Krzysztof Rolbiecki

Implementation of search for squarks and gluinos in CheckMATE (857/D111/2019)

November 28, 2019

1 Summary

This is an implementation of a search for the supersymmetric partners of quarks and gluons (squarks and gluinos) in final states containing hadronic jets and missing transverse momentum. Events with an electron or muon are vetoed. The analysis is performed using data at $\sqrt{s} = 13$ TeV collected with the ATLAS detector during Run 2 of the Large Hadron Collider, corresponding to an integrated luminosity of 139 fb^{-1} [1]. The results are interpreted in the context of various R-parity-conserving models where squarks and gluinos are pair-produced and the neutralino is the lightest supersymmetric particle. An exclusion limit at the 95% confidence level on the mass of the gluino is set at 2.35 TeV for a simplified model incorporating only a gluino and the lightest neutralino, assuming the lightest neutralino is massless. For a simplified model involving the strong production of mass-degenerate first- and second-generation squarks, the results exclude at 95% confidence level squark masses below 1.94 TeV are excluded if the lightest neutralino is massless. This is `CheckMATE` implementation validated for version 2 [2, 3].

2 Validation

Processes analyzed for validation:

- $p\,p \to \tilde{g}\,\tilde{g} \to q\,\bar{q}\,\tilde{\chi}_1^0\,q\,\bar{q}\,\tilde{\chi}_1^0$
  squarks decoupled
  Events generated with `MG5_aMC 2.6.0` [5] interfaced to `Pythia8` [6] with up to two extra partons (CKKW-L).
- $p\,p \to \tilde{g}\,\tilde{g} \to q\,\bar{q}'\,\tilde{\chi}_1^\pm\,q\,\bar{q}'\,\tilde{\chi}_1^\pm;\qquad \tilde{\chi}_1^\pm \to W^\pm\,\tilde{\chi}_1^0$
  squarks decoupled
  Events generated with `MG5_aMC 2.6.0` [5] interfaced to `Pythia8` [6] with up to two extra partons (CKKW-L).
- $p\,p \to \tilde{q}\,\tilde{q} \to q\,\tilde{\chi}_1^0\,q\,\tilde{\chi}_1^0$
  gluinos decoupled
  Events generated with `MG5_aMC 2.6.0` [5] interfaced to `Pythia8` [6] with up to two extra partons (CKKW-L).
- $p\,p \to \tilde{q}\,\tilde{q} \to q'\,\tilde{\chi}_1^\pm\,q'\,\tilde{\chi}_1^\pm;\qquad \tilde{\chi}_1^\pm \to W^\pm\,\tilde{\chi}_1^0$
  gluinos decoupled
  Events generated with `MG5_aMC 2.6.0` [5] interfaced to `Pythia8` [6] with up to two extra partons (CKKW-L).

	Selection	$m_{\tilde{q}} = 1200$ GeV $m_{\tilde{\chi}_1^0} = 600$ GeV		$m_{\tilde{q}} = 1400$ GeV $m_{\tilde{\chi}_1^0} = 600$ GeV		$m_{\tilde{q}} = 1600$ GeV $m_{\tilde{\chi}_1^0} = 400$ GeV	
		ATLAS	CM	ATLAS	CM	ATLAS	CM
Generated MC events		10000	10000	6000	10000	6000	10000
Common Requirements	Preselection, $E_{\mathrm{T}}^{\mathrm{miss}} > 300$ GeV, $p_T(j_1) > 200$ GeV, $m_{\mathrm{eff}} > 800$ GeV	1763	1780	541	546	174	176
	jet multiplicity ≥ 2	1763	1780	541	546	174	176
	Cleaning cuts	1746	–	535	–	173	–
SR-2j-1600	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.8$	1433	1434	431	433	136	139
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1377	1353	411	410	129	130
	$p_T(j_2) > 250$ GeV	853	850	311	310	111	112
	$\lvert\eta(j_{1,2})\rvert < 2.0$	836	832	306	305	109	110
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	568	554	228	227	86.4	87.3
	m_{eff}(incl.) > 1600 GeV	366	362	202	195	83.5	84.2
SR-2j-2200	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1603	1619	483	492	156	158
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	1567	1566	470	476	151	153
	$p_T(j_1) > 600$ GeV	509	514	269	259	120	121
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	337	339	201	188	94.6	95.7
	m_{eff}(incl.) > 2200 GeV	101	96	108	101	76.1	76.4
SR-2j-2800	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.8$	1433	1434	431	433	136	138
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1377	1352	411	410	129	130
	$p_T(j_2) > 250$ GeV	853	850	311	311	111	112
	$\lvert\eta(j_{1,2})\rvert < 1.2$	655	653	235	239	82.3	84.3
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	439	433	173	178	64.6	66.4
	m_{eff}(incl.) > 2800 GeV	15.6	10.5	18.8	15.1	29.1	27.0

Table 1: Events normalized to the nominal cross section at NNLO-NNLL and 139 fb^{-1}.

