FIT TO AN ANALYTIC FORM OF THE MEASURED CENTRAL CMS MAGNETIC FIELD

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Vassili Maroussov

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

May 2008

Purdue University

West Lafayette, Indiana

To Natalia. Without her influence and support this work could neither have been started nor completed.

ACKNOWLEDGMENTS

In 1996 I was fortunate enough to be invited to CERN to work in the Magnet Test and Analysis group of LHC division which was led by Peter Sievers. I wish to thank Peter for inviting me and introducing to the other people who were working in his group. I was lucky once again to receive a delay of my French visa and to be involved in the measurement analysis instead of participation in routine magnet measurements (which were on the French side of CERN). I have found that despite the fact that a very long time has passed since James Clerk Maxwell wrote down his equations in 1861 it is still possible to find new beauties in this field.

In autumn of 2006 I became involved with the CMS field mapping project. I would like to thank Slava Klyukhin, who first accepted my assistance in doing the round-the-clock measurements and writing the raw data conversion software. Slava introduced me to Martijn Mulders. Very special thanks to Martijn who trusted in my analysis which initially I was doing on my own in parallel with an “official” analysis group. Without Martijn support my work could never have been converted to a part of the CMS software for the offline tracking.

I thank my advisor at Purdue University, Laszlo Gutay, who tolerated me and my sharp turns for a long time. Laszlo is the one who initiated the contact with the CMS magnet group and supported a radical change of my thesis topic.

Last, but not least I sincerely thank Natalia Batalova for not letting me give up.

TABLE OF CONTENTS

Page

LIST OF FIGURES vi
LIST OF TABLES xvi
ABSTRACT xvii
1 INTRODUCTION 1
1.1 CMS Detector Brief Description 2
1.2 CMS Magnet 4
1.3 Field Uncertainty and the Momentum Resolution 5
2 CMS FIELD MEASUREMENT 6
2.1 CMS coordinate system 6
2.2 Fieldmapper Description 6
2.3 Standard Measurement Procedure 7
2.4 Field Maps Obtained 7
3 FIT IN THE AXISYMMETRIC MODEL (THEORY) 8
3.1 Reducing Dimensionality: Field ϕ–folding 9
3.2 Linear Fit: General Remarks 10
3.3 Linear Fit: Choice of the Basis 10
3.4 Linear Fit: Evaluation of the Expansion, Gain Shifts and Offsets 12
3.5 Non-linear Fit 15
3.6 Mixing Coefficient Recovery 15
4 FIT IN THE AXISYMMETRIC MODEL (REALIZATION) 16
4.1 Notation used 16
4.2 Choice of the expansion degree 16
4.3 Fit Dependence on the Offset Constraint Weight 20
5 FIT IN THE AXISYMMETRIC MODEL (RESULTS) 22

Page

5.1 BFit: a C++ Code That Knows the CMS Field 22
5.2 BFit: verification 22
5.3 BFit: current status 26
6 FIT IN 3 DIMENSIONS (THEORY) 27
6.1 Relation between Harmonic Homogeneous Polynomials and Spherical Harmonics 27
6.2 Derivation of a Complete 3D Basis 28
7 FIT IN 3 DIMENSIONS (PREPARATION) 31
7.1 Fieldmapper Tilt Correction (Theory) 31
7.2 Fieldmapper Tilt Correction (Results) 33
7.3 Fieldmapper Tilt Correction (Errors) 38
7.4 Fieldmapper Tilt Correction (Verification) 39
7.5 Data Smoothing and Elimination of Outliers 41
8 FIT IN 3 DIMENSIONS (RESULTS AND DISCUSSION) 48
8.1 Normal Equation Matrix Evaluation 48
8.2 Results and Discussion 48
9 CONCLUSIONS AND PLANS 51
APPENDIX 78
poly2d_base 78
File: Poly2dbase.h 78
File: Poly2dbase.cxx 80
rz_harm_poly 84
File: rz_harm_poly.h 84
File: rz_harm_poly.cxx 86
HarmBasis3DCyl 89
File: HarmBasis3DCyl.h 89
File: HarmBasis3DCyl.cxx 90
VITA 93
PUBLICATIONS 94

LIST OF FIGURES

Figure Page

Fig. 1.1 LHC layout (north to south view). 1
Fig. 1.2 CMS perspective view with “disks” moved apart. 2
Fig. 1.3 CMS slice in the barrel area. 4
Fig. 1.4 Comparison of detector magnets 4
Fig. 2.1 The fieldmapper schematic design 7
Fig. 2.2 The fieldmapper inside HB at SX5. 7
Fig. 4.1 B_z fit error vs. the expansion degree for nominal field 3.8 T 16
Fig. 4.2 B_z fit error vs. the expansion degree for nominal field 4.0 T 16
Fig. 4.3 Radial corrections for the N-arm for the nominal field value 3.8 T 17
Fig. 4.4 Radial corrections for the N-arm for the nominal field value 4.0 T 17
Fig. 4.5 Radial corrections for the P-arm for the nominal field value 3.8 T 17
Fig. 4.6 Radial corrections for the P-arm for the nominal field value 4.0 T 17
Fig. 4.7 Longitudinal corrections for the N-arm for the nominal field value 3.8 T 18
Fig. 4.8 Longitudinal corrections for the N-arm for the nominal field value 4.0 T 18
Fig. 4.9 Longitudinal corrections for the P-arm for the nominal field value 3.8 T 18
Fig. 4.10 Longitudinal corrections for the P-arm for the nom. field value 4.0 T 18
Fig. 4.11 z-profiles of the fieldmapper arms for the nom. field value 3.8 T and N=16 18
Fig. 4.12 z-profiles of the fieldmapper arms for the nom. field value 4.0 T and N=16 18
Fig. 4.13 Gain corrections for the P-arm for the nominal field value 4.0 T 19
Fig. 4.14 Offset corrections for the P-arm for the nominal field value 4.0 T 19
Fig. 4.15 B_r deviation from average value for the offset constraint weight 0.01 20
Fig. 4.16 B_r deviation from average value for the offset constraint weight 10.0 20
Fig. 4.17 B_z deviation from average value for the offset constraint weight 0.01 20
Fig. 4.18 B_z deviation from average value for the offset constraint weight 10.0 20

Figure Page

Fig. 5.1 B_z field map obtained with *BFit* 22
Fig. 5.2 B_r field map obtained with *BFit* 22
Fig. 5.3 Measured B_z minus *BFit* output at for N-Arm at 2.0 T 23
Fig. 5.4 Measured B_z minus *BFit* output at for P-Arm at 2.0 T 23
Fig. 5.5 Measured B_z minus *BFit* output at for N-Arm at 3.5 T 23
Fig. 5.6 Measured B_z minus *BFit* output at for P-Arm at 3.5 T 23
Fig. 5.7 Measured B_z minus *BFit* output at for N-Arm at 3.8 T 23
Fig. 5.8 Measured B_z minus *BFit* output at for P-Arm at 3.8 T 23
Fig. 5.9 Measured B_z minus *BFit* output at for N-Arm at 3.8 T (2nd run) 24
Fig. 5.10 Measured B_z minus *BFit* output at for P-Arm at 3.8 T (2nd run) 24
Fig. 5.11 Measured B_z minus *BFit* output at for N-Arm at 4.0 T 24
Fig. 5.12 Measured B_z minus *BFit* output at for P-Arm at 4.0 T 24
Fig. 5.13 Measured B_r minus *BFit* output at for N-Arm at 2.0 T 24
Fig. 5.14 Measured B_r minus *BFit* output at for P-Arm at 2.0 T 24
Fig. 5.15 Measured B_r minus *BFit* output at for N-Arm at 3.5 T 25
Fig. 5.16 Measured B_r minus *BFit* output at for P-Arm at 3.5 T 25
Fig. 5.17 Measured B_r minus *BFit* output at for N-Arm at 3.8 T 25
Fig. 5.18 Measured B_r minus *BFit* output at for P-Arm at 3.8 T 25
Fig. 5.19 Measured B_r minus *BFit* output at for N-Arm at 3.8 T (2nd run) 25
Fig. 5.20 Measured B_r minus *BFit* output at for P-Arm at 3.8 T (2nd run) 25
Fig. 5.21 Measured B_r minus *BFit* output at for N-Arm at 4.0 T 26
Fig. 5.22 Measured B_r minus *BFit* output at for P-Arm at 4.0 T 26
Fig. 7.1 B_ϕ dependence on angular and z-index for the N4 probe 33
Fig. 7.2: B_ϕ dependence on angular and z-index for the P4 probe 33
Fig. 7.3 Azimuthal angle ϕ_0 vs. z-index for the N-arm probes. 34
Fig. 7.4 Azimuthal angle ϕ_0 vs. z-index for the P-arm probes. 34
Fig. 7.5 Polar angle θ vs. z-index for the N-arm probes. 34
Fig. 7.6 Polar angle θ vs. z-index for the P-arm probes. 34

Figure Page

Fig. 7.7 θ–ϕ_0 distribution for all z-indices for the N-arm obtained from the B_ϕ..... 35
Fig. 7.8 θ–ϕ_0 distribution for all z-indices for the N-arm obtained from the B_r 35
Fig. 7.9 θ–ϕ_0 distribution for all z-indices for the P-arm obtained from the B_ϕ..... 36
Fig. 7.10 θ–ϕ_0 distribution for all z-indices for the P-arm obtained from the B_r..... 36
Fig. 7.11 X-Y trajectory for all z-indices for the N-arm obtained from the B_ϕ..... 36
Fig. 7.12 X-Y trajectory for all z-indices for the P-arm obtained from the B_ϕ..... 36
Fig. 7.13 X-Y trajectory for all z-indices for the N-arm obtained from the B_r..... 37
Fig. 7.14 X-Y trajectory for all z-indices for the P-arm obtained from the B_r..... 37
Fig. 7.15 *X*-misalignment for N-arm (from B_ϕ) 38
Fig. 7.16 *X*-misalignment for P-arm (from B_r) 38
Fig. 7.17 *Y*-misalignment for N-arm (from B_ϕ)..... 38
Fig. 7.18 *Y*-misalignment for P-arm (from B_r)..... 38
Fig. 7.19 RMS error of the misalignment vs. *z*-index for the N-arm (from B_ϕ) 39
Fig. 7.20 RMS error of the misalignment vs. *z*-index for the N-arm (from B_r)..... 39
Fig. 7.21 RMS error of the misalignment vs. *z*-index for the P-arm (from B_ϕ) 39
Fig. 7.22 RMS error of the misalignment vs. *z*-index for the P-arm (from B_r)..... 39
Fig. 7.23 Comparison of the optical survey results to the X-tilt predicted from the magnetic field data from the 4 T measurements (published by permission of Martijn Mulders, CERN). 40
Fig. 7.24 Comparison of the optical survey results to the Y-tilt predicted from the magnetic field data from the 4 T measurements (published by permission of Martijn Mulders, CERN). 41
Fig. 7.25 Filter error for the N4 probe at $n_\phi = 6$ 42
Fig. 7.26 Filter error for the N4 probe at $n_\phi = 7$ 42
Fig. 7.27 Filter error for the N4 probe at $n_\phi = 8$ 42
Fig. 7.28 Filter error for the N4 probe at $n_\phi = 9$ 42
Fig. 7.29 Filter error for the N4 probe at $n_\phi = 0$ 42
Fig. 7.30 Filter error for the N4 probe at $n_\phi = 11$ 42

Figure Page

Fig. 7.31 Filter error for the N4 probe at $n_\phi = 12$ 43
Fig. 7.32 Filter error for the N4 probe at $n_\phi = 13$ 43
Fig. 7.33 Filter error for the P4 probe at $n_\phi = 0$ 43
Fig. 7.34 Filter error for the P4 probe at $n_\phi = 1$ 43
Fig. 7.35 Filter error for the P4 probe at $n_\phi = 2$ 43
Fig. 7.36 Filter error for the P4 probe at $n_\phi = 3$ 43
Fig. 7.37 Filter error for the P4 probe at $n_\phi = 4$ 44
Fig. 7.38 Filter error for the P4 probe at $n_\phi = 5$ 44
Fig. 7.39 Filter error for the P4 probe at $n_\phi = 6$ 44
Fig. 7.40 Filter error for the P4 probe at $n_\phi = 7$ 44
Fig. 7.41 Filter error for the N5 probe at $n_\phi = 0$ 44
Fig. 7.42 Filter error for the N5 probe at $n_\phi = 1$ 44
Fig. 7.43 Filter error for the N5 probe at $n_\phi = 2$ 45
Fig. 7.44 Filter error for the N5 probe at $n_\phi = 3$ 45
Fig. 7.45 Filter error for the N5 probe at $n_\phi = 4$ 45
Fig. 7.46 Filter error for the N5 probe at $n_\phi = 5$ 45
Fig. 7.47 Filter error for the N5 probe at $n_\phi = 6$ 45
Fig. 7.48 Filter error for the N5 probe at $n_\phi = 7$ 45
Fig. 7.49 Filter error for the P5 probe at $n_\phi = 0$ 46
Fig. 7.50 Filter error for the P5 probe at $n_\phi = 1$ 46
Fig. 7.51 Filter error for the P5 probe at $n_\phi = 2$ 46
Fig. 7.52 Filter error for the P5 probe at $n_\phi = 3$ 46
Fig. 7.53 Filter error for the P5 probe at $n_\phi = 4$ 46
Fig. 7.54 Filter error for the P5 probe at $n_\phi = 5$ 46
Fig. 7.55 Filter error for the P5 probe at $n_\phi = 6$ 47
Fig. 7.56 Filter error for the P5 probe at $n_\phi = 7$ 47
Fig. 8.1 B_z map for the N1 probe at 2 T 52

