih_bin

the distribution overlaid over binary collisions

5.9.3.45 TH1D * tr_his_c::wpro

the wounding profile

5.9.3.46 TH2D* tr_his_c::xyhist

cartesian fixed-axes distribution

5.9.3.47 TH2D * tr_his_c::xyhist_core

cartesian fixed-axes core distribution

5.9.3.48 TH2D * tr_his_c::xyhist_mantle

cartesian fixed-axes mantle distribution

5.9.3.49 TH2D * tr_his_c::xyhist_nuclA

cartesian fixed-axes distribution of density in nucleus A

5.9.3.50 TH2D * tr_his_c::xyhist_nuclB

cartesian fixed-axes distribution of density in nucleus B

5.9.3.51 TH2D * tr_his_c::xyhistr

cartesian variable-axes distribution

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

- build/include/[functions.h](#)

Chapter 6

File Documentation

6.1 build/include/collision.h File Reference

```
#include "distrib.h"  
#include <TMath.h>
```

Classes

- class [collision](#)
collision class

6.1.1 Detailed Description

Part of GLISSANDO 3

6.2 build/include/counter.h File Reference

Classes

- class [counter](#)
simplest counting class
- class [counter2](#)
counting class with variance
- class [counter_2D](#)
2-dimensional counting class

6.2.1 Detailed Description

Part of GLISSANDO 3

6.3 build/include/distrib.h File Reference

```
#include <TH1D.h>  
#include <TH3D.h>  
#include "counter.h"
```

Classes

- class [distr](#)
Distribution of sources in space.
- class [nucleus](#)
nucleus class

6.3.1 Detailed Description

Part of GLISSANDO 3

6.4 build/include/functions.h File Reference

```
#include <math.h>
#include <time.h>
#include <string.h>
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <TH1D.h>
#include <TH2D.h>
#include <TFile.h>
#include <TTree.h>
#include <TRandom3.h>
#include "counter.h"
```

Classes

- struct [SOURCE](#)
structure for output of the full event - transverse coordinates, weight, number of the event
- struct [bin_t](#)
structure for storing observables in subsequent rapidity bins
- class [tr_his_c](#)
class storing the trees and histograms

Functions

- Double_t [fsNN](#) (Double_t ecm)
NN inelastic cross section.
- Double_t [fsQQ](#) (Double_t ecm)
parton-parton inelastic cross section
- Double_t [fqscale](#) (Double_t ecm)
parton separation parameter
- void [readpar](#) (TString inpfiler)
process the input file containing parameters
- void [echopar](#) ()
echo parameters to the output

- void `reset_counters` ()
reset the counters used to store physical quantities in the event
- void `helper` (Int_t argc, char *str)
print the version or brief help
- void `header` ()
print the header in the console output
- void `epilog` ()
print epilog to the output
- Int_t `time_start` ()
start the time measurement and print the stamp
- void `time_stop` (Int_t ts)
stop the time measurement and print the stamp
- Float_t `los` ()
random number generator using the built-in ROOT generator, uniform on (0,1)
- Float_t `rlosA` ()
random number generator for the Woods-Saxon (or Fermi) distribution - nucleus A
- Float_t `rlosB` ()
random number generator for the Woods-Saxon (or Fermi) distribution - nucleus B
- Float_t `rlosA_def` (Float_t *cth_pointerA, Float_t beta2, Float_t beta4)
random number generator for the deformed Woods-Saxon or Fermi distribution
- Float_t `rlosB_def` (Float_t *cth_pointerB, Float_t beta2, Float_t beta4)
random number generator for the deformed Woods-Saxon or Fermi distribution
- Float_t `rlos_hole` (Float_t s)
random number generator for distribution with a hole in the middle
- Float_t `rlos_alpha` (Float_t s, Float_t bp, Float_t dp)
random number generator for distribution in the alpha nucleus - not used
- Float_t `rlos_abf` (Float_t sc, Float_t scp)
random number generator for distribution of nucleons in the alpha cluster
- Float_t `rlosA_hos` ()
random number generator for the harmonic oscillator shell model density - nucleus A
- Float_t `rlosB_hos` ()
random number generator for the harmonic oscillator shell model density - nucleus B
- Float_t `rlos_hult` ()
random number generator for the Hulthen distribution
- Double_t `gamgen` (Float_t a)
random number generator for the Gamma distribution
- Int_t `negbin` (Double_t m, Double_t v)
random number generator for the negative binomial distribution
- Float_t `rlos_parton` ()
random radial coordinate of a parton in the nucleon
- Float_t `fg` (Float_t rr)
rapidity distribution with the plateau and half-Gaussians
- Float_t `fpm` (Float_t rr)
function f +/- from Bozek, arXiv:1002.4999v2 [nucl-th]
- Float_t `los_rap_A` ()
random number generator for the rapidity distribution - wounded nucleons from nucleus A
- Float_t `los_rap_B` ()
random number generator for the rapidity distribution - wounded nucleons from nucleus B
- Float_t `los_rap_bin` ()
random number generator for the rapidity distribution - binary collisions
- Float_t `dist` (Int_t m, Float_t u, Float_t v)
statistical distribution overlaid on RDS
- Float_t `disp` (Float_t w)
random shift of the source location

Variables

- Int_t **PARTONS**
- Int_t **EVENTS** =1000
number of generated events
- Int_t **NBIN** =40
number of bins for histogramming in x, y, and r
- Int_t **FBIN** =72
number of bins for histogramming in the azimuthal angle
- Int_t **NUMA** =208
mass number of nucleus A
- Int_t **NUMB** =208
mass number of nucleus B
- Int_t **NCS** =3
number of partons in the nucleon
- Int_t **WMIN** =2
minimum number of wounded nucleons to record the event
- Int_t **MODEL** =0
switch for the superimposed multiplicity distribution: 0 - scale, 1 - Poisson, 2 - Gamma, 3 - Negative Binomial
- Int_t **DOBIN** =0
1 - count binary collisions even in the pure wounded-nucleon model. 0 - do not
- Int_t **W0** =2
minimum allowed number of wounded objects in the acceptance window
- Int_t **W1** =10000
maximum allowed number of wounded objects in the acceptance window
- Int_t **SHIFT** =1
1 - shift the coordinates of the fireball to c.m., 0 - do not
- Int_t **RET** =0
0 - fix-last algorithm (preferred), 1 - return-to-beginning algorithm for the generation of the nuclear distribution
- Int_t **FULL** =0
1 - generate the full event tree (large output file), 0 - do not
- Int_t **FILES** =0
1 - read distribution from files, 0 - do not
- Int_t **NNWP** =0
0 - hard-sphere wounding profile, 1 - Gaussian wounding profile, 2 - gamma wounding profile
- Int_t **NUMRAP** =10
number of particles per unit weight generated in the whole rapidity range
- Int_t **ARANK** =2
rank of the Fourier moment for the forward-backward analysis
- Int_t **PP** =-1
power of the transverse radius in the Fourier moments
- Int_t **RO** =0
rank of the rotation axes (0 - rotation rank = rank of the Fourier moment)
- UInt_t **ISEED**
read seed for the ROOT random number generator, if 0 - random seed generated
- UInt_t **ISEED1**
copy of ISEED
- Float_t **BMIN** =0.
minimum value of the impact parameter in the acceptance window
- Float_t **BMAX** =25.
maximum value of the impact parameter in the acceptance window

- Float_t **RDS0** =0.
minimum value of the relative deposited strength (RDS) in the acceptance window
- Float_t **RDS1** =100000
maximum value of the relative deposited strength (RDS) in the acceptance window
- Float_t **RWSA** =6.407
Woods-Saxon radius for nucleus A.
- Float_t **AWSA** =0.459
Woods-Saxon width for nucleus A.
- Float_t **BETA2A** =0.0
Deformation parameter beta2 of deformed Woods-Saxon distribution for nucleus A.
- Float_t **BETA4A** =0.0
Deformation parameter beta4 of deformed Woods-Saxon distribution for nucleus A.
- Float_t **ROTA_THETA** =-1.0
Parameter controlling the rotation of nucleus A in XZ plane (polar angle THETA, -1 means random rotation)
- Float_t **ROTA_PHI** =-1.0
Parameter controlling the rotation of nucleus A in XY plane (azimuthal angle PHI, -1 means random rotation)
- Float_t **RWSB** =6.407
Woods-Saxon radius for nucleus B.
- Float_t **AWSB** =0.459
Woods-Saxon width for nucleus B.
- Float_t **BETA2B** =0.0
Deformation parameter beta2 of deformed Woods-Saxon distribution for nucleus B.
- Float_t **BETA4B** =0.0
Deformation parameter beta4 of deformed Woods-Saxon distribution for nucleus B.
- Float_t **ROTB_THETA** =-1.0
Parameter controlling the rotation of nucleus B in XZ plane (polar angle THETA, -1 means random rotation)
- Float_t **ROTB_PHI** =-1.0
Parameter controlling the rotation of nucleus B in XY plane (azimuthal angle PHI, -1 means random rotation)
- Float_t **BTOT** = fmax(**RWSA**,**RWSB**)+**AWSA**+**AWSB**
maximum coordinate value for some histograms
- Float_t **WFA** =0.
the w parameter for the Fermi distribution for nucleus A
- Float_t **WFB** =0.
the w parameter for the Fermi distribution for nucleus B
- Float_t **SNN** =73.5
NN "wounding" cross section in milibarns.
- Float_t **SBIN** =73.5
NN binary cross section in milibarns.
- Float_t **ALPHA** =0.15
the mixing parameter: 0 - wounded, 1 - binary, 0.145 - mixed (PHOBOS)
- Float_t **Uw** =2.
Poisson or Gamma parameters for superimposed distribution, wounded nucleons.
- Float_t **Ubin** =2.
Poisson or Gamma parameters for superimposed distribution, binary collisions.
- Float_t **Vw** =4.
Negative binomial variance, wounded nucleons.
- Float_t **Vbin** =4.
Negative binomial variance, binary collisions.
- Float_t **PI** =4.*atan(1.)
the number pi
- Float_t **CD** =0.9

- closest allowed distance (expulsion distance) between nucleons in the nucleus in fm (simulation of repulsion)*

 - Float_t **DW** =0.
- dispersion of the location of the source for wounded nucleons (in fm)*

 - Float_t **DBIN** =0.
- dispersion of the location of the source for binary collisions (in fm)*

 - Float_t **GA** =0.92
- Gaussian wounding profile parameter (height at the origin)*

 - Float_t **RAPRANGE** =5.
- range in rapidity*

 - Float_t **ETA0** =2.5
- 2*ETA0 is the width of the plateau in eta*

 - Float_t **ETAM** =8.58
- parameter of the Bialas-Czyz-Bozek model*

 - Float_t **SIGETA** =1.4
- parameter controlling the width of the rapidity distribution*

 - Float_t **MAXYRAP** =10.
- maximum absolute value of the y coordinate in the x-y-rapidity histogram*

 - Float_t **FBRAP** =2.5
- forward rapidity for the forward-backward analysis (backward rapidity = - FBRAP)*

 - Float_t **RCHA** =5.66
- harmonic oscillator shell model density mean squared charge radii of 12C-nucleus*

 - Float_t **RCHB** =5.66
- harmonic oscillator shell model density mean squared charge radii of 12C-nucleus*

 - Float_t **RCHP** =0.7714
- harmonic oscillator shell model density mean squared charge radii of proton*

 - Float_t **OMEGA** =0.4
- relative variance of cross-section fluctuations*

 - Float_t **GAMA** =1.0
- gamma wounding profile parameter (height at the origin)*

 - Float_t **SCALEA** =3.7
- scale parameter for the size of the nucleus (cluster version)*

 - Float_t **SIGMAA** =1.05
- standard deviation of x,y,z coordinates of nucleons in the alpha cluster*

 - Float_t **SIGMABISA** =1.3
- standard deviation of x,y,z coordinates of nucleons in the 3He cluster or nucleon no. 9*

 - Float_t **SCALEB** =3.7
- scale parameter for the size of the nucleus (cluster version)*

 - Float_t **SIGMAB** =1.05
- standard deviation of x,y,z coordinates of nucleons in the alpha cluster*

 - Float_t **SIGMABISB** =1.3
- standard deviation of x,y,z coordinates of nucleons in the 3He cluster or nucleon no. 9*

 - Float_t **QSCALE** =0.286825
- scale parameter in the parton distribution function, $f(r)=r^2 \cdot \exp(-r/QSCALE)$, $QSCALE=1/(4.27/\text{Sqrt}(3/2))$*

 - Float_t **DS** =0.4
- source smearing parameter*

 - Float_t **ETACC** =2.4
- multiplicity acceptance in rapidity: (-ETACC,ETACC)*

 - Float_t **RAPSHIFT** =0.
- pseudorapidity shift (in p-Pb collisions)*

 - Float_t **YB** =8.58
- rapidity of the beam*

- `Float_t ECM` = -1.
center of mass energy of the colliding system [GeV]
- `counter2 epart1`
counter for epsilon participant (variable-axes), $\langle r^3 \cos(\phi) \rangle / \langle r^3 \rangle$
- `counter2 epart2`
counter for epsilon participant (variable-axes), $\langle r^n \cos(2 \phi) \rangle / \langle r^n \rangle$
- `counter2 epart3`
counter for variable-axes $\langle r^n \cos(3 \phi) \rangle / \langle r^n \rangle$
- `counter2 epart4`
counter for variable-axes $\langle r^n \cos(4 \phi) \rangle / \langle r^n \rangle$
- `counter2 epart5`
counter for variable-axes $\langle r^n \cos(5 \phi) \rangle / \langle r^n \rangle$
- `counter2 epart6`
counter for variable-axes $\langle r^n \cos(6 \phi) \rangle / \langle r^n \rangle$
- `counter2 nwounded`
counter for number of wounded objects
- `counter2 nbinary`
counter for number of binary collisions
- `counter2 nhot`
counter for number of hot-spots
- `counter2 nweight`
counter for relative deposited strength (RDS)
- `counter_2D angles`
counter for forward-backward reaction-plane angle correlations
- `Int_t evall`
number of all attempted event
- `Int_t roo`
Fourier rank for the rotation axis.
- `Int_t ppp`
power of the weight in eccentricity definition
- `Int_t kk`
number of the current event
- `Float_t d`
the wounding distance
- `Float_t dbin`
the binary-collision distance
- `Float_t b`
impact parameter
- `Float_t sitot`
the total A+B cross section in the acceptance window
- `Float_t sirad`
equivalent hard-sphere radius for the cross section
- `Float_t rwA`
number of wounded objects in A
- `Float_t rwB`
number of wounded objects in B
- `Float_t rwAB`
number of all wounded objects
- `Float_t rbin`
number of binary collisions
- `Float_t rhotspot`