	Selection	$m_{\tilde{g}} = 1400$ GeV $m_{\tilde{\chi}_1^0} = 1000$ GeV		$m_{\tilde{q}} = 1800$ GeV $m_{\tilde{\chi}_1^0} = 1000$ GeV		$m_{\tilde{q}} = 2200$ GeV $m_{\tilde{\chi}_1^0} = 600$ GeV	
		ATLAS	CM	ATLAS	CM	ATLAS	CM
Generated MC events		60000	60000	60000	10000	50000	5000
Common Requirements	Preselection, $E_{\mathrm{T}}^{\mathrm{miss}} > 300$ GeV, $p_T(j_1) > 200$ GeV, $m_{\mathrm{eff}} > 800$ GeV	2562	2561	467	467	57.6	57.7
	jet multiplicity ≥ 2	2562	2561	467	467	57.6	57.7
	Cleaning cuts	2532	–	461	–	56.8	–
SR4j-1000	jet multiplicity ≥ 4	1931	1991	410	421	53.5	54.4
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1718	1778	357	365	44.7	45.6
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	1583	1629	322	330	39.8	40.2
	$p_T(j_4) > 100$ GeV	661	697	234	234	35.3	34.9
	$\lvert\eta(j_{1,2,3,4})\rvert < 2.0$	574	600	214	215	32.1	31.9
	Aplanarity > 0.04	429	484	159	168	22.3	22.7
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	149	164	82.7	86.0	13.9	14.0
	m_{eff}(incl.) > 1000 GeV	149	163	82.7	86.0	13.9	14.0
SR4j-2200	jet multiplicity ≥ 4	1931	1991	410	421	53.5	54.4
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1718	1778	357	365	44.7	45.6
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	1583	1629	322	330	39.8	40.2
	$p_T(j_4) > 100$ GeV	661	697	234	233	35.3	34.9
	$\lvert\eta(j_{1,2,3,4})\rvert < 2.0$	574	600	214	215	32.1	31.9
	Aplanarity > 0.04	429	484	159	168	22.3	22.7
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	149	164	82.7	86.0	13.9	14.0
	m_{eff}(incl.) > 2200 GeV	13.7	13.4	34.9	37.6	13.6	13.7
SR4j-3400	jet multiplicity ≥ 4	1931	1991	410	421	53.5	54.4
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1718	1778	357	365	44.7	45.6
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	1583	1629	322	330	39.8	40.2
	$p_T(j_4) > 100$ GeV	661	697	234	233	35.3	34.9
	$\lvert\eta(j_{1,2,3,4})\rvert < 2.0$	574	600	214	215	32.1	31.9
	Aplanarity > 0.04	429	484	159	168	22.3	22.7
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 10$ GeV$^{1/2}$	398	446	142	151	19.6	20.0
	m_{eff}(incl.) > 3400 GeV	0.279	0	1.43	2.52	8.04	7.6

Table 2: ATLAS cross section normalization seems to be wrong. Events for `CheckMATE` normalized to ATLAS preselection row. The $\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}})$ cut in Table 17 of [1] does not match Table 8: $\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$ vs. $\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$. The numbers here were produced according to prescription in this table, ie. not with the cut implemented in the final version.

	Selection	$m_{\tilde{g}} = 1400$ GeV $m_{\tilde{\chi}_1^\pm} = 1100$ GeV $m_{\tilde{\chi}_1^0} = 800$ GeV		$m_{\tilde{g}} = 2000$ GeV $m_{\tilde{\chi}_1^\pm} = 1500$ GeV $m_{\tilde{\chi}_1^0} = 1000$ GeV		$m_{\tilde{g}} = 2200$ GeV $m_{\tilde{\chi}_1^\pm} = 1200$ GeV $m_{\tilde{\chi}_1^0} = 200$ GeV	
		ATLAS	CM	ATLAS	CM	ATLAS	CM
Generated MC events		30000	30000	30000	20000	30000	5000
Common Requirements	Preselection, $E_{\mathrm{T}}^{\mathrm{miss}} > 300$ GeV, $p_T(j_1) > 200$ GeV, $m_{\mathrm{eff}} > 800$ GeV	1160	1131	64.3	64.0	25.4	26.0
	jet multiplicity ≥ 2	1160	1131	64.3	64.0	25.4	26.0
	Cleaning cuts	1147	–	63.5	–	25.1	–
SR5j-1600	jet multiplicity ≥ 5	1022	1026	60.2	60.1	24.2	25.4
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	895	892	52.0	52.5	20.4	21.2
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	783	765	43.6	43.3	16.5	16.9
	$p_T(j_1) > 600$ GeV	46.2	43.1	10.7	9.9	13.1	13.4
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	18.6	15.4	4.86	4.27	6.38	6.33
	m_{eff}(incl.) > 1600 GeV	18.4	15.2	4.86	4.27	6.38	6.33
SR6j-1000	jet multiplicity ≥ 6	798	824	50.7	52.4	21.7	23.0
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	700	717	43.6	45.3	18.1	19.2
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	600	604	35.9	36.7	14.4	15.2
	$p_T(j_6) > 75$ GeV	313	329	25.7	26.0	12.3	13.0
	$\lvert\eta(j_{1,2,3,4,5,6})\rvert < 2.0$	260	277	22.6	22.6	10.5	11.1
	Aplanarity > 0.08	171	199	16.0	16.8	7.28	7.80
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	42.8	47.1	6.91	6.7	3.58	3.74
	m_{eff}(incl.) > 1000 GeV	42.8	47.1	6.91	6.7	3.58	3.74
SR6j-2200	jet multiplicity ≥ 6	798	823	50.7	52.4	21.7	23.0
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	700	717	43.6	45.3	18.1	19.2
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	600	604	35.9	36.7	14.4	15.2
	$p_T(j_6) > 75$ GeV	313	329	25.7	26.0	12.3	13.0
	$\lvert\eta(j_{1,2,3,4,5,6})\rvert < 2.0$	260	277	22.6	22.6	10.5	11.1
	Aplanarity > 0.08	171	199	16.0	16.8	7.28	7.80
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	42.8	47.1	6.91	6.7	3.58	3.74
	m_{eff}(incl.) > 2200 GeV	4.96	5.6	4.87	4.6	3.57	3.73
SR6j-3400	jet multiplicity ≥ 6	798	823	50.7	52.4	21.7	23.0
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	700	717	43.6	45.3	18.1	19.2
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	600	604	35.9	36.7	14.4	15.2
	$p_T(j_6) > 75$ GeV	313	329	25.7	26.0	12.3	13.0
	$\lvert\eta(j_{1,2,3,4,5,6})\rvert < 2.0$	260	277	22.6	22.6	10.5	11.1
	Aplanarity > 0.08	171	199	16.0	16.8	7.28	7.80
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 10$ GeV$^{1/2}$	143	165	13.5	14.1	6.03	6.44
	m_{eff}(incl.) > 3400 GeV	0.152	0	0.260	0.294	3.56	3.53