Figure Page

Fig. 8.2 B_z residuals for the N1 probe at 2 T 52
Fig. 8.3 B_r map for the N1 probe at 2 T 52
Fig. 8.4 B_r residuals for the N1 probe at 2 T 52
Fig. 8.5 B_ϕ map for the N1 probe at 2 T 52
Fig. 8.6 δB_z spread for the N1 probe at 2 T 52
Fig. 8.7 B_z map for the N2 probe at 2 T 53
Fig. 8.8 B_z residuals for the N2 probe at 2 T 53
Fig. 8.9 B_r map for the N2 probe at 2 T 53
Fig. 8.10 B_r residuals for the N2 probe at 2 T 53
Fig. 8.11 B_ϕ map for the N2 probe at 2 T 53
Fig. 8.12 δB_z spread for the N2 probe at 2 T 53
Fig. 8.13 B_z map for the N3 probe at 2 T 54
Fig. 8.14 B_z residuals for the N3 probe at 2 T 54
Fig. 8.15 B_r map for the N3 probe at 2 T 54
Fig. 8.16 B_r residuals for the N3 probe at 2 T 54
Fig. 8.17 B_ϕ map for the N3 probe at 2 T 54
Fig. 8.18 δB_z spread for the N3 probe at 2 T 54
Fig. 8.19 B_z map for the N4 probe at 2 T 55
Fig. 8.20 B_z residuals for the N4 probe at 2 T 55
Fig. 8.21 B_r map for the N4 probe at 2 T 55
Fig. 8.22 B_r residuals for the N4 probe at 2 T 55
Fig. 8.23 B_ϕ map for the N4 probe at 2 T 55
Fig. 8.24 δB_z spread for the N4 probe at 2 T 55
Fig. 8.25 B_z map for the N5 probe at 2 T 56
Fig. 8.26 B_z residuals for the N5 probe at 2 T 56
Fig. 8.27 B_r map for the N5 probe at 2 T 56
Fig. 8.28 B_r residuals for the N5 probe at 2 T 56
Fig. 8.29 B_ϕ map for the N5 probe at 2 T 56

Figure Page

Fig. 8.30 δB_z spread for the N5 probe at 2 T 56
Fig. 8.31 B_z map for the P1 probe at 2 T 57
Fig. 8.32 B_z residuals for the P1 probe at 2 T 57
Fig. 8.33 B_r map for the P1 probe at 2 T 57
Fig. 8.34 B_r residuals for the P1 probe at 2 T 57
Fig. 8.35 B_ϕ map for the P1 probe at 2 T 57
Fig. 8.36 δB_z spread for the P1 probe at 2 T 57
Fig. 8.37 B_z map for the P2 probe at 2 T 58
Fig. 8.38 B_z residuals for the P2 probe at 2 T 58
Fig. 8.39 B_r map for the P2 probe at 2 T 58
Fig. 8.40 B_r residuals for the P2 probe at 2 T 58
Fig. 8.41 B_ϕ map for the P2 probe at 2 T 58
Fig. 8.42 δB_z spread for the P2 probe at 2 T 58
Fig. 8.43 B_z map for the P3 probe at 2 T 59
Fig. 8.44 B_z residuals for the P3 probe at 2 T 59
Fig. 8.45 B_r map for the P3 probe at 2 T 59
Fig. 8.46 B_r residuals for the P3 probe at 2 T 59
Fig. 8.47 B_ϕ map for the P3 probe at 2 T 59
Fig. 8.48 δB_z spread for the P3 probe at 2 T 59
Fig. 8.49 B_z map for the P4 probe at 2 T 60
Fig. 8.50 B_z residuals for the P4 probe at 2 T 60
Fig. 8.51 B_r map for the P4 probe at 2 T 60
Fig. 8.52 B_r residuals for the P4 probe at 2 T 60
Fig. 8.53 B_ϕ map for the P4 probe at 2 T 60
Fig. 8.54 δB_z spread for the P4 probe at 2 T 60
Fig. 8.55 B_z map for the P5 probe at 2 T 61
Fig. 8.56 B_z residuals for the P5 probe at 2 T 61
Fig. 8.57 B_r map for the P5 probe at 2 T 61

Figure Page

Fig. 8.58 B_r residuals for the P5 probe at 2 T 61
Fig. 8.59 B_ϕ map for the P5 probe at 2 T 61
Fig. 8.60 δB_z spread for the P5 probe at 2 T 61
Fig. 8.61 B_z map for the N1 probe at 4 T 62
Fig. 8.62 B_z residuals for the N1 probe at 4 T 62
Fig. 8.63 B_r map for the N1 probe at 4 T 62
Fig. 8.64 B_r residuals for the N1 probe at 4 T 62
Fig. 8.65 B_ϕ map for the N1 probe at 4 T 62
Fig. 8.66 δB_z spread for the N1 probe at 4 T 62
Fig. 8.67 B_z map for the N2 probe at 4 T 63
Fig. 8.68 B_z residuals for the N2 probe at 4 T 63
Fig. 8.69 B_r map for the N2 probe at 4 T 63
Fig. 8.70 B_r residuals for the N2 probe at 4 T 63
Fig. 8.71 B_ϕ map for the N2 probe at 4 T 63
Fig. 8.72 δB_z spread for the N2 probe at 4 T 63
Fig. 8.73 B_z map for the N3 probe at 4 T 64
Fig. 8.74 B_z residuals for the N3 probe at 4 T 64
Fig. 8.75 B_r map for the N3 probe at 4 T 64
Fig. 8.76 B_r residuals for the N3 probe at 4 T 64
Fig. 8.77 B_ϕ map for the N3 probe at 4 T 64
Fig. 8.78 δB_z spread for the N3 probe at 4 T 64
Fig. 8.79 B_z map for the N4 probe at 4 T 65
Fig. 8.80 B_z residuals for the N4 probe at 4 T 65
Fig. 8.81 B_r map for the N4 probe at 4 T 65
Fig. 8.82 B_r residuals for the N4 probe at 4 T 65
Fig. 8.83 B_ϕ map for the N4 probe at 4 T 65
Fig. 8.84 δB_z spread for the N4 probe at 4 T 65
Fig. 8.85 B_z map for the N5 probe at 4 T 66

Figure Page

Fig. 8.86 B_z residuals for the N5 probe at 4 T 66
Fig. 8.87 B_r map for the N5 probe at 4 T 66
Fig. 8.88 B_r residuals for the N5 probe at 4 T 66
Fig. 8.89 B_ϕ map for the N5 probe at 4 T 66
Fig. 8.90 δB_z spread for the N5 probe at 4 T 66
Fig. 8.91 B_z map for the P1 probe at 4 T 67
Fig. 8.92 B_z residuals for the P1 probe at 4 T 67
Fig. 8.93 B_r map for the P1 probe at 4 T 67
Fig. 8.94 B_r residuals for the P1 probe at 4 T 67
Fig. 8.95 B_ϕ map for the P1 probe at 4 T 67
Fig. 8.96 δB_z spread for the P1 probe at 4 T 67
Fig. 8.97 B_z map for the P2 probe at 4 T 68
Fig. 8.98 B_z residuals for the P2 probe at 4 T 68
Fig. 8.99 B_r map for the P2 probe at 4 T 68
Fig. 8.100 B_r residuals for the P2 probe at 4 T 68
Fig. 8.101 B_ϕ map for the P2 probe at 4 T 68
Fig. 8.102 δB_z spread for the P2 probe at 4 T 68
Fig. 8.103 B_z map for the P3 probe at 4 T 69
Fig. 8.104 B_z residuals for the P3 probe at 4 T 69
Fig. 8.105 B_r map for the P3 probe at 4 T 69
Fig. 8.106 B_r residuals for the P3 probe at 4 T 69
Fig. 8.107 B_ϕ map for the P3 probe at 4 T 69
Fig. 8.108 B_ϕ residuals for the P3 probe at 4 T 69
Fig. 8.109 B_z map for the P4 probe at 4 T 70
Fig. 8.110 B_z residuals for the P4 probe at 4 T 70
Fig. 8.111 B_r map for the P4 probe at 4 T 70
Fig. 8.112 B_r residuals for the P4 probe at 4 T 70
Fig. 8.113 B_ϕ map for the P4 probe at 4 T 70

Figure Page

Fig. 8.114 δB_z spread for the P4 probe at 4 T 70
Fig. 8.115 B_z map for the P5 probe at 4 T 71
Fig. 8.116 B_z residuals for the P5 probe at 4 T 71
Fig. 8.117 B_r map for the P5 probe at 4 T 71
Fig. 8.118 B_r residuals for the P5 probe at 4 T 71
Fig. 8.119 B_ϕ map for the P5 probe at 4 T 71
Fig. 8.120 δB_z spread for the P5 probe at 4 T 71
Fig. 8.121 B_z residuals for the N5 probe at 2 T, the outermost *z*-positions are excluded 72
Fig. 8.122 δB_z spread for the N5 probe at 2 T, the outermost *z*-positions are excluded . 72
Fig. 8.123 B_r residuals for the N5 probe at 2 T, the outermost *z*-positions are excluded 72
Fig. 8.124 δB_r spread for the N5 probe at 2 T, the outermost *z*-positions are excluded . 72
Fig. 8.125 The "ridge-to-canyon" structure of the B_r error around ϕ–index = 24 for the N5 probe at 2 T 72
Fig. 8.126 The "double ridge" structure of the B_z error around ϕ–index = 24 for the N5 probe at 2 T 72
Fig. 8.127 B_z residuals for the P5 probe at 2 T, the outermost *z*-positions are excluded 73
Fig. 8.128 δB_z spread for the P5 probe at 2 T, the outermost *z*-positions are excluded .. 73
Fig. 8.129 B_r map error the P5 probe at 2 T, the outermost *z*-positions are excluded 73
Fig. 8.130 δB_r spread for the P5 probe at 2 T, the outermost *z*-positions are excluded .. 73
Fig. 8.131 The "ridge-to-canyon" structure of the B_r error around ϕ–index = 0 for the P5 probe at 2 T 73
Fig. 8.132 The "double ridge" structure of the B_z error around ϕ–index = 0 for the P5 probe at 2 T 73
Fig. 8.133 B_z residuals for the N5 probe at 4 T, the outermost *z*-positions are excluded 74
Fig. 8.134 δB_z spread for the N5 probe at 4 T, the outermost *z*-positions are excluded . 74
Fig. 8.135 B_r residuals for the N5 probe at 4 T, the outermost *z*-positions are excluded 74
Fig. 8.136 δB_r spread for the N5 probe at 4 T, the outermost *z*-positions are excluded . 74

Figure Page

Fig. 8.137 The "ridge-to-canyon" structure of the B_r error around ϕ–index = 24 for the N5 probe at 4 T 74
Fig. 8.138 The "double ridge" structure of the B_z error around ϕ–index = 24 for the N5 probe at 4 T 74
Fig. 8.139 B_z residuals for the P5 probe at 4 T, the outermost z-positions are excluded 75
Fig. 8.140 δB_z spread for the P5 probe at 4 T, the outermost z-positions are excluded .. 75
Fig. 8.141 B_r residuals for the P5 probe at 4 T, the outermost z-positions are excluded 75
Fig. 8.142 δB_r spread for the P5 probe at 4 T, the outermost z-positions are excluded .. 75
Fig. 8.143 The "ridge-to-canyon" structure of the B_r error around ϕ–index = 0 for the P5 probe at 4 T 75
Fig. 8.144 The "double ridge" structure of the B_z error around ϕ–index = 0 for the P5 probe at 4 T 75

LIST OF TABLES

Table Page

Table 1.1 CMS solenoid parameters 4

Table 8.1 The average and RMS of the B_z fit residuals for the N-arm probes at 2 T 49

Table 8.2 B_z The average and RMS of the fit residuals for the P-arm probes at 2 T 49

Table 8.3 The average and RMS of the B_z fit residuals for the N-arm probes at 4 T 49

Table 8.4 B_z The average and RMS of the fit residuals for the P-arm probes at 4 T 49

Table 8.5 The average and RMS of the B_z fit residuals for the N5 and P5 probes at 4 T at shortened z-range 50

ABSTRACT

Maroussov, Vassili Ph.D., Purdue University, May, 2008. Fit to an Analytic Form of the Measured Central CMS Magnetic Field. Major Professor: Laszlo J. Gutay.

The CMS magnetic field measurements inside the hadronic calorimeter barrel (HB) had been carried out at CERN SX5 in November 2006 using the "fieldmapper" device developed at FNAL. The goal of these measurements was to provide field maps for several nominal field values with the precision of $\pm 5 \cdot 10^{-4}$ T, usable for charged particle tracking.

An initial analysis of the measured data indicated presence of large systematic errors, which supposedly were related to fieldmapper misalignments and errors of calibration of the Hall probes.

An algorithm had been developed that allows to eliminate the systematic errors and to recover the actual field. The algorithm is based on the idea of making the measurements self-consistent (that is: the field must satisfy the Maxwell equations) by introducing fitted parameters for amplifier gain shifts and the probes position shifts.

With the help of this algorithm, an analytic field representation, both for the axially symmetric and complete 3D field models, had been obtained which is within $\pm 4 \cdot 10^{-4}$ T agreement with the measured field values in whole volume spanned in measurements except the most remote off center part of it, which makes only 6% of total volume.

Based on this analytic axially symmetric fit, a C++ code (BFit) is developed intended to be used in the CMS offline tracking. This code is recently integrated into the CMSSW software package.

1 INTRODUCTION

The world's highest energy (2×7 TeV) Large Hadron Collider (LHC) [1,2] is under construction in the 27 km (~17 miles) long former LEP [3] tunnel by CERN [4]. Two "general purpose" large detectors, ATLAS (**A** **T**oroidal **L**HC **A**pparatu**s**) [5] and CMS (**C**ompact **M**uon **S**olenoid) [6-8] have been built[1] to study products of the high energy proton-proton collisions. Both entered the commissioning stage at the time of writing.