- number of hot-spots*
- Float_t [rpa](#)
relative deposited strength (RDS)
- Float_t [sizeav](#)
size
- Float_t [es](#)
epsilon standard (fixed-axes), $\langle r^2 \cos(2 \phi) \rangle / \langle r^2 \rangle$
- Float_t [ess](#)
fixed-axes sine moment, $\langle r^2 \sin(2 \phi) \rangle / \langle r^2 \rangle$
- Float_t [ep1](#)
epsilon participant (variable-axes), $\langle r^3 \cos(\phi) \rangle / \langle r^3 \rangle$
- Float_t [ep1s](#)
variable-axes sine moment, $\langle r^3 \sin(\phi) \rangle / \langle r^3 \rangle$
- Float_t [ep](#)
epsilon participant (variable-axes), $\langle r^n \cos(2 \phi) \rangle / \langle r^n \rangle$
- Float_t [eps](#)
variable-axes sine moment, $\langle r^n \sin(2 \phi) \rangle / \langle r^n \rangle$
- Float_t [es3](#)
fixed-axes $\langle r^n \cos(3 \phi) \rangle / \langle r^n \rangle$
- Float_t [es3s](#)
fixed-axes sine moment, $\langle r^n \sin(3 \phi) \rangle / \langle r^n \rangle$
- Float_t [ep3](#)
variable-axes $\langle r^2 \cos(3 \phi) \rangle / \langle r^2 \rangle$
- Float_t [ep3s](#)
variable-axes sine moment, $\langle r^n \sin(3 \phi) \rangle / \langle r^n \rangle$
- Float_t [es4](#)
fixed-axes $\langle r^n \cos(4 \phi) \rangle / \langle r^n \rangle$
- Float_t [es4s](#)
fixed-axes sine moment, $\langle r^n \sin(4 \phi) \rangle / \langle r^n \rangle$
- Float_t [ep4](#)
variable-axes $\langle r^n \cos(4 \phi) \rangle / \langle r^n \rangle$
- Float_t [ep4s](#)
variable-axes sine moment, $\langle r^n \sin(4 \phi) \rangle / \langle r^n \rangle$
- Float_t [es5](#)
fixed-axes $\langle r^n \cos(5 \phi) \rangle / \langle r^n \rangle$
- Float_t [es5s](#)
fixed-axes sine moment, $\langle r^n \sin(5 \phi) \rangle / \langle r^n \rangle$
- Float_t [ep5](#)
variable-axes $\langle r^n \cos(5 \phi) \rangle / \langle r^n \rangle$
- Float_t [ep5s](#)
variable-axes sine moment, $\langle r^n \sin(5 \phi) \rangle / \langle r^n \rangle$
- Float_t [es6](#)
fixed-axes $\langle r^n \cos(6 \phi) \rangle / \langle r^n \rangle$
- Float_t [es6s](#)
fixed-axes sine moment, $\langle r^n \sin(6 \phi) \rangle / \langle r^n \rangle$
- Float_t [ep6](#)
variable-axes $\langle r^n \cos(6 \phi) \rangle / \langle r^n \rangle$
- Float_t [ep6s](#)
variable-axes sine moment, $\langle r^n \sin(6 \phi) \rangle / \langle r^n \rangle$
- Float_t [epA2](#)
- Float_t [epA3](#)

- Float_t [epA4](#)
- Float_t [tiltA](#)
tilt angle of the triangle
- Float_t [surfA](#)
surface of the triangle projected on the transverse plane
- Float_t [ep22](#)
- Float_t [ep32](#)
- Float_t [ep42](#)
- Float_t [phirot](#)
rotation angle maximizing the second Fourier moment
- Float_t [phirot_plus](#)
rotation angle maximizing the second Fourier moment - increased rapidity
- Float_t [phirot_minus](#)
rotation angle maximizing the second Fourier moment - decreased rapidity
- Float_t [phirot3](#)
rotation angle maximizing the third Fourier moment
- Float_t [phirot4](#)
rotation angle maximizing the fourth Fourier moment
- Float_t [phirot5](#)
rotation angle maximizing the fifth Fourier moment
- Float_t [phirot6](#)
rotation angle maximizing the sixth Fourier moment
- Float_t [xx](#)
center-of-mass x coordinate
- Float_t [yy](#)
center-of-mass y coordinate
- Float_t [xepp](#)
average ep
- Float_t [xsepp](#)
standard deviation of ep
- Float_t [wfqA](#)
number of wounded nucleons evaluated from the number of wounded partons in nucleus A
- Float_t [wfqB](#)
number of wounded nucleons evaluated from the number of wounded partons in nucleus B
- Float_t [wfq](#)
total number of wounded nucleons evaluated from the number of wounded partons

6.4.1 Detailed Description

Part of GLISSANDO 3

6.4.2 Function Documentation

6.4.2.1 Float_t disp (Float_t w)

random shift of the source location

The location of the source may be shifted randomly when $DW > 0$ or $DBIN > 0$, with the Gaussian distribution of the width w .

Parameters

<i>w</i>	average shift of the source location, Gaussian distribution
----------	---

6.4.2.2 `Float_t dist ( Int_t m, Float_t u, Float_t v )`

statistical distribution overlaid on RDS

Parameters

<i>m</i>	case: 0 - none, 1 - Poisson, 2 - Gamma distribution, 3 - Negative binomial distribution
<i>u</i>	average of the distribution in cases 1, 2 and 3
<i>v</i>	variance of the distribution in case 3, $v > u$

6.4.2.3 `void echopar ( )`

echo parameters to the output

6.4.2.4 `void epilog ( )`

print epilog to the output

6.4.2.5 `Float_t fg ( Float_t rr )`

rapidity distribution with the plateau and half-Gaussians

Parameters

<i>rr</i>	spatial rapidity
-----------	------------------

6.4.2.6 `Float_t fpm ( Float_t rr )`

function f +/- from Bozek, arXiv:1002.4999v2 [nucl-th]

Parameters

<i>rr</i>	spatial rapidity
-----------	------------------

6.4.2.7 `Double_t fqscale ( Double_t ecm )`

parton separation parameter

parton-parton separation that (together with fsQQ) reproduces the NN inelastic cross section and the NN wounding profile

Parameters

<i>ecm</i>	$\sqrt{s_{NN}}$
------------	-----------------

6.4.2.8 `Double_t fsNN ( Double_t ecm )`

NN inelastic cross section.

NN inelastic cross section from our parametrization of the COMPETE model

Parameters

<i>ecm</i>	sqrt(s_NN)
------------	------------

6.4.2.9 Double_t fsQQ (Double_t *ecm*)

parton-parton inelastic cross section

parton-parton inelastic cross section that (together with fqscale) reproduces the NN inelastic cross section and the NN wounding profile

Parameters

<i>ecm</i>	sqrt(s_NN)
------------	------------

6.4.2.10 Double_t gamgen (Float_t *a*)

random number generator for the Gamma distribution

Parameters

<i>a</i>	the parameter a in $f(x) = x^{a-1} \exp(-x)/\Gamma(a)$
----------	--

6.4.2.11 void header ()

print the header in the console output

6.4.2.12 void helper (Int_t *argc*, char * *str*)

print the version or brief help

Parameters

<i>argc</i>	number of command line parameters
<i>str</i>	string parameter (-v for version, -h for brief help)

6.4.2.13 Float_t los ()

random number generator using the built-in ROOT generator, uniform on (0,1)

6.4.2.14 Float_t los_rap_A ()

random number generator for the rapidity distribution - wounded nucleons from nucleus A

Formula for the wounded quark rapidity profile from Bozek, arXiv:1002.4999v2 [nucl-th].

6.4.2.15 Float_t los_rap_B ()

random number generator for the rapidity distribution - wounded nucleons from nucleus B

Formula for the wounded quark rapidity profile from Bozek, arXiv:1002.4999v2 [nucl-th].

6.4.2.16 Float_t los_rap_bin ()

random number generator for the rapidity distribution - binary collisions

Formula for the wounded quark rapidity profile from Bozek, arXiv:1002.4999v2 [nucl-th].

6.4.2.17 Int_t negbin (Double_t *m*, Double_t *v*)

random number generator for the negative binomial distribution

6.4.2.18 void readpar (TString *infile*)

process the input file containing parameters

scan the input file for the parameters reset from the default values

reset number of constituents to 1 in the wounded nucleon case

correct wrong input

see the paper for the discussion of parametrizations of the nuclear distributions

set the range for the histograms

Parameters

<i>infile</i>	name of the input file
---------------	------------------------

6.4.2.19 void reset_counters ()

reset the counters used to store physical quantities in the event

6.4.2.20 Float_t rlos_abf (Float_t *sc*, Float_t *scp*)

random number generator for distribution of nucleons in the alpha cluster

Parameters

<i>sc</i>	scale parameter
<i>scp</i>	scale parameter

6.4.2.21 Float_t rlos_alpha (Float_t *s*, Float_t *bp*, Float_t *dp*)

random number generator for distribution in the alpha nucleus - not used

6.4.2.22 Float_t rlos_hole (Float_t *s*)

random number generator for distribution with a hole in the middle

Parameters

<i>s</i>	size scale parameter
----------	----------------------

6.4.2.23 Float_t rlos_hult ()

random number generator for the Hulthen distribution

The Hulthen distribution used to generate the distance between nucleons in the deuteron

6.4.2.24 Float_t rlos_parton ()

random radial coordinate of a parton in the nucleon

gamma distribution $f(r)=r^2 \cdot \exp(-r/wid)$

6.4.2.25 Float_t rlosA ()

random number generator for the Woods-Saxon (or Fermi) distribution - nucleus A

6.4.2.26 Float_t rlosA_def (Float_t * cth_pointerA, Float_t beta2, Float_t beta4)

random number generator for the deformed Woods-Saxon or Fermi distribution

random number generator for the Woods-Saxon or Fermi distribution (deformed with beta2, beta4 parameters of the spherical harmonics Y20, Y40) - nucleus A

Parameters

<i>cth_pointerA</i>	cos(theta)
<i>beta2</i>	beta_2 nuclear deformation parameter
<i>beta4</i>	beta_4 nuclear deformation parameter

6.4.2.27 Float_t rlosA_hos ()

random number generator for the harmonic oscillator shell model density - nucleus A

The harmonic oscillator shell distribution used to generate the distance between nucleons in light ($2 < \text{NUMA} < 17$) nuclei (Nuclear Sizes. L. R. B. Elton, Oxford University Press, New York, 1961.)

6.4.2.28 Float_t rlosB ()

random number generator for the Woods-Saxon (or Fermi) distribution - nucleus B

6.4.2.29 Float_t rlosB_def (Float_t * cth_pointerB, Float_t beta2, Float_t beta4)

random number generator for the deformed Woods-Saxon or Fermi distribution

random number generator for the Woods-Saxon or Fermi distribution (deformed with beta2, beta4 parameters of the spherical harmonics Y20, Y40) - nucleus B

Parameters

<i>cth_pointerB</i>	cos(theta)
<i>beta2</i>	beta_2 nuclear deformation parameter
<i>beta4</i>	beta_4 nuclear deformation parameter

6.4.2.30 Float_t rlosB_hos ()

random number generator for the harmonic oscillator shell model density - nucleus B

The harmonic oscillator shell distribution used to generate the distance between nucleons in light ($2 < \text{NUMB} < 17$) nuclei (Nuclear Sizes. L. R. B. Elton, Oxford University Press, New York, 1961.)

6.4.2.31 Int_t time_start ()

start the time measurement and print the stamp

6.4.2.32 void time_stop (Int_t ts)

stop the time measurement and print the stamp

Parameters

<i>ts</i>	time at start
-----------	---------------

6.4.3 Variable Documentation**6.4.3.1 Float_t ALPHA =0.15**

the mixing parameter: 0 - wounded, 1 - binary, 0.145 - mixed (PHOBOS)

6.4.3.2 counter_2D angles

counter for forward-backward reaction-plane angle correlations

6.4.3.3 Int_t ARANK =2

rank of the Fourier moment for the forward-backward analysis

6.4.3.4 Float_t AWSA =0.459

Woods-Saxon width for nucleus A.

6.4.3.5 Float_t AWSB =0.459

Woods-Saxon width for nucleus B.

6.4.3.6 Float_t b

impact parameter

6.4.3.7 Float_t BETA2A =0.0

Deformation parameter beta2 of deformed Woods-Saxon distribution for nucleus A.

6.4.3.8 Float_t BETA2B =0.0

Deformation parameter beta2 of deformed Woods-Saxon distribution for nucleus B.

6.4.3.9 Float_t BETA4A =0.0

Deformation parameter beta4 of deformed Woods-Saxon distribution for nucleus A.

6.4.3.10 Float_t BETA4B =0.0

Deformation parameter beta4 of deformed Woods-Saxon distribution for nucleus B.

6.4.3.11 Float_t BMAX =25.

maximum value of the impact parameter in the acceptance window

6.4.3.12 Float_t BMIN =0.

minimum value of the impact parameter in the acceptance window

6.4.3.13 Float_t BTOT = fmax(RWSA,RWSB)+AWSA+AWSB

maximum coordinate value for some histograms

6.4.3.14 Float_t CD =0.9

closest allowed distance (expulsion distance) between nucleons in the nucleus in fm (simulation of repulsion)

6.4.3.15 Float_t d

the wounding distance

6.4.3.16 Float_t DBIN =0.

dispersion of the location of the source for binary collisions (in fm)

6.4.3.17 Float_t dbin

the binary-collision distance

6.4.3.18 Int_t DOBIN =0

1 - count binary collisions even in the pure wounded-nucleon model. 0 - do not

6.4.3.19 Float_t DS =0.4

source smearing parameter

6.4.3.20 Float_t DW =0.

dispersion of the location of the source for wounded nucleons (in fm)

6.4.3.21 Float_t ECM =-1.

center of mass energy of the colliding system [GeV]

6.4.3.22 Float_t ep

epsilon participant (variable-axes), $\langle r^n \cos(2 \phi) \rangle / \langle r^n \rangle$

6.4.3.23 Float_t ep1

epsilon participant (variable-axes), $\langle r^3 \cos(\phi) \rangle / \langle r^3 \rangle$

6.4.3.24 Float_t ep1s

variable-axes sine moment, $\langle r^3 \sin(\phi) \rangle / \langle r^3 \rangle$

6.4.3.25 Float_t ep22**6.4.3.26 Float_t ep3**

variable-axes $\langle r^2 \cos(3 \phi) \rangle / \langle r^2 \rangle$

6.4.3.27 Float_t ep32**6.4.3.28 Float_t ep3s**

variable-axes sine moment, $\langle r^n \sin(3 \phi) \rangle / \langle r^n \rangle$

6.4.3.29 Float_t ep4

variable-axes $\langle r^n \cos(4 \phi) \rangle / \langle r^n \rangle$

6.4.3.30 Float_t ep42**6.4.3.31 Float_t ep4s**

variable-axes sine moment, $\langle r^n \sin(4 \phi) \rangle / \langle r^n \rangle$

6.4.3.32 Float_t ep5

variable-axes $\langle r^n \cos(5 \phi) \rangle / \langle r^n \rangle$

6.4.3.33 Float_t ep5s

variable-axes sine moment, $\langle r^n \sin(5 \phi) \rangle / \langle r^n \rangle$

6.4.3.34 Float_t ep6

variable-axes $\langle r^n \cos(6 \phi) \rangle / \langle r^n \rangle$

6.4.3.35 Float_t ep6s

variable-axes sine moment, $\langle r^n \sin(6 \phi) \rangle / \langle r^n \rangle$

6.4.3.36 Float_t epA2**6.4.3.37 Float_t epA3****6.4.3.38 Float_t epA4****6.4.3.39 counter2 epart**

counter for epsilon participant (variable-axes), $\langle r^n \cos(2 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.40 counter2 epart1

counter for epsilon participant (variable-axes), $\langle r^3 \cos(\text{phi}) \rangle / \langle r^3 \rangle$

6.4.3.41 counter2 epart3

counter for variable-axes $\langle r^n \cos(3 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.42 counter2 epart4

counter for variable-axes $\langle r^n \cos(4 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.43 counter2 epart5

counter for variable-axes $\langle r^n \cos(5 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.44 counter2 epart6

counter for variable-axes $\langle r^n \cos(6 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.45 Float_t eps

variable-axes sine moment, $\langle r^n \sin(2 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.46 Float_t es

epsilon standard (fixed-axes), $\langle r^2 \cos(2 \text{ phi}) \rangle / \langle r^2 \rangle$

6.4.3.47 Float_t es3

fixed-axes $\langle r^n \cos(3 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.48 Float_t es3s

fixed-axes sine moment, $\langle r^n \sin(3 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.49 Float_t es4

fixed-axes $\langle r^n \cos(4 \text{ phi}) \rangle / \langle r^n \rangle$

6.4.3.50 Float_t es4s

fixed-axes sine moment, $\langle r^n \sin(4 \phi) \rangle / \langle r^n \rangle$

6.4.3.51 Float_t es5

fixed-axes $\langle r^n \cos(5 \phi) \rangle / \langle r^n \rangle$

6.4.3.52 Float_t es5s

fixed-axes sine moment, $\langle r^n \sin(5 \phi) \rangle / \langle r^n \rangle$

6.4.3.53 Float_t es6

fixed-axes $\langle r^n \cos(6 \phi) \rangle / \langle r^n \rangle$

6.4.3.54 Float_t es6s

fixed-axes sine moment, $\langle r^n \sin(6 \phi) \rangle / \langle r^n \rangle$

6.4.3.55 Float_t ess

fixed-axes sine moment, $\langle r^2 \sin(2 \phi) \rangle / \langle r^2 \rangle$

6.4.3.56 Float_t ETA0 =2.5

$2 \cdot \text{ETA0}$ is the width of the plateau in eta

6.4.3.57 Float_t ETACC =2.4

multiplicity acceptance in rapidity: $(-\text{ETACC}, \text{ETACC})$

6.4.3.58 Float_t ETAM =8.58

parameter of the Bialas-Czyz-Bozek model

6.4.3.59 Int_t evall

number of all attempted event

6.4.3.60 Int_t EVENTS =1000

number of generated events

6.4.3.61 Int_t FBIN =72

number of bins for histogramming in the azimuthal angle

6.4.3.62 Float_t FBRAP =2.5

forward rapidity for the forward-backward analysis (backward rapidity = - FBRAP)