Table 3: Events normalized to the nominal cross section at NNLO-NNLL and 139 fb^{-1}.

	Selection	$m_{\tilde{q}} = 800$ GeV $m_{\tilde{\chi}_1^\pm} = 600$ GeV $m_{\tilde{\chi}_1^0} = 400$ GeV	
		ATLAS	CM
Generated MC events		30000	50000
Common Requirements	Preselection, $E_{\mathrm{T}}^{\mathrm{miss}} > 300$ GeV,		
	$p_T(j_1) > 200$ GeV, $m_{\mathrm{eff}} > 800$ GeV	6101	6095
	jet multiplicity ≥ 2	6101	6095
	Cleaning cuts	6039	–
SR5j-1600	jet multiplicity ≥ 5	3513	3914
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	2985	3280
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	2669	2887
	$p_T(j_1) > 600$ GeV	240	284
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	68.4	101
	m_{eff}(incl.) > 1600 GeV	68.4	101
SR6j-1000	jet multiplicity ≥ 6	1752	2311
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1448	1910
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	1252	1608
	$p_T(j_6) > 75$ GeV	388	561
	$\lvert\eta(j_{1,2,3,4,5,6})\rvert < 2.0$	250	411
	Aplanarity > 0.08	123	216
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	10.4	16.3
	m_{eff}(incl.) > 1000 GeV	10.4	16.3
SR6j-2200	jet multiplicity ≥ 6	1752	2311
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1448	1910
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	1252	1608
	$p_T(j_6) > 75$ GeV	388	561
	$\lvert\eta(j_{1,2,3,4,5,6})\rvert < 2.0$	250	411
	Aplanarity > 0.08	123	216
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 16$ GeV$^{1/2}$	10.4	16.3
	m_{eff}(incl.) > 2200 GeV	3.31	1.22
SR6j-3400	jet multiplicity ≥ 6	1752	2311
	$\Delta\phi(j_{1,2,(3)}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.4$	1448	1910
	$\Delta\phi(j_{i>3}, E_{\mathrm{T}}^{\mathrm{miss}}) > 0.2$	1252	1608
	$p_T(j_6) > 75$ GeV	388	561
	$\lvert\eta(j_{1,2,3,4,5,6})\rvert < 2.0$	250	411
	Aplanarity > 0.08	123	216
	$E_{\mathrm{T}}^{\mathrm{miss}}/\sqrt{H_T} > 10$ GeV$^{1/2}$	84.6	177
	m_{eff}(incl.) > 3400 GeV	0	2.4

Table 4: Events normalized to the nominal cross section at NNLO-NNLL and 139 fb^{-1}. This does not seem to fit well. With other cutflows in agreement this might be due to modeling problems.

