

## GLISSANDO 3

Generated by Doxygen 1.8.6

Mon Nov 12 2018 10:36:01



# Contents

<b>1</b>	<b>Main Page</b>	<b>1</b>
<b>2</b>	<b>Hierarchical Index</b>	<b>3</b>
2.1	Class Hierarchy	3
<b>3</b>	<b>Class Index</b>	<b>5</b>
3.1	Class List	5
<b>4</b>	<b>File Index</b>	<b>7</b>
4.1	File List	7
<b>5</b>	<b>Class Documentation</b>	<b>9</b>
5.1	collision Class Reference	9
5.1.1	Detailed Description	10
5.1.2	Constructor & Destructor Documentation	10
5.1.2.1	collision	10
5.1.2.2	collision	10
5.1.2.3	collision	10
5.1.3	Member Function Documentation	10
5.1.3.1	gen_RDS	10
5.1.3.2	operator=	11
5.1.3.3	shift_cmx_w_c	11
5.1.3.4	shift_cmy_w_c	11
5.1.4	Member Data Documentation	11
5.1.4.1	nbin	11
5.1.4.2	nhotspot	11
5.1.4.3	nwA	11
5.1.4.4	nwAB	11
5.1.4.5	nwB	11
5.1.4.6	nzw	11
5.1.4.7	rd2	11
5.1.4.8	rpa	12
5.1.4.9	specA	12

5.1.4.10	specB	12
5.1.4.11	wc	12
5.1.4.12	wwA	12
5.1.4.13	wwAB	12
5.1.4.14	wwB	12
5.1.4.15	xcmA	12
5.1.4.16	xcmB	12
5.1.4.17	ycmA	12
5.1.4.18	ycmB	12
5.2	counter Class Reference	12
5.2.1	Detailed Description	13
5.2.2	Constructor & Destructor Documentation	13
5.2.2.1	counter	13
5.2.2.2	~counter	13
5.2.3	Member Function Documentation	13
5.2.3.1	add	13
5.2.3.2	get	14
5.2.3.3	getN	14
5.2.3.4	mean	14
5.2.3.5	reset	14
5.2.4	Member Data Documentation	14
5.2.4.1	count	14
5.2.4.2	value	14
5.3	counter2 Class Reference	14
5.3.1	Detailed Description	15
5.3.2	Constructor & Destructor Documentation	15
5.3.2.1	counter2	15
5.3.2.2	~counter2	15
5.3.3	Member Function Documentation	15
5.3.3.1	add	15
5.3.3.2	get	15
5.3.3.3	get2	16
5.3.3.4	getN	16
5.3.3.5	mean	16
5.3.3.6	reset	16
5.3.3.7	var	16
5.3.3.8	vara	16
5.3.4	Member Data Documentation	16
5.3.4.1	count	16
5.3.4.2	value	16

5.3.4.3	value2	16
5.4	counter_2D Class Reference	16
5.4.1	Detailed Description	18
5.4.2	Constructor & Destructor Documentation	18
5.4.2.1	counter_2D	18
5.4.2.2	~counter_2D	18
5.4.3	Member Function Documentation	18
5.4.3.1	add	18
5.4.3.2	corr	18
5.4.3.3	cov	18
5.4.3.4	get_x	18
5.4.3.5	get_x2	18
5.4.3.6	get_xy	18
5.4.3.7	get_y	18
5.4.3.8	get_y2	19
5.4.3.9	getN	19
5.4.3.10	mean_x	19
5.4.3.11	mean_y	19
5.4.3.12	reset	19
5.4.3.13	var_x	19
5.4.3.14	var_y	19
5.4.4	Member Data Documentation	19
5.4.4.1	count	19
5.4.4.2	valuex	19
5.4.4.3	valuex2	19
5.4.4.4	valuexy	19
5.4.4.5	valuey	19
5.4.4.6	valuey2	20
5.5	distr Class Reference	20
5.5.1	Detailed Description	22
5.5.2	Constructor & Destructor Documentation	22
5.5.2.1	distr	22
5.5.2.2	distr	22
5.5.2.3	distr	22
5.5.3	Member Function Documentation	23
5.5.3.1	cmx	23
5.5.3.2	cmx_w	23
5.5.3.3	cmy	23
5.5.3.4	cmy_w	23
5.5.3.5	cmz	23

5.5.3.6	cmz_w	23
5.5.3.7	eps	23
5.5.3.8	eps	23
5.5.3.9	eps	23
5.5.3.10	eps	24
5.5.3.11	fill_xy	24
5.5.3.12	fill_xy	24
5.5.3.13	msrad	24
5.5.3.14	msrad_t	24
5.5.3.15	msrad_t_w	24
5.5.3.16	msrad_w	25
5.5.3.17	msx	25
5.5.3.18	msy	25
5.5.3.19	mxy	25
5.5.3.20	operator=	25
5.5.3.21	phrot	25
5.5.3.22	phrot	25
5.5.3.23	qx	25
5.5.3.24	qy	26
5.5.3.25	rotate	26
5.5.3.26	rotate_polar	26
5.5.3.27	shift_cmx	26
5.5.3.28	shift_cmx_w	26
5.5.3.29	shift_cmy	26
5.5.3.30	shift_cmy_w	26
5.5.3.31	shift_cmz	26
5.5.3.32	shift_cmz_w	27
5.5.3.33	shift_x	27
5.5.3.34	shift_y	27
5.5.3.35	shift_z	27
5.5.3.36	size	27
5.5.3.37	sum_w	27
5.5.3.38	uncluster	27
5.5.4	Member Data Documentation	27
5.5.4.1	c	27
5.5.4.2	msr	27
5.5.4.3	msrt	28
5.5.4.4	n	28
5.5.4.5	sumw	28
5.5.4.6	w	28

5.5.4.7	x	28
5.5.4.8	xcm	28
5.5.4.9	y	28
5.5.4.10	ycm	28
5.5.4.11	z	28
5.5.4.12	zcm	28
5.6	nucleus Class Reference	29
5.6.1	Detailed Description	30
5.6.2	Constructor & Destructor Documentation	30
5.6.2.1	nucleus	30
5.6.2.2	nucleus	30
5.6.3	Member Function Documentation	31
5.6.3.1	dist2	31
5.6.3.2	good_all	31
5.6.3.3	good_down	31
5.6.3.4	good_pair	31
5.6.3.5	operator=	31
5.6.3.6	set_alpha_cluster	31
5.6.3.7	set_berillium_7_cluster	32
5.6.3.8	set_berillium_8_cluster	32
5.6.3.9	set_berillium_9_cluster	32
5.6.3.10	set_carbon_cluster	32
5.6.3.11	set_deuteron	32
5.6.3.12	set_deuteron_s0	32
5.6.3.13	set_deuteron_s1	33
5.6.3.14	set_file	33
5.6.3.15	set_file_uncor	33
5.6.3.16	set_oxygen_cluster	33
5.6.3.17	set_oxygen_cluster_square	33
5.6.3.18	set_parton_A	34
5.6.3.19	set_parton_B	34
5.6.3.20	set_proton	34
5.6.3.21	set_random_A	34
5.6.3.22	set_random_A	34
5.6.3.23	set_random_A_def	34
5.6.3.24	set_random_A_def	34
5.6.3.25	set_random_A_hos	35
5.6.3.26	set_random_A_hos	35
5.6.3.27	set_random_B	35
5.6.3.28	set_random_B	35

5.6.3.29	set_random_B_def . . . . .	35
5.6.3.30	set_random_B_def . . . . .	35
5.6.3.31	set_random_B_hos . . . . .	35
5.6.3.32	set_random_B_hos . . . . .	35
5.6.3.33	set_tritium . . . . .	36
5.6.4	Member Data Documentation . . . . .	36
5.6.4.1	cth . . . . .	36
5.6.4.2	g . . . . .	36
5.6.4.3	phi . . . . .	36
5.6.4.4	r . . . . .	36
5.6.4.5	sth . . . . .	36
5.7	SOURCE Struct Reference . . . . .	36
5.7.1	Detailed Description . . . . .	36
5.7.2	Member Data Documentation . . . . .	37
5.7.2.1	KK . . . . .	37
5.7.2.2	W . . . . .	37
5.7.2.3	X . . . . .	37
5.7.2.4	Y . . . . .	37
5.8	tr_his_c Class Reference . . . . .	37
5.8.1	Detailed Description . . . . .	39
5.8.2	Member Function Documentation . . . . .	39
5.8.2.1	fill . . . . .	39
5.8.2.2	fill_res . . . . .	39
5.8.2.3	fill_tr . . . . .	40
5.8.2.4	gen . . . . .	40
5.8.2.5	init . . . . .	40
5.8.2.6	write . . . . .	40
5.8.2.7	write_d . . . . .	40
5.8.2.8	write_r . . . . .	40
5.8.2.9	write_w . . . . .	40
5.8.2.10	write_wpro . . . . .	40
5.8.3	Member Data Documentation . . . . .	40
5.8.3.1	full_event . . . . .	40
5.8.3.2	nbinar . . . . .	40
5.8.3.3	nbinar2 . . . . .	41
5.8.3.4	nemsp . . . . .	41
5.8.3.5	nemsp2 . . . . .	41
5.8.3.6	nemsp4 . . . . .	41
5.8.3.7	nsize . . . . .	41
5.8.3.8	nsize2 . . . . .	41

5.8.3.9	ntarg	41
5.8.3.10	ntarg2	41
5.8.3.11	nuni	41
5.8.3.12	nunib	41
5.8.3.13	nuniRDS	41
5.8.3.14	nunp	41
5.8.3.15	nw2b	42
5.8.3.16	nwb	42
5.8.3.17	nwei	42
5.8.3.18	nwei2	42
5.8.3.19	nx	42
5.8.3.20	nx2	42
5.8.3.21	ny	42
5.8.3.22	ny2	42
5.8.3.23	param	42
5.8.3.24	phys	42
5.8.3.25	radA	42
5.8.3.26	radB	42
5.8.3.27	rcostheta_nuclA	43
5.8.3.28	rcostheta_nuclB	43
5.8.3.29	rrel_u	43
5.8.3.30	rrelA	43
5.8.3.31	rrelB	43
5.8.3.32	tree	43
5.8.3.33	weih	43
5.8.3.34	weih_bin	43
5.8.3.35	wpro	43
5.8.3.36	xyhist	43
5.8.3.37	xyhist_core	43
5.8.3.38	xyhist_mantle	43
5.8.3.39	xyhist_nuclA	44
5.8.3.40	xyhist_nuclB	44
5.8.3.41	xyhistr	44
<b>6</b>	<b>File Documentation</b>	<b>45</b>
6.1	build/include/collision.h File Reference	45
6.1.1	Detailed Description	45
6.2	build/include/counter.h File Reference	45
6.2.1	Detailed Description	45
6.3	build/include/distrib.h File Reference	45

6.3.1	Detailed Description	46
6.4	build/include/functions.h File Reference	46
6.4.1	Detailed Description	52
6.4.2	Function Documentation	52
6.4.2.1	disp	52
6.4.2.2	dist	52
6.4.2.3	echopar	53
6.4.2.4	epilog	53
6.4.2.5	fgama	53
6.4.2.6	fomega	53
6.4.2.7	fqscale	53
6.4.2.8	fsNN	53
6.4.2.9	fsQQ	53
6.4.2.10	gamgen	53
6.4.2.11	header	54
6.4.2.12	helper	54
6.4.2.13	los	54
6.4.2.14	los12	54
6.4.2.15	los32	54
6.4.2.16	negbin	54
6.4.2.17	readpar	54
6.4.2.18	reset_counters	54
6.4.2.19	rlos_abf	55
6.4.2.20	rlos_alpha	56
6.4.2.21	rlos_hole	56
6.4.2.22	rlos_hult	56
6.4.2.23	rlos_parton	56
6.4.2.24	rlosA	56
6.4.2.25	rlosA_def	56
6.4.2.26	rlosA_hos	56
6.4.2.27	rlosB	57
6.4.2.28	rlosB_def	57
6.4.2.29	rlosB_hos	57
6.4.2.30	time_start	57
6.4.2.31	time_stop	57
6.4.3	Variable Documentation	57
6.4.3.1	ALPHA	57
6.4.3.2	angles	57
6.4.3.3	ARANK	57
6.4.3.4	ARGC	57