6.4.3.63 Int_t FILES =0

1 - read distribution from files, 0 - do not

6.4.3.64 Int_t FULL =0

1 - generate the full event tree (large output file), 0 - do not

6.4.3.65 Float_t GA =0.92

Gaussian wounding profile parameter (height at the origin)

6.4.3.66 Float_t GAMA =1.0

gamma wounding profile parameter (height at the origin)

6.4.3.67 UInt_t ISEED

read seed for the ROOT random number generator, if 0 - random seed generated

6.4.3.68 UInt_t ISEED1

copy of ISEED

6.4.3.69 Int_t kk

number of the current event

6.4.3.70 Float_t MAXYRAP =10.

maximum absolute value of the y coordinate in the x-y-rapidity histogram

6.4.3.71 Int_t MODEL =0

switch for the superimposed multiplicity distribution: 0 - scale, 1 - Poisson, 2 - Gamma, 3 - Negative Binomial

6.4.3.72 Int_t NBIN =40

number of bins for histogramming in x, y, and r

6.4.3.73 counter2 nbinary

counter for number of binary collisions

6.4.3.74 Int_t NCS =3

number of partons in the nucleon

6.4.3.75 counter2 nhot

counter for number of hot-spots

6.4.3.76 Int_t NNWP =0

0 - hard-sphere wounding profile, 1 - Gaussian wounding profile, 2 - gamma wounding profile

6.4.3.77 Int_t NUMA =208

mass number of nucleus A

6.4.3.78 Int_t NUMB =208

mass number of nucleus B

6.4.3.79 Int_t NUMRAP =10

number of particles per unit weight generated in the whole rapidity range

6.4.3.80 counter2 nweight

counter for relative deposited strength (RDS)

6.4.3.81 counter2 nwounded

counter for number of wounded objects

6.4.3.82 Float_t OMEGA =0.4

relative variance of cross-section fluctuations

6.4.3.83 Int_t PARTONS**6.4.3.84 Float_t phirot**

rotation angle maximizing the second Fourier moment

6.4.3.85 Float_t phirot3

rotation angle maximizing the third Fourier moment

6.4.3.86 Float_t phirot4

rotation angle maximizing the fourth Fourier moment

6.4.3.87 Float_t phiro5

rotation angle maximizing the fifth Fourier moment

6.4.3.88 Float_t phiro6

rotation angle maximizing the sixth Fourier moment

6.4.3.89 Float_t phiro_minus

rotation angle maximizing the second Fourier moment - decreased rapidity

6.4.3.90 Float_t phiro_plus

rotation angle maximizing the second Fourier moment - increased rapidity

6.4.3.91 Float_t PI=4.*atan(1.)

the number pi

6.4.3.92 Int_t PP=-1

power of the transverse radius in the Fourier moments

6.4.3.93 Int_t ppp

power of the weight in eccentricity definition

6.4.3.94 Float_t QSCALE=0.286825

scale parameter in the parton distribution function, $f(r)=r^2*\exp(-r/QSCALE)$, $QSCALE=1/(4.27/\text{Sqrt}(3/2))$

6.4.3.95 Float_t RAPRANGE=5.

range in rapidity

6.4.3.96 Float_t RAPSHIFT=0.

pseudorapidity shift (in p-Pb collisions)

6.4.3.97 Float_t rbin

number of binary collisions

6.4.3.98 Float_t RCHA=5.66

harmonic oscillator shell model density mean squared charge radii of 12C-nucleus

6.4.3.99 Float_t RCHB =5.66

harmonic oscillator shell model density mean squared charge radii of 12C-nucleus

6.4.3.100 Float_t RCHP =0.7714

harmonic oscillator shell model density mean squared charge radii of proton

6.4.3.101 Float_t RDS0 =0.

minimum value of the relative deposited strength (RDS) in the acceptance window

6.4.3.102 Float_t RDS1 =100000

maximum value of the relative deposited strength (RDS) in the acceptance window

6.4.3.103 Int_t RET =0

0 - fix-last algorithm (preferred), 1 - return-to-beginning algorithm for the generation of the nuclear distribution

6.4.3.104 Float_t rhotspot

number of hot-spots

6.4.3.105 Int_t RO =0

rank of the rotation axes (0 - rotation rank = rank of the Fourier moment)

6.4.3.106 Int_t roo

Fourier rank for the rotation axis.

6.4.3.107 Float_t ROTA_PHI =-1.0

Parameter controlling the rotation of nucleus A in XY plane (azimuthal angle PHI, -1 means random rotation)

6.4.3.108 Float_t ROTA_THETA =-1.0

Parameter controlling the rotation of nucleus A in XZ plane (polar angle THETA, -1 means random rotation)

6.4.3.109 Float_t ROTB_PHI =-1.0

Parameter controlling the rotation of nucleus B in XY plane (azimuthal angle PHI, -1 means random rotation)

6.4.3.110 Float_t ROTB_THETA =-1.0

Parameter controlling the rotation of nucleus B in XZ plane (polar angle THETA, -1 means random rotation)

6.4.3.111 Float_t rpa

relative deposited strength (RDS)

6.4.3.112 Float_t rwA

number of wounded objects in A

6.4.3.113 Float_t rwAB

number of all wounded objects

6.4.3.114 Float_t rwB

number of wounded objects in B

6.4.3.115 Float_t RWSA =6.407

Woods-Saxon radius for nucleus A.

6.4.3.116 Float_t RWSB =6.407

Woods-Saxon radius for nucleus B.

6.4.3.117 Float_t SBIN =73.5

NN binary cross section in milibarns.

6.4.3.118 Float_t SCALEA =3.7

scale parameter for the size of the nucleus (cluster version)

6.4.3.119 Float_t SCALEB =3.7

scale parameter for the size of the nucleus (cluster version)

6.4.3.120 Int_t SHIFT =1

1 - shift the coordinates of the fireball to c.m., 0 - do not

6.4.3.121 Float_t SIGETA =1.4

parameter controlling the width of the rapidity distribution

6.4.3.122 Float_t SIGMAA =1.05

standard deviation of x,y,z coordinates of nucleons in the alpha cluster

6.4.3.123 Float_t SIGMAB =1.05

standard deviation of x,y,z coordinates of nucleons in the alpha cluster

6.4.3.124 Float_t SIGMABISA =1.3

standard deviation of x,y,z coordinates of nucleons in the 3He cluster or nucleon no. 9

6.4.3.125 Float_t SIGMABISB =1.3

standard deviation of x,y,z coordinates of nucleons in the 3He cluster or nucleon no. 9

6.4.3.126 Float_t sirad

equivalent hard-sphere radius for the cross section

6.4.3.127 Float_t sitot

the total A+B cross section in the acceptance window

6.4.3.128 Float_t sizeav

size

6.4.3.129 Float_t SNN =73.5

NN "wounding" cross section in milibarns.

6.4.3.130 Float_t surfA

surface of the triangle projected on the transverse plane

6.4.3.131 Float_t tiltA

tilt angle of the triangle

6.4.3.132 Float_t Ubin =2.

Poisson or Gamma parameters for superimposed distribution, binary collisions.

6.4.3.133 Float_t Uw =2.

Poisson or Gamma parameters for superimposed distribution, wounded nucleons.

6.4.3.134 Float_t Vbin =4.

Negative binomial variance, binary collisions.

6.4.3.135 `Float_t Vw =4.`

Negative binomial variance, wounded nucleons.

6.4.3.136 `Int_t W0 =2`

minimum allowed number of wounded objects in the acceptance window

6.4.3.137 `Int_t W1 =10000`

maximum allowed number of wounded objects in the acceptance window

6.4.3.138 `Float_t WFA =0.`

the w parameter for the Fermi distribution for nucleus A

6.4.3.139 `Float_t WFB =0.`

the w parameter for the Fermi distribution for nucleus B

6.4.3.140 `Float_t wfq`

total number of wounded nucleons evaluated from the number of wounded partons

6.4.3.141 `Float_t wfqA`

number of wounded nucleons evaluated from the number of wounded partons in nucleus A

6.4.3.142 `Float_t wfqB`

number of wounded nucleons evaluated from the number of wounded partons in nucleus B

6.4.3.143 `Int_t WMIN =2`

minimum number of wounded nucleons to record the event

6.4.3.144 `Float_t xep`

average ep

6.4.3.145 `Float_t xsepp`

standard deviation of ep

6.4.3.146 `Float_t xx`

center-of-mass x coordinate

6.4.3.147 Float_t YB =8.58

rapidity of the beam

6.4.3.148 Float_t yy

center-of-mass y coordinate

6.5 build/src/glissando3.cxx File Reference

```
#include <math.h>
#include <time.h>
#include <string.h>
#include <iostream>
#include <iomanip>
#include <fstream>
#include <TH1D.h>
#include <TH2D.h>
#include <TH3D.h>
#include <TFile.h>
#include <TTree.h>
#include <TRandom3.h>
#include "counter.h"
#include "functions.h"
#include "distrib.h"
#include "collision.h"
```

Macros

- #define `_VER_` 3.102

Functions

- `Int_t main` (`Int_t argc`, `char *argv[]`)
the main function of GLISSANDO 3

Variables

- `Float_t ver = _VER_`
current version of the code
- `TRandom3 raa`
ROOT random number generator.

6.5.1 Detailed Description

The main file of GLISSANDO 3

6.5.2 Macro Definition Documentation

6.5.2.1 `#define _VER_ 3.102`

6.5.3 Function Documentation

6.5.3.1 `Int_t main ( Int_t argc, char * argv[] )`

the main function of GLISSANDO 3

The main function of GLISSANDO 3 contains the basic structure of the Glauber Monte Carlo simulation, i.e., declarations and definitions of basic objects of the nucleus and collision classes, the main loop over events, evaluation of basic quantities, etc. It is meant to be tailored by the user to meet his needs. For speed of the execution, some switches of the code are controlled by the preprocessor variables (*nnwp*, *files*, *profile*, *weight*, etc.). (the units for all dimensionful quantities in GLISSANDO are powers of fm)

start time

print basic info

print header

set the input file

process input parameters

set the ROOT output file

seed the ROOT random-number generator

reset counters used for some basic physical quantities

declare and initialize trees and histograms for storage of data

set the minimum wounding and binary-collision distances (hard-sphere profile) or the Gaussian wounding parameters (Gaussian profile)

echo basic parameters to the console

`#if(files)` then initialize the nucleon distributions from external tables

declare nuclei A and B

declare the collision

— start the main loop over events

generate the distributions of nucleons in nuclei A and B

shift nuclei to the center-of-mass frame

rotate randomly in the azimuth

rotate the nuclei by theta (zx plane) and phi (xy plane) angles The proper order of rotations is important

generate the impact parameter b with the distribution proportional to b^2 in the range (BMAX, BMIN)

shift the coordinates of the nucleons in nucleus A such that the center of mass is at the point $(b \cdot \text{NUMB} / (\text{NUMA} + \text{NUMB}), 0)$

shift the coordinates of the nucleons in nucleus B such that the center of mass is at the point $(-b \cdot \text{NUMA} / (\text{NUMA} + \text{NUMB}), 0)$

collide the nuclei, create the sources (wounded objects, binary collisions) and RDS

`#if(weight)` generate the histograms for the NN collision profiles

generate various 2-dim histograms with the distributions of sources

output of the flow vector at centers of the CMS bins

generate the variable-axes Fourier moments (up to 6-th moment)

get some basic properties of the event
 fill the data in trees and histograms
 if(FULL) write the full event info to the file (added for comparability reasons, takes a lot of space)
 — end of main loop over events
 output some final results to the console
 the total cross section and the equivalent hard-sphere radius
 project out the marginal distribution in the radial variable (generate the radial Fourier profiles)
 generate and write some histograms with physical quantities
 #if(*weight*) normalize to the wounding and the binary cross sections and write
 closing ROOT file
 write exit info
 stop time and print stamp

Parameters

<i>argc</i>	number of command line parameters
<i>argv</i>	used for passing input and output file names first argument: <input.dat> second argument: <output.root> third argument: <nucl_A.dat> (present only when <i>files</i> =1) fourth argument: <nucl_B.dat> (esent pronly when <i>files</i> =1) command line parameters (file names)

6.5.4 Variable Documentation

6.5.4.1 TRandom3 raa

ROOT random number generator.

6.5.4.2 Float_t ver = _VER_

current version of the code

6.6 macro/angles.C File Reference

```
#include "label.C"
```

Functions

- void [angles](#) (char *p)
produces plots of the correlation between principal axes in the forward and backward rapidities

6.6.1 Detailed Description

Script generating the plot of the correlation between principal axes in the forward and backward rapidities (part of GLISSANDO 2)

6.6.2 Function Documentation

6.6.2.1 void angles (char * p)

produces plots of the correlation between principal axes in the forward and backward rapidities

Produces plots of the correlation between principal axes in the forward and backward rapidities.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.7 macro/cen_exa.C File Reference

```
#include "label.C"
```

Functions

- void [cen_exa](#) (char *p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

6.7.1 Function Documentation

6.7.1.1 void cen_exa (char * p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

Produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality. The centrality is determined from the number of wounded nucleons.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.8 macro/cen_exa_RDS.C File Reference

```
#include "label.C"
```

Functions

- void [cen_exa_RDS](#) (char *p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

6.8.1 Function Documentation

6.8.1.1 void cen_exa_RDS (char * p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

Produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality. The centrality is determined from the number of wounded nucleons.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.9 macro/demo_3/cen_exa_RDS.C File Reference

```
#include "label.C"
```

Functions

- void [cen_exa_RDS](#) (char *p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

6.9.1 Function Documentation

6.9.1.1 void [cen_exa_RDS](#) (char * p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

Produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality. The centrality is determined from the number of wounded nucleons.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.10 macro/cen_exa_RDS_moj.C File Reference

Functions

- void [cen_exa_RDS_moj](#) (char *p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

6.10.1 Function Documentation

6.10.1.1 void [cen_exa_RDS_moj](#) (char * p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

Produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality. The centrality is determined from the number of wounded nucleons.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.11 macro/cen_ratio.C File Reference

```
#include "label.C"
```

Functions

- void [cen_ratio](#) (char *p)
generates the centrality classes

6.11.1 Function Documentation

6.11.1.1 void cen_ratio (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strenth (RDS), and the impact parameter. Plots of distributions divided into classes are produced. The script makes sense for the minimum-bias simulations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.12 macro/cenPb.C File Reference

```
#include "label.C"
```

Functions

- void [cenPb](#) (char *p)
generates the centrality classes

6.12.1 Detailed Description

Script generating the centrality classes (alternative to centrality.C) (part of GLISSANDO 2)

6.12.2 Function Documentation

6.12.2.1 void cenPb (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strenth (RDS), and the impact parameter. Plots of distributions divided into classes are produced. The script makes sense for the minimum-bias simulations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.13 macro/centrality.C File Reference

```
#include "label.C"
#include <iostream>
#include <fstream>
```

Functions

- void [centrality](#) (char *p)
generates the centrality classes

6.13.1 Function Documentation

6.13.1.1 void centrality (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strength (RDS), and the impact parameter. Plots of centrality vs. these variables are generated. The script makes sense for the minimum-bias simulations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.14 macro/demo_2/centrality.C File Reference

```
#include "label.C"
#include <iostream>
#include <fstream>
```

Functions

- void [centrality](#) (char *p)
generates the centrality classes

6.14.1 Function Documentation

6.14.1.1 void centrality (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strength (RDS), and the impact parameter. Plots of centrality vs. these variables are generated. The script makes sense for the minimum-bias simulations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.15 macro/centrality2.C File Reference

```
#include "label.C"
```

Functions

- void [centrality2](#) (char *p)
generates the centrality classes

6.15.1 Function Documentation

6.15.1.1 void centrality2 (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strenth (RDS), and the impact parameter. Plots of distributions divided into classes are produced. The script makes sense for the minimum-bias simulations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.16 macro/demo_2/centrality2.C File Reference

```
#include "label.C"
```

Functions

- void [centrality2](#) (char *p)
generates the centrality classes

6.16.1 Function Documentation

6.16.1.1 void centrality2 (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strenth (RDS), and the impact parameter. Plots of distributions divided into classes are produced. The script makes sense for the minimum-bias simulations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.17 macro/centrality2l.C File Reference

```
#include "label.C"
```

Functions

- void [centrality2l](#) (char *p)
generates the centrality classes

6.17.1 Function Documentation

6.17.1.1 void centrality2l (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strength (RDS), and the impact parameter. Plots of distributions divided into classes are produced. The script makes sense for the minimum-bias simulations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.18 macro/centrality3.C File Reference

```
#include "label.C"
```

Functions

- void [centrality3](#) (char *p)
generates the centrality classes

6.18.1 Detailed Description

Script generating the centrality classes (alternative to centrality.C) (part of GLISSANDO 2)

6.18.2 Function Documentation

6.18.2.1 void centrality3 (char * p)

generates the centrality classes

Centrality classes are generated in the total number of wounded nucleons, relative deposited strength (RDS), and the impact parameter. Plots of distributions divided into classes are produced. The script makes sense for the minimum-bias simulations.