3 Code

```
#include "atlas_conf_2019_040.h"
// AUTHOR: K. Rolbiecki
//  EMAIL: krolb@fuw.edu.pl
void Atlas_conf_2019_040::initialize() {
  setAnalysisName("atlas_conf_2019_040");
  setInformation(""
    "# search for squarks and gluinos in MET_jet final states\n"
  "");
  setLuminosity(139.0*units::INVFB);
  bookSignalRegions("MB-SSd-2-1000-10;MB-SSd-2-1000-16;MB-SSd-2-1000-22;MB-SSd-2-1600-10;MB-SSd-2-1600-16;
  MB-SSd-2-1600-22;MB-SSd-2-2200-16;MB-SSd-2-2200-22;MB-SSd-2-2800-16;MB-SSd-2-2800-22;MB-SSd-2-3400-22;
  MB-SSd-2-3400-28;MB-SSd-2-4000-22;MB-SSd-2-4000-28;MB-SSd-4-1000-10;MB-SSd-4-1000-16;MB-SSd-4-1000-22;
  MB-SSd-4-1600-10;MB-SSd-4-1600-16;MB-SSd-4-1600-22;MB-SSd-4-2200-16;MB-SSd-4-2200-22;MB-SSd-2-2800-16;
  MB-SSd-2-2800-22;MB-GGd-4-1000-10;MB-GGd-4-1000-16;MB-GGd-4-1000-22;MB-GGd-4-1600-10;MB-GGd-4-1600-16;
  MB-GGd-4-1600-22;MB-GGd-4-2200-10;MB-GGd-4-2200-16;MB-GGd-4-2200-22;MB-GGd-4-2800-10;MB-GGd-4-2800-16;
  MB-GGd-4-2800-22;MB-GGd-4-3400-10;MB-GGd-4-3400-16;MB-GGd-4-3400-22;MB-GGd-4-4000-10;MB-GGd-4-4000-16;
  MB-GGd-4-4000-22;MB-C-2-1600-16;MB-C-2-1600-22;MB-C-2-2200-16;MB-C-2-2200-22;MB-C-2-2800-16;
  MB-C-2-2800-22;MB-C-4-1600-16;MB-C-4-1600-22;MB-C-4-2200-16;MB-C-4-2200-22;MB-C-4-2800-16;
  MB-C-4-2800-22;MB-C-5-1600-16;MB-C-5-1600-22;MB-C-5-2200-16;MB-C-5-2200-22;MB-C-5-2800-16;
  MB-C-5-2800-22;SR-2j-1600;SR-2j-2200;SR-2j-2800;SR-4j-1000;SR-4j-2200;SR-4j-3400;SR-5j-1600;SR-6j-1000;
  SR-6j-2200;SR-6j-3400");
}

void Atlas_conf_2019_040::analyze() {

  missingET->addMuons(muonsCombined);

  electronsLoose = filterPhaseSpace(electronsLoose, 7., -2.47, 2.47);
  muonsCombined = filterPhaseSpace(muonsCombined, 6., -2.7, 2.7);
  jets = filterPhaseSpace(jets, 20., -2.8, 2.8);

  jets = overlapRemoval(jets, electronsLoose, 0.2, "y");
  electronsLoose = specialoverlap(electronsLoose, jets);
  muonsCombined = specialoverlap(muonsCombined, jets);
  jets = overlapRemoval_muon_jet_tracks(jets, muonsCombined, 0.2, 2);

  std::vector<Jet*> sigjets = filterPhaseSpace(jets, 50., -2.8, 2.8);

  countCutflowEvent("00_all");

  double met = missingET->P4().Et();

  if(electronsLoose.size() > 0) return;
  if(muonsCombined.size() > 0) return;
  if (met < 300.) return;
  if ( sigjets.size() < 2 || sigjets[0]->PT < 200. ) return;
  if ( M_eff(sigjets, 0) < 800. ) return;

  countCutflowEvent("01_Preselection");

//  if( dPhi(sigjets, 0) < 0.4 ) return;

//                         PTj1  PTj2  Nj  Eta Phi1 Phi2 ET/HT Apl   Meff
  if (Passes_Cuts(sigjets, 250., 250., 2, 2.0, 0.8, 0.4, 16., 0.,   1600., true, "2j-1600") )
```

```
countSignalEvent("SR-2j-1600");
  if (Passes_Cuts(sigjets, 600., 50.,  2, 2.8, 0.4, 0.2, 16., 0.,   2200., true, "2j-2200") )
countSignalEvent("SR-2j-2200");
  if (Passes_Cuts(sigjets, 250., 250., 2, 1.2, 0.8, 0.4, 16., 0.,   2800., true, "2j-2800") )
countSignalEvent("SR-2j-2800");
  if (Passes_Cuts(sigjets, 200., 100., 4, 2.0, 0.4, 0.4, 16., 0.04, 1000., true, "4j-1000") )
countSignalEvent("SR-4j-1000");
  if (Passes_Cuts(sigjets, 200., 100., 4, 2.0, 0.4, 0.4, 16., 0.04, 2200., true, "4j-2200") )
countSignalEvent("SR-4j-2200");
  if (Passes_Cuts(sigjets, 200., 100., 4, 2.0, 0.4, 0.4, 10., 0.04, 3400., true, "4j-3400") )
countSignalEvent("SR-4j-3400");
  if (Passes_Cuts(sigjets, 600., 50.,  5, 2.8, 0.4, 0.2, 16., 0.,   1600., true, "5j-1600") )
countSignalEvent("SR-5j-1600");
  if (Passes_Cuts(sigjets, 200., 75.,  6, 2.0, 0.4, 0.2, 16., 0.08, 1000., true, "6j-1000") )
countSignalEvent("SR-6j-1000");
  if (Passes_Cuts(sigjets, 200., 75.,  6, 2.0, 0.4, 0.2, 16., 0.08, 2200., true, "6j-2200") )
countSignalEvent("SR-6j-2200");
  if (Passes_Cuts(sigjets, 200., 75.,  6, 2.0, 0.4, 0.2, 10., 0.08, 3400., true, "6j-3400") )
countSignalEvent("SR-6j-3400");

  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 10., 16., 0., 1000., 1600., false, "MB-SSd"))
countSignalEvent("MB-SSd-2-1000-10");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 16., 22., 0., 1000., 1600., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-1000-16");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 22., -1., 0., 1000., 1600., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-1000-22");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 10., 16., 0., 1600., -1.0 , false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-1600-10");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 16., 22., 0., 1600., 2200., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-1600-16");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 22., -1., 0., 1600., 2200., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-1600-22");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 16., 22., 0., 2200., 2800., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-2200-16");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 22., -1., 0., 2200., 2800., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-2200-22");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 16., 22., 0., 2800., -1.0 , false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-2800-16");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 22., -1., 0., 2800., 3400., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-2800-22");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 22., 28., 0., 3400., 4000., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-3400-22");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 28., -1., 0., 3400., 4000., false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-3400-28");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 22., 28., 0., 4000., -1.0 , false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-4000-22");