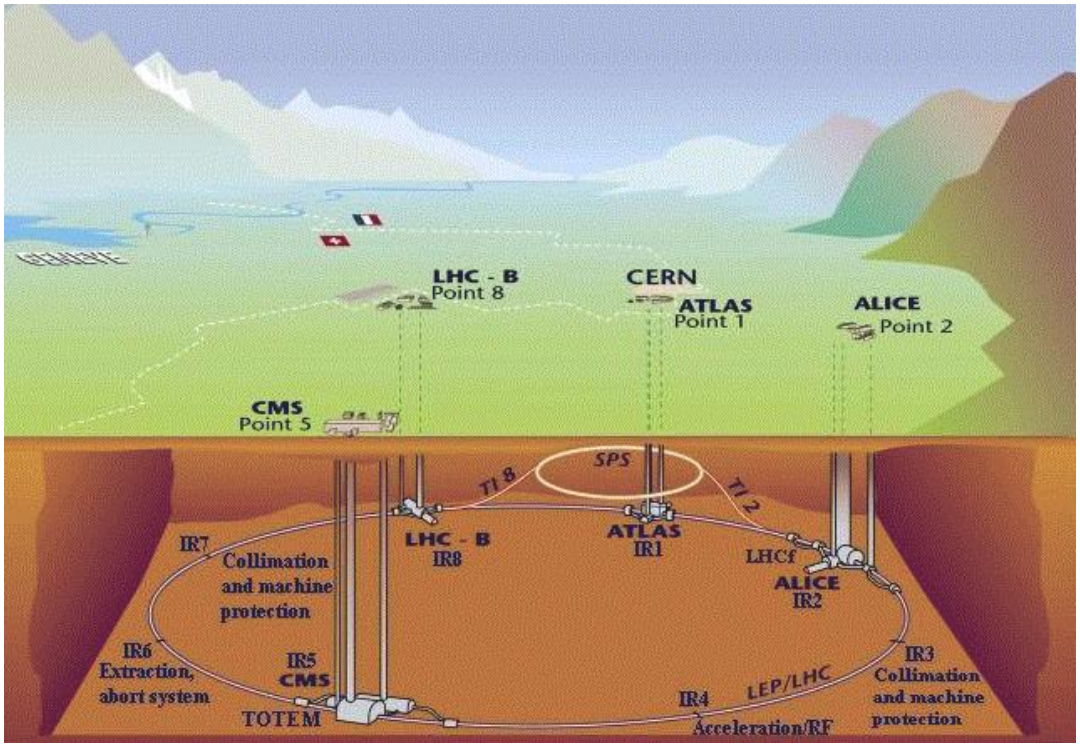


Fig. 1.1 LHC layout (north to south view).

From the beginning Purdue Task D has participated in the construction of the endcaps of CMS. Since it was expected in 2006, that the accelerator would provide colliding beams

[1] Other big LHC experiments are:
ALICE: lead ion colliding beam (quark-gluon plasma study);
LHCb: specialized for B-quark physics study

at the end of 2007, there was a rush to measure and analyze the detector magnetic field before the accelerator started. My advisor suggested I join the CMS magnet group for the duration of measurements to accomplish this goal. As the measurements were completed and the analysis of the data began, it turned out that due to the complexity of removing systematic errors, this analysis became a full-time job. It left no time for other activities and thus became a topic of my thesis.

1.1 CMS Detector Brief Description

The CMS detector has been built by an international collaboration, which involved 155 scientific institutes from 37 countries (Fig. 1.2). Unlike other LHC detectors, which were assembled right in their caverns, large blocks ("disks") of the CMS of up to 2000 tons of weight were constructed at the ground level and then lowered into the cavern.

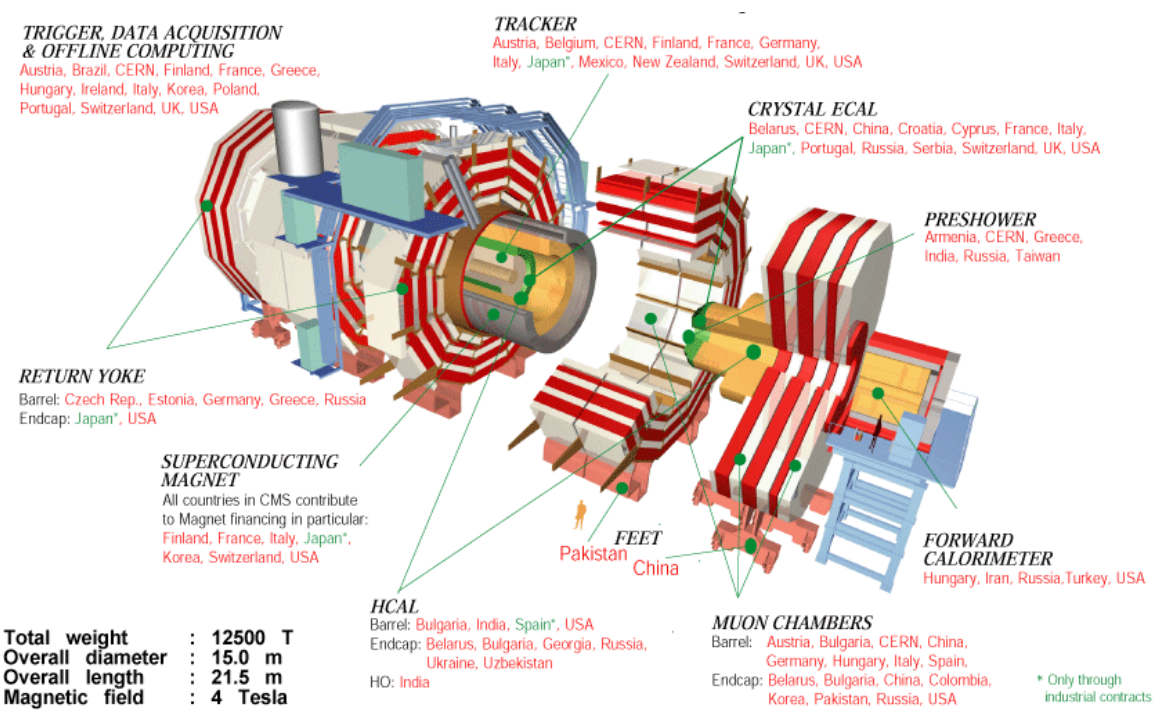


Fig. 1.2 CMS perspective view with "disks" moved apart.

As is common for general purpose detectors, the CMS resembles a Russian nested doll ("matrioshka") composed of layers of particle detectors of different types (Fig. 1.3), each specialized for its task. These cylindrical layers, centered on the beam line and symmetric w.r.t. the collision point, are listed below:

- **Tracker**. Contributes to precise momentum measurement of charged particles; reveals the position of primary vertex (actual collision point) and secondary vertices (points of decay of long lived unstable particles). Made of finely segmented silicon sensors (strips and pixels). The world's largest silicon detector. It has 205 m^2 of silicon sensors comprising 9.3 million microstrips and 66 million pixels [9].
- **Electromagnetic calorimeter (ECAL)**. Primarily is used to detect photons and measure their energy. Together with the tracker it enables $e^{\pm}$ identification. Made of nearly 80.000 crystals of scintillating lead tungstate ($PbWO_4$). A “preshower” detector, based on silicon sensors, further helps particle identification in the endcaps [10].
- **Hadron calorimeter**. Allow the determination of the energy of hadrons. It is made of layers of dense material (brass) interleaved with plastic scintillators [11].
- **The Magnet**. Although it doesn’t detect any particles, I listed it here since it is the field of the magnet that allows the particle momentum calculation by measuring the particle track curvature [12].
- **Muon detectors (and return yoke)**. Intended to identify muons and measure their momenta. CMS uses three types of detectors: drift tubes (DT), cathode strip chambers (CSC) and resistive plate chambers (RPC). The DT's are used for precise trajectory measurements in the central barrel region, while the CSC's are used in the endcaps for the same purpose. The RPC's provide a fast signal when a muon passes through the muon detector, they are installed in both the barrel and the endcaps and are used as a trigger. The return yoke, interleaved with layers of detectors, serves as the muon pass-through filter [13].

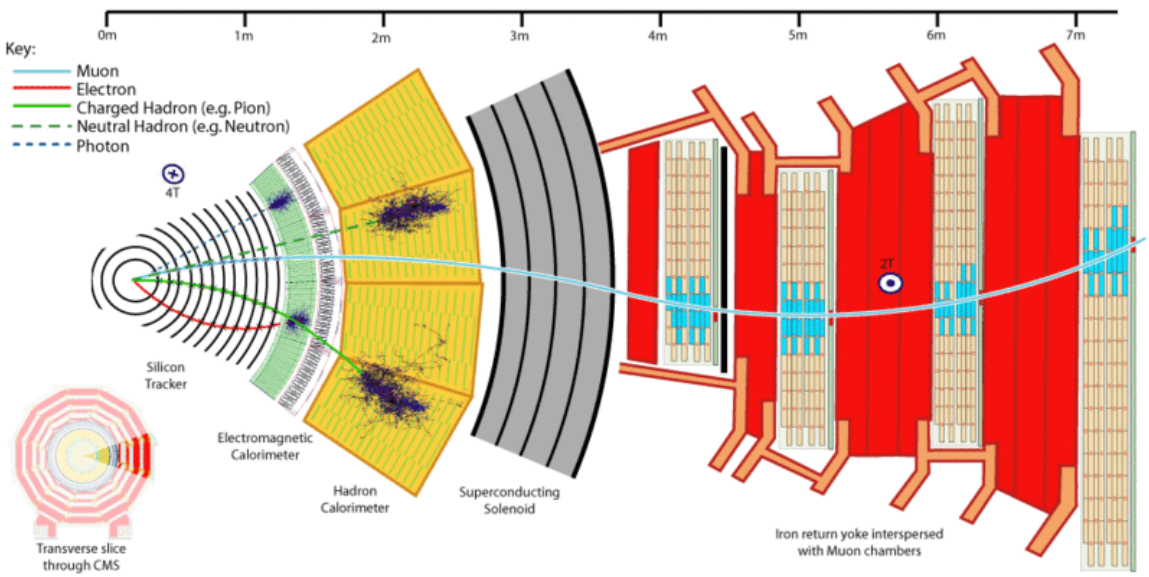


Fig. 1.3 CMS slice in the barrel area

1.2 CMS Magnet

Despite it is called "compact", the CMS magnet is the biggest solenoidal superconducting magnet in the world [12]. Some of the magnet parameters are listed in the Table 1.1

Table 1.1 CMS solenoid parameters

Magnetic length	12.5 m
Free bore diameter	6 m
Central magnetic induction	4 T
Nominal current	20 kA
Stored energy	2.7 GJ
Magnetic Pressure	64 atm
Total conductor length	53 km

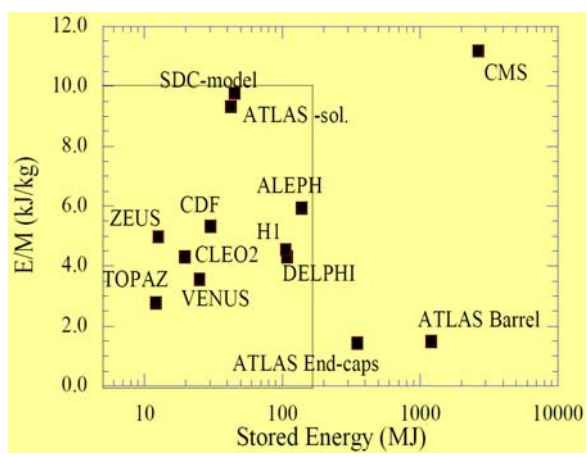


Fig. 1.4 Comparison of detector magnets (by Martijn Mulders, CERN)

In Fig. 1.4 the CMS magnet is compared with several large (both conventional and superconducting) detector magnets. There is the total stored energy at x-axis and the ratio

of stored energy to the magnet mass at y-axis. One can see at the time of writing there is nothing close to this “compact” solenoid in the world.

The magnet return yoke of the barrel has 12-fold rotational symmetry and is assembled of three sections along the *z*-axis; each is split into 4 layers (holding the muon chambers in the gaps). Most of the iron volume is saturated or nearly saturated, so the field in the yoke is just a few times lower than the field in the central volume.

1.3 Field Uncertainty and the Momentum Resolution

For a uniform field in the tracker, momentum resolution would be entirely defined by the spatial resolution of the tracker. However, the CMS magnet has neither iron nor winding shims to achieve the field uniformity; the field strength varies by more than 5% in the tracker volume; meaning an exact knowledge of the magnetic field map is vital for high resolution momentum measurement.

2 CMS FIELD MEASUREMENT

The CMS magnetic field mapping inside the hadronic calorimeter barrel (HB) was carried out at CERN SX5 in November 2006 using the "fieldmapper" device [14]. I participated as an operator in these round the clock measurements, which took more two weeks, and I also developed software that was used for conversion of the raw data (probes readings) to readable format.

2.1 CMS coordinate system

The CMS detector is located at the northernmost point from the LHC center. The origin of the CMS coordinate system is the CMS collision point. Neglecting the small tilt of the LEP/LHC plane, the CMS coordinate system is defined as follows:

- The *x* axis is horizontal, pointing to the LHC center (south).
- The *y* axis is vertical pointing upwards.
- The *z* axis is pointing along the west beam direction, forming a right-handed coordinate system.

The magnetic field of the solenoid is supposed to point into the +z direction. The corresponding cylindrical coordinate system is defined the usual way so the azimuthal angle $\phi = 0$ along the positive *x* direction and ϕ increases as the radius vector *r* rotates toward positive *y* direction. In a corresponding spherical system the polar angle θ is the angle between radial vector *R* and +z direction.