6.4.3.5	AWSA	58
6.4.3.6	AWSB	58
6.4.3.7	b	58
6.4.3.8	BETA2A	58
6.4.3.9	BETA2B	58
6.4.3.10	BETA4A	58
6.4.3.11	BETA4B	58
6.4.3.12	BMAX	58
6.4.3.13	BMIN	58
6.4.3.14	BTOT	58
6.4.3.15	CD	58
6.4.3.16	CLUSTERS	58
6.4.3.17	d	58
6.4.3.18	DBIN	59
6.4.3.19	dbin	59
6.4.3.20	DOBIN	59
6.4.3.21	DS	59
6.4.3.22	DW	59
6.4.3.23	ECM	59
6.4.3.24	ep1s	59
6.4.3.25	ep3s	59
6.4.3.26	ep4s	59
6.4.3.27	ep5s	59
6.4.3.28	ep6s	59
6.4.3.29	epart	59
6.4.3.30	epart1	59
6.4.3.31	epart3	59
6.4.3.32	epart4	59
6.4.3.33	epart5	60
6.4.3.34	epart6	60
6.4.3.35	eps	60
6.4.3.36	ETA0	60
6.4.3.37	ETAM	60
6.4.3.38	evall	60
6.4.3.39	EVENTS	60
6.4.3.40	FBIN	60
6.4.3.41	FBRAP	60
6.4.3.42	FILES	60
6.4.3.43	FULL	60
6.4.3.44	GA	60

6.4.3.45	GAMA	60
6.4.3.46	ISEED	61
6.4.3.47	ISEED1	61
6.4.3.48	kk	61
6.4.3.49	MAXYRAP	61
6.4.3.50	MODEL	61
6.4.3.51	NBIN	61
6.4.3.52	nbinary	61
6.4.3.53	NCS	61
6.4.3.54	nhot	61
6.4.3.55	NNWP	61
6.4.3.56	NUMA	61
6.4.3.57	NUMB	61
6.4.3.58	NUMRAP	62
6.4.3.59	nweight	62
6.4.3.60	nwounded	62
6.4.3.61	OMEGA	62
6.4.3.62	PARTONS	62
6.4.3.63	phirot	62
6.4.3.64	phirot1	62
6.4.3.65	phirot3	62
6.4.3.66	phirot4	62
6.4.3.67	phirot5	62
6.4.3.68	phirot6	62
6.4.3.69	PI	62
6.4.3.70	PP	62
6.4.3.71	ppp	62
6.4.3.72	QSCALE	62
6.4.3.73	RAPRANGE	62
6.4.3.74	rbin	63
6.4.3.75	RCHA	63
6.4.3.76	RCHB	63
6.4.3.77	RCHP	63
6.4.3.78	RDS0	63
6.4.3.79	RDS1	63
6.4.3.80	RET	63
6.4.3.81	rhotspot	63
6.4.3.82	RO	63
6.4.3.83	roo	63
6.4.3.84	ROTA_PHI	63

6.4.3.85	ROTA_THETA	63
6.4.3.86	ROTB_PHI	64
6.4.3.87	ROTB_THETA	64
6.4.3.88	rpa	64
6.4.3.89	rwA	64
6.4.3.90	rwAB	64
6.4.3.91	rwB	64
6.4.3.92	RWSA	64
6.4.3.93	RWSB	64
6.4.3.94	SBIN	64
6.4.3.95	SCALEA	64
6.4.3.96	SCALEB	64
6.4.3.97	SHIFT	64
6.4.3.98	SIGETA	65
6.4.3.99	SIGMAA	65
6.4.3.100	SIGMAB	65
6.4.3.101	SIGMABISA	65
6.4.3.102	SIGMABISB	65
6.4.3.103	sirad	65
6.4.3.104	sitot	65
6.4.3.105	sizeav	65
6.4.3.106	SNN	65
6.4.3.107	Ubin	65
6.4.3.108	Uw	65
6.4.3.109	Vbin	65
6.4.3.110	Vw	66
6.4.3.111	W0	66
6.4.3.112	W1	66
6.4.3.113	WFA	66
6.4.3.114	WFB	66
6.4.3.115	wfq	66
6.4.3.116	wfqA	66
6.4.3.117	wfqB	66
6.4.3.118	WMIN	66
6.4.3.119	xepp	66
6.4.3.120	xsepp	66
6.4.3.121	xx	66
6.4.3.122	yy	67
6.5	build/src/glissando3.cxx File Reference	67
6.5.1	Detailed Description	67

6.5.2	Macro Definition Documentation	67
6.5.2.1	_VER_	67
6.5.3	Function Documentation	67
6.5.3.1	main	68
6.5.4	Variable Documentation	69
6.5.4.1	raa	69
6.5.4.2	ver	69
6.6	macro/demo_2/centrality2.C File Reference	69
6.6.1	Detailed Description	69
6.6.2	Function Documentation	69
6.6.2.1	centrality2	69
6.7	macro/demo_2/core_mantle.C File Reference	70
6.7.1	Detailed Description	70
6.7.2	Function Documentation	70
6.7.2.1	core_mantle	70
6.8	macro/demo_2/corr.C File Reference	70
6.8.1	Detailed Description	70
6.8.2	Function Documentation	70
6.8.2.1	corr	70
6.9	macro/demo_2/density.C File Reference	71
6.9.1	Detailed Description	71
6.9.2	Function Documentation	71
6.9.2.1	density	71
6.10	macro/demo_2/epsilon.C File Reference	71
6.10.1	Detailed Description	71
6.10.2	Function Documentation	72
6.10.2.1	epsilon	72
6.11	macro/demo_2/fourier.C File Reference	72
6.11.1	Detailed Description	72
6.11.2	Function Documentation	72
6.11.2.1	fourier	72
6.12	macro/demo_2/info.C File Reference	72
6.12.1	Detailed Description	72
6.12.2	Function Documentation	73
6.12.2.1	info	73
6.13	macro/demo_2/label.C File Reference	73
6.13.1	Function Documentation	73
6.13.1.1	label	73
6.13.1.2	label_fit	73
6.14	macro/demo_3/label.C File Reference	73

6.14.1	Function Documentation	74
6.14.1.1	label	74
6.14.1.2	label_fit	74
6.15	macro/demo_2/mult.C File Reference	74
6.15.1	Detailed Description	74
6.15.2	Function Documentation	74
6.15.2.1	mult	74
6.16	macro/demo_2/profile2_deformation_63Cu.C File Reference	74
6.16.1	Detailed Description	75
6.16.2	Function Documentation	75
6.16.2.1	profile2_deformation_63Cu	75
6.17	macro/demo_2/profile2_deformation_U.C File Reference	75
6.17.1	Detailed Description	75
6.17.2	Function Documentation	75
6.17.2.1	profile2_deformation_U	75
6.18	macro/demo_2/size.C File Reference	76
6.18.1	Detailed Description	76
6.18.2	Function Documentation	76
6.18.2.1	size	76
6.19	macro/demo_2/wounding_profile.C File Reference	76
6.19.1	Detailed Description	76
6.19.2	Function Documentation	76
6.19.2.1	wounding_profile	76
6.20	macro/demo_3/b_dist.C File Reference	77
6.20.1	Function Documentation	77
6.20.1.1	b_dist	77
6.21	macro/demo_3/b_distr.C File Reference	77
6.21.1	Function Documentation	77
6.21.1.1	b_distr	77
6.22	macro/demo_3/cent.C File Reference	77
6.22.1	Detailed Description	78
6.22.2	Function Documentation	78
6.22.2.1	cent	78
6.23	macro/demo_3/cent_comp.C File Reference	78
6.23.1	Function Documentation	78
6.23.1.1	cent_comp	78
6.24	macro/demo_3/clusters.C File Reference	78
6.24.1	Function Documentation	78
6.24.1.1	clusters	78
6.25	macro/demo_3/eps_dist.C File Reference	79

6.25.1	Function Documentation	79
6.25.1.1	eps_dist	79
6.26	macro/demo_3/epsilon_c.C File Reference	79
6.26.1	Detailed Description	79
6.26.2	Function Documentation	79
6.26.2.1	epsilon_c	79
6.27	macro/demo_3/fourier_Q.C File Reference	79
6.27.1	Function Documentation	80
6.27.1.1	fourier_Q	80
6.28	macro/demo_3/inel_prof.C File Reference	80
6.28.1	Function Documentation	80
6.28.1.1	inel_prof	80
6.29	macro/demo_3/mult_pPb_Q.C File Reference	80
6.29.1	Function Documentation	80
6.29.1.1	mult_pPb_Q	80
6.30	macro/demo_3/mult_Q.C File Reference	81
6.30.1	Function Documentation	81
6.30.1.1	mult_Q	81
6.31	macro/demo_3/part_dist_pp.C File Reference	81
6.31.1	Function Documentation	81
6.31.1.1	part_dist_pp	81
6.32	macro/demo_3/rad_dis.C File Reference	81
6.32.1	Detailed Description	82
6.32.2	Function Documentation	82
6.32.2.1	rad_dis	82
6.33	macro/demo_3/sc_comp.C File Reference	82
6.33.1	Detailed Description	82
6.33.2	Function Documentation	82
6.33.2.1	sc_comp	82
6.34	macro/demo_3/w_prof_norm_AA.C File Reference	82
6.34.1	Function Documentation	83
6.34.1.1	w_prof_norm_AA	83
6.35	macro/demo_3/w_prof_norm_pp.C File Reference	84
6.35.1	Function Documentation	84
6.35.1.1	nn	84
6.35.1.2	nn_1	84
6.35.1.3	w_prof_norm_pp	84

# Chapter 1

## Main Page

GLISSANDO 3 - GLauber Initial State Simulation AND mOre... ver. 3.110, 28 October 2018

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

Authors:

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

counter . . . . .	12
counter2 . . . . .	14
counter_2D . . . . .	16
distr . . . . .	20
collision . . . . .	9
nucleus . . . . .	29
SOURCE . . . . .	36
tr_his_c . . . . .	37



## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">collision</a>	Collision class . . . . .	9
<a href="#">counter</a>	Simplest counting class . . . . .	12
<a href="#">counter2</a>	Counting class with variance . . . . .	14
<a href="#">counter_2D</a>	2-dimensional counting class . . . . .	16
<a href="#">distr</a>	Distribution of sources in space . . . . .	20
<a href="#">nucleus</a>	Nucleus class . . . . .	29
<a href="#">SOURCE</a>	Structure for output of the full event - transverse coordinates, weight, number of the event . . .	36
<a href="#">tr_his_c</a>	Class storing the trees and histograms . . . . .	37



## Chapter 4

# File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

build/include/collision.h	45
build/include/counter.h	45
build/include/distrib.h	45
build/include/functions.h	46
build/src/glissando3.cxx	67
macro/demo_2/centrality2.C	69
macro/demo_2/core_mantle.C	70
macro/demo_2/corr.C	70
macro/demo_2/density.C	71
macro/demo_2/epsilon.C	71
macro/demo_2/fourier.C	72
macro/demo_2/info.C	72
macro/demo_2/label.C	73
macro/demo_2/mult.C	74
macro/demo_2/profile2_deformation_63Cu.C	74
macro/demo_2/profile2_deformation_U.C	75
macro/demo_2/size.C	76
macro/demo_2/wounding_profile.C	76
macro/demo_3/b_dist.C	77
macro/demo_3/b_distr.C	77
macro/demo_3/cent.C	77
macro/demo_3/cent_comp.C	78
macro/demo_3/clusters.C	78
macro/demo_3/eps_dist.C	79
macro/demo_3/epsilon_c.C	79
macro/demo_3/fourier_Q.C	79
macro/demo_3/inel_prof.C	80
macro/demo_3/label.C	73
macro/demo_3/mult_pPb_Q.C	80
macro/demo_3/mult_Q.C	81
macro/demo_3/part_dist_pp.C	81
macro/demo_3/rad_dis.C	81
macro/demo_3/sc_comp.C	82
macro/demo_3/w_prof_norm_AA.C	82
macro/demo_3/w_prof_norm_pp.C	84



## Chapter 5

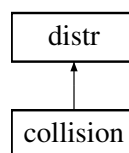
# Class Documentation

### 5.1 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 spactator coordinates*
- `void shift_cmy_w_c` ()

#### 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](#)

### 5.1.1 Detailed Description

collision class

Class performing the collision of two nuclei.