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.19 macro/col_chain_8_q.C File Reference

Functions

- Double_t [f2](#) (Int_t x)
- Double_t [f3](#) (Int_t x)
- Float_t [krot](#) (Float_t alfa, Float_t beta, Float_t a, Float_t b, Float_t nbin, Float_t nwA)
- Float_t [los](#) ()
- void [col_chain_8_q](#) (char *p, int n)

Variables

- TRandom3 [raa](#)

6.19.1 Function Documentation

6.19.1.1 void [col_chain_8_q](#) (char * *p*, int *n*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.19.1.2 Double_t [f2](#) (Int_t *x*)

6.19.1.3 Double_t [f3](#) (Int_t *x*)

6.19.1.4 Float_t [krot](#) (Float_t *alfa*, Float_t *beta*, Float_t *a*, Float_t *b*, Float_t *nbin*, Float_t *nwA*)

6.19.1.5 Float_t [los](#) ()

6.19.2 Variable Documentation

6.19.2.1 TRandom3 [raa](#)

6.20 macro/core_mantle.C File Reference

```
#include "label.C"
```

Functions

- void [core_mantle](#) (char *p)
generates the core and corona distributions

6.20.1 Function Documentation

6.20.1.1 `void core_mantle ( char * p )`

generates the core and corona distributions

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.21 macro/demo_2/core_mantle.C File Reference

```
#include "label.C"
```

Functions

- void `core_mantle` (char *p)
generates the core and corona distributions

6.21.1 Function Documentation

6.21.1.1 void `core_mantle` (char * p)

generates the core and corona distributions

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.22 macro/corr.C File Reference

```
#include "label.C"
```

Functions

- void `corr` (char *p)
generates the plot of the radial NN correlation function, $C(r)$, for nucleus A

6.22.1 Function Documentation

6.22.1.1 void `corr` (char * p)

generates the plot of the radial NN correlation function, $C(r)$, for nucleus A

$C(r)$ is obtained via division of the correlated and uncorrelated distributions of the relative distance in the nucleon pairs. Requires *profile*=1.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.23 macro/demo_2/corr.C File Reference

```
#include "label.C"
```

Functions

- void `corr` (char *p)

generates the plot of the radial NN correlation function, $C(r)$, for nucleus A

6.23.1 Function Documentation

6.23.1.1 void `corr` (char * p)

generates the plot of the radial NN correlation function, $C(r)$, for nucleus A

$C(r)$ is obtained via division of the correlated and uncorrelated distributions of the relative distance in the nucleon pairs. Requires *profile=1*.

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.24 macro/demo_2/density.C File Reference

```
#include "label.C"
```

Functions

- void `density` (char *p)

produces plots of the fixed-axes and variable-axes densities

6.24.1 Function Documentation

6.24.1.1 void `density` (char * p)

produces plots of the fixed-axes and variable-axes densities

Produces plots of the fixed-axes and variable-axes (participant) densities

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.25 macro/density.C File Reference

```
#include "label.C"
```

Functions

- void `density` (char *p)
produces plots of the fixed-axes and variable-axes densities

6.25.1 Function Documentation

6.25.1.1 void density (char * p)

produces plots of the fixed-axes and variable-axes densities

Produces plots of the fixed-axes and variable-axes (participant) densities

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.26 macro/demo_2/dxdy.C File Reference

```
#include "label.C"
```

Functions

- void `dxdy` (char *p)
produces the plot of the dispersion of the center-of-mass x and y coordinates as a function of the number of wounded nucleons.

6.26.1 Function Documentation

6.26.1.1 void dxdy (char * p)

produces the plot of the dispersion of the center-of-mass x and y coordinates as a function of the number of wounded nucleons.

Produces the plot of the standard deviation of the x and y center-of-mass coordinates as a function of the number of wounded nucleons.

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.27 macro/dxdy.C File Reference

```
#include "label.C"
```

Functions

- void `dxdy` (char *p)
produces the plot of the dispersion of the center-of-mass x and y coordinates as a function of the number of wounded nucleons.

6.27.1 Function Documentation

6.27.1.1 void dxdy (char * p)

produces the plot of the dispersion of the center-of-mass x and y coordinates as a function of the number of wounded nucleons.

Produces the plot of the standard deviation of the x and y center-of-mass coordinates as a function of the number of wounded nucleons.

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.28 macro/demo_2/epsilon.C File Reference

```
#include "label.C"
```

Functions

- void [epsilon](#) (char *p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the number of wounded nucleons

6.28.1 Function Documentation

6.28.1.1 void epsilon (char * p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the number of wounded nucleons

Produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the number of wounded nucleons

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.29 macro/epsilon.C File Reference

```
#include "label.C"
```

Functions

- void [epsilon](#) (char *p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the number of wounded nucleons

6.29.1 Function Documentation

6.29.1.1 void epsilon (char * p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the number of wounded nucleons

Produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the number of wounded nucleons

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.30 macro/demo_2/epsilon_b.C File Reference

```
#include "label.C"
```

Functions

- void [epsilon_b](#) (char *p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the impact parameter b

6.30.1 Function Documentation

6.30.1.1 void epsilon_b (char * p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the impact parameter b

Produces plots of the mean and the scaled standard deviation of the fixed- and variable-axes eccentricities as functions of the impact parameter b

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.31 macro/epsilon_b.C File Reference

```
#include "label.C"
```

Functions

- void [epsilon_b](#) (char *p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the impact parameter b

6.31.1 Function Documentation

6.31.1.1 void epsilon_b (char * p)

produces plots of the mean and the scaled standard deviation of the eccentricities as functions of the impact parameter b

Produces plots of the mean and the scaled standard deviation of the fixed- and variable-axes eccentricities as functions of the impact parameter b

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.32 macro/demo_2/epsilon_c.C File Reference

```
#include "label.C"
```

Functions

- void [epsilon_c](#) (char *p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

6.32.1 Function Documentation

6.32.1.1 void epsilon_c (char * p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

Produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality. The centrality is determined from the number of wounded nucleons.

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.33 macro/epsilon_c.C File Reference

```
#include "label.C"
```

Functions

- void [epsilon_c](#) (char *p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

6.33.1 Function Documentation

6.33.1.1 void epsilon_c (char * p)

produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality

Produces plots of the mean and the scaled standard deviation of the participant eccentricities as functions of centrality. The centrality is determined from the number of wounded nucleons. determination of centrality bins

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.34 macro/demo_2/fitr.C File Reference

```
#include "label.C"
```

Functions

- Double_t [saxon](#) (Double_t *x, Double_t *par)
Woods-Saxon radial density function.
- void [fitr](#) (char *p)
fits the obtained density to the Woods-Saxon form and plot the result

6.34.1 Function Documentation

6.34.1.1 void fitr (char * p)

fits the obtained density to the Woods-Saxon form and plot the result

Fits the obtained density of nucleons in nucleus A to the Woods-Saxon form and plots the result. Requires *profile*=1.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.34.1.2 Double_t saxon (Double_t * x, Double_t * par)

Woods-Saxon radial density function.

Parameters

x	radial coordinate
par	Woods-Saxon parameters (par[0]=norm, par[1]=R, par[2]=a)

6.35 macro/fitr.C File Reference

```
#include "label.C"
```

Functions

- Double_t [saxon](#) (Double_t *x, Double_t *par)
Woods-Saxon radial density function.
- void [fitr](#) (char *p)
fits the obtained density to the Woods-Saxon form and plot the result

6.35.1 Function Documentation

6.35.1.1 void `fitr` (char * *p*)

fits the obtained density to the Woods-Saxon form and plot the result

Fits the obtained density of nucleons in nucleus A to the Woods-Saxon form and plots the result. Requires *profile*=1.

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.35.1.2 Double_t `saxon` (Double_t * *x*, Double_t * *par*)

Woods-Saxon radial density function.

Parameters

<i>x</i>	radial coordinate
<i>par</i>	Woods-Saxon parameters (par[0]=norm, par[1]=R, par[2]=a)

6.36 macro/demo_2/fourier.C File Reference

```
#include "label.C"
```

Functions

- void `fourier` (char *p)
generates the plot of ϵ_n^ vs. N_w , $n=1,2,3,4,5,6$*

6.36.1 Function Documentation

6.36.1.1 void `fourier` (char * *p*)

generates the plot of ϵ_n^* vs. N_w , $n=1,2,3,4,5,6$

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.37 macro/fourier.C File Reference

```
#include "label.C"
```

Functions

- void `fourier` (char *p)
generates the plot of ϵ_n^ vs. N_w , $n=1,2,3,4,5,6$*

6.37.1 Function Documentation

6.37.1.1 void `fourier` (char * *p*)

generates the plot of ϵ_n^* vs. N_w , $n=1,2,3,4,5,6$

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.38 macro/demo_2/info.C File Reference

Functions

- void `info` (char **p*)

prints info on the stored GLISSANDO 2 ROOT file

6.38.1 Function Documentation

6.38.1.1 void `info` (char * *p*)

prints info on the stored GLISSANDO 2 ROOT file

Prints the parameters of the simulation stored in the ROOT file. < relative variance of cross-section fluctuations

< gamma approximation wounding profile parameter (height at the origin)

Parameters

<i>p</i>	name of the ROOT file
----------	-----------------------

6.39 macro/info.C File Reference

Functions

- void `info` (char **p*)

prints info on the stored GLISSANDO 2 ROOT file

6.39.1 Function Documentation

6.39.1.1 void `info` (char * *p*)

prints info on the stored GLISSANDO 2 ROOT file

Prints the parameters of the simulation stored in the ROOT file. < relative variance of cross-section fluctuations

< gamma approximation wounding profile parameter (height at the origin)

Parameters

<i>p</i>	name of the ROOT file
----------	-----------------------

6.40 macro/demo_2/label.C File Reference

Functions

- void `label` (char *infile)
generates the label for graphics, containing the basic information on the simulation
- void `label_fit` (char *infile)
generate the label for plots referring to distributions within the nucleus

6.40.1 Function Documentation

6.40.1.1 void `label` (char * *infile*)

generates the label for graphics, containing the basic information on the simulation

Generates the label for graphics, containing the basic information on the simulation.

Parameters

<i>infile</i>	name of the ROOT input file
---------------	-----------------------------

6.40.1.2 void `label_fit` (char * *infile*)

generate the label for plots referring to distributions within the nucleus

Parameters

<i>infile</i>	name of the ROOT input file
---------------	-----------------------------

6.41 macro/demo_3/label.C File Reference

Functions

- void `label` (char *infile)
generates the label for graphics, containing the basic information on the simulation
- void `label_fit` (char *infile)
generate the label for plots referring to distributions within the nucleus

6.41.1 Function Documentation

6.41.1.1 void `label` (char * *infile*)

generates the label for graphics, containing the basic information on the simulation

Generates the label for graphics, containing the basic information on the simulation.

Parameters

<i>infile</i>	name of the ROOT input file
---------------	-----------------------------

6.41.1.2 void `label_fit` (char * *infile*)

generate the label for plots referring to distributions within the nucleus

Parameters

<i>infile</i>	name of the ROOT input file
---------------	-----------------------------

6.42 macro/label.C File Reference

Functions

- void `label` (char *infile)
generates the label for graphics, containing the basic information on the simulation
- void `label_fit` (char *infile)
generate the label for plots referring to distributions within the nucleus

6.42.1 Function Documentation

6.42.1.1 void label (char * *infile*)

generates the label for graphics, containing the basic information on the simulation

Generates the label for graphics, containing the basic information on the simulation.

Parameters

<i>infile</i>	name of the ROOT input file
---------------	-----------------------------

6.42.1.2 void label_fit (char * *infile*)

generate the label for plots referring to distributions within the nucleus

Parameters

<i>infile</i>	name of the ROOT input file
---------------	-----------------------------

6.43 macro/demo_2/mult.C File Reference

```
#include "label.C"
```

Functions

- void `mult` (char *p)
produces plots of the scaled variance of RDS vs. the number of wounded nucleons in the projectile

6.43.1 Function Documentation

6.43.1.1 void mult (char * *p*)

produces plots of the scaled variance of RDS vs. the number of wounded nucleons in the projectile

Produce the plot of the scaled variance of RDS vs. the number of wounded nucleons in the projectile.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.44 macro/mult.C File Reference

```
#include "label.C"
```

Functions

- void [mult](#) (char *p)

produces plots of the scaled variance of RDS vs. the number of wounded nucleons in the projectile

6.44.1 Function Documentation

6.44.1.1 void mult (char * p)

produces plots of the scaled variance of RDS vs. the number of wounded nucleons in the projectile

Produce the plot of the scaled variance of RDS vs. the number of wounded nucleons in the projectile.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.45 macro/demo_2/overlay.C File Reference

```
#include "label.C"
```

Functions

- void [overlay](#) (char *p)

generate the overlaid distribution for the wounded and binary-collision sources

6.45.1 Function Documentation

6.45.1.1 void overlay (char * p)

generate the overlaid distribution for the wounded and binary-collision sources

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.46 macro/overlay.C File Reference

```
#include "label.C"
```

Functions

- void [overlay](#) (char *p)

generate the overlaid distribution for the wounded and binary-collision sources

6.46.1 Function Documentation

6.46.1.1 void overlay (char * p)

generate the overlaid distribution for the wounded and binary-collision sources

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.47 macro/demo_2/profile2.C File Reference

```
#include "label.C"
```

Functions

- void [profile2](#) (char *p)

produces plots of the fixed-axes and variable-axes Fourier profiles of the density of sources.

6.47.1 Function Documentation

6.47.1.1 void profile2 (char * p)

produces plots of the fixed-axes and variable-axes Fourier profiles of the density of sources.

The plots are obtained by Fourier-projecting the 2-dim distribution of sources.

Parameters

<i>p</i>	name of the GLISSANDO input ROOT file
----------	---------------------------------------

6.48 macro/profile2.C File Reference

```
#include "label.C"
```

Functions

- void [profile2](#) (char *p)

produces plots of the fixed-axes and variable-axes Fourier profiles of the density of sources.

6.48.1 Function Documentation

6.48.1.1 void profile2 (char * *p*)

produces plots of the fixed-axes and variable-axes Fourier profiles of the density of sources.