2.2 Fieldmapper Description

The fieldmapper (Fig. 2.1), developed at FNAL, is a magnetic field measuring device that has two mechanical degrees of freedom. It can move along z-axis and rotate its' "propeller" which has two arms on the axle that oriented along z-direction. There are 5 Hall sensor assemblies (each sensor assembly contains 3 mutually orthogonal Hall

probes) and 2 NMR probes installed on each arm of the "propeller". The fieldmapper, driven by a pneumatic gear, can be positioned at one of 122 equidistant (by 5 cm) points in z-direction and has 48 fixed equidistant angular positions. Thus with the Hall probes the fieldmapper provides probe readings in a cylindrical mesh of 122(Z)×48(ϕ)×5(R) dimension for each arm. The distance between arm rotation planes is 95 cm (19 z-steps). The outermost Hall probes are located at radii r = 1.724 m, so the spanned volume during a scan is the cylinder $|z| \leq 3.5$ m, $r \leq 1.724$ m.

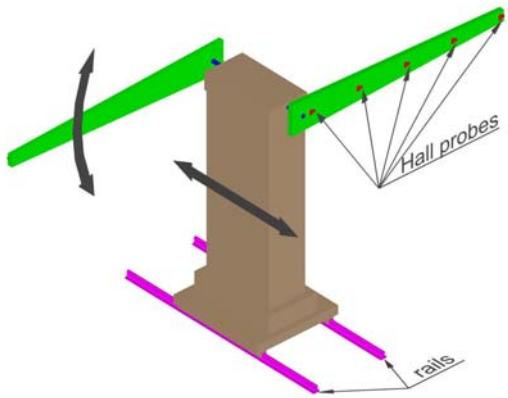


Fig. 2.1 The fieldmapper schematic design

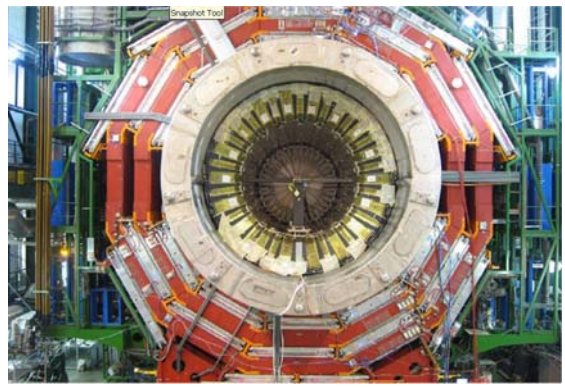


Fig. 2.2 The fieldmapper inside HB at SX5.

2.3 Standard Measurement Procedure

During a standard measurement the fieldmapper starts from the most negative *z*-position and makes first ϕ–scan in the increasing ϕ–direction. After repositioning to the next *z*-point it makes ϕ–scan in the decreasing ϕ–direction, etc., so the rotation direction is alternated at each following step. Full scan with this procedure takes about 60 hours.

2.4 Field Maps Obtained

Using the standard procedure, the field was measured for 2, 3.5, 3.8 (2 times) and 4 T. The map for 3 T was made, by skipping odd *z*-positions, but with 2 ϕ–scans, both clockwise and counter-clockwise for all even *z*-positions. Due to technical problems, which were solved near the end of the field mapping campaign, NMR probe data is available only for the 4 T scan.

3 FIT IN THE AXISYMMETRIC MODEL (THEORY)

Most of the data in the CMS magnetic field mapping had been obtained by means of the Hall probes, which were installed on "arms" of the fieldmapper. Unlike a NMR probe, the Hall probe doesn't yield absolute field value. It is nonlinear, temperature dependent and requires a careful calibration. Such calibration was made for each probe before the installation on the fieldmapper. After the installation probes are moved into a different environment. The wiring is not the same as had been used at the calibration bench. What is more important, all electronics, signal amplifiers for example, operate under different conditions. Therefore, one cannot expect the gain in each channel to be exactly the same as at calibration. That leads to systematic errors.

Other sources of systematic errors are different kinds of misalignments, the fieldmapper axle tilt, deviations of the probes radial and longitudinal positions from design values and probes' tilts relative to the fieldmapper rotation planes. One may note that tilts lead to mixing of the field components, where the B_r and B_ϕ measurements are the most affected, because the B_z is much larger. As for the B_z measurement, in first order it is insensitive to tilts.

Below I describe a method of the systematic error recovery, which gives a field approximation as a byproduct. This method is based on analysis of 2D field obtained by folding the ϕ-dimension. Only the B_z data is used for the fit and systematic errors recovery.

3.1 Reducing Dimensionality: Field ϕ–folding

I average over the ϕ-dimension and consider the "folded" 2D field $\tilde{\mathbf{B}}(r,z)$ defined as:

$$\tilde{\mathbf{B}}(r,z)=\frac{1}{2\pi}\int_0^{2\pi} d\phi\,\mathbf{B}(r,\phi,z) \tag{3.1}$$

The $\tilde{\mathbf{B}}(r,z)$ has only r and z-components, since the "parent" 3D field $\mathbf{B}$ satisfies the Maxwell's equations [15, pp.60-70; 16, p.237]; thus it has both $\nabla\cdot\mathbf{B}(r,z,\phi)=0$ and $\nabla\times\mathbf{B}(r,z,\phi)=0$. It may be shown that the same holds (in 2 dimensions) for the folded 2D field. I first show that $\nabla\cdot\tilde{\mathbf{B}}(r,z)=0$:

$$\begin{aligned}\nabla\cdot\tilde{\mathbf{B}}(r,z)&=\frac{1}{r}\frac{\partial}{\partial r}\left(r\tilde{B}_r\right)+\frac{\partial}{\partial z}\tilde{B}_z=\frac{1}{2\pi}\int_0^{2\pi}d\phi\left(\frac{1}{r}\frac{\partial}{\partial r}\left(rB_r\right)+\frac{\partial}{\partial z}B_z\right)\\&=\frac{1}{2\pi}\int_0^{2\pi}d\phi\left(\underbrace{\nabla\cdot\mathbf{B}}_{=0}-\frac{1}{r}\frac{\partial}{\partial\phi}B_\phi\right)=-\frac{1}{2\pi r}\int_0^{2\pi}d\phi\frac{\partial}{\partial\phi}B_\phi\\&=-\frac{1}{2\pi r}\left(B_\phi(r,2\pi,z)-B_\phi(r,0,z)\right)=0\end{aligned} \tag{3.2}$$

To show that $\nabla\times\tilde{\mathbf{B}}(r,z)=0$, it is enough to consider ϕ-component of $\nabla\times\mathbf{B}(r,z,\phi)$:

$$\nabla\times\tilde{\mathbf{B}}(r,z)=\frac{\partial}{\partial z}\tilde{B}_r-\frac{\partial}{\partial r}\tilde{B}_z=\frac{1}{2\pi}\int_0^{2\pi}d\phi\underbrace{\left(\frac{\partial}{\partial z}B_r-\frac{\partial}{\partial r}B_z\right)}_{=0}=0 \tag{3.3}$$

Equations (3.2) and (3.3) show that $\tilde{\mathbf{B}}$ can be expressed as the gradient of a scalar function of r,z.

Hereafter in this chapter I consider only 2D field and drop the *tilde* over $\mathbf{B}$ in the notation.

In the case if the "parent" 3D field is (nearly[2]) axially symmetric the folded 2D field is insensitive to the radial offset and tilt of the fieldmapper axle in linear order.

The origin of the first order cancellation can be explained as follows: if to tilt the rotation plane, every pair of opposite points on the probe trajectory is displaced by the

[2] We will specify the meaning of "nearly" more precisely while describing the 3D basis for the field fit.

same distance dz in opposite directions; derivatives dB_z/dz are equal, Thus after integration over ϕ there is no change of the 2D ϕ-folded field in first order by the tilt angle. The consideration for radial displacement of the probe circular trajectory is similar. Therefore small errors in the fieldmapper's radial position and axial orientation do not introduce an error in the 2D ϕ-folded field in first order.

3.2 Linear Fit: General Remarks

The linear fit problem is to find a linear combination of basis functions which approximates the experimental data in the best way. Any complete functional basis is of infinite dimension, but in fitting of an experimental data, only a finite number of terms may be used. Therefore the linear fit problem beside the evaluation of the expansion coefficients requires at least two more choices to be made before:

- choice of the basis
- choice of a criterion for an optimal basis truncation

3.3 Linear Fit: Choice of the Basis

In the present work I use the scalar potential approach: I am are looking for a linear combination of the basis scalar functions (each satisfies the Laplace equation) and the gradient of this linear combination that approximates well the field measurement results.

The cylindrical coordinate system is used for the CMS tracking. Keeping in mind that the fit obtained is supposed to be converted to software for the tracking and must provide a fast field calculation, I decided to use homogeneous polynomials of r, z as the basis. Polynomial calculations are fast and the calculus (differentiation, etc.) for them is reduced to simple arithmetic and easy to program.

A general homogeneous polynomial of n-th degree of variables r,z is:

$$p_n(r,z) = \sum_{k=0}^{n} a_{n,k} r^{n-k} z^k \tag{3.4}$$

I require the polynomial to satisfy the Laplace equation in cylindrical coordinates:

$$\begin{aligned}\Delta p_n(r,z) &= \left(\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial}{\partial r}\right)+\frac{\partial^2}{\partial z^2}\right)\sum_{k=0}^{n} a_{n,k} r^{n-k} z^k \\ &= \sum_{k=0}^{n-2}(n-k)^2 a_{n,k} r^{n-k-2} z^k + \sum_{k=2}^{n} k(k-1) a_{n,k} r^{n-k} z^{k-2} \\ &= \sum_{k=0}^{n-2}\left((n-k)^2 a_{n,k} + (k+1)(k+2) a_{n,k+2}\right) r^{n-k-2} z^k \equiv 0\end{aligned} \tag{3.5}$$

Equation (3.5) leads to the following recurrent relation for the polynomial coefficients:

$$a_{n,k+2} = \frac{(n-k)^2}{(k+1)(k+2)} a_{n,k} \tag{3.6}$$

The relation (3.6) has to be accomplished with the condition that the potential must not depend on the r sign, so only even powers of r may be presented. We can then generate all the basis set.

Several basis polynomials for the scalar potential are presented below:

$$\begin{aligned}p_0(r,z) &= 1 \\ p_1(r,z) &= z \\ p_2(r,z) &= \tfrac{1}{2}z^2 - \tfrac{1}{4}r^2 \\ p_3(r,z) &= \tfrac{1}{3}z^3 - \tfrac{1}{2}z\,r^2 \\ p_4(r,z) &= \tfrac{1}{4}z^4 - \tfrac{3}{4}z^2 r^2 + \tfrac{3}{32}r^4\end{aligned} \tag{3.7}$$

Let us define $b_n(r,z) \equiv \frac{\partial}{\partial z} p_{n+1}(r,z)$ The b_n set is the basis for B_z. Several basis polynomials for the B_z are presented below:

$$\begin{aligned}b_0(r,z) &= 1 \\ b_1(r,z) &= z \\ b_2(r,z) &= z^2 - \tfrac{1}{2}r^2 \\ b_3(r,z) &= z^3 - \tfrac{3}{2}z\,r^2 \\ b_4(r,z) &= z^4 - 3z^2 r^2 + \tfrac{3}{8}r^4\end{aligned} \tag{3.8}$$

Overall normalization coefficients in (3.7) are chosen in such a way that at $r = 0$ the B_z basis reduces to $\{1, z, z^2, z^3, \ldots\}$. From the latest it can be seen that the basis is complete and the basis functions are linearly independent.

Similar to (3.8), the set $d_n(r,z) \equiv \frac{\partial}{\partial r} p_{n+1}(r,z)$ gives the basis functions for B_r. Several basis polynomials for the B_r are presented below:

$$\begin{aligned} d_0(r,z) &= 0 \\ d_1(r,z) &= -\tfrac{1}{2} r \\ d_2(r,z) &= -z\,r \\ d_3(r,z) &= -\tfrac{3}{2} z^2 r + \tfrac{3}{8} r^3 \\ d_4(r,z) &= -2z^3 r + \tfrac{3}{2} z\,r^3 \end{aligned} \tag{3.9}$$

3.4 Linear Fit: Evaluation of the Expansion, Gain Shifts and Offsets

If the B_z is measured in points $\vec{r}_k = (r_k, z_k)$, $k = 0,1,\ldots,K$ and the basis $\{b_n(r,z)\}$ is truncated at $n = N$, then for each measurement point we have the following equation:

$$\sum_{n=0}^{N} b_n(r_k, z_k) c_n = B_z(r_k, z_k) \tag{3.10}$$

It leads to the following system of linear equations for finding the vector of the expansion coefficients **C**:

$$\mathbf{AC} = \mathbf{B} \quad \text{here} \quad \mathbf{A} = \begin{pmatrix} b_0(\vec{r}_0) & b_1(\vec{r}_0) & \cdots & b_N(\vec{r}_0) \\ b_0(\vec{r}_1) & b_1(\vec{r}_1) & \cdots & b_N(\vec{r}_1) \\ \vdots & \vdots & \ddots & \vdots \\ b_0(\vec{r}_K) & b_1(\vec{r}_K) & \cdots & b_N(\vec{r}_K) \end{pmatrix}; \quad \mathbf{B} = \begin{pmatrix} B_z(\vec{r}_0) \\ B_z(\vec{r}_1) \\ \vdots \\ B_z(\vec{r}_K) \end{pmatrix} \tag{3.11}$$

With $K > N$ (what we have, indeed) the system above has no exact solution. Instead, we are looking for a solution for vector **C** that minimizes the Euclidian norm of the residual vector, $(\mathbf{AC}\text{-}\mathbf{B})^2$. It can be shown that such a solution is unique, provided all the **A** columns are linearly independent. **C** can be obtained as solution of the following equation with a non-degenerated square matrix:

$$\left(\mathbf{A}^{\mathbf{T}}\mathbf{A}\right)\mathbf{C} = \mathbf{A}^{\mathbf{T}}\mathbf{B} \tag{3.12}$$

The equation in the form above is known as a "normal equation" [17,18].