### 5.1.2 Constructor & Destructor Documentation

#### 5.1.2.1 `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

#### 5.1.2.2 `collision::collision ( )` [\[inline\]](#)

default constructor

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

#### 5.1.2.3 `collision::collision ( const collision & w1 )` [\[inline\]](#)

copying constructor

### 5.1.3 Member Function Documentation

#### 5.1.3.1 `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)

**5.1.3.2** `collision& collision::operator= ( const collision & w1 )` `[inline]`

substitution overloading

**5.1.3.3** `void collision::shift_cmx_w_c ( )` `[inline]`

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

**5.1.3.4** `void collision::shift_cmy_w_c ( )` `[inline]`

## 5.1.4 Member Data Documentation

**5.1.4.1** `Int_t collision::nbin`

number of binary collisions in the event

**5.1.4.2** `Int_t collision::nhotspot`

number of hot spots

**5.1.4.3** `Int_t collision::nWA`

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

**5.1.4.4** `Int_t collision::nWAB`

total number of wounded objects ( $nWA+nWB$ ) generated in the event

**5.1.4.5** `Int_t collision::nWB`

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

**5.1.4.6** `Int_t collision::nzw`

number of sources with non-zero weight

**5.1.4.7** `Float_t collision::rd2`

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

**5.1.4.8 Float\_t collision::rpa**

weight of the source (RDS)

**5.1.4.9 Int\_t collision::specA**

number of spectators in A

**5.1.4.10 Int\_t collision::specB**

number of spectators in B

**5.1.4.11 Int\_t collision::wc**

counter of the sources

**5.1.4.12 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

**5.1.4.13 Int\_t\* collision::wwAB****5.1.4.14 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

**5.1.4.15 Float\_t collision::xcmA**

x cm cordinate of the spectators from the A nucleus

**5.1.4.16 Float\_t collision::xcmB**

x cm cordinate of the spectators from the B nucleus

**5.1.4.17 Float\_t collision::ycmA**

y cm cordinate of the spectators from the A nucleus

**5.1.4.18 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

## 5.2 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*

### 5.2.1 Detailed Description

simplest counting class

Class counting the mean.

### 5.2.2 Constructor & Destructor Documentation

#### 5.2.2.1 `counter::counter()` `[inline]`

constructor

#### 5.2.2.2 `counter::~~counter()` `[inline]`

destructor

### 5.2.3 Member Function Documentation

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

add entry to the counter

Parameters

---

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

#### 5.2.3.2 `Double_t counter::get ( )` [inline]

get the sum of values

#### 5.2.3.3 `Int_t counter::getN ( )` [inline]

get the number of entries

#### 5.2.3.4 `Double_t counter::mean ( )` [inline]

get the mean value

#### 5.2.3.5 `void counter::reset ( )` [inline]

reset the counter

### 5.2.4 Member Data Documentation

#### 5.2.4.1 `Int_t counter::count` [private]

number of entries

#### 5.2.4.2 `Double_t counter::value` [private]

sum of values counted

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

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

## 5.3 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*

### 5.3.1 Detailed Description

counting class with variance

Class counting the mean and variance.

### 5.3.2 Constructor & Destructor Documentation

#### 5.3.2.1 counter2::counter2 ( ) [inline]

constructor

#### 5.3.2.2 counter2::~~counter2 ( ) [inline]

destructor

### 5.3.3 Member Function Documentation

#### 5.3.3.1 void counter2::add ( Double\_t s ) [inline]

add entry

Parameters

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

#### 5.3.3.2 Double\_t counter2::get ( ) [inline]

get the sum of values

**5.3.3.3** `Double_t counter2::get2 ( ) [inline]`

get the sum of squares of values

**5.3.3.4** `Int_t counter2::getN ( ) [inline]`

get the number of entries

**5.3.3.5** `Double_t counter2::mean ( ) [inline]`

get the mean value

**5.3.3.6** `void counter2::reset ( ) [inline]`

reset the counter

**5.3.3.7** `Double_t counter2::var ( ) [inline]`

get the variance

**5.3.3.8** `Double_t counter2::vara ( ) [inline]`

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

## 5.3.4 Member Data Documentation

**5.3.4.1** `Int_t counter2::count [private]`

number of entries

**5.3.4.2** `Double_t counter2::value [private]`

sum of values

**5.3.4.3** `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](#)

## 5.4 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 x 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  $x*y$*

### 5.4.1 Detailed Description

2-dimensional counting class

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

### 5.4.2 Constructor & Destructor Documentation

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

constructor

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

destructor

### 5.4.3 Member Function Documentation

5.4.3.1 `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

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

get the Pearson correlation coefficient

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

get the covariance

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

get the sum of x values

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

get the sum of x<sup>2</sup>

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

get the sum of xy

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

get the sum of y values

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

get the sum of y2

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

get the number of entries

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

get the mean x value

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

get the mean y value

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

reset the counter

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

get the variance of x

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

get the variance of y

## 5.4.4 Member Data Documentation

5.4.4.1 `Int_t counter_2D::count [private]`

number of entries

5.4.4.2 `Double_t counter_2D::valuex [private]`

sum of x values

5.4.4.3 `Double_t counter_2D::valuex2 [private]`

sum of  $x^2$

5.4.4.4 `Double_t counter_2D::valuexy [private]`

sum of  $x*y$

5.4.4.5 `Double_t counter_2D::valuey [private]`

sum of y values

#### 5.4.4.6 Double\_t counter\_2D::valuey2 [private]

sum of  $y^2$

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

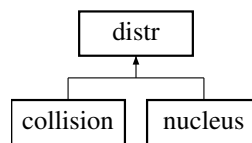
- [build/include/counter.h](#)

## 5.5 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 [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](#) ()  
*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](#) ()  
*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 [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](#) (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](#) (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 [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 [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*
- 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*
- [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*
- `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*

### 5.5.1 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.

### 5.5.2 Constructor & Destructor Documentation

#### 5.5.2.1 `distr::distr( Int_t k )` `[inline]`

constructor

Parameters

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

#### 5.5.2.2 `distr::distr( void )` `[inline]`

default constructor, 208 sources

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

#### 5.5.2.3 `distr::distr( const distr & w1 )` `[inline]`

copying constructor

### 5.5.3 Member Function Documentation

#### 5.5.3.1 `Float_t distr::cmx ( ) [inline]`

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

#### 5.5.3.2 `Float_t distr::cmx_w ( ) [inline]`

generate the center-of-mass x coordinate with weights

#### 5.5.3.3 `Float_t distr::cmx ( ) [inline]`

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

#### 5.5.3.4 `Float_t distr::cmx_w ( ) [inline]`

generate the center-of-mass y coordinate with weights

#### 5.5.3.5 `Float_t distr::cmz ( ) [inline]`

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

#### 5.5.3.6 `Float_t distr::cmz_w ( ) [inline]`

generate the center-of-mass z coordinate with weights

#### 5.5.3.7 `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

#### 5.5.3.8 `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,...
----------	---

#### 5.5.3.9 `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,...
<i>k</i>	power of transverse radius in the weight

5.5.3.10 `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

5.5.3.11 `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

5.5.3.12 `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

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

mean squared radius, no weights

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

mean squared transverse radius, no weights

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

mean squared transverse radius with weights

**5.5.3.16** `Float_t distr::msrad_w( ) [inline]`

mean squared radius with weights

**5.5.3.17** `Float_t distr::msx( ) [inline]`

mean squared x, no weights

**5.5.3.18** `Float_t distr::msy( ) [inline]`

mean squared y, no weights

**5.5.3.19** `Float_t distr::mxy( ) [inline]`

mean x\*y, no weights

**5.5.3.20** `distr & distr::operator=( const distr & w1 )`

substitution overloading

**5.5.3.21** `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,...
----------	---

**5.5.3.22** `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,...
<i>k</i>	power of transverse radius in the weight

**5.5.3.23** `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. See the Glissando 3 paper for details

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>d</i>	smearing parameter

#### 5.5.3.24 `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  $k=3$ . See the Glissando 3 paper for details

##### 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

#### 5.5.3.25 `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
-----------	--

#### 5.5.3.26 `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
--------------	--

#### 5.5.3.27 `void distr::shift_cmx ( ) [inline]`

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

#### 5.5.3.28 `void distr::shift_cmx_w ( ) [inline]`

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

#### 5.5.3.29 `void distr::shift_cmy ( ) [inline]`

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

#### 5.5.3.30 `void distr::shift_cmy_w ( ) [inline]`

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

#### 5.5.3.31 `void distr::shift_cmz ( ) [inline]`

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

**5.5.3.32** `void distr::shift_cmz_w( )` `[inline]`

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

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

translate in the x direction

Parameters

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

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

translate in the y direction

Parameters

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

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

translate in the z direction

Parameters

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

**5.5.3.36** `Float_t distr::size( )` `[inline]`

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

**5.5.3.37** `Float_t distr::sum_w( )` `[inline]`

destructor

generate sum of weights

**5.5.3.38** `void distr::uncluster( )` `[inline]`

uncluster

## 5.5.4 Member Data Documentation

**5.5.4.1** `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.

**5.5.4.2** `Float_t distr::msr`

mean squared radius of the distribution

#### 5.5.4.3 `Float_t distr::msrt`

mean squared transverse radius of the distribution

#### 5.5.4.4 `Int_t distr::n`

number of sources (points)

#### 5.5.4.5 `Float_t distr::sumw`

sum of weights of the distribution

#### 5.5.4.6 `Float_t* distr::w`

weight

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

#### 5.5.4.7 `Float_t* distr::x`

x space coordinate

#### 5.5.4.8 `Float_t distr::xcm`

center-of-mass x coordinate of the distribution

#### 5.5.4.9 `Float_t* distr::y`

y space coordinate

#### 5.5.4.10 `Float_t distr::ycm`

center-of-mass y coordinate of the distribution

#### 5.5.4.11 `Float_t* distr::z`

z space coordinate

#### 5.5.4.12 `Float_t distr::zcm`

center-of-mass z coordinate of the distribution

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

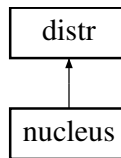
- [build/include/distrib.h](#)

## 5.6 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\\_deuteron\\_s1](#) ()
- void [set\\_deuteron\\_s0](#) ()
- 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

### 5.6.1 Detailed Description

nucleus class

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

### 5.6.2 Constructor & Destructor Documentation

#### 5.6.2.1 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
----------	--

#### 5.6.2.2 nucleus::nucleus ( const nucleus & w1 ) [inline]

copying constructor

### 5.6.3 Member Function Documentation

#### 5.6.3.1 `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

#### 5.6.3.2 `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
----------	--------------------

#### 5.6.3.3 `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

#### 5.6.3.4 `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

#### 5.6.3.5 `nucleus& nucleus::operator= ( const nucleus & w1 ) [inline]`

substitution overloading

#### 5.6.3.6 `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

5.6.3.7 `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

5.6.3.8 `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

5.6.3.9 `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

5.6.3.10 `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

5.6.3.11 `void nucleus::set_deuteron ( )` `[inline]`

set the distribution for the deuteron

Use the Hulthen distribution.

5.6.3.12 `void nucleus::set_deuteron_s0 ( )` `[inline]`

#### 5.6.3.13 void nucleus::set\_deuteron\_s1 ( ) [inline]

set the distributions of the polarized deuteron

#### 5.6.3.14 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

#### 5.6.3.15 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

#### 5.6.3.16 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

#### 5.6.3.17 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

**5.6.3.18** `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

**5.6.3.19** `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

**5.6.3.20** `void nucleus::set_proton ( )` `[inline]`

set the distribution for the proton

The proton is just placed at the origin

**5.6.3.21** `void nucleus::set_random_A ( )` `[inline]`

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

**5.6.3.22** `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

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

**5.6.3.23** `void nucleus::set_random_A_def ( )` `[inline]`

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

**5.6.3.24** `void nucleus::set_random_A_def ( Float_t d )` `[inline]`

## Parameters

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

**5.6.3.25** `void nucleus::set_random_A_hos ( ) [inline]`

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

**5.6.3.26** `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 $d$
-----	--

**5.6.3.27** `void nucleus::set_random_B ( ) [inline]`

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

**5.6.3.28** `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 $d$
-----	--

**5.6.3.29** `void nucleus::set_random_B_def ( ) [inline]`

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

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

## Parameters

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

**5.6.3.31** `void nucleus::set_random_B_hos ( ) [inline]`

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

**5.6.3.32** `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
----------	--

5.6.3.33 void nucleus::set\_tritium ( Float\_t *scale* ) [inline]

set triangular tritium

## Parameters

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

## 5.6.4 Member Data Documentation

5.6.4.1 Float\_t nucleus::cth [private]

5.6.4.2 bool nucleus::g [private]

5.6.4.3 Float\_t nucleus::phi [private]

5.6.4.4 Float\_t nucleus::r [private]

5.6.4.5 Float\_t nucleus::sth [private]

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

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

## 5.7 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*

## 5.7.1 Detailed Description

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

## 5.7.2 Member Data Documentation

### 5.7.2.1 UInt\_t SOURCE::KK

number of the event

### 5.7.2.2 Float\_t SOURCE::W

weight

### 5.7.2.3 Float\_t SOURCE::X

x coordinate

### 5.7.2.4 Float\_t SOURCE::Y

y coordinate

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

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

## 5.8 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 participant-plane distribution*
- TH1D \* [nx](#)  
*center-of-mass x coordinate of the source distribution vs. Nw*
- TH1D \* [nx2](#)  
*square of cm x coordinate, then its variance, vs. Nw*
- TH1D \* [ny](#)  
*center-of-mass y coordinate of the source distribution vs. Nw*
- TH1D \* [ny2](#)  
*square of cm y coordinate, then its variance, vs. Nw*
- TH1D \* [nsize](#)  
*size vs. Nw*
- TH1D \* [nsize2](#)  
*square of size, then its variance/size<sup>2</sup>, vs. Nw*
- TH1D \* [nepsp](#)  
*participant-plane eccentricity vs. Nw*
- TH1D \* [nepsp2](#)  
*square of participant-plane eccentricity, then its variance, vs. Nw*
- TH1D \* [nepsp4](#)  
*participant-plane fourth moment vs. Nw*
- TH1D \* [nuni](#)  
*frequency of Nw, i.e. histogram of unity vs. Nw*
- TH1D \* [nuniRDS](#)  
*frequency of RDS, i.e. histogram of unity vs. RDS*
- 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 \* [nump](#)  
*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*