  if (Passes_Cuts_MB(sigjets, 200., 100., 2, 3, 2.0, 0.8, 0.4, 28., -1., 0., 4000., -1.0 , false, "MB-SSd"))
  countSignalEvent("MB-SSd-2-4000-28");

  if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 10., 16., 0., 1000., 1600., false, "MB-SSd"))
  countSignalEvent("MB-SSd-4-1000-10");
  if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 16., 22., 0., 1000., 1600., false, "MB-SSd"))
  countSignalEvent("MB-SSd-4-1000-16");
  if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 22., -1., 0., 1000., 1600., false, "MB-SSd"))
  countSignalEvent("MB-SSd-4-1000-22");
  if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 10., 16., 0., 1600., -1.0 , false, "MB-SSd"))
  countSignalEvent("MB-SSd-4-1600-10");
```

```
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 16., 22., 0., 1600., 2200., false, "MB-SSd"))
countSignalEvent("MB-SSd-4-1600-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 22., -1., 0., 1600., 2200., false, "MB-SSd"))
countSignalEvent("MB-SSd-4-1600-22");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 16., 22., 0., 2200., 2800., false, "MB-SSd"))
countSignalEvent("MB-SSd-4-2200-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 22., -1., 0., 2200., 2800., false, "MB-SSd"))
countSignalEvent("MB-SSd-4-2200-22");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 16., 22., 0., 2800., -1.0 , false, "MB-SSd"))
countSignalEvent("MB-SSd-4-2800-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.8, 0.4, 22., -1., 0., 2800., -1.0 , false, "MB-SSd"))
countSignalEvent("MB-SSd-4-2800-22");

if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 10., 16., 0.04, 1000., 1600., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-1000-10");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 16., 22., 0.04, 1000., 1600., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-1000-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 22., -1., 0.04, 1000., 1600., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-1000-22");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 10., 16., 0.04, 1600., 2200., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-1600-10");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 16., 22., 0.04, 1600., 2200., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-1600-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 22., -1., 0.04, 1600., 2200., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-1600-22");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 10., 16., 0.04, 2200., 2800., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-2200-10");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 16., 22., 0.04, 2200., 2800., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-2200-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 22., -1., 0.04, 2200., 2800., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-2200-22");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 10., 16., 0.04, 2800., 3400., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-2800-10");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 16., 22., 0.04, 2800., 3400., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-2800-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 22., -1., 0.04, 2800., 3400., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-2800-22");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 10., 16., 0.04, 3400., 4000., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-3400-10");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 16., 22., 0.04, 3400., 4000., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-3400-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 22., -1., 0.04, 3400., 4000., false, "MB-GGd"))
countSignalEvent("MB-GGd-4-3400-22");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 10., 16., 0.04, 4000., -1.0 , false, "MB-GGd"))
countSignalEvent("MB-GGd-4-4000-10");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 16., 22., 0.04, 4000., -1.0 , false, "MB-GGd"))
countSignalEvent("MB-GGd-4-4000-16");
if (Passes_Cuts_MB(sigjets, 200., 100., 4, 0, 2.0, 0.4, 0.2, 22., -1., 0.04, 4000., -1.0 , false, "MB-GGd"))
countSignalEvent("MB-GGd-4-4000-22");

if (Passes_Cuts_MB(sigjets, 600.,  50., 2, 3, 2.8, 0.4, 0.2, 16., 22., 0.0, 1600., 2200., false, "MB-C"))
countSignalEvent("MB-C-2-1600-16");
if (Passes_Cuts_MB(sigjets, 600.,  50., 2, 3, 2.8, 0.4, 0.2, 22., -1., 0.0, 1600., 2200., false, "MB-C"))
countSignalEvent("MB-C-2-1600-22");
if (Passes_Cuts_MB(sigjets, 600.,  50., 2, 3, 2.8, 0.4, 0.2, 16., 22., 0.0, 2200., 2800., false, "MB-C"))
countSignalEvent("MB-C-2-2200-16");