We now assume that the measurement channels are miscalibrated and the fieldmapper is misaligned, so we have gain shifts $dA_{p(k)}$, offsets $dB_{p(k)}$ and probe position errors $dr_{p(k)}$, $dz_{p(k)}$. Hereafter the $p(k)$ means "index of probe used for the measurement in the k-th point". We then modify the equation (3.10) as follows:

$$\begin{gathered} 1\cdot c_0 + \sum_{n=2}^{N} b_n\left(r_k - dr_{p(k)}, z_k - dz_{p(k)}\right)c_n = \left(1 + dA_{p(k)}\right)\left(B_z\right)_k + dB_{p(k)} \rightarrow \\ \left(1\cdot c_0 + \sum_{n=2}^{N} b_n\left(r_k - dr_{p(k)}, z_k - dz_{p(k)}\right)c_n\right) - \left(B_z\right)_k dA_{p(k)} - 1\cdot dB_{p(k)} = \left(B_z\right)_k \end{gathered} \tag{3.13}$$

Unknowns are the expansion coefficients c_n, gain shifts $dA_{p(k)}$ and offsets $dB_{p(k)}$; the probe position errors $dr_{p(k)}$, $dz_{p(k)}$ are treated as fixed, otherwise the equation turns to non-linear. To obtain a unique, physical solution to the system of equations (3.13), we must impose additional constraints:

- To avoid the trivial non-physical solution $dA_{p(k)} = -1$, $dB_{p(k)} = 0$, $c_n = 0$, we require $\Sigma dA_{p(k)} = 0$; it follows that we recover the field to within a factor. This is natural; there is no way to carry out an *absolute* field post-calibration by means of mathematical tricks.
- Since a simultaneous shift of all $dB_{p(k)}$ is equivalent to shift of the expansion coefficient c_0, we require $\Sigma dB_{p(k)} = 0$. It will be shown later, that this constraint does not introduce any error.
- We set $c_1 = 0$ in (3.13) for reasons for that will be explained in Section 3.5.

The conditions above are sufficient to ensure that the system of equations (3.13) has a unique solution. However, we do more steps.

One must take into account that the field measured is nearly homogeneous; B_k varies slightly from point to point. Loosely speaking, the coefficients of dB_p and dA_p are almost proportional in all equations and the residuals stay almost unchanged under the transformations

$$dB_p \to dB_p + \varepsilon\,; \quad dA_p \to dA_p - \overline{(B_z)_p} \cdot \varepsilon \tag{3.14}$$

Here $\overline{(B_z)_p}$ denotes the mean of B_z over all measurement points for the p-th probe.

The "weak invariance" of residuals under the transformation (3.14) leads to large errors in the determination of dB_p and dA_p unless an additional condition is applied. Such a condition cannot arise from the mathematical formalism, rather physical reasons should be used instead. We discuss them below.

The parameter actually measured is the voltage on a Hall probe. There are 2 main reasons for voltage offsets to appear. The first one is possible thermo-galvanic contacts in the wiring. To the best our knowledge, all connectors used in the wiring between probes and differential amplifier inputs are symmetric, therefore thermo-compensated. Thus we do not expect offsets due to the thermo-galvanic effect. The second reason is an intrinsic input amplifier offset. For the amplifiers used this is approximately 1 μV. The voltage on the Hall probes at nominal CMS field is ~0.3 V. Thus a rough estimate of the relative offset is $\sim 3\times 10^{-6}$.

As for gain shift, the main reason for its appearance is a deviation of resistors from their nominal values in the amplifier feedback circuit, the (double integration type) ADC circuit and the current sources of the probes. In our measurements neither amplifiers nor the ADC's were thermo-conditioned. Experience with electronics design suggests that the expected long-term stability of the resistance may be on the order of 10^{-3} to 10^{-4}.

Summarizing all the above we expect the gain shift to be the main source of an error. Therefore we augmented the system (3.13) with the additional set of equations $W_{dB} \cdot dB_p = 0$, where W_{dB} is a weight coefficient, which defines how important is to keep dB_p close to zero. In other words, with by $W_{dB} \neq 0$ we instruct our solver to use more dA_p, not dB_p, while finding a solution that minimizes the residual vector norm.

While the choice of the weight coefficient W_{dB} is highly ambiguous, it will be shown later that the influence of this choice on the fitted field map is very weak. By introducing this additional set of equations we obtain more realistic corrections.

3.5 Non-linear Fit

For any choice of $dr_{p(k)}$, $dz_{p(k)}$ the system of equations (3.13) with additional constraints discussed in previous chapter, yields a unique solution and a unique norm of the residual vector. Thus we may consider this norm as a function of $dr_{p(k)}$, $dz_{p(k)}$ and minimize it. In order to make this procedure stable we apply the following constraints:

- The expansion coefficient for $b_1(r,z)$, the only basis function with non-zero derivative at $z=0$, is equal to 0; that is $z=0$ corresponds to the field maximum on the axis. If such a constraint were not applied, the fit error minimum would not be unique; its value would remain constant under a simultaneous shift of all probes. This is why we take $c_1=0$ in (3.13)
- The radii of innermost probes (N1,P1) are fixed. This constraint is applied because the innermost probes are very close to the axis, where the r-derivative of the field vanishes, thus the minimum value is almost insensitive to the N1 and P1 radii.

In the implementation of the non-linear fit we used the MIGRAD method of the MINUIT package [19] which is embedded in the ROOT framework [20].

3.6 Mixing Coefficient Recovery

Due to a local (relative to the arm rotation plane) misalignment each B_r probe picks up a fraction of the B_z; We refer to this as local $B_z \rightarrow B_r$ mixing. Assume that after the B_z fit we have fitted functions $F_r(r,z)$ and $F_z(r,z)$ for the r and z components, respectively. Then the mixing coefficients $m_{p(k)}$,the gain shifts $dA_{p(k)}$ and offsets $dB_{p(k)}$ for B_r may be recovered by solving the following system of equations:

$$\begin{aligned} &\left(1+dA_{p(k)}\right)\left(B_r\right)_k + m_{p(k)}F_z\left(r_k,z_k\right)+dB_{p(k)} = F_r\left(r_k,z_k\right) \quad \rightarrow \\ &F_z\left(r_k,z_k\right)m_{p(k)} + \left(B_r\right)_k dA_{p(k)} + 1\cdot dB_{p(k)} = F_r\left(r_k,z_k\right) - \left(B_r\right)_k \end{aligned} \tag{3.15}$$

To improve the evaluation precision of $m_{p(k)}$, $dA_{p(k)}$ and $dB_{p(k)}$, only the data for the (r, z) area of the highest quality B_z fit must be used.

4 FIT IN THE AXISYMMETRIC MODEL (REALIZATION)

In this chapter we describe how the program outlined in Chapter 3 was realized. Some results are presented only for certain nominal field values, however this program was carried out for all field maps measured.

4.1 Notation used

There are 5 field probes on each fieldmapper arm. The probes on the "negative" arm will be referred as N1,N2,…,N5 and those on the "positive" arm as P1,P2,…,P5, where N1 and P1 are the probes on the innermost radius. Each physical parameter related to a probe has a subscript with the probe name.

4.2 Choice of the expansion degree

The B_z non-linear field fit was carried out for polynomials of degree 6 through 21. The dependence of the fit error on the expansion degree N for nominal field values of 3.8 T and 4.0 T is shown in Fig. 4.1 and Fig. 4.2, respectively.

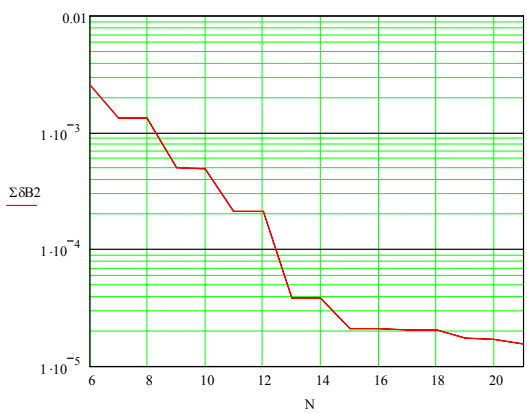


Fig. 4.1 B_z **fit error vs. the expansion degree for nominal field 3.8 T**

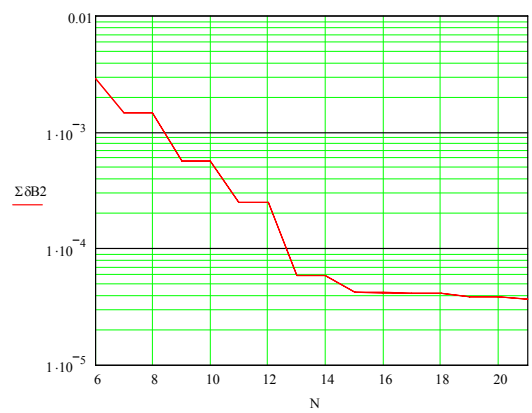


Fig. 4.2 B_z **fit error vs. the expansion degree for nominal field 4.0 T**

As can be seen from the above figures, the fit quality stays almost constant for $N \geq 15$. Below we consider the dependence of various correction parameters on N to specify an optimal expansion degree more precisely.

The dependence of correction to a probe's radial positions for each arm for nominal field values of 3.8 T and 4.0 T is shown in Fig. 4.3 through Fig. 4.6

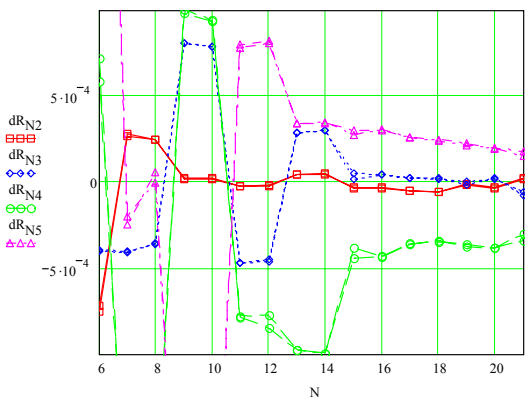


Fig. 4.3 Radial corrections for the N-arm for the nominal field value 3.8 T

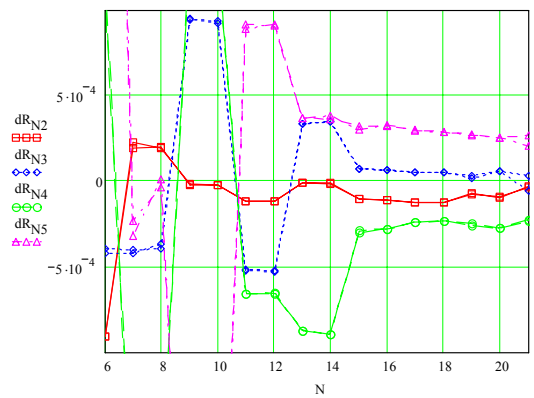


Fig. 4.4 Radial corrections for the N-arm for the nominal field value 4.0 T

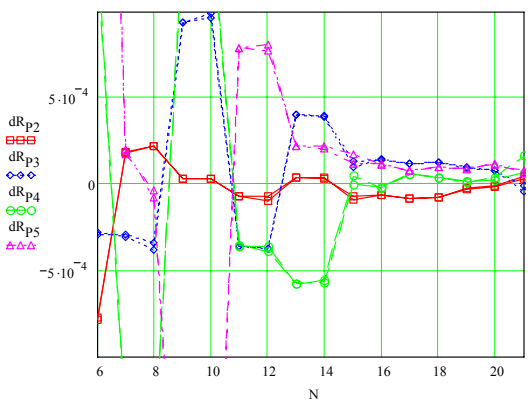


Fig. 4.5 Radial corrections for the P-arm for the nominal field value 3.8 T

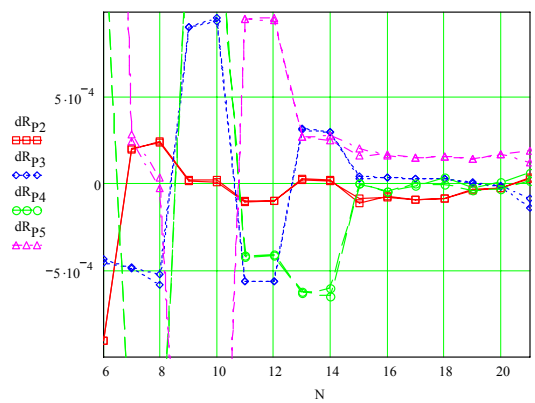


Fig. 4.6 Radial corrections for the P-arm for the nominal field value 4.0 T

A "plateau" can be seen in the plots above for expansion degree in the range 15-18. Though the relative correction errors, which can be roughly estimated both from variations on the plateau and from difference between two field values, are quite large, the corrections themselves are rather small, all being below 0.5 mm, smaller than probe's size.

A much more pronounced plateau for the same expansion degree range of 15-18 is observed for longitudinal corrections (Fig. 4.7 through Fig. 4.10).