### 5.8.1 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

### 5.8.2 Member Function Documentation

#### 5.8.2.1 void tr\_his\_c::fill ( ) [inline]

fill trees param and phys

#### 5.8.2.2 void tr\_his\_c::fill\_res ( ) [inline]

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

**5.8.2.3 void tr\_his\_c::fill\_tr ( ) [inline]**

fill the main tree

**5.8.2.4 void tr\_his\_c::gen ( ) [inline]**

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

**5.8.2.5 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

**5.8.2.6 void tr\_his\_c::write ( ) [inline]**

write out trees param, phys, full\_event, and the main tree

**5.8.2.7 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

**5.8.2.8 void tr\_his\_c::write\_r ( ) [inline]**

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

**5.8.2.9 void tr\_his\_c::write\_w ( ) [inline]**

write out the overlaid distributions

**5.8.2.10 void tr\_his\_c::write\_wpro ( ) [inline]**

write out the wounding profile

**5.8.3 Member Data Documentation****5.8.3.1 TTree \* tr\_his\_c::full\_event**

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

**5.8.3.2 TH1D \* tr\_his\_c::nbinar**

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

**5.8.3.3 TH1D \* tr\_his\_c::nbinar2**

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

**5.8.3.4 TH1D \* tr\_his\_c::nepsp**

participant-plane eccentricity vs. Nw

**5.8.3.5 TH1D \* tr\_his\_c::nepsp2**

square of participant-plane eccentricity, then its variance, vs. Nw

**5.8.3.6 TH1D \* tr\_his\_c::nepsp4**

participant-plane fourth moment vs. Nw

**5.8.3.7 TH1D \* tr\_his\_c::nsize**

size vs. Nw

**5.8.3.8 TH1D \* tr\_his\_c::nsize2**

square of size, then its variance/size<sup>2</sup>, vs. Nw

**5.8.3.9 TH1D \* tr\_his\_c::ntarg**

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

**5.8.3.10 TH1D \* tr\_his\_c::ntarg2**

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

**5.8.3.11 TH1D \* tr\_his\_c::nuni**

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

**5.8.3.12 TH1D \* tr\_his\_c::nunib**

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

**5.8.3.13 TH1D \* tr\_his\_c::nuniRDS**

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

**5.8.3.14 TH1D \* tr\_his\_c::nunp**

frequency of the number of wounded objects in nucleus A

**5.8.3.15 TH1D \* tr\_his\_c::nw2b**

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

**5.8.3.16 TH1D \* tr\_his\_c::nwb**

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

**5.8.3.17 TH1D\* tr\_his\_c::nwei**

RDS vs. number of wounded objects in nucleus A.

**5.8.3.18 TH1D \* tr\_his\_c::nwei2**

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

**5.8.3.19 TH1D\* tr\_his\_c::nx**

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

**5.8.3.20 TH1D \* tr\_his\_c::nx2**

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

**5.8.3.21 TH1D \* tr\_his\_c::ny**

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

**5.8.3.22 TH1D \* tr\_his\_c::ny2**

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

**5.8.3.23 TTree\* tr\_his\_c::param**

parameters

**5.8.3.24 TTree \* tr\_his\_c::phys**

A+B cross section and other physical results.

**5.8.3.25 TH1D\* tr\_his\_c::radA**

one-body radial distribution in the nucleus A

**5.8.3.26 TH1D \* tr\_his\_c::radB**

one-body radial distribution in the nucleus B

**5.8.3.27 TH2D \* tr\_his\_c::rcostheta\_nuclA**

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

**5.8.3.28 TH2D \* tr\_his\_c::rcostheta\_nuclB**

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

**5.8.3.29 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)

**5.8.3.30 TH1D \* tr\_his\_c::rrelA**

distance between the pair of objects in the nucleus A

**5.8.3.31 TH1D \* tr\_his\_c::rrelB**

distance between the pair of objects in the nucleus B

**5.8.3.32 TTree \* tr\_his\_c::tree**

basic physical results

**5.8.3.33 TH1D \* tr\_his\_c::weih**

the distribution overlaid on the wounded objects

**5.8.3.34 TH1D \* tr\_his\_c::weih\_bin**

the distribution overlaid over binary collisions

**5.8.3.35 TH1D \* tr\_his\_c::wpro**

the wounding profile

**5.8.3.36 TH2D \* tr\_his\_c::xyhist**

cartesian fixed-axes distribution

**5.8.3.37 TH2D \* tr\_his\_c::xyhist\_core**

cartesian fixed-axes core distribution

**5.8.3.38 TH2D \* tr\_his\_c::xyhist\_mantle**

cartesian fixed-axes mantle distribution

**5.8.3.39 TH2D \* tr\_his\_c::xyhist\_nuclA**

cartesian fixed-axes distribution of density in nucleus A

**5.8.3.40 TH2D \* tr\_his\_c::xyhist\_nuclB**

cartesian fixed-axes distribution of density in nucleus B

**5.8.3.41 TH2D \* tr\_his\_c::xyhistr**

cartesian participant-plane 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*
- 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*
- Double\_t [fgama](#) (Double\_t ecm)
- Double\_t [fomega](#) (Double\_t ecm)
- 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 `los32` ()  
*random number generator for spin (3/2,3/2) projection*
- Float\_t `los12` ()  
*random number generator for spin (3/2,1/2) projection*
- 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 `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 **CLUSTERS**
- Int\_t **ARGC**
- 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** =-1.  
*Woods-Saxon radius for nucleus A.*
- Float\_t **AWSA** =0.459  
*Woods-Saxon width for nucleus A.*
- Float\_t **BETA2A** =-1  
*Deformation parameter beta2 of deformed Woods-Saxon distribution for nucleus A.*
- Float\_t **BETA4A** =-1  
*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** =-1.  
*Woods-Saxon radius for nucleus B.*
- Float\_t **AWSB** =0.459  
*Woods-Saxon width for nucleus B.*
- Float\_t **BETA2B** =-1  
*Deformation parameter beta2 of deformed Woods-Saxon distribution for nucleus B.*
- Float\_t **BETA4B** =-1  
*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** =-1  
*NN "wounding" cross section in milibarns.*
- Float\_t **SBIN** =-1  
*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** =1.  
*Poisson or Gamma parameters for superimposed distribution, wounded nucleons.*
- Float\_t **Ubin** =1.  
*Poisson or Gamma parameters for superimposed distribution, binary collisions.*
- Float\_t **Vw** =2.  
*Negative binomial variance, wounded nucleons.*
- Float\_t **Vbin** =2.  
*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** =1
  - 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** =-1
  - relative variance of cross-section fluctuations*
- Float\_t **GAMA** =-1
  - gamma wounding profile parameter (height at the origin)*
- Float\_t **SCALEA** =1.
  - scale parameter for the size of the nucleus (cluster version)*
- Float\_t **SIGMAA** =1.
  - standard deviation of x,y,z coordinates of nucleons in the alpha cluster*
- Float\_t **SIGMABISA** =1.
  - standard deviation of x,y,z coordinates of nucleons in the 3He cluster or nucleon no. 9*
- Float\_t **SCALEB** =1.
  - scale parameter for the size of the nucleus (cluster version)*
- Float\_t **SIGMAB** =1.
  - standard deviation of x,y,z coordinates of nucleons in the alpha cluster*
- Float\_t **SIGMABISB** =1.
  - standard deviation of x,y,z coordinates of nucleons in the 3He cluster or nucleon no. 9*
- Float\_t **QSCALE** =-1
  - scale parameter in the parton distribution function,  $f(r)=r^{\wedge}2*\exp(-r/QSCALE)$ ,  $QSCALE=1/(4.27/\text{Sqrt}(3/2))$*
- Float\_t **DS** =-1
  - source smearing parameter*
- Float\_t **ECM** =-1.
  - center of mass energy of the colliding system [GeV]*
- **counter2 epart1**
  - counter for participant-plane  $\langle r^{\wedge}3 \cos(\phi) \rangle / \langle r^{\wedge}3 \rangle$*

- [counter2 epart](#)  
*counter for participant-plane  $\langle r^n \cos(2 \phi) \rangle / \langle r^n \rangle$*
- [counter2 epart3](#)  
*counter for participant-plane  $\langle r^n \cos(3 \phi) \rangle / \langle r^n \rangle$*
- [counter2 epart4](#)  
*counter for participant-plane  $\langle r^n \cos(4 \phi) \rangle / \langle r^n \rangle$*
- [counter2 epart5](#)  
*counter for participant-plane  $\langle r^n \cos(5 \phi) \rangle / \langle r^n \rangle$*
- [counter2 epart6](#)  
*counter for participant-plane  $\langle r^n \cos(6 \phi) \rangle / \langle r^n \rangle$*
- [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 [ep1s](#)
- participant-plane eccentricities*
- Float\_t [eps](#)
- Float\_t [ep3s](#)
- Float\_t [ep4s](#)
- Float\_t [ep5s](#)
- Float\_t [ep6s](#)
- Float\_t [phirot1](#)
- angles of the principal axis*
- Float\_t [phirot](#)
- Float\_t [phirot3](#)
- Float\_t [phirot4](#)
- Float\_t [phirot5](#)
- Float\_t [phirot6](#)
- 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

$w$	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

set the deformation parameters

## 6.4.2.4 void epilog ( )

print epilog to the output

6.4.2.5 Double\_t fgama ( Double\_t *ecm* )6.4.2.6 Double\_t fomega ( Double\_t *ecm* )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 $f(x) = x^{(a-1)} \exp(-x)/\text{Gamma}(a)$
----------	--

## 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 los12 ( )

random number generator for spin (3/2,1/2) projection

## 6.4.2.15 Float\_t los32 ( )

random number generator for spin (3/2,3/2) projection

6.4.2.16 Int\_t negbin ( Double\_t *m*, Double\_t *v* )

random number generator for the negative binomial distribution

6.4.2.17 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
---------------	------------------------

## 6.4.2.18 void reset\_counters ( )

reset the counters used to store physical quantities in the event

#### 6.4.2.19 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

6.4.2.20 `Float_t rlos_alpha ( Float_t s, Float_t bp, Float_t dp )`

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

6.4.2.21 `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
----------	----------------------

6.4.2.22 `Float_t rlos_hult ( )`

random number generator for the Hulthen distribution

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

6.4.2.23 `Float_t rlos_parton ( )`

random radial coordinate of a parton in the nucleon

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

6.4.2.24 `Float_t rlosA ( )`

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

6.4.2.25 `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

6.4.2.26 `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 ( $2 < N_{\text{A}} < 17$ ) nuclei (Nuclear Sizes. L. R. B. Elton, Oxford University Press, New York, 1961.)

**6.4.2.27 Float\_t rlosB ( )**

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

**6.4.2.28 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

**6.4.2.29 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.)

**6.4.2.30 Int\_t time\_start ( )**

start the time measurement and print the stamp

**6.4.2.31 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 Int\_t ARGC**

**6.4.3.5 Float\_t AWSA =0.459**

Woods-Saxon width for nucleus A.

**6.4.3.6 Float\_t AWSB =0.459**

Woods-Saxon width for nucleus B.