if (Passes_Cuts_MB(sigjets, 600.,  50., 2, 3, 2.8, 0.4, 0.2, 22., -1., 0.0, 2200., 2800., false, "MB-C"))
```

```
  countSignalEvent("MB-C-2-2200-22");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 2, 3, 2.8, 0.4, 0.2, 16., 22., 0.0, 2800., -1.0 , false, "MB-C"))
  countSignalEvent("MB-C-2-2800-16");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 2, 3, 2.8, 0.4, 0.2, 22., -1., 0.0, 2800., -1.0 , false, "MB-C"))
  countSignalEvent("MB-C-2-2800-22");

  if (Passes_Cuts_MB(sigjets, 600.,  50., 4, 4, 2.8, 0.4, 0.2, 16., 22., 0.0, 1600., 2200., false, "MB-C"))
  countSignalEvent("MB-C-4-1600-16");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 4, 4, 2.8, 0.4, 0.2, 22., -1., 0.0, 1600., 2200., false, "MB-C"))
  countSignalEvent("MB-C-4-1600-22");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 4, 4, 2.8, 0.4, 0.2, 16., 22., 0.0, 2200., 2800., false, "MB-C"))
  countSignalEvent("MB-C-4-2200-16");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 4, 4, 2.8, 0.4, 0.2, 22., -1., 0.0, 2200., 2800., false, "MB-C"))
  countSignalEvent("MB-C-4-2200-22");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 4, 4, 2.8, 0.4, 0.2, 16., 22., 0.0, 2800., -1.0 , false, "MB-C"))
  countSignalEvent("MB-C-4-2800-16");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 4, 4, 2.8, 0.4, 0.2, 22., -1., 0.0, 2800., -1.0 , false, "MB-C"))
  countSignalEvent("MB-C-4-2800-22");

  if (Passes_Cuts_MB(sigjets, 600.,  50., 5, 0, 2.8, 0.4, 0.2, 16., 22., 0.0, 1600., 2200., false, "MB-C"))
  countSignalEvent("MB-C-5-1600-16");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 5, 0, 2.8, 0.4, 0.2, 22., -1., 0.0, 1600., 2200., false, "MB-C"))
  countSignalEvent("MB-C-5-1600-22");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 5, 0, 2.8, 0.4, 0.2, 16., 22., 0.0, 2200., 2800., false, "MB-C"))
  countSignalEvent("MB-C-5-2200-16");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 5, 0, 2.8, 0.4, 0.2, 22., -1., 0.0, 2200., 2800., false, "MB-C"))
  countSignalEvent("MB-C-5-2200-22");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 5, 0, 2.8, 0.4, 0.2, 16., 22., 0.0, 2800., -1.0 , false, "MB-C"))
  countSignalEvent("MB-C-5-2800-16");
  if (Passes_Cuts_MB(sigjets, 600.,  50., 5, 0, 2.8, 0.4, 0.2, 22., -1., 0.0, 2800., -1.0 , false, "MB-C"))
  countSignalEvent("MB-C-5-2800-22");

  return;

}

void Atlas_conf_2019_040::finalize() {
  // Whatever should be done after the run goes here
}


bool Atlas_conf_2019_040::sortByPT(Jet *i, Jet *j) { return (i->PT > j->PT); }

double Atlas_conf_2019_040::Aplanarity(std::vector<Jet*> input_jets) {

  double mag = 0.;
  for (int k = 0; k < input_jets.size(); k++)
    mag += pow(input_jets[k]->P4().Rho(),2);

  TMatrixD st(TMatrixD::kZero, TMatrixD(3,3) );
  for (int k = 0; k < input_jets.size(); k++) {
    st(0,0) += input_jets[k]->P4().X()*input_jets[k]->P4().X()/mag;
    st(0,1) += input_jets[k]->P4().X()*input_jets[k]->P4().Y()/mag;
    st(0,2) += input_jets[k]->P4().X()*input_jets[k]->P4().Z()/mag;
    st(1,1) += input_jets[k]->P4().Y()*input_jets[k]->P4().Y()/mag;
    st(1,2) += input_jets[k]->P4().Y()*input_jets[k]->P4().Z()/mag;
    st(2,2) += input_jets[k]->P4().Z()*input_jets[k]->P4().Z()/mag;
```

```
  }
  st(1,0) = st(0,1);
  st(2,0) = st(0,2);
  st(2,1) = st(1,2);

  TMatrixDEigen eigen(st);
  TMatrixD diag = eigen.GetEigenValues();

  std::vector<double> lambdas;
  lambdas.push_back( diag(0,0) );
  lambdas.push_back( diag(1,1) );
  lambdas.push_back( diag(2,2) );
  std::sort (lambdas.begin(), lambdas.end());

  return 1.5*lambdas[0];
}

bool Atlas_conf_2019_040::check_nTrack_jet(Jet* jet, std::vector<Track*> tracks, int nTracksMin) {

  int nTracks = 0;
  for (std::vector<Track*>::iterator it=tracks.begin(); it!=tracks.end(); it++)
    for (int part = 0; part < jet->Particles.GetEntries(); part++)
      if (jet->Particles.At(part) == (*it)->Particle && (*it)->PT > 0.5) nTracks++;

    return nTracks > nTracksMin;
}

double Atlas_conf_2019_040::check_track_pt(Jet* jet, std::vector<Track*> tracks) {

  double track_pt = 0.;

  for (std::vector<Track*>::iterator it=tracks.begin(); it!=tracks.end(); it++)
    for (int part = 0; part < jet->Particles.GetEntries(); part++)
       if (jet->Particles.At(part) == (*it)->Particle && (*it)->PT > 0.5) track_pt += (*it)->PT;

  return track_pt;

}

std::vector<Jet*> Atlas_conf_2019_040::overlapRemoval_muon_jet_tracks(std::vector<Jet*> cand_jets,
                                     std::vector<Muon*> cand_muons, double deltaR, int nTracks){

  std::vector<Jet*> passed;
  for (std::vector<Jet*>::iterator jet = cand_jets.begin(); jet != cand_jets.end(); jet++) {

    if (check_nTrack_jet(*jet, tracks, nTracks)) {
      passed.push_back(*jet);
      continue;
    }
    double dR = deltaR;
    bool iso = true;

    for (std::vector<Muon*>::iterator mu=cand_muons.begin(); mu!=cand_muons.end(); mu++)
      if ((*jet)->P4().DeltaR((*mu)->P4()) < dR) {
iso = false;
break;
      }
```

```
    if (iso) passed.push_back(*jet);
  }

  return passed;
}

std::vector<Jet*> Atlas_conf_2019_040::overlapRemoval_muon_jet_tracks2(std::vector<Jet*> cand_jets,