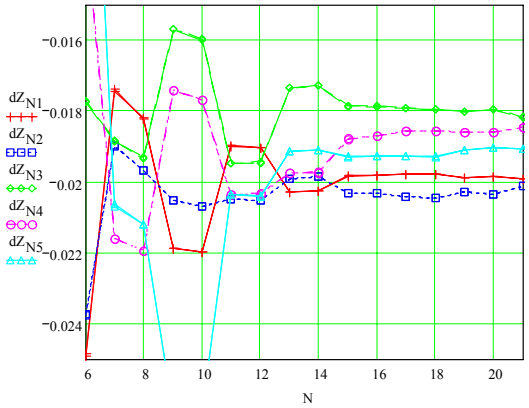


Fig. 4.7 Longitudinal corrections for the N-arm for the nominal field value 3.8 T

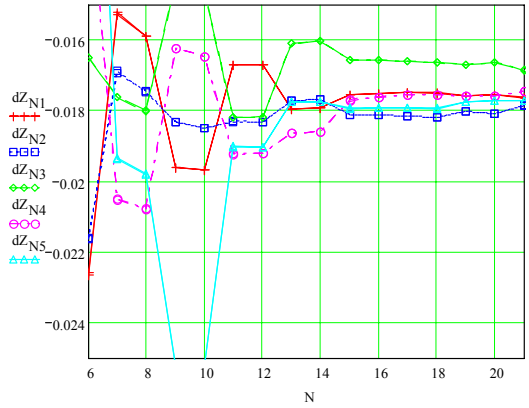


Fig. 4.8 Longitudinal corrections for the N-arm for the nominal field value 4.0 T

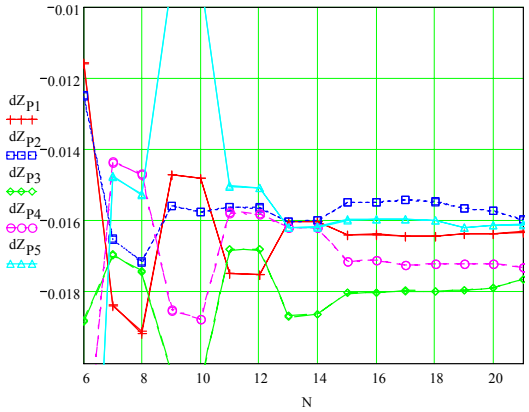


Fig. 4.9 Longitudinal corrections for the P-arm for the nominal field value 3.8 T

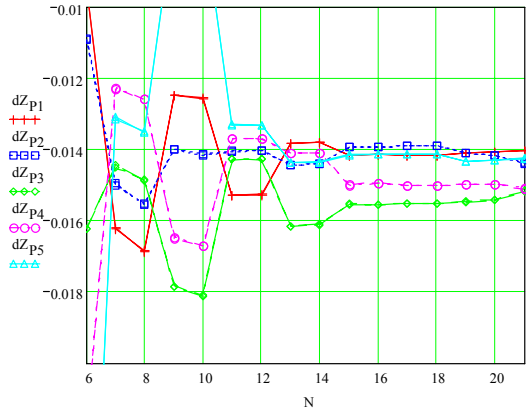


Fig. 4.10 Longitudinal corrections for the P-arm for the nom. field value 4.0 T

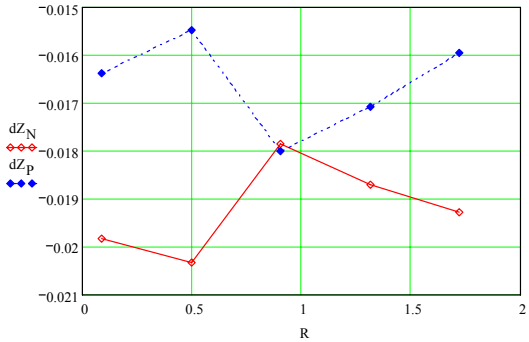


Fig. 4.11 z-profiles of the fieldmapper arms for the nom. field value 3.8 T and *N*=16

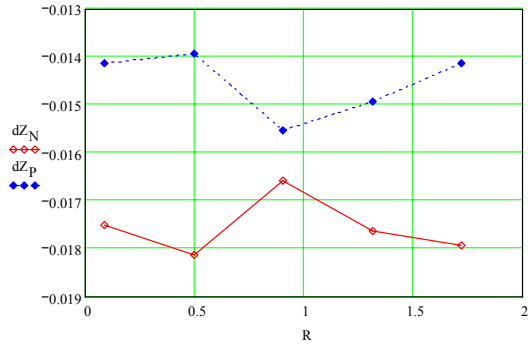


Fig. 4.12 z-profiles of the fieldmapper arms for the nom. field value 4.0 T and *N*=16

Z-profiles of the fieldmapper arms for the expansion degree $N = 16$ and for nominal field values of 3.8 T and 4.0 T are shown in Fig. 4.11 and Fig. 4.12, respecpectively. There is nothing wrong with both arms having a common shift of longitudinal corrections with a change in field. In the coordinate system used, $z = 0$ corresponds to the maximum of the field on the axis ($r = 0$), so with a change in the field, the coordinate origin can "slip" in general.

Gain and offset corrections for the P-arm a for nominal field values of 4.0 T are shown in Fig. 4.13 and Fig. 4.14

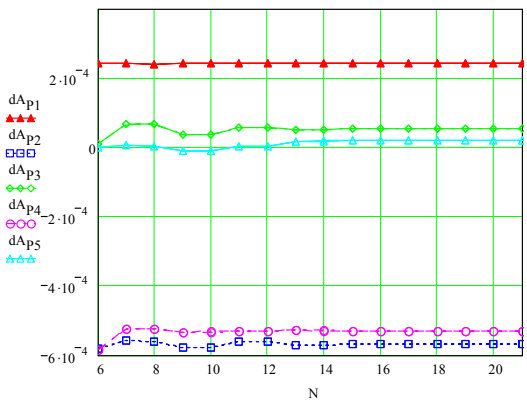


Fig. 4.13 Gain corrections for the P-arm for the nominal field value 4.0 T

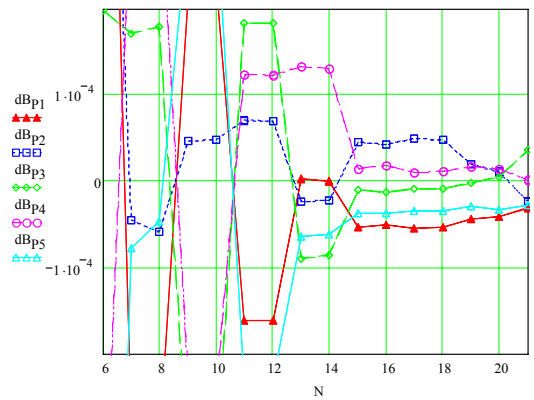


Fig. 4.14 Offset corrections for the P-arm for the nominal field value 4.0 T

As can be seen from Fig. 4.14, the offset corrections have a pronounced plateau for expansion degree in the range 15-18. As for the gain corrections (Fig. 4.13), they are almost independent of the expansion degree for all the range $N = 6$-21, which gives us confidence in the results.

<u>Conclusions to this chapter:</u>

- The expansion degree N should be chosen in the range 15-18.
- The radial and offset corrections are negligibly small; it may be assumed that they are equal to zero. It implies that condition $\Sigma dB_{p(k)} = 0$ imposed in Section 3.4 to obtain a unique solution does not change the results.

There are no strong arguments that would tell us which value in the optimal expansion range to choose. From now on we choose $N = 16$.

4.3 Fit Dependence on the Offset Constraint Weight

In Section 3.4 we introduced a set of equations that constrain the magnitude of the offset corrections. We noted that this constraint, which is based more on qualitative considerations of sources of errors in the measurement system, is somewhat ambiguous. In this chapter we investigate how the fit depends on the offset constraint weight W_{dB}.

For a nominal field of 3.8 T we made 20 fits, varying W_{dB} from 0.01 to 10 in equidistantly spaced $log(W_{dB})$ increments. We then calculated an averaged field map over all fits. The deviations from the averaged map for the fits obtained with the smallest and the largest values of W_{dB} are shown in Fig. 4.15 - Fig. 4.18

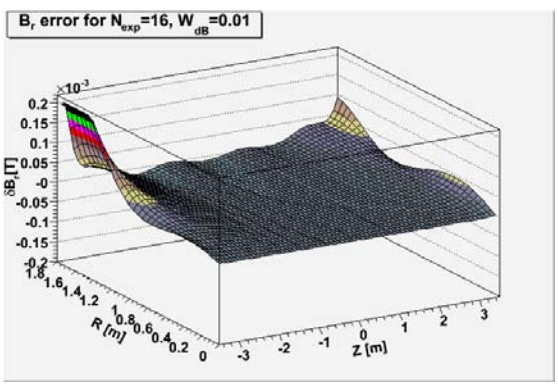


Fig. 4.15 B_r **deviation from average value for the offset constraint weight 0.01**

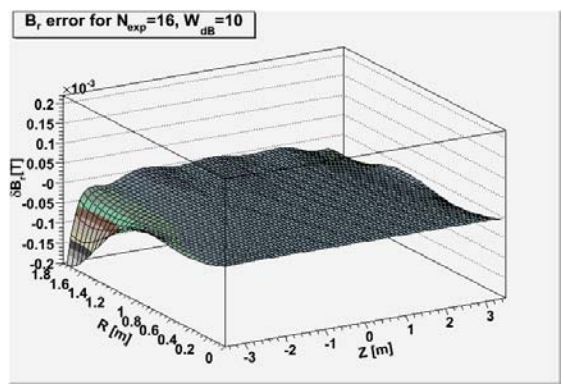


Fig. 4.16 B_r **deviation from average value for the offset constraint weight 10.0**

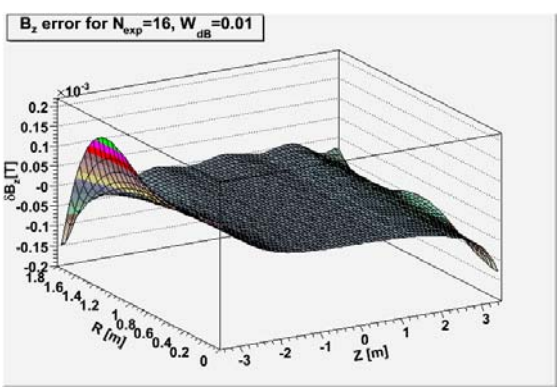


Fig. 4.17 B_z **deviation from average value for the offset constraint weight 0.01**

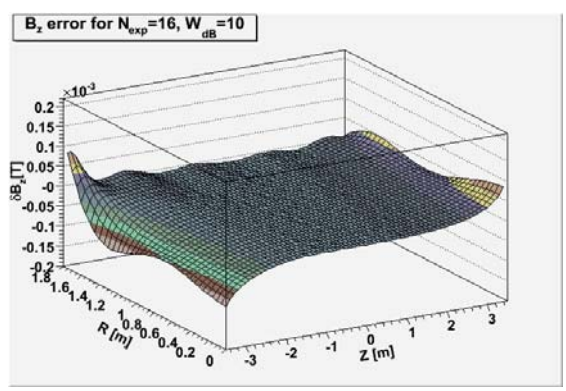


Fig. 4.18 B_z **deviation from average value for the offset constraint weight 10.0**

As it can be seen from these figures, varying W_{dB} by 4 orders of magnitude changes the "corners" (and only the "corners") by less than ± 2 Gauss. However we keep these constraints (in the ***BFit*** code, mentioned later, $W_{dB} = 1$ is used) for reasons that were explained in Section 3.4. We believe that with this constraint the gain corrections and offsets are much more realistic. Variation of W_{dB} does not change much for the axisymmetric model, but may become important for a 3D fit.

5 FIT IN THE AXISYMMETRIC MODEL (RESULTS)

The goal of the present work is to provide a field model (a software code) to be used in the CMS offline tracking. Such a code has been developed and verified.

5.1 BFit: a C++ Code That Knows the CMS Field

After the expansion coefficients have been obtained for each measured field, each expansion coefficient was fitted by a rational polynomial of second order as a function of the nominal field. This provided a basis the *BFit* C++ code, which return the B_r and B_z field components at any point of the tracker volume not just for the measured nominal fields, but for an arbitrary field.

The map for 4 T, the B_z and B_r maps obtained using *BFit*, is show in Fig. 5.1 and Fig. 5.2.

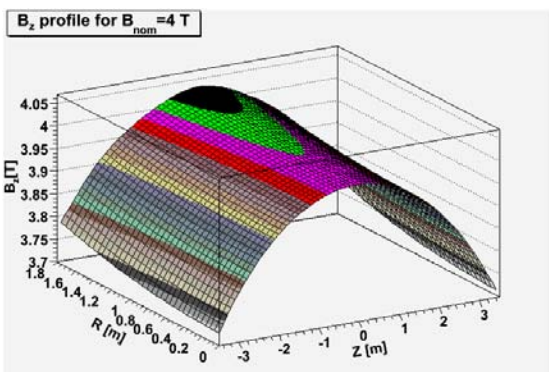


Fig. 5.1 B_z **field map obtained with *BFit***

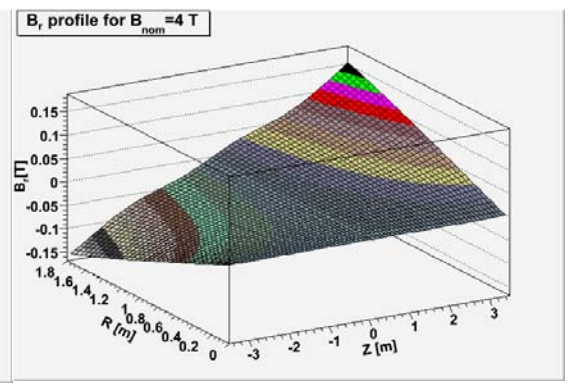


Fig. 5.2 B_r **field map obtained with *BFit***

5.2 BFit: verification

We tested the *BFit* code by comparing its output with the measurement results.