The plots are obtained by Fourier-projecting the 2-dim distribution of sources.

Parameters

<i>p</i>	name of the GLISSANDO input ROOT file
----------	---------------------------------------

6.49 macro/demo_2/profile2_deformation_63Cu.C File Reference

```
#include "label.C"
```

Functions

- void [profile2_deformation_63Cu](#) (char **p*, char **ps*)
produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

6.49.1 Function Documentation

6.49.1.1 void profile2_deformation_63Cu (char * *p*, char * *ps*)

produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

The plots are obtained as 2D r-cos(theta) histograms of nucleons positions

Parameters

<i>p</i>	name of the GLISSANDO input ROOT for the deformed case
<i>ps</i>	name of the GLISSANDO input ROOT for the spherical case

6.50 macro/profile2_deformation_63Cu.C File Reference

```
#include "label.C"
```

Functions

- void [profile2_deformation_63Cu](#) (char **p*, char **ps*)
produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

6.50.1 Function Documentation

6.50.1.1 void profile2_deformation_63Cu (char * *p*, char * *ps*)

produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

The plots are obtained as 2D r-cos(theta) histograms of nucleons positions

Parameters

<i>p</i>	name of the GLISSANDO input ROOT for the deformed case
<i>ps</i>	name of the GLISSANDO input ROOT for the spherical case

6.51 macro/demo_2/profile2_deformation_Au.C File Reference

```
#include "label.C"
```

Functions

- void [profile2_deformation_Au](#) (char *p, char *ps)
produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

6.51.1 Function Documentation

6.51.1.1 void [profile2_deformation_Au](#) (char * *p*, char * *ps*)

produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

The plots are obtained as 2D r-cos(theta) histograms of nucleons positions

Parameters

<i>p</i>	name of the GLISSANDO input ROOT for the deformed case
<i>ps</i>	name of the GLISSANDO input ROOT for the spherical case

6.52 macro/profile2_deformation_Au.C File Reference

```
#include "label.C"
```

Functions

- void [profile2_deformation_Au](#) (char *p, char *ps)
produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

6.52.1 Function Documentation

6.52.1.1 void [profile2_deformation_Au](#) (char * *p*, char * *ps*)

produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

The plots are obtained as 2D r-cos(theta) histograms of nucleons positions

Parameters

<i>p</i>	name of the GLISSANDO input ROOT for the deformed case
<i>ps</i>	name of the GLISSANDO input ROOT for the spherical case

6.53 macro/demo_2/profile2_deformation_U.C File Reference

```
#include "label.C"
```

Functions

- void [profile2_deformation_U](#) (char *p, char *ps)
produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

6.53.1 Function Documentation

6.53.1.1 void [profile2_deformation_U](#) (char * *p*, char * *ps*)

produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

The plots are obtained as 2D r-cos(theta) histograms of nucleons positions

Parameters

<i>p</i>	name of the GLISSANDO input ROOT for the deformed case
<i>ps</i>	name of the GLISSANDO input ROOT for the spherical case

6.54 macro/profile2_deformation_U.C File Reference

```
#include "label.C"
```

Functions

- void [profile2_deformation_U](#) (char *p, char *ps)
produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

6.54.1 Function Documentation

6.54.1.1 void [profile2_deformation_U](#) (char * *p*, char * *ps*)

produces plots of the fixed-axes r-cos(theta) distributions of the nucleons in the nucleus.

The plots are obtained as 2D r-cos(theta) histograms of nucleons positions

Parameters

<i>p</i>	name of the GLISSANDO input ROOT for the deformed case
<i>ps</i>	name of the GLISSANDO input ROOT for the spherical case

6.55 macro/demo_2/size.C File Reference

```
#include "label.C"
```

Functions

- void [size](#) (char *p)

Produces the plot of the scaled standard deviation of the size parameter.

6.55.1 Function Documentation

6.55.1.1 void size (char * p)

Produces the plot of the scaled standard deviation of the size parameter.

Produces the plot of the scaled standard deviation of the size parameter. Used for the transverse momentum fluctuations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.56 macro/size.C File Reference

```
#include "label.C"
```

Functions

- void [size](#) (char *p)

Produces the plot of the scaled standard deviation of the size parameter.

6.56.1 Function Documentation

6.56.1.1 void size (char * p)

Produces the plot of the scaled standard deviation of the size parameter.

Produces the plot of the scaled standard deviation of the size parameter. Used for the transverse momentum fluctuations.

Parameters

p	name of the ROOT input file
-----	-----------------------------

6.57 macro/demo_2/wounding_profile.C File Reference

```
#include "label.C"
```

Functions

- void [wounding_profile](#) (char *p)
generates the plots of the wounding and binary-collision profiles

6.57.1 Function Documentation

6.57.1.1 void wounding_profile (char * p)

generates the plots of the wounding and binary-collision profiles

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.58 macro/wounding_profile.C File Reference

```
#include "label.C"
```

Functions

- void [wounding_profile](#) (char *p)
generates the plots of the wounding and binary-collision profiles

6.58.1 Function Documentation

6.58.1.1 void wounding_profile (char * p)

generates the plots of the wounding and binary-collision profiles

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.59 macro/demo_3/cent.C File Reference

```
#include "label.C"
```

Functions

- void [cent](#) (char *p)
generates the centrality distribution of wounded objects

6.59.1 Detailed Description

Script generating the centrality distribution of wounded objects (part of GLISSANDO 3)

6.59.2 Function Documentation

6.59.2.1 void cent (char * p)

generates the centrality distribution of wounded objects

generates the centrality distribution of wounded objects (nucleons or quarks)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.60 macro/demo_3/inel_prof.C File Reference

```
#include "label.C"
```

Functions

- void [inel_prof](#) (char *p)
generates the plots of the wounding profile

6.60.1 Function Documentation

6.60.1.1 void inel_prof (char * p)

generates the plots of the wounding profile

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.61 macro/demo_3/rad_dis.C File Reference

```
#include "label.C"
```

Functions

- void [rad_dis](#) (char *p)
radial distribution of nucleons or partons

6.61.1 Detailed Description

Script yielding the radial distribution of nucleons or partons (part of GLISSANDO 3)

6.61.2 Function Documentation

6.61.2.1 void rad_dis (char * p)

radial distribution of nucleons or partons

for *profile*=0 - nucleons, for *profile*=1 - partons

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.62 macro/eps_fluct.C File Reference

Functions

- void [eps_fluct](#) (char *p, int n)

6.62.1 Detailed Description

(part of GLISSANDO 2)

6.62.2 Function Documentation

6.62.2.1 void [eps_fluct](#) (char * *p*, int *n*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.63 macro/fourier_sm_eps_chain.C File Reference

```
#include "label.C"
```

Functions

- void [fourier_sm_eps_chain](#) (char *p, int n)

6.63.1 Function Documentation

6.63.1.1 void [fourier_sm_eps_chain](#) (char * *p*, int *n*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.64 macro/fourier_sm_n2_chain.C File Reference

```
#include "label.C"
```

Functions

- void [fourier_sm_n2_chain](#) (char *p, int n)

6.64.1 Function Documentation

6.64.1.1 void `fourier_sm_n2_chain` (char * *p*, int *n*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.65 macro/fourier_sm_n4_chain.C File Reference

```
#include "label.C"
```

Functions

- void `fourier_sm_n4_chain` (char **p*, int *n*)

6.65.1 Function Documentation

6.65.1.1 void `fourier_sm_n4_chain` (char * *p*, int *n*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.66 macro/g_005.C File Reference

Macros

- #define `PI` 3.141592653589793

Functions

- Double_t `lin` (Double_t **x*, Double_t **par*)
fit function
- void `g_005` ()

6.66.1 Macro Definition Documentation

6.66.1.1 #define `PI` 3.141592653589793

6.66.2 Function Documentation

6.66.2.1 void `g_005` ()

6.66.2.2 Double_t `lin` (Double_t * *x*, Double_t * *par*)

fit function

6.67 macro/g_005_Q.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_005_Q](#) ()

6.67.1 Macro Definition Documentation

6.67.1.1 #define [PI](#) 3.141592653589793

6.67.2 Function Documentation

6.67.2.1 void [g_005_Q](#) ()

6.67.2.2 Double_t [lin](#) (Double_t * *x*, Double_t * *par*)

fit function

6.68 macro/g_005t.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_005](#) ()

6.68.1 Macro Definition Documentation

6.68.1.1 #define [PI](#) 3.141592653589793

6.68.2 Function Documentation

6.68.2.1 void [g_005](#) ()

6.68.2.2 Double_t [lin](#) (Double_t * *x*, Double_t * *par*)

fit function

6.69 macro/g_2030.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_2030](#) ()

6.69.1 Macro Definition Documentation

6.69.1.1 #define [PI](#) 3.141592653589793

6.69.2 Function Documentation

6.69.2.1 void [g_2030](#) ()

6.69.2.2 Double_t [lin](#) (Double_t * x, Double_t * *par*)

fit function

6.70 macro/g_2030_Q.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_2030_Q](#) ()

6.70.1 Macro Definition Documentation

6.70.1.1 #define [PI](#) 3.141592653589793

6.70.2 Function Documentation

6.70.2.1 void [g_2030_Q](#) ()

6.70.2.2 Double_t [lin](#) (Double_t * x, Double_t * *par*)

fit function

6.71 macro/g_2030t.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_2030](#) ()

6.71.1 Macro Definition Documentation

6.71.1.1 #define [PI](#) 3.141592653589793

6.71.2 Function Documentation

6.71.2.1 void [g_2030](#) ()

6.71.2.2 Double_t [lin](#) (Double_t * *x*, Double_t * *par*)

fit function

6.72 macro/g_5060.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_5060](#) ()

6.72.1 Macro Definition Documentation

6.72.1.1 #define [PI](#) 3.141592653589793

6.72.2 Function Documentation

6.72.2.1 void [g_5060](#) ()

6.72.2.2 Double_t [lin](#) (Double_t * *x*, Double_t * *par*)

fit function

6.73 macro/g_5060_Q.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_5060_Q](#) ()

6.73.1 Macro Definition Documentation

6.73.1.1 #define [PI](#) 3.141592653589793

6.73.2 Function Documentation

6.73.2.1 void [g_5060_Q](#) ()

6.73.2.2 Double_t [lin](#) (Double_t * x, Double_t * *par*)

fit function

6.74 macro/g_5060t.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_5060](#) ()

6.74.1 Macro Definition Documentation

6.74.1.1 #define [PI](#) 3.141592653589793

6.74.2 Function Documentation

6.74.2.1 void [g_5060](#) ()

6.74.2.2 Double_t [lin](#) (Double_t * x, Double_t * *par*)

fit function

6.75 macro/g_pp_n.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_pp_n](#) ()

6.75.1 Macro Definition Documentation

6.75.1.1 #define [PI](#) 3.141592653589793

6.75.2 Function Documentation

6.75.2.1 void [g_pp_n](#) ()

6.75.2.2 Double_t [lin](#) (Double_t * x, Double_t * *par*)

fit function

6.76 macro/g_pp_q (Case Conflict).C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_pp_Q](#) ()

6.76.1 Macro Definition Documentation

6.76.1.1 #define [PI](#) 3.141592653589793

6.76.2 Function Documentation

6.76.2.1 void [g_pp_Q](#) ()

6.76.2.2 Double_t [lin](#) (Double_t * x, Double_t * *par*)

fit function

6.77 macro/g_pp_q.C File Reference

```
#include <TRandom3.h>
```

Macros

- #define [PI](#) 3.141592653589793

Functions

- Float_t [los](#) ()
random number generator using the built-in ROOT generator, uniform on (0,1)
- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_pp_q](#) ()

Variables

- TRandom3 [raa](#)
ROOT random number generator.

6.77.1 Macro Definition Documentation

6.77.1.1 #define [PI](#) 3.141592653589793

6.77.2 Function Documentation

6.77.2.1 void [g_pp_q](#) ()

6.77.2.2 Double_t [lin](#) (Double_t * x, Double_t * par)

fit function

6.77.2.3 Float_t [los](#) ()

random number generator using the built-in ROOT generator, uniform on (0,1)

6.77.3 Variable Documentation

6.77.3.1 TRandom3 [raa](#)

ROOT random number generator.

6.78 macro/g_pp_Qt.C File Reference

Macros

- #define [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_pp_Qt](#) ()

6.78.1 Macro Definition Documentation

6.78.1.1 `#define PI 3.141592653589793`

6.78.2 Function Documentation

6.78.2.1 void [g_pp_Qt](#) ()

6.78.2.2 Double_t [lin](#) (Double_t * x, Double_t * par)

fit function

6.79 macro/g_pPb_03_Q.C File Reference

Macros

- `#define PI 3.141592653589793`

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_pPb_03_Q](#) ()

6.79.1 Macro Definition Documentation

6.79.1.1 `#define PI 3.141592653589793`

6.79.2 Function Documentation

6.79.2.1 void [g_pPb_03_Q](#) ()

6.79.2.2 Double_t [lin](#) (Double_t * x, Double_t * par)

fit function

6.80 macro/g_pPb_n21.C File Reference

Macros

- `#define PI 3.141592653589793`

Functions

- `Double_t lin (Double_t *x, Double_t *par)`
fit function
- `void g_pPb_n21 ()`

6.80.1 Macro Definition Documentation

6.80.1.1 `#define PI 3.141592653589793`

6.80.2 Function Documentation

6.80.2.1 `void g_pPb_n21 ( )`

6.80.2.2 `Double_t lin ( Double_t * x, Double_t * par )`

fit function

6.81 macro/g_pPb_n21_Q.C File Reference

Macros

- `#define PI 3.141592653589793`

Functions

- `Double_t lin (Double_t *x, Double_t *par)`
fit function
- `void g_pPb_n21_Q ()`

6.81.1 Macro Definition Documentation

6.81.1.1 `#define PI 3.141592653589793`

6.81.2 Function Documentation

6.81.2.1 `void g_pPb_n21_Q ( )`

6.81.2.2 `Double_t lin ( Double_t * x, Double_t * par )`

fit function

6.82 macro/g_pPb_n25.C File Reference

Macros

- `#define PI 3.141592653589793`

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_pPb_n25](#) ()

6.82.1 Macro Definition Documentation

6.82.1.1 `#define PI 3.141592653589793`

6.82.2 Function Documentation

6.82.2.1 void [g_pPb_n25](#) ()

6.82.2.2 Double_t [lin](#) (Double_t * x, Double_t * par)

fit function

6.83 macro/g_Q_2030.C File Reference

Macros

- `#define` [PI](#) 3.141592653589793

Functions

- Double_t [lin](#) (Double_t *x, Double_t *par)
fit function
- void [g_Q_2030](#) ()

6.83.1 Macro Definition Documentation

6.83.1.1 `#define PI 3.141592653589793`

6.83.2 Function Documentation

6.83.2.1 void [g_Q_2030](#) ()

6.83.2.2 Double_t [lin](#) (Double_t * x, Double_t * par)

fit function

6.84 macro/hydro.C File Reference

Functions

- void [hydro](#) (char *p, char *pp)
generates the grid for input to hydrodynamic calculations

6.84.1 Detailed Description

Script generating grid for input to hydrodynamic calculations, to be used as the initial condition (part of GLISSANDO 2, not supported in GLISSANDO 3!)

6.84.2 Function Documentation

6.84.2.1 void hydro (char * *p*, char * *pp*)

generates the grid for input to hydrodynamic calculations

The grid with the initial condition for hydro is generated from the c0rhist histogram in the format (rho, phi, density).