**6.4.3.7 Float\_t b**

impact parameter

**6.4.3.8 Float\_t BETA2A =-1**

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

**6.4.3.9 Float\_t BETA2B =-1**

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

**6.4.3.10 Float\_t BETA4A =-1**

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

**6.4.3.11 Float\_t BETA4B =-1**

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

**6.4.3.12 Float\_t BMAX =25.**

maximum value of the impact parameter in the acceptance window

**6.4.3.13 Float\_t BMIN =0.**

minimum value of the impact parameter in the acceptance window

**6.4.3.14 Float\_t BTOT = fmax(RWSA,RWSB)+AWSA+AWSB**

maximum coordinate value for some histograms

**6.4.3.15 Float\_t CD =0.9**

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

**6.4.3.16 Int\_t CLUSTERS****6.4.3.17 Float\_t d**

the wounding distance

**6.4.3.18 Float\_t DBIN =0.**

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

**6.4.3.19 Float\_t dbin**

the binary-collision distance

**6.4.3.20 Int\_t DOBIN =0**

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

**6.4.3.21 Float\_t DS =-1**

source smearing parameter

**6.4.3.22 Float\_t DW =0.**

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

**6.4.3.23 Float\_t ECM =-1.**

center of mass energy of the colliding system [GeV]

**6.4.3.24 Float\_t ep1s**

participant-plane eccentricities

**6.4.3.25 Float\_t ep3s****6.4.3.26 Float\_t ep4s****6.4.3.27 Float\_t ep5s****6.4.3.28 Float\_t ep6s****6.4.3.29 counter2 epart**

counter for participant-plane  $\langle r^n \cos(2 \phi) \rangle / \langle r^n \rangle$

**6.4.3.30 counter2 epart1**

counter for participant-plane  $\langle r^3 \cos(\phi) \rangle / \langle r^3 \rangle$

**6.4.3.31 counter2 epart3**

counter for participant-plane  $\langle r^n \cos(3 \phi) \rangle / \langle r^n \rangle$

**6.4.3.32 counter2 epart4**

counter for participant-plane  $\langle r^n \cos(4 \phi) \rangle / \langle r^n \rangle$

**6.4.3.33 counter2 epart5**

counter for participant-plane  $\langle r^n \cos(5 \phi) \rangle / \langle r^n \rangle$

**6.4.3.34 counter2 epart6**

counter for participant-plane  $\langle r^n \cos(6 \phi) \rangle / \langle r^n \rangle$

**6.4.3.35 Float\_t eps****6.4.3.36 Float\_t ETA0 =2.5**

2\*ETA0 is the width of the plateau in eta

**6.4.3.37 Float\_t ETAM =8.58**

parameter of the Bialas-Czyz-Bozek model

**6.4.3.38 Int\_t evall**

number of all attempted event

**6.4.3.39 Int\_t EVENTS =1000**

number of generated events

**6.4.3.40 Int\_t FBIN =72**

number of bins for histogramming in the azimuthal angle

**6.4.3.41 Float\_t FBRAP =2.5**

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

**6.4.3.42 Int\_t FILES =0**

1 - read distribution from files, 0 - do not

**6.4.3.43 Int\_t FULL =0**

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

**6.4.3.44 Float\_t GA =1**

Gaussian wounding profile parameter (height at the origin)

**6.4.3.45 Float\_t GAMA =-1**

gamma wounding profile parameter (height at the origin)

**6.4.3.46 UInt\_t ISEED**

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

**6.4.3.47 UInt\_t ISEED1**

copy of ISEED

**6.4.3.48 Int\_t kk**

number of the current event

**6.4.3.49 Float\_t MAXYRAP =10.**

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

**6.4.3.50 Int\_t MODEL =0**

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

**6.4.3.51 Int\_t NBIN =40**

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

**6.4.3.52 counter2 nbinary**

counter for number of binary collisions

**6.4.3.53 Int\_t NCS =3**

number of partons in the nucleon

**6.4.3.54 counter2 nhot**

counter for number of hot-spots

**6.4.3.55 Int\_t NNWP =0**

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

**6.4.3.56 Int\_t NUMA =208**

mass number of nucleus A

**6.4.3.57 Int\_t NUMB =208**

mass number of nucleus B

**6.4.3.58 Int\_t NUMRAP =10**

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

**6.4.3.59 counter2 nweight**

counter for relative deposited strength (RDS)

**6.4.3.60 counter2 nwounded**

counter for number of wounded objects

**6.4.3.61 Float\_t OMEGA =-1**

relative variance of cross-section fluctuations

**6.4.3.62 Int\_t PARTONS****6.4.3.63 Float\_t phiro1****6.4.3.64 Float\_t phiro1**

angles of the principal axis

**6.4.3.65 Float\_t phiro3****6.4.3.66 Float\_t phiro4****6.4.3.67 Float\_t phiro5****6.4.3.68 Float\_t phiro6****6.4.3.69 Float\_t PI =4.\*atan(1.)**

the number pi

**6.4.3.70 Int\_t PP =-1**

power of the transverse radius in the Fourier moments

**6.4.3.71 Int\_t ppp**

power of the weight in eccentricity definition

**6.4.3.72 Float\_t QSCALE =-1**

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

**6.4.3.73 Float\_t RAPRANGE =5.**

range in rapidity

**6.4.3.74 Float\_t rbin**

number of binary collisions

**6.4.3.75 Float\_t RCHA =5.66**

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

**6.4.3.76 Float\_t RCHB =5.66**

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

**6.4.3.77 Float\_t RCHP =0.7714**

harmonic oscillator shell model density mean squared charge radii of proton

**6.4.3.78 Float\_t RDS0 =0.**

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

**6.4.3.79 Float\_t RDS1 =100000**

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

**6.4.3.80 Int\_t RET =0**

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

**6.4.3.81 Float\_t rhotspot**

number of hot-spots

**6.4.3.82 Int\_t RO =0**

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

**6.4.3.83 Int\_t roo**

Fourier rank for the rotation axis.

**6.4.3.84 Float\_t ROTA\_PHI =-1.0**

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

**6.4.3.85 Float\_t ROTA\_THETA =-1.0**

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

**6.4.3.86 Float\_t ROTB\_PHI =-1.0**

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

**6.4.3.87 Float\_t ROTB\_THETA =-1.0**

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

**6.4.3.88 Float\_t rpa**

relative deposited strength (RDS)

**6.4.3.89 Float\_t rwA**

number of wounded objects in A

**6.4.3.90 Float\_t rwAB**

number of all wounded objects

**6.4.3.91 Float\_t rwB**

number of wounded objects in B

**6.4.3.92 Float\_t RWSA =-1.**

Woods-Saxon radius for nucleus A.

**6.4.3.93 Float\_t RWSE =-1.**

Woods-Saxon radius for nucleus B.

**6.4.3.94 Float\_t SBIN =-1**

NN binary cross section in milibarns.

**6.4.3.95 Float\_t SCALEA =1.**

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

**6.4.3.96 Float\_t SCALEB =1.**

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

**6.4.3.97 Int\_t SHIFT =1**

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

**6.4.3.98 Float\_t SIGETA =1.4**

parameter controlling the width of the rapidity distribution

**6.4.3.99 Float\_t SIGMAA =1.**

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

**6.4.3.100 Float\_t SIGMAB =1.**

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

**6.4.3.101 Float\_t SIGMABISA =1.**

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

**6.4.3.102 Float\_t SIGMABISB =1.**

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

**6.4.3.103 Float\_t sirad**

equivalent hard-sphere radius for the cross section

**6.4.3.104 Float\_t sitot**

the total A+B cross section in the acceptance window

**6.4.3.105 Float\_t sizeav**

size

**6.4.3.106 Float\_t SNN =-1**

NN "wounding" cross section in milibarns.

**6.4.3.107 Float\_t Ubin =1.**

Poisson or Gamma parameters for superimposed distribution, binary collisions.

**6.4.3.108 Float\_t Uw =1.**

Poisson or Gamma parameters for superimposed distribution, wounded nucleons.

**6.4.3.109 Float\_t Vbin =2.**

Negative binomial variance, binary collisions.

6.4.3.110 Float\_t Vw =2.

Negative binomial variance, wounded nucleons.

6.4.3.111 Int\_t W0 =2

minimum allowed number of wounded objects in the acceptance window

6.4.3.112 Int\_t W1 =10000

maximum allowed number of wounded objects in the acceptance window

6.4.3.113 Float\_t WFA =0.

the w parameter for the Fermi distribution for nucleus A

6.4.3.114 Float\_t WFB =0.

the w parameter for the Fermi distribution for nucleus B

6.4.3.115 Float\_t wfq

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

6.4.3.116 Float\_t wfqA

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

6.4.3.117 Float\_t wfqB

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

6.4.3.118 Int\_t WMIN =2

minimum number of wounded nucleons to record the event

6.4.3.119 Float\_t xep

average ep

6.4.3.120 Float\_t xsepp

standard deviation of ep

6.4.3.121 Float\_t xx

center-of-mass x coordinate

## 6.4.3.122 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_ 3.111`

### 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_ 3.111`

### 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*, 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 from other sources the third command line argument is the file for nucleus A and the fourth (optional) for nucleus B - if absent, B is generated randomly in Glissando3. Format of reading is adjusted to the format of files, which is different for A=3 (helium, tritium) and A>3.

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 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

generate the participant-plane 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

<code>argc</code>	number of command line parameters
<code>argv</code>	used for passing input and output file names first argument: <code>&lt;input.dat&gt;</code> second argument: <code>&lt;output.root&gt;</code> third argument: <code>&lt;nucl_A.dat&gt;</code> (present only when <code>files=1</code> ) fourth argument: <code>&lt;nucl_B.dat&gt;</code> (esent pronly when <code>files=1</code> ) 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/demo\_2/centrality2.C File Reference

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

### Functions

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

### 6.6.1 Detailed Description

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

### 6.6.2 Function Documentation

#### 6.6.2.1 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

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

## 6.7 macro/demo\_2/core\_mantle.C File Reference

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

### Functions

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

#### 6.7.1 Detailed Description

Script generating the core-corona densities (part of GLISSANDO 2)

#### 6.7.2 Function Documentation

##### 6.7.2.1 void `core_mantle` ( char \* *p* )

generates the core and corona distributions

## Parameters

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

## 6.8 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*

#### 6.8.1 Detailed Description

Script generating the NN correlation plot for nucleus A (part of GLISSANDO 2)

#### 6.8.2 Function Documentation

##### 6.8.2.1 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
-----	-----------------------------

## 6.9 macro/demo\_2/density.C File Reference

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

## Functions

- void [density](#) (char \*p)  
*produces plots of the fixed-axes and participant-plane densities*

### 6.9.1 Detailed Description

Script generating the fixed- and participant-plane densities (part of GLISSANDO 2)

### 6.9.2 Function Documentation

#### 6.9.2.1 void density ( char \* p )

produces plots of the fixed-axes and participant-plane densities

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

## Parameters

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

## 6.10 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*

### 6.10.1 Detailed Description

Script generating the plots of eccentricities and their scaled standard deviations as functions of the number of wounded nucleons (part of GLISSANDO 2)

## 6.10.2 Function Documentation

### 6.10.2.1 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
----------	-----------------------------

## 6.11 macro/demo\_2/fourier.C File Reference

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

### Functions

- void [fourier](#) (char \*p)  
*generates the plot of epsilon\_n^\* vs. N<sub>w</sub>, n=1,2,3,4,5,6*

### 6.11.1 Detailed Description

Script generating the epsilon\_n^\* vs. N<sub>w</sub> plot (part of GLISSANDO 2)

Script generating the epsilon\_n vs. Q<sub>w</sub> plot (part of GLISSANDO 3)

## 6.11.2 Function Documentation

### 6.11.2.1 void fourier ( char \* p )

generates the plot of epsilon\_n^\* vs. N<sub>w</sub>, n=1,2,3,4,5,6

Parameters

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

## 6.12 macro/demo\_2/info.C File Reference

### Functions

- void [info](#) (char \*p)  
*prints info on the stored GLISSANDO 2 ROOT file*

### 6.12.1 Detailed Description

Script printing info on the GLISSANDO 2 ROOT file (part of GLISSANDO 2)

## 6.12.2 Function Documentation

### 6.12.2.1 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
----------	-----------------------

## 6.13 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*

### 6.13.1 Function Documentation

#### 6.13.1.1 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
---------------	-----------------------------

#### 6.13.1.2 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
---------------	-----------------------------

## 6.14 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*

### 6.14.1 Function Documentation

#### 6.14.1.1 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
---------------	-----------------------------

#### 6.14.1.2 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
---------------	-----------------------------

## 6.15 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*

### 6.15.1 Detailed Description

Script plotting the event-by-event scaled variance of RDS vs. the number of wounded nucleons in the projectile (part of GLISSANDO 2)

### 6.15.2 Function Documentation

#### 6.15.2.1 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

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

## 6.16 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.*

### 6.16.1 Detailed Description

Script generating r-cos(theta) distributions of the nucleons in the nucleus (part of GLISSANDO 2)

### 6.16.2 Function Documentation

#### 6.16.2.1 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

## 6.17 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.*

### 6.17.1 Detailed Description

Script generating r-cos(theta) distributions of the nucleons in the nucleus (part of GLISSANDO 2)

### 6.17.2 Function Documentation

#### 6.17.2.1 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

## 6.18 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.*

#### 6.18.1 Detailed Description

Script plotting the event-by-event scaled standard deviation of the size parameter defined as the average distance of sources from the center of mass (part of GLISSANDO 2)