The difference between the measured B_z and the *BFit* output for B_z is shown in Fig. 5.3 through Fig. 5.12

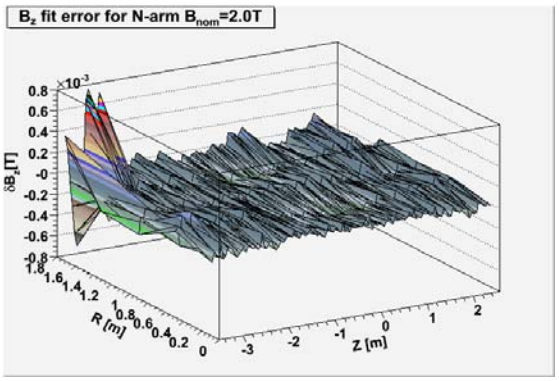


Fig. 5.3 **Measured B_z minus *BFit* output at for N-Arm at 2.0 T**

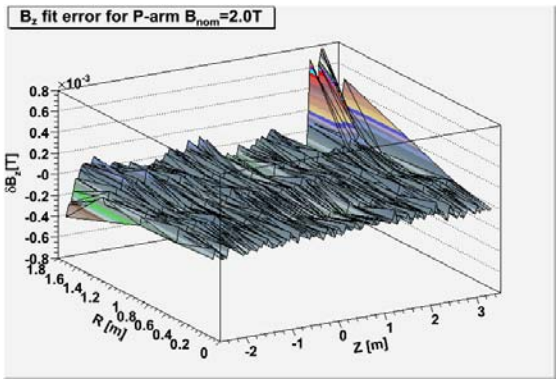


Fig. 5.4 **Measured B_z minus *BFit* output at for P-Arm at 2.0 T**

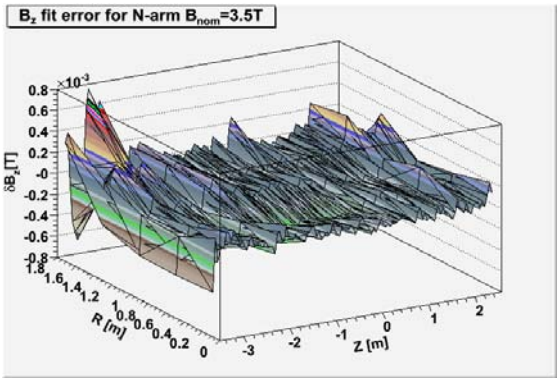


Fig. 5.5 **Measured B_z minus *BFit* output at for N-Arm at 3.5 T**

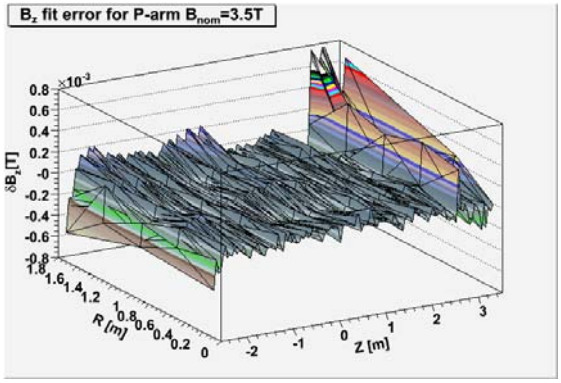


Fig. 5.6 **Measured B_z minus *BFit* output at for P-Arm at 3.5 T**

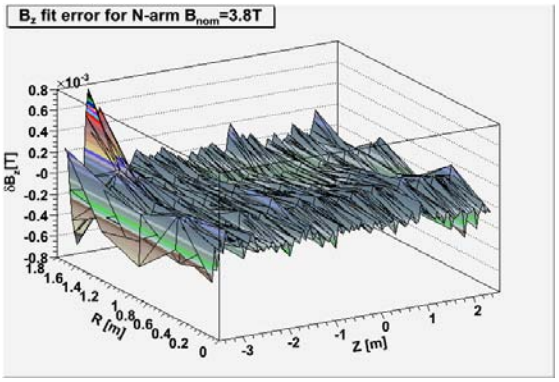


Fig. 5.7 **Measured B_z minus *BFit* output at for N-Arm at 3.8 T**

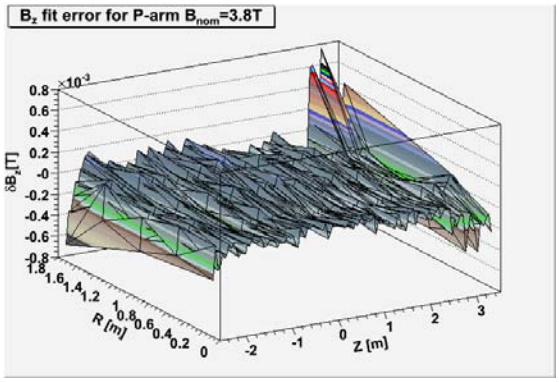


Fig. 5.8 **Measured B_z minus *BFit* output at for P-Arm at 3.8 T**

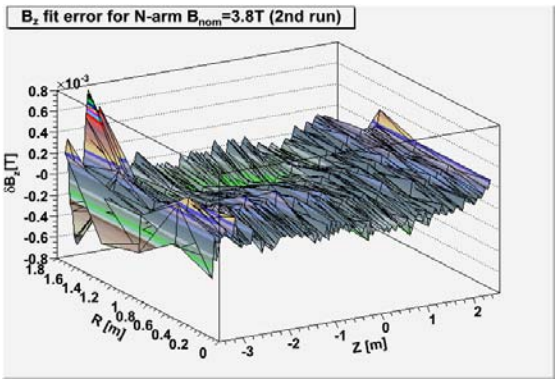


Fig. 5.9 Measured B_z minus *BFit* output at for N-Arm at 3.8 T (2nd run)

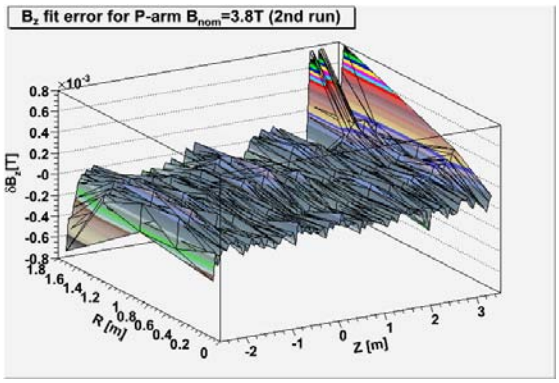


Fig. 5.10 Measured B_z minus *BFit* output at for P-Arm at 3.8 T (2nd run)

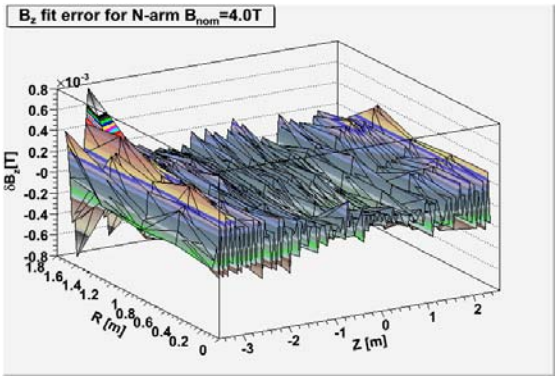


Fig. 5.11 Measured B_z minus *BFit* output at for N-Arm at 4.0 T

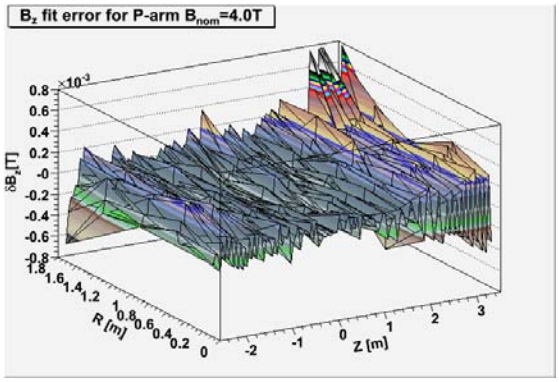


Fig. 5.12 Measured B_z minus *BFit* output at for P-Arm at 4.0 T

The difference between the measured B_r (corrected for $B_z \rightarrow B_r$ mixing) and the *BFit* output for B_r is shown in Fig. 5.13 - Fig. 5.22.

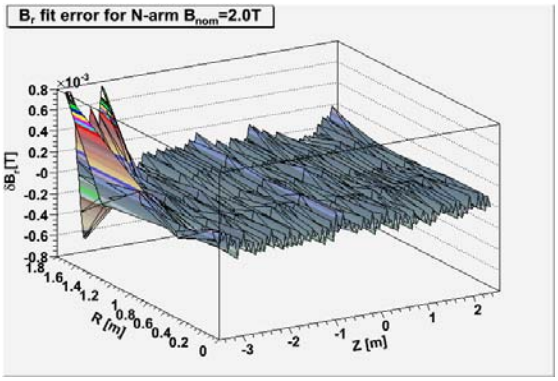


Fig. 5.13 Measured B_r minus *BFit* output at for N-Arm at 2.0 T

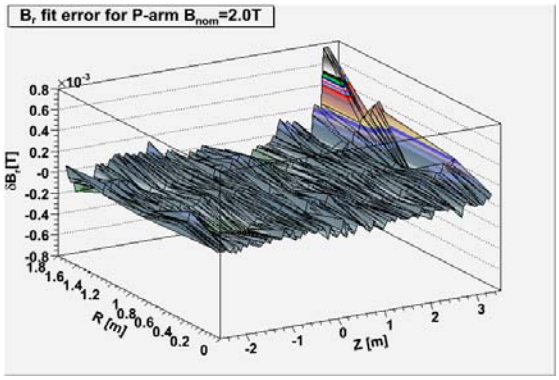


Fig. 5.14 Measured B_r minus *BFit* output at for P-Arm at 2.0 T

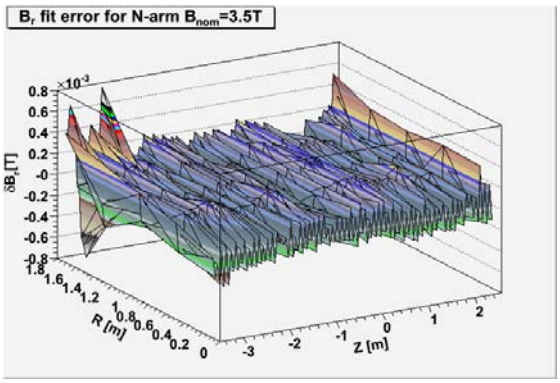


Fig. 5.15 Measured B_r minus *BFit* output at for N-Arm at 3.5 T

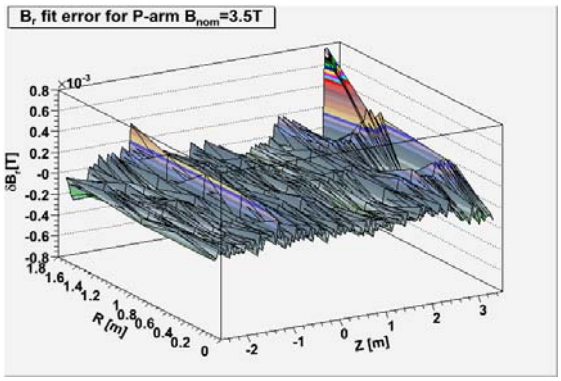


Fig. 5.16 Measured B_r minus *BFit* output at for P-Arm at 3.5 T

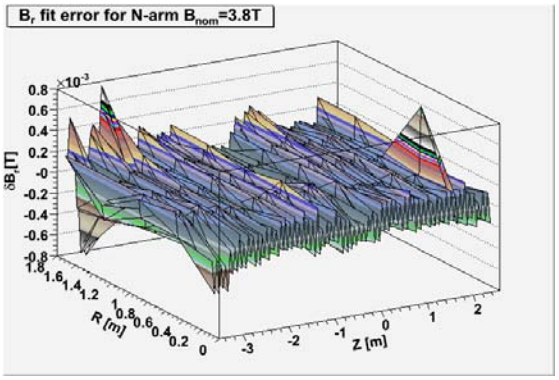


Fig. 5.17 Measured B_r minus *BFit* output at for N-Arm at 3.8 T

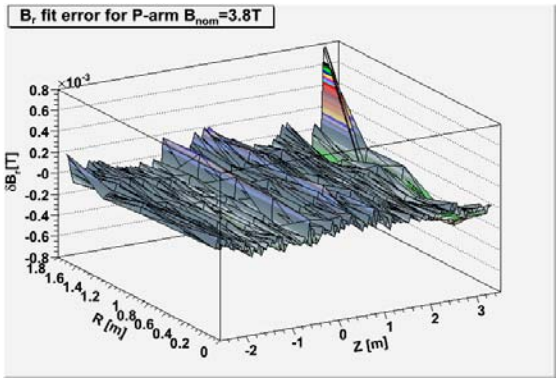


Fig. 5.18 Measured B_r minus *BFit* output at for P-Arm at 3.8 T

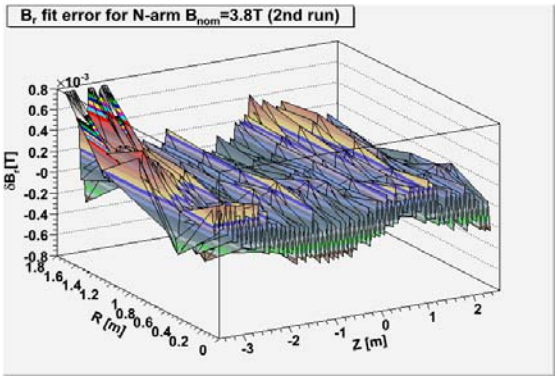


Fig. 5.19 Measured B_r minus *BFit* output at for N-Arm at 3.8 T (2nd run)