Parameters

<i>p</i>	the ROOT input file
<i>pp</i>	an ASCII output file in the format (rho, phi, density)

6.85 macro/npa_dist.C File Reference

Functions

- void [npa_dist](#) (char *p)

6.85.1 Function Documentation

6.85.1.1 void npa_dist (char * *p*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.86 macro/nq.C File Reference

Functions

- void [nq](#) (char *p, char *p1)

6.86.1 Function Documentation

6.86.1.1 void nq (char * *p*, char * *p1*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.87 macro/nw_dist.C File Reference

Functions

- void [nw_dist](#) (char *p)

6.87.1 Function Documentation

6.87.1.1 void nw_dist (char * p)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.88 macro/tilted.C File Reference

```
#include "label.C"
```

Functions

- void [tilted](#) (char *p)
generates the tilted initial profile in the x-rapidity space near y=0

6.88.1 Detailed Description

Script generating the tilted initial profile in the x-rapidity space at y=0 (part of GLISSANDO 2)

6.88.2 Function Documentation

6.88.2.1 void tilted (char * p)

generates the tilted initial profile in the x-rapidity space near y=0

Generates the tilted initial profile in the x-rapidity space near y=0. Useful for testing the initial conditions to hydrodynamics. Requires *rrapidity*=1.

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.89 macro/w_prof_norm.C File Reference

Functions

- Double_t [nn](#) (Double_t *x, Double_t *par)
- void [w_prof_norm](#) (char *p)

6.89.1 Function Documentation

6.89.1.1 Double_t nn (Double_t * x, Double_t * par)

6.89.1.2 void w_prof_norm (char * *p*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.90 macro/w_prof_norm_wb.C File Reference

Functions

- Double_t [nn](#) (Double_t *x, Double_t *par)
- Float_t [nn_1](#) (Float_t x)
- void [w_prof_norm_wb](#) (char *p)

Variables

- Float_t [siexp](#) =74.4354
- Float_t [gama](#) =0.99999
- Float_t [omega](#) =0.514292

6.90.1 Function Documentation

6.90.1.1 Double_t [nn](#) (Double_t * x, Double_t * *par*)

6.90.1.2 Float_t [nn_1](#) (Float_t x)

6.90.1.3 void [w_prof_norm_wb](#) (char * *p*)

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.90.2 Variable Documentation

6.90.2.1 Float_t [gama](#) =0.99999

6.90.2.2 Float_t [omega](#) =0.514292

6.90.2.3 Float_t [siexp](#) =74.4354

6.91 macro/w_prof_norm_wb0.C File Reference

Functions

- Double_t [nn](#) (Double_t *x, Double_t *par)
- Float_t [nn_1](#) (Float_t x)
- void [w_prof_norm_wb0](#) (char *p)

Variables

- Float_t [gama](#) =0.99999
- Float_t [omega](#) =0.514292

6.91.1 Function Documentation

6.91.1.1 `Double_t nn ( Double_t * x, Double_t * par )`

6.91.1.2 `Float_t nn_1 ( Float_t x )`

6.91.1.3 `void w_prof_norm_wb0 ( char * p )`

Parameters

<i>p</i>	name of the ROOT input file
----------	-----------------------------

6.91.2 Variable Documentation

6.91.2.1 `Float_t gama =0.99999`

6.91.2.2 `Float_t omega =0.514292`

Index

- ~counter
 - counter, [14](#)
- ~counter2
 - counter2, [16](#)
- ~counter_2D
 - counter_2D, [18](#)
- _VER_
 - glissando3.cxx, [79](#)
- ALPHA
 - functions.h, [66](#)
- ARANK
 - functions.h, [66](#)
- AWSA
 - functions.h, [66](#)
- AWSB
 - functions.h, [66](#)
- add
 - counter, [14](#)
 - counter2, [16](#)
 - counter_2D, [18](#)
- angles
 - angles.C, [80](#)
 - functions.h, [66](#)
- angles.C
 - angles, [80](#)
- b
 - functions.h, [66](#)
- BETA2A
 - functions.h, [66](#)
- BETA2B
 - functions.h, [66](#)
- BETA4A
 - functions.h, [66](#)
- BETA4B
 - functions.h, [66](#)
- BMAX
 - functions.h, [67](#)
- BMIN
 - functions.h, [67](#)
- BTOT
 - functions.h, [67](#)
- bin
 - bin_t, [9](#)
- bin_t, [9](#)
 - bin, [9](#)
- build/include/collision.h, [53](#)
- build/include/counter.h, [53](#)
- build/include/distrib.h, [53](#)
- build/include/functions.h, [54](#)
- build/src/glissando3.cxx, [78](#)
- c
 - distr, [35](#)
- CD
 - functions.h, [67](#)
- cen_exa
 - cen_exa.C, [81](#)
- cen_exa.C
 - cen_exa, [81](#)
- cen_exa_RDS
 - cen_exa_RDS.C, [81](#)
 - demo_3/cen_exa_RDS.C, [82](#)
- cen_exa_RDS.C
 - cen_exa_RDS, [81](#)
- cen_exa_RDS_moj
 - cen_exa_RDS_moj.C, [82](#)
- cen_exa_RDS_moj.C
 - cen_exa_RDS_moj, [82](#)
- cen_ratio
 - cen_ratio.C, [83](#)
- cen_ratio.C
 - cen_ratio, [83](#)
- cenpPb
 - cenpPb.C, [83](#)
- cenpPb.C
 - cenpPb, [83](#)
- cent
 - cent.C, [107](#)
- cent.C
 - cent, [107](#)
- centrality
 - centrality.C, [84](#)
 - demo_2/centrality.C, [84](#)
- centrality.C
 - centrality, [84](#)
- centrality2
 - centrality2.C, [85](#)
 - demo_2/centrality2.C, [85](#)
- centrality2.C
 - centrality2, [85](#)
- centrality2l
 - centrality2l.C, [86](#)
- centrality2l.C
 - centrality2l, [86](#)
- centrality3
 - centrality3.C, [86](#)
- centrality3.C
 - centrality3, [86](#)

- cmx
 - distr, [24](#)
- cmx_w
 - distr, [24](#)
- cmy
 - distr, [24](#)
- cmy_w
 - distr, [24](#), [25](#)
- cmz
 - distr, [25](#)
- cmz_w
 - distr, [25](#)
- col_chain_8_q
 - col_chain_8_q.C, [87](#)
- col_chain_8_q.C
 - col_chain_8_q, [87](#)
 - f2, [87](#)
 - f3, [87](#)
 - krot, [87](#)
 - los, [87](#)
 - raa, [87](#)
- collision, [9](#)
 - collision, [10](#), [11](#)
 - gen_RDS, [11](#)
 - nbin, [12](#)
 - nhotspot, [12](#)
 - nwA, [12](#)
 - nwAB, [12](#)
 - nwB, [12](#)
 - nzw, [12](#)
 - operator=, [11](#)
 - rd2, [12](#)
 - rpa, [12](#)
 - shift_cmx_w_c, [11](#)
 - shift_cmy_w_c, [11](#)
 - specA, [12](#)
 - specB, [12](#)
 - wc, [12](#)
 - writerds, [11](#)
 - wwA, [12](#)
 - wwAB, [13](#)
 - wwB, [13](#)
 - xcmA, [13](#)
 - xcmB, [13](#)
 - ycmA, [13](#)
 - ycmB, [13](#)
- core_mantle
 - core_mantle.C, [88](#)
 - demo_2/core_mantle.C, [89](#)
- core_mantle.C
 - core_mantle, [88](#)
- corr
 - corr.C, [89](#)
 - counter_2D, [19](#)
 - demo_2/corr.C, [90](#)
- corr.C
 - corr, [89](#)
- count
 - counter, [15](#)
 - counter2, [17](#)
 - counter_2D, [20](#)
- counter, [13](#)
 - ~counter, [14](#)
 - add, [14](#)
 - count, [15](#)
 - counter, [14](#)
 - get, [14](#)
 - getN, [14](#)
 - mean, [14](#)
 - reset, [14](#)
 - value, [15](#)
- counter2, [15](#)
 - ~counter2, [16](#)
 - add, [16](#)
 - count, [17](#)
 - counter2, [16](#)
 - get, [16](#)
 - get2, [16](#)
 - getN, [16](#)
 - mean, [16](#)
 - reset, [16](#)
 - value, [17](#)
 - value2, [17](#)
 - var, [16](#)
 - vara, [16](#)
- counter_2D, [17](#)
 - ~counter_2D, [18](#)
 - add, [18](#)
 - corr, [19](#)
 - count, [20](#)
 - counter_2D, [18](#)
 - counter_2D, [18](#)
 - cov, [19](#)
 - get_x, [19](#)
 - get_x2, [19](#)
 - get_xy, [19](#)
 - get_y, [19](#)
 - get_y2, [19](#)
 - getN, [19](#)
 - mean_x, [19](#)
 - mean_y, [19](#)
 - reset, [19](#)
 - valuex, [20](#)
 - valuex2, [20](#)
 - valuexy, [20](#)
 - valuey, [20](#)
 - valuey2, [20](#)
 - var_x, [19](#)
 - var_y, [20](#)
- cov
- counter_2D, [19](#)
- cth
 - nucleus, [43](#)
- d
 - functions.h, [67](#)
- DBIN

- functions.h, [67](#)
- DOBIN
 - functions.h, [67](#)
- DS
 - functions.h, [67](#)
- DW
 - functions.h, [67](#)
- dbin
 - functions.h, [67](#)
- demo_2/centrality.C
 - centrality, [84](#)
- demo_2/centrality2.C
 - centrality2, [85](#)
- demo_2/core_mantle.C
 - core_mantle, [89](#)
- demo_2/corr.C
 - corr, [90](#)
- demo_2/density.C
 - density, [90](#)
- demo_2/dxdy.C
 - dxdy, [91](#)
- demo_2/epsilon.C
 - epsilon, [92](#)
- demo_2/epsilon_b.C
 - epsilon_b, [93](#)
- demo_2/epsilon_c.C
 - epsilon_c, [94](#)
- demo_2/fitr.C
 - fitr, [95](#)
 - saxon, [95](#)
- demo_2/fourier.C
 - fourier, [96](#)
- demo_2/info.C
 - info, [97](#)
- demo_2/label.C
 - label, [98](#)
 - label_fit, [98](#)
- demo_2/mult.C
 - mult, [99](#)
- demo_2/overlay.C
 - overlay, [100](#)
- demo_2/profile2.C
 - profile2, [101](#)
- demo_2/profile2_deformation_63Cu.C
 - profile2_deformation_63Cu, [102](#)
- demo_2/profile2_deformation_Au.C
 - profile2_deformation_Au, [103](#)
- demo_2/profile2_deformation_U.C
 - profile2_deformation_U, [104](#)
- demo_2/size.C
 - size, [105](#)
- demo_2/wounding_profile.C
 - wounding_profile, [106](#)
- demo_3/cen_exa_RDS.C
 - cen_exa_RDS, [82](#)
- demo_3/label.C
 - label, [98](#)
 - label_fit, [98](#)
- density
 - demo_2/density.C, [90](#)
 - density.C, [91](#)
- density.C
 - density, [91](#)
- disp
 - functions.h, [61](#)
- dist
 - functions.h, [62](#)
- dist2
 - nucleus, [38](#)
- distr, [20](#)
 - c, [35](#)
 - cmx, [24](#)
 - cmx_w, [24](#)
 - cmy, [24](#)
 - cmy_w, [24](#), [25](#)
 - cmz, [25](#)
 - cmz_w, [25](#)
 - distr, [24](#)
 - eps, [25](#), [26](#)
 - eps2p_sq, [26](#)
 - eps_s, [26](#), [27](#)
 - fill_polar, [27](#)
 - fill_polar_s, [27](#)
 - fill_xy, [28](#)
 - mA, [28](#)
 - mAnz, [28](#)
 - mB, [28](#)
 - mBnz, [28](#)
 - msr, [35](#)
 - msrad, [29](#)
 - msrad_t, [29](#)
 - msrad_t_w, [29](#)
 - msrad_w, [29](#)
 - msrt, [35](#)
 - msx, [29](#)
 - msy, [29](#)
 - mxy, [29](#)
 - n, [35](#)
 - operator=, [29](#)
 - phrot, [29](#)
 - qx, [30](#)
 - qy, [30](#)
 - rotate, [30](#)
 - rotate_polar, [32](#)
 - shift_cmx, [32](#)
 - shift_cmx_w, [32](#)
 - shift_cmy, [32](#)
 - shift_cmy_w, [32](#)
 - shift_cmz, [32](#)
 - shift_cmz_w, [32](#)
 - shift_x, [32](#)
 - shift_y, [34](#)
 - shift_z, [34](#)
 - size, [34](#)
 - sum_w, [34](#)
 - sumw, [35](#)

- surf, [34](#)
- tilt, [34](#)
- uncluster, [34](#)
- w, [35](#)
- x, [35](#)
- xcm, [35](#)
- y, [35](#)
- ycm, [35](#)
- ye, [35](#)
- ys, [35](#)
- z, [36](#)
- zcm, [36](#)
- dx dy
 - demo_2/dx dy.C, [91](#)
 - dx dy.C, [92](#)
- dx dy.C
 - dx dy, [92](#)
- ECM
 - functions.h, [67](#)
- ETA0
 - functions.h, [70](#)
- ETACC
 - functions.h, [70](#)
- ETAM
 - functions.h, [70](#)
- EVENTS
 - functions.h, [70](#)
- echopar
 - functions.h, [62](#)
- ep
 - functions.h, [67](#)
- ep1
 - functions.h, [68](#)
- ep1s
 - functions.h, [68](#)
- ep22
 - functions.h, [68](#)
- ep3
 - functions.h, [68](#)
- ep32
 - functions.h, [68](#)
- ep3s
 - functions.h, [68](#)
- ep4
 - functions.h, [68](#)
- ep42
 - functions.h, [68](#)
- ep4s
 - functions.h, [68](#)
- ep5
 - functions.h, [68](#)
- ep5s
 - functions.h, [68](#)
- ep6
 - functions.h, [68](#)
- ep6s
 - functions.h, [68](#)
- epA2
 - functions.h, [68](#)
- epA3
 - functions.h, [69](#)
- epA4
 - functions.h, [69](#)
- epart
 - functions.h, [69](#)
- epart1
 - functions.h, [69](#)
- epart3
 - functions.h, [69](#)
- epart4
 - functions.h, [69](#)
- epart5
 - functions.h, [69](#)
- epart6
 - functions.h, [69](#)
- epilog
 - functions.h, [62](#)
- eps
 - distr, [25](#), [26](#)
 - functions.h, [69](#)
- eps2p_sq
 - distr, [26](#)
- eps_fluct
 - eps_fluct.C, [109](#)
- eps_fluct.C
 - eps_fluct, [109](#)
- eps_s
 - distr, [26](#), [27](#)
- epsilon
 - demo_2/epsilon.C, [92](#)
 - epsilon.C, [92](#)
- epsilon.C
 - epsilon, [92](#)
- epsilon_b
 - demo_2/epsilon_b.C, [93](#)
 - epsilon_b.C, [93](#)
- epsilon_b.C
 - epsilon_b, [93](#)
- epsilon_c
 - demo_2/epsilon_c.C, [94](#)
 - epsilon_c.C, [94](#)
- epsilon_c.C
 - epsilon_c, [94](#)
- es
 - functions.h, [69](#)
- es3
 - functions.h, [69](#)
- es3s
 - functions.h, [69](#)
- es4
 - functions.h, [69](#)
- es4s
 - functions.h, [69](#)
- es5
 - functions.h, [70](#)
- es5s