#### 6.18.2 Function Documentation

##### 6.18.2.1 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
-----	-----------------------------

## 6.19 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*

#### 6.19.1 Detailed Description

Script generating the wounding and binary-collision profiles (part of GLISSANDO 2)

Script generating the wounding (inelasticity) profile (part of GLISSANDO 3)

#### 6.19.2 Function Documentation

##### 6.19.2.1 void wounding\_profile ( char \* p )

generates the plots of the wounding and binary-collision profiles

## Parameters

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

## 6.20 macro/demo\_3/b\_dist.C File Reference

## Functions

- void [b\\_dist](#) (char \*p)

### 6.20.1 Function Documentation

#### 6.20.1.1 void b\_dist ( char \* p )

## Parameters

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

## 6.21 macro/demo\_3/b\_distr.C File Reference

```
#include "TCanvas.h"
#include "TStyle.h"
#include "TH1.h"
```

## Functions

- void [b\\_distr](#) (char \*p)

### 6.21.1 Function Documentation

#### 6.21.1.1 void b\_distr ( char \* p )

## Parameters

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

## 6.22 macro/demo\_3/cent.C File Reference

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

## Functions

- void [cent](#) (char \*p)  
*generates the centrality distribution of wounded objects*

### 6.22.1 Detailed Description

Script generating the centrality distribution of wounded objects (part of GLISSANDO 3)

### 6.22.2 Function Documentation

#### 6.22.2.1 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
----------	-----------------------------

## 6.23 macro/demo\_3/cent\_comp.C File Reference

### Functions

- void [cent\\_comp](#) (char \*p\_qq, char \*p\_NN)  
*generates the centrality distribution of wounded objects*

### 6.23.1 Function Documentation

#### 6.23.1.1 void cent\_comp ( char \* p\_qq, char \* p\_NN )

generates the centrality distribution of wounded objects

generates the centrality distribution of wounded objects (nucleons or quarks)

Parameters

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

## 6.24 macro/demo\_3/clusters.C File Reference

```
#include "TCanvas.h"
#include "TStyle.h"
#include "TH1.h"
```

### Functions

- void [clusters](#) (char \*p)

### 6.24.1 Function Documentation

#### 6.24.1.1 void clusters ( char \* p )

## Parameters

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

## 6.25 macro/demo\_3/eps\_dist.C File Reference

## Functions

- void `eps_dist` (char \*p)

## 6.25.1 Function Documentation

6.25.1.1 void `eps_dist` ( char \*  $p$  )

## Parameters

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

## 6.26 macro/demo\_3/epsilon\_c.C File Reference

## Functions

- void `epsilon_c` (char \*p)

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

## 6.26.1 Detailed Description

Script generating the plots of participant eccentricities and their scaled standard deviations as functions of centrality (part of GLISSANDO 2)

## 6.26.2 Function Documentation

6.26.2.1 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

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

## 6.27 macro/demo\_3/fourier\_Q.C File Reference

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

## Functions

- void `fourier_Q` (char \*p, Int\_t up)  
*generates the plot of epsilon\_n vs. Qw, n=2,3*

### 6.27.1 Function Documentation

6.27.1.1 void `fourier_Q` ( char \* p, Int\_t up )

generates the plot of epsilon\_n vs. Qw, n=2,3

Parameters

<i>p</i>	name of the ROOT input file
<i>up</i>	x-range of the plot

## 6.28 macro/demo\_3/inel\_prof.C File Reference

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

## Functions

- void `inel_prof` (char \*p)  
*generates the plots of the wounding profile*

### 6.28.1 Function Documentation

6.28.1.1 void `inel_prof` ( char \* p )

generates the plots of the wounding profile

Parameters

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

## 6.29 macro/demo\_3/mult\_pPb\_Q.C File Reference

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

## Functions

- void `mult_pPb_Q` (char \*p)  
*generates the plot of epsilon\_n vs. Qw, n=2,3*

### 6.29.1 Function Documentation

6.29.1.1 void `mult_pPb_Q` ( char \* p )

generates the plot of epsilon\_n vs. Qw, n=2,3

## Parameters

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

## 6.30 macro/demo\_3/mult\_Q.C File Reference

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

## Functions

- void [mult\\_Q](#) (char \*p)  
*generates the plot of epsilon\_n vs. Qw, n=2,3*

## 6.30.1 Function Documentation

6.30.1.1 void [mult\\_Q](#) ( char \*  $p$  )

generates the plot of epsilon\_n vs. Qw, n=2,3

## Parameters

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

## 6.31 macro/demo\_3/part\_dist\_pp.C File Reference

## Functions

- void [part\\_dist\\_pp](#) (char \*p)

## 6.31.1 Function Documentation

6.31.1.1 void [part\\_dist\\_pp](#) ( char \*  $p$  )

## Parameters

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

## 6.32 macro/demo\_3/rad\_dis.C File Reference

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

## Functions

- void [rad\\_dis](#) (char \*p)  
*radial distribution of nucleons or partons*

### 6.32.1 Detailed Description

Script yielding the radial distribution of nucleons or partons (part of GLISSANDO 3)

### 6.32.2 Function Documentation

#### 6.32.2.1 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
----------	-----------------------------

## 6.33 macro/demo\_3/sc\_comp.C File Reference

### Functions

- void [sc\\_comp](#) (char \*p\_clust, char \*p\_unif)  
*generates the plot of the symmetric cumulant SC(2,3)*

### 6.33.1 Detailed Description

(part of GLISSANDO 3)

### 6.33.2 Function Documentation

#### 6.33.2.1 void sc\_comp ( char \* p\_clust, char \* p\_unif )

generates the plot of the symmetric cumulant SC(2,3)

< names of the ROOT input files

determination of centrality bins

determination of centrality bins

## 6.34 macro/demo\_3/w\_prof\_norm\_AA.C File Reference

```
#include "TCanvas.h"
#include "TStyle.h"
#include "TH1.h"
```

### Functions

- void [w\\_prof\\_norm\\_AA](#) (char \*p\_qq, char \*p\_NN)

### 6.34.1 Function Documentation

6.34.1.1 void w\_prof\_norm\_AA ( char \* *p\_qq*, char \* *p\_NN* )

## Parameters

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

## 6.35 macro/demo\_3/w\_prof\_norm\_pp.C File Reference

## Functions

- `Double_t nn` (`Double_t *x`, `Double_t *par`)
- `Float_t nn_1` (`Double_t gama`, `Double_t omega`, `Float_t x`)
- `void w_prof_norm_pp` (`char *p`)

### 6.35.1 Function Documentation

6.35.1.1 `Double_t nn ( Double_t * x, Double_t * par )`

6.35.1.2 `Float_t nn_1 ( Double_t gama, Double_t omega, Float_t x )`

6.35.1.3 `void w_prof_norm_pp ( char * p )`

## Parameters

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

# Index

~counter  
    counter, [13](#)  
~counter2  
    counter2, [15](#)  
~counter\_2D  
    counter\_2D, [18](#)  
\_VER\_  
    glissando3.cxx, [67](#)

ALPHA  
    functions.h, [57](#)

ARANK  
    functions.h, [57](#)

ARGC  
    functions.h, [57](#)

AWSA  
    functions.h, [57](#)

AWSB  
    functions.h, [58](#)

add  
    counter, [13](#)  
    counter2, [15](#)  
    counter\_2D, [18](#)

angles  
    functions.h, [57](#)

b  
    functions.h, [58](#)

b\_dist  
    b\_dist.C, [77](#)

b\_dist.C  
    b\_dist, [77](#)

b\_distr  
    b\_distr.C, [77](#)

b\_distr.C  
    b\_distr, [77](#)

BETA2A  
    functions.h, [58](#)

BETA2B  
    functions.h, [58](#)

BETA4A  
    functions.h, [58](#)

BETA4B  
    functions.h, [58](#)

BMAX  
    functions.h, [58](#)

BMIN  
    functions.h, [58](#)

BTOT  
    functions.h, [58](#)

build/include/collision.h, [45](#)  
build/include/counter.h, [45](#)  
build/include/distrib.h, [45](#)  
build/include/functions.h, [46](#)  
build/src/glissando3.cxx, [67](#)

c  
    distr, [27](#)

CD  
    functions.h, [58](#)

CLUSTERS  
    functions.h, [58](#)

cent  
    cent.C, [78](#)

cent.C  
    cent, [78](#)

cent\_comp  
    cent\_comp.C, [78](#)

cent\_comp.C  
    cent\_comp, [78](#)

centrality2  
    centrality2.C, [69](#)

centrality2.C  
    centrality2, [69](#)

clusters  
    clusters.C, [78](#)

clusters.C  
    clusters, [78](#)

cmx  
    distr, [23](#)

cmx\_w  
    distr, [23](#)

cm\_y  
    distr, [23](#)

cm\_y\_w  
    distr, [23](#)

cmz  
    distr, [23](#)

cmz\_w  
    distr, [23](#)

collision, [9](#)  
    collision, [10](#)  
    gen\_RDS, [10](#)  
    nbin, [11](#)  
    nhotspot, [11](#)  
    nwA, [11](#)  
    nwAB, [11](#)  
    nwB, [11](#)  
    nzw, [11](#)  
    operator=, [11](#)

- rd2, 11
- rpa, 11
- shift\_cmx\_w\_c, 11
- shift\_cmy\_w\_c, 11
- specA, 12
- specB, 12
- wc, 12
- wwA, 12
- wwAB, 12
- wwB, 12
- xcmA, 12
- xcmB, 12
- ycmA, 12
- ycmB, 12
- core\_mantle
  - core\_mantle.C, 70
- core\_mantle.C
  - core\_mantle, 70
- corr
  - corr.C, 70
  - counter\_2D, 18
- corr.C
  - corr, 70
- count
  - counter, 14
  - counter2, 16
  - counter\_2D, 19
- counter, 12
  - ~counter, 13
  - add, 13
  - count, 14
  - counter, 13
  - get, 14
  - getN, 14
  - mean, 14
  - reset, 14
  - value, 14
- counter2, 14
  - ~counter2, 15
  - add, 15
  - count, 16
  - counter2, 15
  - get, 15
  - get2, 15
  - getN, 16
  - mean, 16
  - reset, 16
  - value, 16
  - value2, 16
  - var, 16
  - vara, 16
- counter\_2D, 16
  - ~counter\_2D, 18
  - add, 18
  - corr, 18
  - count, 19
  - counter\_2D, 18
  - counter\_2D, 18
- cov, 18
  - get\_x, 18
  - get\_x2, 18
  - get\_xy, 18
  - get\_y, 18
  - get\_y2, 18
  - getN, 19
  - mean\_x, 19
  - mean\_y, 19
  - reset, 19
  - valuex, 19
  - valuex2, 19
  - valuexy, 19
  - valuey, 19
  - valuey2, 19
  - var\_x, 19
  - var\_y, 19
- cov
  - counter\_2D, 18
- cth
  - nucleus, 36
- d
  - functions.h, 58
- DBIN
  - functions.h, 58
- DOBIN
  - functions.h, 59
- DS
  - functions.h, 59
- DW
  - functions.h, 59
- dbin
  - functions.h, 59
- demo\_2/label.C
  - label, 73
  - label\_fit, 73
- demo\_3/label.C
  - label, 74
  - label\_fit, 74
- density
  - density.C, 71
- density.C
  - density, 71
- disp
  - functions.h, 52
- dist
  - functions.h, 52
- dist2
  - nucleus, 31
- distr, 20
  - c, 27
  - cmx, 23
  - cmx\_w, 23
  - cmy, 23
  - cmy\_w, 23
  - cmz, 23
  - cmz\_w, 23
  - distr, 22