                                     std::vector<Muon*> cand_muons, double deltaR){

  std::vector<Jet*> passed;
  for (std::vector<Jet*>::iterator jet = cand_jets.begin(); jet != cand_jets.end(); jet++) {

    double dR = deltaR;
    bool iso = true;

    for (std::vector<Muon*>::iterator mu=cand_muons.begin(); mu!=cand_muons.end(); mu++)
       if ((*jet)->P4().DeltaR((*mu)->P4()) < dR && 0.7*check_track_pt(*jet, tracks) < (*mu)->PT &&
                   (*jet)->PT < 0.5*(*mu)->PT ) {
 iso = false;
 break;
       }

     if (iso) passed.push_back(*jet);
  }

  return passed;
}

template <class X, class Y>
std::vector<X*> Atlas_conf_2019_040::specialoverlap(std::vector<X*> candidates, std::vector<Y*> neighbours) {
      // If neighbours are empty, return candidates
      if(neighbours.size() == 0)
        return candidates;
      std::vector<X*> passed_candidates;
      // Loop over candidates
      for(int i = 0; i < candidates.size(); i++) {
        bool overlap = false;
        // If a neighbour is too close, declare overlap, break and don't save candidate
        for(int j = 0; j < neighbours.size(); j++) {
          if (candidates[i]->P4().DeltaR(neighbours[j]->P4()) > 0.2 and
          candidates[i]->P4().DeltaR(neighbours[j]->P4()) < std::min(0.4, 0.04 + 10./candidates[i]->PT) ) {
            overlap = true;
            break;
          }
        }
        if (!overlap)
          passed_candidates.push_back(candidates[i]);
      }
      return passed_candidates;
    }

double Atlas_conf_2019_040::dPhi(std::vector<Jet*> jets, int j) {

  int kmax = 0; int kmin = 0;
  if ( !j ) {
    kmax = std::min(int(jets.size()), 3);
```

```
    kmin = 0;
  }
  else {
    kmax = jets.size();
    kmin = 3;
  }
  double dphimin = 10.;

  for (int k = kmin; k < kmax; k++)
      dphimin = std::min(fabs(jets[k]->P4().DeltaPhi( missingET->P4() )), dphimin);

  return dphimin;
}

double Atlas_conf_2019_040::M_eff(std::vector<Jet*> jets, int j) {

  double Meff = missingET->P4().Et();
  int kmax = 0;
  if( !j ) kmax = jets.size(); else kmax=j;

  for(int i = 0; i < kmax; i++)
    if(jets[i]->PT > 50)  Meff+=jets[i]->PT;

  return Meff;
}

double Atlas_conf_2019_040::HT(std::vector<Jet*> jets) {

  double  PTSum = 0.;
  for (int i = 0; i < jets.size(); i++) PTSum+=jets[i]->PT;
  return PTSum;
}

bool Atlas_conf_2019_040::Passes_Cuts(std::vector<Jet*> jets, double PT1Cut, double PT2Cut, int NJet,
                double EtaCut, double dPhiCut1, double dPhiCut2, double METHTCut, double AplanarityCut,
                double MeffCut, bool cutflow, std::string sr) {

  if ( jets.size() < NJet ) return false;
  if (cutflow) countCutflowEvent(sr+"_02_jetmulti");

  if( dPhi(jets, 0) < dPhiCut1 ) return false;
  if (cutflow) countCutflowEvent(sr+"_03_dPhilow");

  if( dPhi(jets, 1) < dPhiCut2 ) return false;
  if (cutflow) countCutflowEvent(sr+"_04_dPhihigh");

  if ( jets[0]->PT < PT1Cut || jets[NJet-1]->PT < PT2Cut ) return false;

  if (cutflow) countCutflowEvent(sr+"_05_PTjets");

  for (int i = 0; i < NJet; i++)
    if ( fabs(jets[i]->Eta) > EtaCut ) return false;

  if (cutflow) countCutflowEvent(sr+"_06_Etajets");

//  if ( !Aplanarity_Cut(jets, AplanarityCut) ) return false;
  if ( AplanarityCut > 0. and  aplanarity(jets) < AplanarityCut ) return false;
```

```
  if (cutflow) countCutflowEvent(sr+"_07_aplanarity");

  if( missingET->P4().Et()/sqrt( HT(jets) ) < METHTCut ) return false;

  if (cutflow) countCutflowEvent(sr+"_08_METrelative");

  if( M_eff(jets, 0) < MeffCut) return false;
  if (cutflow) countCutflowEvent(sr+"_09_Meff");

  return true;
}


bool Atlas_conf_2019_040::Passes_Cuts_MB(std::vector<Jet*> jets, double PT1Cut, double PT2Cut, int NJetMin,
            int NJetMax, double EtaCut, double dPhiCut1, double dPhiCut2, double METHTMin, double METHTMax,
            double AplanarityCut, double MeffMin, double MeffMax, bool cutflow, std::string sr) {

  if ( jets.size() < NJetMin ) return false;
  if ( NJetMax and jets.size() > NJetMax ) return false;
  if (cutflow) countCutflowEvent(sr+"_02_jetmulti");

  if( dPhi(jets, 0) < dPhiCut1 ) return false;
  if (cutflow) countCutflowEvent(sr+"_03_dPhilow");

  if( dPhi(jets, 1) < dPhiCut2 ) return false;
  if (cutflow) countCutflowEvent(sr+"_04_dPhihigh");

  if ( jets[0]->PT < PT1Cut || jets[NJetMin-1]->PT < PT2Cut ) return false;

  if (cutflow) countCutflowEvent(sr+"_05_PTjets");

  for (int i = 0; i < NJetMin; i++)
    if ( fabs(jets[i]->Eta) > EtaCut ) return false;

  if (cutflow) countCutflowEvent(sr+"_06_Etajets");

//  if ( !Aplanarity_Cut(jets, AplanarityCut) ) return false;
  if ( AplanarityCut > 0. and  aplanarity(jets) < AplanarityCut ) return false;
  if (cutflow) countCutflowEvent(sr+"_07_aplanarity");

  if( missingET->P4().Et()/sqrt( HT(jets) ) < METHTMin ) return false;
  if( METHTMax > 0. and missingET->P4().Et()/sqrt( HT(jets) ) > METHTMax ) return false;

  if (cutflow) countCutflowEvent(sr+"_08_METrelative");

  if( M_eff(jets, 0) < MeffMin) return false;
  if( MeffMax > 0. and M_eff(jets, 0) > MeffMax) return false;
  if (cutflow) countCutflowEvent(sr+"_09_Meff");

  return true;
}
```