- functions.h, 70
- es6
 - functions.h, 70
- es6s
 - functions.h, 70
- ess
 - functions.h, 70
- evall
 - functions.h, 70
- f2
 - col_chain_8_q.C, 87
- f3
 - col_chain_8_q.C, 87
- FBIN
 - functions.h, 70
- FBRAP
 - functions.h, 70
- FILES
 - functions.h, 71
- FULL
 - functions.h, 71
- fg
 - functions.h, 62
- fill
 - tr_his_c, 47
- fill_polar
 - distr, 27
- fill_polar_s
 - distr, 27
- fill_res
 - tr_his_c, 47
- fill_tr
 - tr_his_c, 47
- fill_xy
 - distr, 28
- fitr
 - demo_2/fitr.C, 95
 - fitr.C, 96
- fitr.C
 - fitr, 96
 - saxon, 96
- fourier
 - demo_2/fourier.C, 96
 - fourier.C, 97
- fourier.C
 - fourier, 97
- fourier_sm_eps_chain
 - fourier_sm_eps_chain.C, 109
- fourier_sm_eps_chain.C
 - fourier_sm_eps_chain, 109
- fourier_sm_n2_chain
 - fourier_sm_n2_chain.C, 110
- fourier_sm_n2_chain.C
 - fourier_sm_n2_chain, 110
- fourier_sm_n4_chain
 - fourier_sm_n4_chain.C, 110
- fourier_sm_n4_chain.C
 - fourier_sm_n4_chain, 110
- fpm
 - functions.h, 62
- fqscale
 - functions.h, 62
- fsNN
 - functions.h, 62
- fsQQ
 - functions.h, 63
- full_event
 - tr_his_c, 48
- functions.h
 - ALPHA, 66
 - ARANK, 66
 - AWSA, 66
 - AWSB, 66
 - angles, 66
 - b, 66
 - BETA2A, 66
 - BETA2B, 66
 - BETA4A, 66
 - BETA4B, 66
 - BMAX, 67
 - BMIN, 67
 - BTOT, 67
 - CD, 67
 - d, 67
 - DBIN, 67
 - DOBIN, 67
 - DS, 67
 - DW, 67
 - dbin, 67
 - disp, 61
 - dist, 62
 - ECM, 67
 - ETA0, 70
 - ETACC, 70
 - ETAM, 70
 - EVENTS, 70
 - echopar, 62
 - ep, 67
 - ep1, 68
 - ep1s, 68
 - ep22, 68
 - ep3, 68
 - ep32, 68
 - ep3s, 68
 - ep4, 68
 - ep42, 68
 - ep4s, 68
 - ep5, 68
 - ep5s, 68
 - ep6, 68
 - ep6s, 68
 - epA2, 68
 - epA3, 69
 - epA4, 69
 - epart, 69
 - epart1, 69

epart3, 69
 epart4, 69
 epart5, 69
 epart6, 69
 epilog, 62
 eps, 69
 es, 69
 es3, 69
 es3s, 69
 es4, 69
 es4s, 69
 es5, 70
 es5s, 70
 es6, 70
 es6s, 70
 ess, 70
 evall, 70
 FBIN, 70
 FBRAP, 70
 FILES, 71
 FULL, 71
 fg, 62
 fpm, 62
 fqscale, 62
 fsNN, 62
 fsQQ, 63
 GA, 71
 GAMA, 71
 gamgen, 63
 header, 63
 helper, 63
 ISEED, 71
 ISEED1, 71
 kk, 71
 los, 63
 los_rap_A, 63
 los_rap_B, 63
 los_rap_bin, 63
 MAXYRAP, 71
 MODEL, 71
 NBIN, 71
 NCS, 71
 NNWP, 72
 NUMA, 72
 NUMB, 72
 NUMRAP, 72
 nbinary, 71
 negbin, 64
 nhot, 72
 nweight, 72
 nwounded, 72
 OMEGA, 72
 PARTONS, 72
 PI, 73
 PP, 73
 phiro, 72
 phiro3, 72
 phiro4, 72
 phiro5, 72
 phiro6, 73
 phiro_minus, 73
 phiro_plus, 73
 ppp, 73
 QSCALE, 73
 RAPRANGE, 73
 RAPSHIFT, 73
 RCHA, 73
 RCHB, 73
 RCHP, 74
 RDS0, 74
 RDS1, 74
 RET, 74
 RO, 74
 ROTA_PHI, 74
 ROTA_THETA, 74
 ROTB_PHI, 74
 ROTB_THETA, 74
 RWSA, 75
 RWSB, 75
 rbin, 73
 readpar, 64
 reset_counters, 64
 rhotspot, 74
 rlos_abf, 64
 rlos_alpha, 64
 rlos_hole, 64
 rlos_hult, 64
 rlos_parton, 65
 rlosA, 65
 rlosA_def, 65
 rlosA_hos, 65
 rlosB, 65
 rlosB_def, 65
 rlosB_hos, 65
 roo, 74
 rpa, 74
 rwA, 75
 rwAB, 75
 rwB, 75
 SBIN, 75
 SCALEA, 75
 SCALEB, 75
 SHIFT, 75
 SIGETA, 75
 SIGMAA, 75
 SIGMAB, 75
 SIGMABISA, 76
 SIGMABISB, 76
 SNN, 76
 sirad, 76
 sitot, 76
 sizeav, 76
 surfA, 76
 tiltA, 76
 time_start, 65
 time_stop, 66

- Ubin, [76](#)
- Uw, [76](#)
- Vbin, [76](#)
- Vw, [76](#)
- W0, [77](#)
- W1, [77](#)
- WFA, [77](#)
- WFB, [77](#)
- WMIN, [77](#)
- wfq, [77](#)
- wfqA, [77](#)
- wfqB, [77](#)
- xepp, [77](#)
- xsepp, [77](#)
- xx, [77](#)
- YB, [77](#)
- yy, [78](#)
- g
 - nucleus, [43](#)
- g_005
 - g_005.C, [110](#)
 - g_005t.C, [111](#)
- g_005.C
 - g_005, [110](#)
 - lin, [110](#)
 - PI, [110](#)
- g_005_Q
 - g_005_Q.C, [111](#)
- g_005_Q.C
 - g_005_Q, [111](#)
 - lin, [111](#)
 - PI, [111](#)
- g_005t.C
 - g_005, [111](#)
 - lin, [111](#)
 - PI, [111](#)
- g_2030
 - g_2030.C, [112](#)
 - g_2030t.C, [113](#)
- g_2030.C
 - g_2030, [112](#)
 - lin, [112](#)
 - PI, [112](#)
- g_2030_Q
 - g_2030_Q.C, [112](#)
- g_2030_Q.C
 - g_2030_Q, [112](#)
 - lin, [112](#)
 - PI, [112](#)
- g_2030t.C
 - g_2030, [113](#)
 - lin, [113](#)
 - PI, [113](#)
- g_5060
 - g_5060.C, [113](#)
 - g_5060t.C, [114](#)
- g_5060.C
 - g_5060, [113](#)
- lin, [113](#)
- PI, [113](#)
- g_5060_Q
 - g_5060_Q.C, [114](#)
- g_5060_Q.C
 - g_5060_Q, [114](#)
 - lin, [114](#)
 - PI, [114](#)
- g_5060t.C
 - g_5060, [114](#)
 - lin, [114](#)
 - PI, [114](#)
- g_Q_2030
 - g_Q_2030.C, [119](#)
- g_Q_2030.C
 - g_Q_2030, [119](#)
 - lin, [119](#)
 - PI, [119](#)
- g_pPb_03_Q
 - g_pPb_03_Q.C, [117](#)
- g_pPb_03_Q.C
 - g_pPb_03_Q, [117](#)
 - lin, [117](#)
 - PI, [117](#)
- g_pPb_n21
 - g_pPb_n21.C, [118](#)
- g_pPb_n21.C
 - g_pPb_n21, [118](#)
 - lin, [118](#)
 - PI, [118](#)
- g_pPb_n21_Q
 - g_pPb_n21_Q.C, [118](#)
- g_pPb_n21_Q.C
 - g_pPb_n21_Q, [118](#)
 - lin, [118](#)
 - PI, [118](#)
- g_pPb_n25
 - g_pPb_n25.C, [119](#)
- g_pPb_n25.C
 - g_pPb_n25, [119](#)
 - lin, [119](#)
 - PI, [119](#)
- g_pp_Q
 - g_pp_q (Case Conflict).C, [115](#)
- g_pp_Qt
 - g_pp_Qt.C, [117](#)
- g_pp_Qt.C
 - g_pp_Qt, [117](#)
 - lin, [117](#)
 - PI, [117](#)
- g_pp_n
 - g_pp_n.C, [115](#)
- g_pp_n.C
 - g_pp_n, [115](#)
 - lin, [115](#)
 - PI, [115](#)
- g_pp_q
 - g_pp_q.C, [116](#)

- g_pp_q (Case Conflict).C
 - g_pp_Q, [115](#)
 - lin, [115](#)
 - PI, [115](#)
- g_pp_q.C
 - g_pp_q, [116](#)
 - lin, [116](#)
 - los, [116](#)
 - PI, [116](#)
 - raa, [116](#)
- GA
 - functions.h, [71](#)
- GAMA
 - functions.h, [71](#)
- gama
 - w_prof_norm_wb.C, [123](#)
 - w_prof_norm_wb0.C, [124](#)
- gamgen
 - functions.h, [63](#)
- gen
 - tr_his_c, [47](#)
- gen_RDS
 - collision, [11](#)
- get
 - counter, [14](#)
 - counter2, [16](#)
- get2
 - counter2, [16](#)
- get_x
 - counter_2D, [19](#)
- get_x2
 - counter_2D, [19](#)
- get_xy
 - counter_2D, [19](#)
- get_y
 - counter_2D, [19](#)
- get_y2
 - counter_2D, [19](#)
- getN
 - counter, [14](#)
 - counter2, [16](#)
 - counter_2D, [19](#)
- glissando3.cxx
 - _VER_, [79](#)
 - main, [79](#)
 - raa, [80](#)
 - ver, [80](#)
- good_all
 - nucleus, [38](#)
- good_down
 - nucleus, [38](#)
- good_pair
 - nucleus, [38](#)
- header
 - functions.h, [63](#)
- helper
 - functions.h, [63](#)
- hydro
 - hydro.C, [120](#)
- hydro.C
 - hydro, [120](#)
- ISEED
 - functions.h, [71](#)
- ISEED1
 - functions.h, [71](#)
- inel_prof
 - inel_prof.C, [107](#)
- inel_prof.C
 - inel_prof, [107](#)
- info
 - demo_2/info.C, [97](#)
 - info.C, [97](#)
- info.C
 - info, [97](#)
- init
 - tr_his_c, [47](#)
- KK
 - SOURCE, [44](#)
- kk
 - functions.h, [71](#)
- krot
 - col_chain_8_q.C, [87](#)
- label
 - demo_2/label.C, [98](#)
 - demo_3/label.C, [98](#)
 - label.C, [99](#)
- label.C
 - label, [99](#)
 - label_fit, [99](#)
- label_fit
 - demo_2/label.C, [98](#)
 - demo_3/label.C, [98](#)
 - label.C, [99](#)
- lin
 - g_005.C, [110](#)
 - g_005_Q.C, [111](#)
 - g_005t.C, [111](#)
 - g_2030.C, [112](#)
 - g_2030_Q.C, [112](#)
 - g_2030t.C, [113](#)
 - g_5060.C, [113](#)
 - g_5060_Q.C, [114](#)
 - g_5060t.C, [114](#)
 - g_pp_n.C, [115](#)
 - g_pp_q (Case Conflict).C, [115](#)
 - g_pp_q.C, [116](#)
 - g_pp_Qt.C, [117](#)
 - g_pPb_03_Q.C, [117](#)
 - g_pPb_n21.C, [118](#)
 - g_pPb_n21_Q.C, [118](#)
 - g_pPb_n25.C, [119](#)
 - g_Q_2030.C, [119](#)
- los
 - col_chain_8_q.C, [87](#)

functions.h, 63
 g_pp_q.C, 116
 los_rap_A
 functions.h, 63
 los_rap_B
 functions.h, 63
 los_rap_bin
 functions.h, 63

 mA
 distr, 28
 MAXYRAP
 functions.h, 71
 mAnz
 distr, 28
 mB
 distr, 28
 mBnz
 distr, 28
 MODEL
 functions.h, 71
 macro/angles.C, 80
 macro/cen_exa.C, 81
 macro/cen_exa_RDS.C, 81
 macro/cen_exa_RDS_moj.C, 82
 macro/cen_ratio.C, 83
 macro/cenpPb.C, 83
 macro/centrality.C, 84
 macro/centrality2.C, 85
 macro/centrality2l.C, 86
 macro/centrality3.C, 86
 macro/col_chain_8_q.C, 87
 macro/core_mantle.C, 87
 macro/corr.C, 89
 macro/demo_2/centrality.C, 84
 macro/demo_2/centrality2.C, 85
 macro/demo_2/core_mantle.C, 89
 macro/demo_2/corr.C, 90
 macro/demo_2/density.C, 90
 macro/demo_2/dxdy.C, 91
 macro/demo_2/epsilon.C, 92
 macro/demo_2/epsilon_b.C, 93
 macro/demo_2/epsilon_c.C, 94
 macro/demo_2/fitr.C, 95
 macro/demo_2/fourier.C, 96
 macro/demo_2/info.C, 97
 macro/demo_2/label.C, 98
 macro/demo_2/mult.C, 99
 macro/demo_2/overlay.C, 100
 macro/demo_2/profile2.C, 101
 macro/demo_2/profile2_deformation_63Cu.C, 102
 macro/demo_2/profile2_deformation_Au.C, 103
 macro/demo_2/profile2_deformation_U.C, 104
 macro/demo_2/size.C, 105
 macro/demo_2/wounding_profile.C, 105
 macro/demo_3/cen_exa_RDS.C, 82
 macro/demo_3/cent.C, 106
 macro/demo_3/inel_prof.C, 107
 macro/demo_3/label.C, 98
 macro/demo_3/rad_dis.C, 107
 macro/density.C, 90
 macro/dxdy.C, 91
 macro/eps_fluct.C, 109
 macro/epsilon.C, 92
 macro/epsilon_b.C, 93
 macro/epsilon_c.C, 94
 macro/fitr.C, 95
 macro/fourier.C, 96
 macro/fourier_sm_eps_chain.C, 109
 macro/fourier_sm_n2_chain.C, 109
 macro/fourier_sm_n4_chain.C, 110
 macro/g_005.C, 110
 macro/g_005_Q.C, 111
 macro/g_005t.C, 111
 macro/g_2030.C, 112
 macro/g_2030_Q.C, 112
 macro/g_2030t.C, 113
 macro/g_5060.C, 113
 macro/g_5060_Q.C, 114
 macro/g_5060t.C, 114
 macro/g_Q_2030.C, 119
 macro/g_pPb_03_Q.C, 117
 macro/g_pPb_n21.C, 117
 macro/g_pPb_n21_Q.C, 118
 macro/g_pPb_n25.C, 118
 macro/g_pp_Qt.C, 116
 macro/g_pp_n.C, 115
 macro/g_pp_q (Case Conflict).C, 115
 macro/g_pp_q.C, 116
 macro/hydro.C, 119
 macro/info.C, 97
 macro/label.C, 99
 macro/mult.C, 100
 macro/npa_dist.C, 120
 macro/nq.C, 120
 macro/nw_dist.C, 120
 macro/overlay.C, 100
 macro/profile2.C, 101
 macro/profile2_deformation_63Cu.C, 102
 macro/profile2_deformation_Au.C, 103
 macro/profile2_deformation_U.C, 104
 macro/size.C, 105
 macro/tilted.C, 121
 macro/w_prof_norm.C, 121
 macro/w_prof_norm_wb.C, 123
 macro/w_prof_norm_wb0.C, 123
 macro/wounding_profile.C, 106
 main
 glissando3.cxx, 79
 mean
 counter, 14
 counter2, 16
 mean_x
 counter_2D, 19
 mean_y
 counter_2D, 19
 msr