- eps, [23](#), [24](#)
- fill\_xy, [24](#)
- msr, [27](#)
- msrad, [24](#)
- msrad\_t, [24](#)
- msrad\_t\_w, [24](#)
- msrad\_w, [24](#)
- msrt, [27](#)
- msx, [25](#)
- msy, [25](#)
- mxy, [25](#)
- n, [28](#)
- operator=, [25](#)
- phrot, [25](#)
- qx, [25](#)
- qy, [26](#)
- rotate, [26](#)
- rotate\_polar, [26](#)
- shift\_cmx, [26](#)
- shift\_cmx\_w, [26](#)
- shift\_cmy, [26](#)
- shift\_cmy\_w, [26](#)
- shift\_cmz, [26](#)
- shift\_cmz\_w, [26](#)
- shift\_x, [27](#)
- shift\_y, [27](#)
- shift\_z, [27](#)
- size, [27](#)
- sum\_w, [27](#)
- sumw, [28](#)
- uncluster, [27](#)
- w, [28](#)
- x, [28](#)
- xcm, [28](#)
- y, [28](#)
- ycm, [28](#)
- z, [28](#)
- zcm, [28](#)
- ECM
  - [functions.h](#), [59](#)
- ETA0
  - [functions.h](#), [60](#)
- ETAM
  - [functions.h](#), [60](#)
- EVENTS
  - [functions.h](#), [60](#)
- echopar
  - [functions.h](#), [53](#)
- ep1s
  - [functions.h](#), [59](#)
- ep3s
  - [functions.h](#), [59](#)
- ep4s
  - [functions.h](#), [59](#)
- ep5s
  - [functions.h](#), [59](#)
- ep6s
  - [functions.h](#), [59](#)
- epart
  - [functions.h](#), [59](#)
- epart1
  - [functions.h](#), [59](#)
- epart3
  - [functions.h](#), [59](#)
- epart4
  - [functions.h](#), [59](#)
- epart5
  - [functions.h](#), [59](#)
- epart6
  - [functions.h](#), [60](#)
- epilog
  - [functions.h](#), [53](#)
- eps
  - distr, [23](#), [24](#)
  - [functions.h](#), [60](#)
- eps\_dist
  - [eps\\_dist.C](#), [79](#)
- eps\_dist.C
  - [eps\\_dist](#), [79](#)
- epsilon
  - [epsilon.C](#), [72](#)
- epsilon.C
  - [epsilon](#), [72](#)
- epsilon\_c
  - [epsilon\\_c.C](#), [79](#)
- epsilon\_c.C
  - [epsilon\\_c](#), [79](#)
- evall
  - [functions.h](#), [60](#)
- FBIN
  - [functions.h](#), [60](#)
- FBRAP
  - [functions.h](#), [60](#)
- FILES
  - [functions.h](#), [60](#)
- FULL
  - [functions.h](#), [60](#)
- fgama
  - [functions.h](#), [53](#)
- fill
  - [tr\\_his\\_c](#), [39](#)
- fill\_res
  - [tr\\_his\\_c](#), [39](#)
- fill\_tr
  - [tr\\_his\\_c](#), [39](#)
- fill\_xy
  - distr, [24](#)
- fomega
  - [functions.h](#), [53](#)
- fourier
  - [fourier.C](#), [72](#)
- fourier.C
  - [fourier](#), [72](#)
- fourier\_Q
  - [fourier\\_Q.C](#), [80](#)
- fourier\_Q.C

- fourier\_Q, [80](#)
- fqscale
  - functions.h, [53](#)
- fsNN
  - functions.h, [53](#)
- fsQQ
  - functions.h, [53](#)
- full\_event
  - tr\_his\_c, [40](#)
- functions.h
  - ALPHA, [57](#)
  - ARANK, [57](#)
  - ARGC, [57](#)
  - AWSA, [57](#)
  - AWSB, [58](#)
  - angles, [57](#)
  - b, [58](#)
  - BETA2A, [58](#)
  - BETA2B, [58](#)
  - BETA4A, [58](#)
  - BETA4B, [58](#)
  - BMAX, [58](#)
  - BMIN, [58](#)
  - BTOT, [58](#)
  - CD, [58](#)
  - CLUSTERS, [58](#)
  - d, [58](#)
  - DBIN, [58](#)
  - DOBIN, [59](#)
  - DS, [59](#)
  - DW, [59](#)
  - dbin, [59](#)
  - disp, [52](#)
  - dist, [52](#)
  - ECM, [59](#)
  - ETA0, [60](#)
  - ETAM, [60](#)
  - EVENTS, [60](#)
  - echopar, [53](#)
  - ep1s, [59](#)
  - ep3s, [59](#)
  - ep4s, [59](#)
  - ep5s, [59](#)
  - ep6s, [59](#)
  - epart, [59](#)
  - epart1, [59](#)
  - epart3, [59](#)
  - epart4, [59](#)
  - epart5, [59](#)
  - epart6, [60](#)
  - epilog, [53](#)
  - eps, [60](#)
  - evall, [60](#)
  - FBIN, [60](#)
  - FBRAP, [60](#)
  - FILES, [60](#)
  - FULL, [60](#)
  - fgama, [53](#)
  - fomega, [53](#)
  - fqscale, [53](#)
  - fsNN, [53](#)
  - fsQQ, [53](#)
  - GA, [60](#)
  - GAMA, [60](#)
  - gamgen, [53](#)
  - header, [54](#)
  - helper, [54](#)
  - ISEED, [60](#)
  - ISEED1, [61](#)
  - kk, [61](#)
  - los, [54](#)
  - los12, [54](#)
  - los32, [54](#)
  - MAXYRAP, [61](#)
  - MODEL, [61](#)
  - NBIN, [61](#)
  - NCS, [61](#)
  - NNWP, [61](#)
  - NUMA, [61](#)
  - NUMB, [61](#)
  - NUMRAP, [61](#)
  - nbinary, [61](#)
  - negbin, [54](#)
  - nhot, [61](#)
  - nweight, [62](#)
  - nwounded, [62](#)
  - OMEGA, [62](#)
  - PARTONS, [62](#)
  - PI, [62](#)
  - PP, [62](#)
  - phirot, [62](#)
  - phirot1, [62](#)
  - phirot3, [62](#)
  - phirot4, [62](#)
  - phirot5, [62](#)
  - phirot6, [62](#)
  - ppp, [62](#)
  - QSCALE, [62](#)
  - RAPRANGE, [62](#)
  - RCHA, [63](#)
  - RCHB, [63](#)
  - RCHP, [63](#)
  - RDS0, [63](#)
  - RDS1, [63](#)
  - RET, [63](#)
  - RO, [63](#)
  - ROTA\_PHI, [63](#)
  - ROTA\_THETA, [63](#)
  - ROTB\_PHI, [63](#)
  - ROTB\_THETA, [64](#)
  - RWSA, [64](#)
  - RWSB, [64](#)
  - rbin, [62](#)
  - readpar, [54](#)
  - reset\_counters, [54](#)
  - rhotspot, [63](#)

- rlos\_abf, [54](#)
- rlos\_alpha, [56](#)
- rlos\_hole, [56](#)
- rlos\_hult, [56](#)
- rlos\_parton, [56](#)
- rlosA, [56](#)
- rlosA\_def, [56](#)
- rlosA\_hos, [56](#)
- rlosB, [56](#)
- rlosB\_def, [57](#)
- rlosB\_hos, [57](#)
- roo, [63](#)
- rpa, [64](#)
- rwA, [64](#)
- rwAB, [64](#)
- rwB, [64](#)
- SBIN, [64](#)
- SCALEA, [64](#)
- SCALEB, [64](#)
- SHIFT, [64](#)
- SIGETA, [64](#)
- SIGMAA, [65](#)
- SIGMAB, [65](#)
- SIGMABISA, [65](#)
- SIGMABISB, [65](#)
- SNN, [65](#)
- sirad, [65](#)
- sitot, [65](#)
- sizeav, [65](#)
- time\_start, [57](#)
- time\_stop, [57](#)
- Ubin, [65](#)
- Uw, [65](#)
- Vbin, [65](#)
- Vw, [65](#)
- W0, [66](#)
- W1, [66](#)
- WFA, [66](#)
- WFB, [66](#)
- WMIN, [66](#)
- wfq, [66](#)
- wfqA, [66](#)
- wfqB, [66](#)
- xepp, [66](#)
- xsepp, [66](#)
- xx, [66](#)
- yy, [66](#)
- g
  - nucleus, [36](#)
- GA
  - functions.h, [60](#)
- GAMA
  - functions.h, [60](#)
- gamgen
  - functions.h, [53](#)
- gen
  - tr\_his\_c, [40](#)
- gen\_RDS
  - collision, [10](#)
- get
  - counter, [14](#)
  - counter2, [15](#)
- get2
  - counter2, [15](#)
- get\_x
  - counter\_2D, [18](#)
- get\_x2
  - counter\_2D, [18](#)
- get\_xy
  - counter\_2D, [18](#)
- get\_y
  - counter\_2D, [18](#)
- get\_y2
  - counter\_2D, [18](#)
- getN
  - counter, [14](#)
  - counter2, [16](#)
  - counter\_2D, [19](#)
- glissando3.cxx
  - \_VER\_, [67](#)
  - main, [67](#)
  - raa, [69](#)
  - ver, [69](#)
- good\_all
  - nucleus, [31](#)
- good\_down
  - nucleus, [31](#)
- good\_pair
  - nucleus, [31](#)
- header
  - functions.h, [54](#)
- helper
  - functions.h, [54](#)
- ISEED
  - functions.h, [60](#)
- ISEED1
  - functions.h, [61](#)
- inel\_prof
  - inel\_prof.C, [80](#)
- inel\_prof.C
  - inel\_prof, [80](#)
- info
  - info.C, [73](#)
- info.C
  - info, [73](#)
- init
  - tr\_his\_c, [40](#)
- KK
  - SOURCE, [37](#)
- kk
  - functions.h, [61](#)
- label
  - demo\_2/label.C, [73](#)

- demo\_3/label.C, 74
- label\_fit
  - demo\_2/label.C, 73
  - demo\_3/label.C, 74
- los
  - functions.h, 54
- los12
  - functions.h, 54
- los32
  - functions.h, 54
- MAXYRAP
  - functions.h, 61
- MODEL
  - functions.h, 61
- macro/demo\_2/centrality2.C, 69
- macro/demo\_2/core\_mantle.C, 70
- macro/demo\_2/corr.C, 70
- macro/demo\_2/density.C, 71
- macro/demo\_2/epsilon.C, 71
- macro/demo\_2/fourier.C, 72
- macro/demo\_2/info.C, 72
- macro/demo\_2/label.C, 73
- macro/demo\_2/mult.C, 74
- macro/demo\_2/profile2\_deformation\_63Cu.C, 74
- macro/demo\_2/profile2\_deformation\_U.C, 75
- macro/demo\_2/size.C, 76
- macro/demo\_2/wounding\_profile.C, 76
- macro/demo\_3/b\_dist.C, 77
- macro/demo\_3/b\_distr.C, 77
- macro/demo\_3/cent.C, 77
- macro/demo\_3/cent\_comp.C, 78
- macro/demo\_3/clusters.C, 78
- macro/demo\_3/eps\_dist.C, 79
- macro/demo\_3/epsilon\_c.C, 79
- macro/demo\_3/fourier\_Q.C, 79
- macro/demo\_3/inel\_prof.C, 80
- macro/demo\_3/label.C, 73
- macro/demo\_3/mult\_Q.C, 81
- macro/demo\_3/mult\_pPb\_Q.C, 80
- macro/demo\_3/part\_dist\_pp.C, 81
- macro/demo\_3/rad\_dis.C, 81
- macro/demo\_3/sc\_comp.C, 82
- macro/demo\_3/w\_prof\_norm\_AA.C, 82
- macro/demo\_3/w\_prof\_norm\_pp.C, 84
- main
  - glissando3.cxx, 67
- mean
  - counter, 14
  - counter2, 16
- mean\_x
  - counter\_2D, 19
- mean\_y
  - counter\_2D, 19
- msr
  - distr, 27
- msrad
  - distr, 24
- msrad\_t
  - distr, 24
- msrad\_t\_w
  - distr, 24
- msrad\_w
  - distr, 24
- msrt
  - distr, 27
- msx
  - distr, 25
- msy
  - distr, 25
- mult
  - mult.C, 74
- mult.C
  - mult, 74
- mult\_Q
  - mult\_Q.C, 81
- mult\_Q.C
  - mult\_Q, 81
- mult\_pPb\_Q
  - mult\_pPb\_Q.C, 80
- mult\_pPb\_Q.C
  - mult\_pPb\_Q, 80
- mxy
  - distr, 25
- n
  - distr, 28
- NBIN
  - functions.h, 61
- NCS
  - functions.h, 61
- NNWP
  - functions.h, 61
- NUMA
  - functions.h, 61
- NUMB
  - functions.h, 61
- NUMRAP
  - functions.h, 61
- nbin
  - collision, 11
- nbinar
  - tr\_his\_c, 40
- nbinar2
  - tr\_his\_c, 40
- nbinary
  - functions.h, 61
- negbin
  - functions.h, 54
- nepsp
  - tr\_his\_c, 41
- nepsp2
  - tr\_his\_c, 41
- nepsp4
  - tr\_his\_c, 41
- nhot
  - functions.h, 61
- nhotspot