Acknowledgements

This research was supported by the National Science Centre, Poland, the Beethoven project under contract DEC-2016/23/G/ST2/04301.

References

[1] The ATLAS collaboration [ATLAS Collaboration], "Search for squarks and gluinos in final states with jets and missing transverse momentum using 139 fb^{-1} of $\sqrt{s}$ =13 TeV pp collision data with the ATLAS detector," ATLAS-CONF-2019-040.

[2] J. S. Kim, D. Schmeier, J. Tattersall and K. Rolbiecki, "A framework to create customised LHC analyses within CheckMATE," Comput. Phys. Commun. **196** (2015) 535 [arXiv:1503.01123 [hep-ph]].

[3] D. Dercks, N. Desai, J. S. Kim, K. Rolbiecki, J. Tattersall and T. Weber, "CheckMATE 2: From the model to the limit," Comput. Phys. Commun. **221** (2017) 383 [arXiv:1611.09856 [hep-ph]].

[4] M. Aaboud *et al.* [ATLAS Collaboration], "Jet energy scale measurements and their systematic uncertainties in proton-proton collisions at $\sqrt{s} = 13$ TeV with the ATLAS detector," Phys. Rev. D **96** (2017) no.7, 072002 [arXiv:1703.09665 [hep-ex]].

[5] J. Alwall *et al.*, "The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations," JHEP **1407** (2014) 079 [arXiv:1405.0301 [hep-ph]].

[6] T. Sjostrand, S. Mrenna and P. Z. Skands, "A Brief Introduction to PYTHIA 8.1," Comput. Phys. Commun. **178** (2008) 852 [arXiv:0710.3820 [hep-ph]].