- distr, [35](#)
- msrad
 - distr, [29](#)
- msrad_t
 - distr, [29](#)
- msrad_t_w
 - distr, [29](#)
- msrad_w
 - distr, [29](#)
- msrt
 - distr, [35](#)
- msx
 - distr, [29](#)
- msy
 - distr, [29](#)
- mult
 - demo_2/mult.C, [99](#)
 - mult.C, [100](#)
- mult.C
 - mult, [100](#)
- mxy
 - distr, [29](#)
- n
 - distr, [35](#)
- NBIN
 - functions.h, [71](#)
- NCS
 - functions.h, [71](#)
- NNWP
 - functions.h, [72](#)
- NUMA
 - functions.h, [72](#)
- NUMB
 - functions.h, [72](#)
- NUMRAP
 - functions.h, [72](#)
- nbin
 - collision, [12](#)
- nbinar
 - tr_his_c, [48](#)
- nbinar2
 - tr_his_c, [48](#)
- nbinary
 - functions.h, [71](#)
- negbin
 - functions.h, [64](#)
- neps
 - tr_his_c, [48](#)
- neps2
 - tr_his_c, [48](#)
- neps2b
 - tr_his_c, [48](#)
- neps4
 - tr_his_c, [48](#)
- nepsb
 - tr_his_c, [48](#)
- nepsp
 - tr_his_c, [48](#)
- nepsp2
 - tr_his_c, [49](#)
- nepsp22
 - tr_his_c, [49](#)
- nepsp2b
 - tr_his_c, [49](#)
- nepsp4
 - tr_his_c, [49](#)
- nepspb
 - tr_his_c, [49](#)
- nhot
 - functions.h, [72](#)
- nhotspot
 - collision, [12](#)
- nn
 - w_prof_norm.C, [121](#)
 - w_prof_norm_wb.C, [123](#)
 - w_prof_norm_wb0.C, [124](#)
- nn_1
 - w_prof_norm_wb.C, [123](#)
 - w_prof_norm_wb0.C, [124](#)
- npa_dist
 - npa_dist.C, [120](#)
- npa_dist.C
 - npa_dist, [120](#)
- nq
 - nq.C, [120](#)
- nq.C
 - nq, [120](#)
- nsize
 - tr_his_c, [49](#)
- nsize2
 - tr_his_c, [49](#)
- ntarg
 - tr_his_c, [49](#)
- ntarg2
 - tr_his_c, [49](#)
- nucleus, [36](#)
 - cth, [43](#)
 - dist2, [38](#)
 - g, [43](#)
 - good_all, [38](#)
 - good_down, [38](#)
 - good_pair, [38](#)
 - nucleus, [38](#)
 - operator=, [39](#)
 - phi, [43](#)
 - r, [43](#)
 - set_alpha_cluster, [39](#)
 - set_berillium_7_cluster, [39](#)
 - set_berillium_8_cluster, [39](#)
 - set_berillium_9_cluster, [39](#)
 - set_carbon_cluster, [39](#)
 - set_deuteron, [40](#)
 - set_file, [40](#)
 - set_file_uncor, [40](#)
 - set_oxygen_cluster, [40](#)
 - set_oxygen_cluster_square, [41](#)

- set_parton_A, [41](#)
- set_parton_B, [41](#)
- set_proton, [41](#)
- set_random_A, [41](#)
- set_random_A_def, [42](#)
- set_random_A_hos, [42](#)
- set_random_B, [42](#)
- set_random_B_def, [42](#)
- set_random_B_hos, [43](#)
- set_tritium, [43](#)
- sth, [43](#)
- nuni
 - tr_his_c, [49](#)
- nuniRDS
 - tr_his_c, [49](#)
- nunib
 - tr_his_c, [49](#)
- nunp
 - tr_his_c, [50](#)
- nw2b
 - tr_his_c, [50](#)
- nw_dist
 - nw_dist.C, [121](#)
- nw_dist.C
 - nw_dist, [121](#)
- nwA
 - collision, [12](#)
- nwAB
 - collision, [12](#)
- nwB
 - collision, [12](#)
- nwb
 - tr_his_c, [50](#)
- nwei
 - tr_his_c, [50](#)
- nwei2
 - tr_his_c, [50](#)
- nweight
 - functions.h, [72](#)
- nwounded
 - functions.h, [72](#)
- nx
 - tr_his_c, [50](#)
- nx2
 - tr_his_c, [50](#)
- ny
 - tr_his_c, [50](#)
- ny2
 - tr_his_c, [50](#)
- nzw
 - collision, [12](#)
- OMEGA
 - functions.h, [72](#)
- omega
 - w_prof_norm_wb.C, [123](#)
 - w_prof_norm_wb0.C, [124](#)
- operator=
 - collision, [11](#)
- distr, [29](#)
- nucleus, [39](#)
- overlay
 - demo_2/overlay.C, [100](#)
 - overlay.C, [101](#)
- overlay.C
 - overlay, [101](#)
- PARTONS
 - functions.h, [72](#)
- PI
 - functions.h, [73](#)
 - g_005.C, [110](#)
 - g_005_Q.C, [111](#)
 - g_005t.C, [111](#)
 - g_2030.C, [112](#)
 - g_2030_Q.C, [112](#)
 - g_2030t.C, [113](#)
 - g_5060.C, [113](#)
 - g_5060_Q.C, [114](#)
 - g_5060t.C, [114](#)
 - g_pp_n.C, [115](#)
 - g_pp_q (Case Conflict).C, [115](#)
 - g_pp_q.C, [116](#)
 - g_pp_Qt.C, [117](#)
 - g_pPb_03_Q.C, [117](#)
 - g_pPb_n21.C, [118](#)
 - g_pPb_n21_Q.C, [118](#)
 - g_pPb_n25.C, [119](#)
 - g_Q_2030.C, [119](#)
- PP
 - functions.h, [73](#)
- param
 - tr_his_c, [50](#)
- phi
 - nucleus, [43](#)
- phirot
 - functions.h, [72](#)
- phirot3
 - functions.h, [72](#)
- phirot4
 - functions.h, [72](#)
- phirot5
 - functions.h, [72](#)
- phirot6
 - functions.h, [73](#)
- phirot_minus
 - functions.h, [73](#)
- phirot_plus
 - functions.h, [73](#)
- phrot
 - distr, [29](#)
- phys
 - tr_his_c, [50](#)
- ppp
 - functions.h, [73](#)
- profile2
 - demo_2/profile2.C, [101](#)
 - profile2.C, [102](#)

- profile2.C
 - profile2, [102](#)
- profile2_deformation_63Cu
 - demo_2/profile2_deformation_63Cu.C, [102](#)
 - profile2_deformation_63Cu.C, [102](#)
- profile2_deformation_63Cu.C
 - profile2_deformation_63Cu, [102](#)
- profile2_deformation_Au
 - demo_2/profile2_deformation_Au.C, [103](#)
 - profile2_deformation_Au.C, [103](#)
- profile2_deformation_Au.C
 - profile2_deformation_Au, [103](#)
- profile2_deformation_U
 - demo_2/profile2_deformation_U.C, [104](#)
 - profile2_deformation_U.C, [104](#)
- profile2_deformation_U.C
 - profile2_deformation_U, [104](#)
- QSCALE
 - functions.h, [73](#)
- qx
 - distr, [30](#)
- qy
 - distr, [30](#)
- r
 - nucleus, [43](#)
- RAPRANGE
 - functions.h, [73](#)
- RAPSHIFT
 - functions.h, [73](#)
- RCHA
 - functions.h, [73](#)
- RCHB
 - functions.h, [73](#)
- RCHP
 - functions.h, [74](#)
- RDS0
 - functions.h, [74](#)
- RDS1
 - functions.h, [74](#)
- RET
 - functions.h, [74](#)
- RO
 - functions.h, [74](#)
- ROTA_PHI
 - functions.h, [74](#)
- ROTA_THETA
 - functions.h, [74](#)
- ROTB_PHI
 - functions.h, [74](#)
- ROTB_THETA
 - functions.h, [74](#)
- RWSA
 - functions.h, [75](#)
- RWSB
 - functions.h, [75](#)
- raa
 - col_chain_8_q.C, [87](#)
- g_pp_q.C, [116](#)
- glissando3.cxx, [80](#)
- rad_dis
 - rad_dis.C, [107](#)
- rad_dis.C
 - rad_dis, [107](#)
- radA
 - tr_his_c, [50](#)
- radA2
 - tr_his_c, [51](#)
- radA3
 - tr_his_c, [51](#)
- radB
 - tr_his_c, [51](#)
- rbin
 - functions.h, [73](#)
- rcostheta_nuclA
 - tr_his_c, [51](#)
- rcostheta_nuclB
 - tr_his_c, [51](#)
- rd2
 - collision, [12](#)
- readpar
 - functions.h, [64](#)
- reset
 - counter, [14](#)
 - counter2, [16](#)
 - counter_2D, [19](#)
- reset_counters
 - functions.h, [64](#)
- rhotspot
 - functions.h, [74](#)
- rlos_abf
 - functions.h, [64](#)
- rlos_alpha
 - functions.h, [64](#)
- rlos_hole
 - functions.h, [64](#)
- rlos_hult
 - functions.h, [64](#)
- rlos_parton
 - functions.h, [65](#)
- rlosA
 - functions.h, [65](#)
- rlosA_def
 - functions.h, [65](#)
- rlosA_hos
 - functions.h, [65](#)
- rlosB
 - functions.h, [65](#)
- rlosB_def
 - functions.h, [65](#)
- rlosB_hos
 - functions.h, [65](#)
- roo
 - functions.h, [74](#)
- rotate
 - distr, [30](#)

rotate_polar
 distr, 32
 rpa
 collision, 12
 functions.h, 74
 rrel_u
 tr_his_c, 51
 rrelA
 tr_his_c, 51
 rrelB
 tr_his_c, 51
 rwA
 functions.h, 75
 rwAB
 functions.h, 75
 rwB
 functions.h, 75

 SBIN
 functions.h, 75
 SCALEA
 functions.h, 75
 SCALEB
 functions.h, 75
 SHIFT
 functions.h, 75
 SIGETA
 functions.h, 75
 SIGMAA
 functions.h, 75
 SIGMAB
 functions.h, 75
 SIGMABISA
 functions.h, 76
 SIGMABISB
 functions.h, 76
 SNN
 functions.h, 76
 SOURCE, 43
 KK, 44
 W, 44
 X, 44
 Y, 44
 saxon
 demo_2/fitr.C, 95
 fitr.C, 96
 set_alpha_cluster
 nucleus, 39
 set_berillium_7_cluster
 nucleus, 39
 set_berillium_8_cluster
 nucleus, 39
 set_berillium_9_cluster
 nucleus, 39
 set_carbon_cluster
 nucleus, 39
 set_deuteron
 nucleus, 40
 set_file
 nucleus, 40
 set_file_uncor
 nucleus, 40
 set_oxygen_cluster
 nucleus, 40
 set_oxygen_cluster_square
 nucleus, 41
 set_parton_A
 nucleus, 41
 set_parton_B
 nucleus, 41
 set_proton
 nucleus, 41
 set_random_A
 nucleus, 41
 set_random_A_def
 nucleus, 42
 set_random_A_hos
 nucleus, 42
 set_random_B
 nucleus, 42
 set_random_B_def
 nucleus, 42
 set_random_B_hos
 nucleus, 43
 set_tritium
 nucleus, 43
 shift_cmx
 distr, 32
 shift_cmx_w
 distr, 32
 shift_cmx_w_c
 collision, 11
 shift_cmy
 distr, 32
 shift_cmy_w
 distr, 32
 shift_cmy_w_c
 collision, 11
 shift_cmz
 distr, 32
 shift_cmz_w
 distr, 32
 shift_x
 distr, 32
 shift_y
 distr, 34
 shift_z
 distr, 34
 siexp
 w_prof_norm_wb.C, 123
 sirad
 functions.h, 76
 sitot
 functions.h, 76
 size
 demo_2/size.C, 105
 distr, 34

- size.C, 105
- size.C
 - size, 105
- sizeav
 - functions.h, 76
- specA
 - collision, 12
- specB
 - collision, 12
- sth
 - nucleus, 43
- sum_w
 - distr, 34
- sumw
 - distr, 35
- surf
 - distr, 34
- surfA
 - functions.h, 76
- tilt
 - distr, 34
- tiltA
 - functions.h, 76
- tilted
 - tilted.C, 121
- tilted.C
 - tilted, 121
- time_start
 - functions.h, 65
- time_stop
 - functions.h, 66
- tr_his_c, 44
 - fill, 47
 - fill_res, 47
 - fill_tr, 47
 - full_event, 48
 - gen, 47
 - init, 47
 - nbinar, 48
 - nbinar2, 48
 - neps, 48
 - neps2, 48
 - neps2b, 48
 - neps4, 48
 - nepsb, 48
 - nensp, 48
 - nensp2, 49
 - nensp22, 49
 - nensp2b, 49
 - nensp4, 49
 - nepsb, 49
 - nsize, 49
 - nsize2, 49
 - ntarg, 49
 - ntarg2, 49
 - nuni, 49
 - nuniRDS, 49
 - nunib, 49
- nunp, 50
- nw2b, 50
- nwb, 50
- nwei, 50
- nwei2, 50
- nx, 50
- nx2, 50
- ny, 50
- ny2, 50
- param, 50
- phys, 50
- radA, 50
- radA2, 51
- radA3, 51
- radB, 51
- rcostheta_nuclA, 51
- rcostheta_nuclB, 51
- rrel_u, 51
- rrelA, 51
- rrelB, 51
- tree, 51
- weih, 51
- weih_bin, 51
- wpro, 51
- write, 47
- write_d, 47
- write_r, 48
- write_w, 48
- write_wpro, 48
- xyhist, 51
- xyhist_core, 52
- xyhist_mantle, 52
- xyhist_nuclA, 52
- xyhist_nuclB, 52
- xyhistr, 52
- tree
 - tr_his_c, 51
- Ubin
 - functions.h, 76
- uncluster
 - distr, 34
- Uw
 - functions.h, 76
- value
 - counter, 15
 - counter2, 17
- value2
 - counter2, 17
- valuex
 - counter_2D, 20
- valuex2
 - counter_2D, 20
- valuexy
 - counter_2D, 20
- valuey
 - counter_2D, 20
- valuey2

- counter_2D, 20
- var
 - counter2, 16
- var_x
 - counter_2D, 19
- var_y
 - counter_2D, 20
- vara
 - counter2, 16
- Vbin
 - functions.h, 76
- ver
 - glissando3.cxx, 80
- Vw
 - functions.h, 76
- W
 - SOURCE, 44
- w
 - distr, 35
- W0
 - functions.h, 77
- W1
 - functions.h, 77
- w_prof_norm
 - w_prof_norm.C, 121
- w_prof_norm.C
 - nn, 121
 - w_prof_norm, 121
- w_prof_norm_wb
 - w_prof_norm_wb.C, 123
- w_prof_norm_wb.C
 - gama, 123
 - nn, 123
 - nn_1, 123
 - omega, 123
 - siexp, 123
 - w_prof_norm_wb, 123
- w_prof_norm_wb0
 - w_prof_norm_wb0.C, 124
- w_prof_norm_wb0.C
 - gama, 124
 - nn, 124
 - nn_1, 124
 - omega, 124
 - w_prof_norm_wb0, 124
- WFA
 - functions.h, 77
- WFB
 - functions.h, 77
- WMIN
 - functions.h, 77
- wc
 - collision, 12
- weih
 - tr_his_c, 51
- weih_bin
 - tr_his_c, 51
- wfq
 - functions.h, 77
- wfqA
 - functions.h, 77
- wfqB
 - functions.h, 77
- wounding_profile
 - demo_2/wounding_profile.C, 106
 - wounding_profile.C, 106
- wounding_profile.C
 - wounding_profile, 106
- wpro
 - tr_his_c, 51
- write
 - tr_his_c, 47
- write_d
 - tr_his_c, 47
- write_r
 - tr_his_c, 48
- write_w
 - tr_his_c, 48
- write_wpro
 - tr_his_c, 48
- writerds
 - collision, 11
- wwA
 - collision, 12
- wwAB
 - collision, 13
- wwB
 - collision, 13
- X
 - SOURCE, 44
- x
 - distr, 35
- xcm
 - distr, 35
- xcmA
 - collision, 13
- xcmB
 - collision, 13
- xepp
 - functions.h, 77
- xsepp
 - functions.h, 77
- xx
 - functions.h, 77
- xyhist
 - tr_his_c, 51
- xyhist_core
 - tr_his_c, 52
- xyhist_mantle
 - tr_his_c, 52
- xyhist_nuclA
 - tr_his_c, 52
- xyhist_nuclB
 - tr_his_c, 52
- xyhistr
 - tr_his_c, 52

Y

SOURCE, [44](#)

y

distr, [35](#)

YB

functions.h, [77](#)

ycm

distr, [35](#)

ycmA

collision, [13](#)

ycmB

collision, [13](#)

ye

distr, [35](#)

ys

distr, [35](#)

yy

functions.h, [78](#)

z

distr, [36](#)

zcm

distr, [36](#)