- collision, 11
- nn
  - w\_prof\_norm\_pp.C, 84
- nn\_1
  - w\_prof\_norm\_pp.C, 84
- nsize
  - tr\_his\_c, 41
- nsize2
  - tr\_his\_c, 41
- ntarg
  - tr\_his\_c, 41
- ntarg2
  - tr\_his\_c, 41
- nucleus, 29
  - cth, 36
  - dist2, 31
  - g, 36
  - good\_all, 31
  - good\_down, 31
  - good\_pair, 31
  - nucleus, 30
  - operator=, 31
  - phi, 36
  - r, 36
  - set\_alpha\_cluster, 31
  - set\_berillium\_7\_cluster, 32
  - set\_berillium\_8\_cluster, 32
  - set\_berillium\_9\_cluster, 32
  - set\_carbon\_cluster, 32
  - set\_deuteron, 32
  - set\_deuteron\_s0, 32
  - set\_deuteron\_s1, 32
  - set\_file, 33
  - set\_file\_uncor, 33
  - set\_oxygen\_cluster, 33
  - set\_oxygen\_cluster\_square, 33
  - set\_parton\_A, 34
  - set\_parton\_B, 34
  - set\_proton, 34
  - set\_random\_A, 34
  - set\_random\_A\_def, 34
  - set\_random\_A\_hos, 35
  - set\_random\_B, 35
  - set\_random\_B\_def, 35
  - set\_random\_B\_hos, 35
  - set\_tritium, 36
  - sth, 36
- nuni
  - tr\_his\_c, 41
- nuniRDS
  - tr\_his\_c, 41
- nunib
  - tr\_his\_c, 41
- nunp
  - tr\_his\_c, 41
- nw2b
  - tr\_his\_c, 41
- nwA
  - collision, 11
- nwAB
  - collision, 11
- nwB
  - collision, 11
- nwb
  - tr\_his\_c, 42
- nwei
  - tr\_his\_c, 42
- nwei2
  - tr\_his\_c, 42
- nweight
  - functions.h, 62
- nwounded
  - functions.h, 62
- nx
  - tr\_his\_c, 42
- nx2
  - tr\_his\_c, 42
- ny
  - tr\_his\_c, 42
- ny2
  - tr\_his\_c, 42
- nzw
  - collision, 11
- OMEGA
  - functions.h, 62
- operator=
  - collision, 11
  - distr, 25
  - nucleus, 31
- PARTONS
  - functions.h, 62
- PI
  - functions.h, 62
- PP
  - functions.h, 62
- param
  - tr\_his\_c, 42
- part\_dist\_pp
  - part\_dist\_pp.C, 81
- part\_dist\_pp.C
  - part\_dist\_pp, 81
- phi
  - nucleus, 36
- phirot
  - functions.h, 62
- phirot1
  - functions.h, 62
- phirot3
  - functions.h, 62
- phirot4
  - functions.h, 62
- phirot5
  - functions.h, 62
- phirot6
  - functions.h, 62

phrot  
     distr, 25  
 phys  
     tr\_his\_c, 42  
 ppp  
     functions.h, 62  
 profile2\_deformation\_63Cu  
     profile2\_deformation\_63Cu.C, 75  
 profile2\_deformation\_63Cu.C  
     profile2\_deformation\_63Cu, 75  
 profile2\_deformation\_U  
     profile2\_deformation\_U.C, 75  
 profile2\_deformation\_U.C  
     profile2\_deformation\_U, 75  
  
 QSCALE  
     functions.h, 62  
 qx  
     distr, 25  
 qy  
     distr, 26  
  
 r  
     nucleus, 36  
 RAPRANGE  
     functions.h, 62  
 RCHA  
     functions.h, 63  
 RCHB  
     functions.h, 63  
 RCHP  
     functions.h, 63  
 RDS0  
     functions.h, 63  
 RDS1  
     functions.h, 63  
 RET  
     functions.h, 63  
 RO  
     functions.h, 63  
 ROTA\_PHI  
     functions.h, 63  
 ROTA\_THETA  
     functions.h, 63  
 ROTB\_PHI  
     functions.h, 63  
 ROTB\_THETA  
     functions.h, 64  
 RWSA  
     functions.h, 64  
 RWSB  
     functions.h, 64  
 raa  
     glissando3.cxx, 69  
 rad\_dis  
     rad\_dis.C, 82  
 rad\_dis.C  
     rad\_dis, 82  
 radA  
     tr\_his\_c, 42  
 radB  
     tr\_his\_c, 42  
 rbin  
     functions.h, 62  
 rcostheta\_nuclA  
     tr\_his\_c, 42  
 rcostheta\_nuclB  
     tr\_his\_c, 43  
 rd2  
     collision, 11  
 readpar  
     functions.h, 54  
 reset  
     counter, 14  
     counter2, 16  
     counter\_2D, 19  
 reset\_counters  
     functions.h, 54  
 rhotspot  
     functions.h, 63  
 rlos\_abf  
     functions.h, 54  
 rlos\_alpha  
     functions.h, 56  
 rlos\_hole  
     functions.h, 56  
 rlos\_hult  
     functions.h, 56  
 rlos\_parton  
     functions.h, 56  
 rlosA  
     functions.h, 56  
 rlosA\_def  
     functions.h, 56  
 rlosA\_hos  
     functions.h, 56  
 rlosB  
     functions.h, 56  
 rlosB\_def  
     functions.h, 57  
 rlosB\_hos  
     functions.h, 57  
 roo  
     functions.h, 63  
 rotate  
     distr, 26  
 rotate\_polar  
     distr, 26  
 rpa  
     collision, 11  
     functions.h, 64  
 rrel\_u  
     tr\_his\_c, 43  
 rrelA  
     tr\_his\_c, 43  
 rrelB  
     tr\_his\_c, 43

rwA  
     functions.h, 64  
 rwAB  
     functions.h, 64  
 rwB  
     functions.h, 64  
  
 SBIN  
     functions.h, 64  
 SCALEA  
     functions.h, 64  
 SCALEB  
     functions.h, 64  
 SHIFT  
     functions.h, 64  
 SIGETA  
     functions.h, 64  
 SIGMAA  
     functions.h, 65  
 SIGMAB  
     functions.h, 65  
 SIGMABISA  
     functions.h, 65  
 SIGMABISB  
     functions.h, 65  
 SNN  
     functions.h, 65  
 SOURCE, 36  
     KK, 37  
     W, 37  
     X, 37  
     Y, 37  
 sc\_comp  
     sc\_comp.C, 82  
 sc\_comp.C  
     sc\_comp, 82  
 set\_alpha\_cluster  
     nucleus, 31  
 set\_berillium\_7\_cluster  
     nucleus, 32  
 set\_berillium\_8\_cluster  
     nucleus, 32  
 set\_berillium\_9\_cluster  
     nucleus, 32  
 set\_carbon\_cluster  
     nucleus, 32  
 set\_deuteron  
     nucleus, 32  
 set\_deuteron\_s0  
     nucleus, 32  
 set\_deuteron\_s1  
     nucleus, 32  
 set\_file  
     nucleus, 33  
 set\_file\_uncor  
     nucleus, 33  
 set\_oxygen\_cluster  
     nucleus, 33  
 set\_oxygen\_cluster\_square  
     nucleus, 33  
  
 set\_parton\_A  
     nucleus, 34  
 set\_parton\_B  
     nucleus, 34  
 set\_proton  
     nucleus, 34  
 set\_random\_A  
     nucleus, 34  
 set\_random\_A\_def  
     nucleus, 34  
 set\_random\_A\_hos  
     nucleus, 35  
 set\_random\_B  
     nucleus, 35  
 set\_random\_B\_def  
     nucleus, 35  
 set\_random\_B\_hos  
     nucleus, 35  
 set\_tritium  
     nucleus, 36  
 shift\_cmx  
     distr, 26  
 shift\_cmx\_w  
     distr, 26  
 shift\_cmx\_w\_c  
     collision, 11  
 shift\_cmy  
     distr, 26  
 shift\_cmy\_w  
     distr, 26  
 shift\_cmy\_w\_c  
     collision, 11  
 shift\_cmz  
     distr, 26  
 shift\_cmz\_w  
     distr, 26  
 shift\_x  
     distr, 27  
 shift\_y  
     distr, 27  
 shift\_z  
     distr, 27  
 sirad  
     functions.h, 65  
 sitot  
     functions.h, 65  
 size  
     distr, 27  
     size.C, 76  
 size.C  
     size, 76  
 sizeav  
     functions.h, 65  
 specA  
     collision, 12  
 specB  
     collision, 12

- sth
  - nucleus, 36
- sum\_w
  - distr, 27
- sumw
  - distr, 28
- time\_start
  - functions.h, 57
- time\_stop
  - functions.h, 57
- tr\_his\_c, 37
  - fill, 39
  - fill\_res, 39
  - fill\_tr, 39
  - full\_event, 40
  - gen, 40
  - init, 40
  - nbinar, 40
  - nbinar2, 40
  - nepsp, 41
  - nepsp2, 41
  - nepsp4, 41
  - nsiz, 41
  - nsiz2, 41
  - ntarg, 41
  - ntarg2, 41
  - nuni, 41
  - nuniRDS, 41
  - nunib, 41
  - nunp, 41
  - nw2b, 41
  - nwb, 42
  - nwei, 42
  - nwei2, 42
  - nx, 42
  - nx2, 42
  - ny, 42
  - ny2, 42
  - param, 42
  - phys, 42
  - radA, 42
  - radB, 42
  - rcostheta\_nuclA, 42
  - rcostheta\_nuclB, 43
  - rrel\_u, 43
  - rrelA, 43
  - rrelB, 43
  - tree, 43
  - weih, 43
  - weih\_bin, 43
  - wpro, 43
  - write, 40
  - write\_d, 40
  - write\_r, 40
  - write\_w, 40
  - write\_wpro, 40
  - xyhist, 43
  - xyhist\_core, 43
  - xyhist\_mantle, 43
  - xyhist\_nuclA, 43
  - xyhist\_nuclB, 44
  - xyhistr, 44
- tree
  - tr\_his\_c, 43
- Ubin
  - functions.h, 65
- uncluster
  - distr, 27
- Uw
  - functions.h, 65
- value
  - counter, 14
  - counter2, 16
- value2
  - counter2, 16
- valuex
  - counter\_2D, 19
- valuex2
  - counter\_2D, 19
- valuexy
  - counter\_2D, 19
- valuey
  - counter\_2D, 19
- valuey2
  - counter\_2D, 19
- var
  - counter2, 16
- var\_x
  - counter\_2D, 19
- var\_y
  - counter\_2D, 19
- vara
  - counter2, 16
- Vbin
  - functions.h, 65
- ver
  - glissando3.cxx, 69
- Vw
  - functions.h, 65
- W
  - SOURCE, 37
- w
  - distr, 28
- W0
  - functions.h, 66
- W1
  - functions.h, 66
- w\_prof\_norm\_AA
  - w\_prof\_norm\_AA.C, 83
- w\_prof\_norm\_AA.C
  - w\_prof\_norm\_AA, 83
- w\_prof\_norm\_pp
  - w\_prof\_norm\_pp.C, 84
- w\_prof\_norm\_pp.C

- nn, [84](#)
- nn\_1, [84](#)
- w\_prof\_norm\_pp, [84](#)
- WFA
  - functions.h, [66](#)
- WFB
  - functions.h, [66](#)
- WMIN
  - functions.h, [66](#)
- wc
  - collision, [12](#)
- weih
  - tr\_his\_c, [43](#)
- weih\_bin
  - tr\_his\_c, [43](#)
- wfq
  - functions.h, [66](#)
- wfqA
  - functions.h, [66](#)
- wfqB
  - functions.h, [66](#)
- wounding\_profile
  - wounding\_profile.C, [76](#)
- wounding\_profile.C
  - wounding\_profile, [76](#)
- wpro
  - tr\_his\_c, [43](#)
- write
  - tr\_his\_c, [40](#)
- write\_d
  - tr\_his\_c, [40](#)
- write\_r
  - tr\_his\_c, [40](#)
- write\_w
  - tr\_his\_c, [40](#)
- write\_wpro
  - tr\_his\_c, [40](#)
- wwA
  - collision, [12](#)
- wwAB
  - collision, [12](#)
- wwB
  - collision, [12](#)
- X
  - SOURCE, [37](#)
- x
  - distr, [28](#)
- xcm
  - distr, [28](#)
- xcmA
  - collision, [12](#)
- xcmB
  - collision, [12](#)
- xepp
  - functions.h, [66](#)
- xsepp
  - functions.h, [66](#)
- xx
  - functions.h, [66](#)
- xyhist
  - tr\_his\_c, [43](#)
- xyhist\_core
  - tr\_his\_c, [43](#)
- xyhist\_mantle
  - tr\_his\_c, [43](#)
- xyhist\_nuclA
  - tr\_his\_c, [43](#)
- xyhist\_nuclB
  - tr\_his\_c, [44](#)
- xyhistr
  - tr\_his\_c, [44](#)
- Y
  - SOURCE, [37](#)
- y
  - distr, [28](#)
- ycm
  - distr, [28](#)
- ycmA
  - collision, [12](#)
- ycmB
  - collision, [12](#)
- yy
  - functions.h, [66](#)
- z
  - distr, [28](#)
- zcm
  - distr, [28](#)