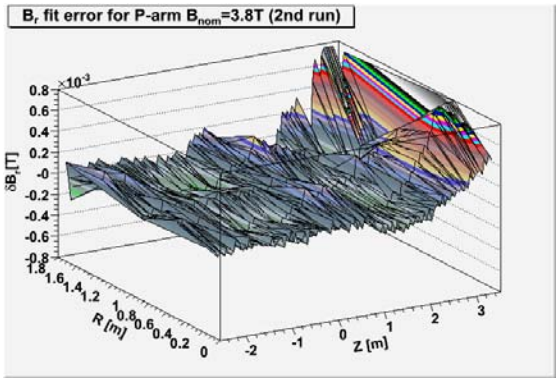


Fig. 5.20 Measured B_r minus *BFit* output at for P-Arm at 3.8 T (2nd run)

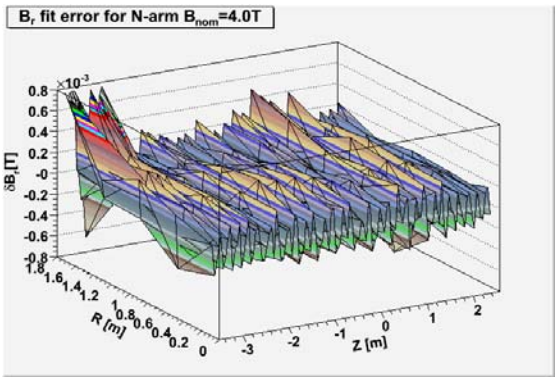


Fig. 5.21 Measured B_r minus *BFit* output at for N-Arm at 4.0 T

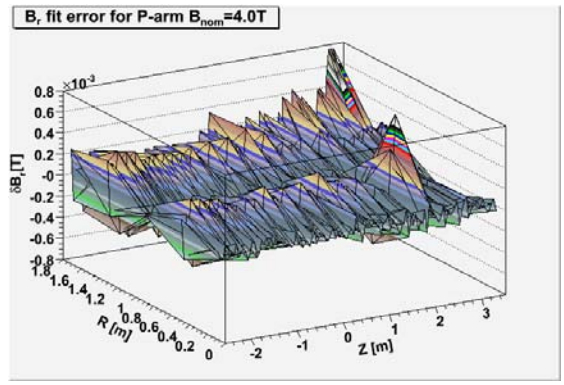


Fig. 5.22 Measured B_r minus *BFit* output at for P-Arm at 4.0 T

As can be seen from the figures above, the *BFit* output matches the measured field with a precision better than $\pm 4\times 10^{-4}$ T over whole scanned volume $|z| \leq 3.5$ m, $r \leq 1.7$ m except "corners", which are approximately described by the equation $|z| + 2.5\,r > 6.7$, (all distances are in meters) and which make up about 6% of the total volume.

5.3 BFit: current status

The *BFit* code is being integrated into the CMSSW (**CMS** **s**oft**w**are) package [21].

6 FIT IN 3 DIMENSIONS (THEORY)

Although most people involved in the CMS tracking consider the axially symmetric field model as sufficient for practical purposes, we feel the work is incomplete until the full 3D field model is developed. Such a 3D field model needs the basis derived in Chapter 3.3 to be extended to span azimuthally dependent functions. The idea of the basis extension comes from a relation between the harmonic homogeneous polynomials and axially symmetric spherical harmonics.

6.1 Relation between Harmonic Homogeneous Polynomials and Spherical Harmonics

It can be shown that the harmonic homogeneous polynomials derived in Chapter 3.3 are in fact, representations of azimuthally symmetric solutions of the Laplace equation [22-23; 25, p.2] with separated variables in spherical coordinates:

$$\Phi_n\left(R,\cos\theta\right)=R^n P_n\left(\cos\theta\right) \tag{6.1}$$

Here P_n is the Legendre polynomial [22-24; 25, pp.9-10; 26] of n-th order. With the substitution of $r=R\sin\theta$, $z=R\cos\theta$ into (3.4) $p_n(r,z)$ can be presented in the form

$$p_n\left(R\sin\theta,R\cos\theta\right)=R^n f_n\left(\cos\theta\right) \tag{6.2}$$

Solution with separated R,θ variables and the same dependence on R must be unique (apart from a factor). Therefore, f_n in (6.2) is proportional to P_n in (6.1). We illustrate this below:

$$
\begin{aligned}
&R^1 P_1(\cos\theta) = R\cos\theta = z \\
&R^2 P_2(\cos\theta) = \tfrac{1}{2}R^2\left(3\cos^2\theta - 1\right) = \tfrac{1}{2}\left(3z^2 - \left(r^2+z^2\right)\right) = z^2 - \tfrac{1}{2}r^2 \\
&R^3 P_3(\cos\theta) = \tfrac{1}{2}R^3\left(5\cos^3\theta - 3\cos\theta\right) = \tfrac{1}{2}\left(5z^3 - 3\left(r^2+z^2\right)z\right) = z^3 - \tfrac{3}{2}r^2 z \\
&R^4 P_4(\cos\theta) = \tfrac{1}{8}R^4\left(35\cos^4\theta - 30\cos^2\theta + 3\right) = \tfrac{1}{8}\left(35z^4 - 30z^2\left(r^2+z^2\right) + 3\left(r^2+z^2\right)^2\right) \\
&\qquad = \tfrac{1}{8}\left(8z^4 - 24z^2r^2 + 6r^4\right) = z^4 - 3z^2r^2 + \tfrac{3}{8}r^4
\end{aligned}
\tag{6.3}
$$

Expressions in (6.3) and (3.7) coincide, within a common factor.

6.2 Derivation of a Complete 3D Basis

A general solution of the Laplace equation $\Delta\Phi = 0$ in spherical coordinates R,θ,ϕ can be written in terms of spherical harmonics [25, p.198; 16, p.110] and powers of R as:

$$
\begin{aligned}
&\Phi(R,\theta,\phi) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l}\left(A_{lm}R^l + B_{lm}R^{-(l+1)}\right)Y_{lm}(\theta,\phi) \\
&\text{here} \quad Y_{lm}(\theta,\phi) = \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_{lm}(\cos\theta)e^{im\phi}
\end{aligned}
\tag{6.4}
$$

Here A_{lm} and B_{lm} are the expansion coefficients, $P_{lm}(\cos\theta)$ are associated Legendre polynomials [25, p.197; 26] and $Y_{lm}(\theta,\phi)$ are the spherical harmonics. We have no singularity at $R = 0$, thus $B_{lm} = 0$ and functions $R^l Y_{lm}(\theta,\phi)$ form a complete basis.

In accordance with the previous section the harmonic homogeneous polynomials $p_l(r, z)$ in spherical coordinates can be expressed through spherical functions as:

$$
p_l(r,z) = p_l(R\sin\theta, R\cos\theta) = N\,R^l P_{l,m=0}(\cos\theta)e^{i(m=0)\phi} = R^l Y_{l,m=0}(\theta,\phi) \tag{6.5}
$$

Here N is just a number (see (6.4)). The ϕ-dependent subset ($m \neq 0$) of the complete basis $\tilde{p}_{lm}(r,z,\phi) = p_{lm}(r,z)e^{im\phi} = R^l Y_{lm}(\theta,\phi)$ can be derived using the ladder operators [27, p.85, (26.15)], which for our purpose can be expressed in cylindrical coordinates as:

$$
\hat{l}_{\pm} = e^{\pm i\phi}\left(\pm\frac{\partial}{\partial\theta} + i\cot\theta\frac{\partial}{\partial\phi}\right) = e^{\pm i\phi}\left(\pm\left(z\frac{\partial}{\partial r} - r\frac{\partial}{\partial z}\right) + i\frac{z}{r}\frac{\partial}{\partial\phi}\right) \tag{6.6}
$$

The overall normalization of basis polynomials $p_{lm}(r, z)$ is not important, but prefer to keep the same normalization for each group of p_{lm} with the same l. For ladder operators it holds:

$$\begin{aligned} \hat{l}_+ Y_{lm} &= \sqrt{(l-m)(l+m+1)}\, Y_{l,m+1} \\ \hat{l}_- Y_{lm} &= \sqrt{(l+m)(l-m+1)}\, Y_{l,m-1} \end{aligned} \tag{6.7}$$

Thus we obtain $p_{l,m\geq 0}(r, z)$ recursively as follows (we omit arguments of p_{lm}):

$$\tilde{p}_{l,0} = p_l \;; \quad \tilde{p}_{l,m+1} = \frac{1}{\sqrt{(l-m)(l+m+1)}} \hat{l}_+ \tilde{p}_{l,m} \;; \quad 0 \leq m < l \tag{6.8}$$

For negative m's the basis elements are (±) complex conjugates of $\tilde{p}_{l,|m|}$.

Note that the basis polynomials $\tilde{p}_{lm}(r, z, \phi)$ are complex due to the ϕ–dependent factor $e^{\pm im\phi}$. In actual work we didn't use the complex basis, but real linear combinations of basis elements with $\pm m$, which give ϕ-dependent factors $\cos m\phi$, $\sin m\phi$.

As mentioned before, the polynomial calculus reduces to elementary arithmetic of coefficients and degrees, which is easy to program. Let us consider for instance, how the $\hat{l}_+$ acts at the polynomial term, multiplied by the phase factor $e^{im\phi}$:

$$\hat{l}_+ \left(z^p r^q e^{im\phi} \right) = e^{i\phi} \left(z \frac{\partial}{\partial r} - r \frac{\partial}{\partial z} + i \frac{z}{r} \frac{\partial}{\partial \phi} \right) z^p r^q e^{im\phi} = \left((q-m) z^{p+1} r^{q-1} - p z^{p-1} r^{q+1} \right) e^{i(m+1)\phi} \tag{6.9}$$

We know what the operator makes of each term of the polynomial p_{lm}, then it is easy to derive the polynomial $p_{l,m+1}$.

However, we did not do any "manual" analytical calculations. The polynomial calculus: differentiation, integration, collecting of terms, "ladder promotion" and the polynomial evaluation we programmed in C++. The developed code, *HarmBasis3DCyl* library, did everything for us. Below is a fragment of diagnostic printout, made with the use of *HarmBasis3DCyl* :

```
-----------------------------------------------------------------------
Basis subset 8
-----------------------------------------------------------------------
L=8, M=0; 5 terms; max. degree = 8:
0.125*z^8 - 1.75*r^2*z^6 + 3.2812*r^4*z^4 - 1.0938*r^6*z^2 + 0.03418*r^8
L=8, M=1; 4 terms; max. degree = 8:
-0.53033*r*z^7 + 2.7842*r^3*z^5 - 2.3202*r^5*z^3 + 0.29002*r^7*z
L=8, M=2; 4 terms; max. degree = 8:
1.1093*r^2*z^6 - 2.7732*r^4*z^4 + 1.0399*r^6*z^2 - 0.034665*r^8
L=8, M=3; 3 terms; max. degree = 8:
-1.502*r^3*z^5 + 1.8774*r^5*z^3 - 0.28162*r^7*z
L=8, M=4; 3 terms; max. degree = 8:
1.4543*r^4*z^4 - 0.87256*r^6*z^2 + 0.036356*r^8
L=8, M=5; 2 terms; max. degree = 8:
-1.0487*r^5*z^3 + 0.26217*r^7*z
L=8, M=6; 2 terms; max. degree = 8:
0.56635*r^6*z^2 - 0.040454*r^8
L=8, M=7; 1 terms; max. degree = 8:
-0.22157*r^7*z
L=8, M=8; 1 terms; max. degree = 8:
0.055394*r^8
-----------------------------------------------------------------------
```

It may be checked that *HarmBasis3DCyl* printed above right coefficients for the basis subset with $l = 8$.

Although *HarmBasis3DCyl* is a little fraction of the program code written during this work, we think this piece of programming deserves to be presented in *Appendix*.

We are now ready to explain what the "nearly axially symmetric" expression, used in Section 3.1, means. In terms of the 3D basis described above it means: the field "nearly does not contain" basis components with $l > 1$ and odd m's, they are very small compared to the dipole field component. In that case deviations of the probe readings caused by a small tilt or radial displacement of the fieldmapper will cancel each other for every pair of opposite points on the circular probe trajectory at the ϕ-folding.

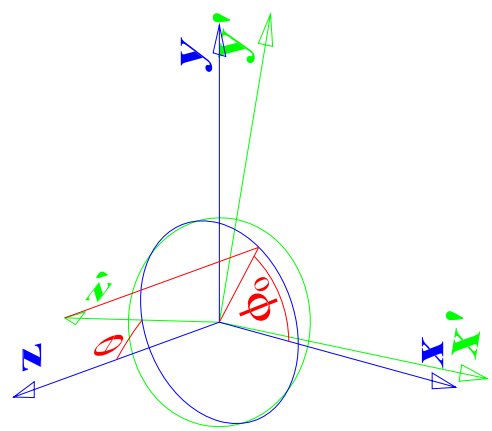
After completion of this work we found that the relation between harmonic polynomials in the Cartesian coordinate system and spherical harmonics is a known fact [28, pp.253-262], [29, pp.232-243]. However, in present work we derived an *explicit representation* of spherical harmonics in the *cylindrical* coordinate system. We have not find references related to such a representation.

7 FIT IN 3 DIMENSIONS (PREPARATION)

Before performing the axisymmetric fit, the measurement data was only visually checked for "big outliers" (which were found and removed, indeed). A 3D fit requires the measurement data to be corrected for the fieldmapper axle tilt.

7.1 Fieldmapper Tilt Correction (Theory)

Let us assume the field is uniform and there is a slight misalignment of fieldmapper so that the fieldmapper axle, which is the z' axis of the local coordinate system, has the polar angle θ and the azimuthal angle ϕ_0 in the CMS spherical coordinate system. We assume the θ is small.



We also assume the azimuthal misalignment is small. That means the orientation of x' and y' axes of the local Cartesian coordinate system for the fieldmapper rotation angle $\phi = 0$ is approximately given by:

$$\mathbf{n}_x \cdot \mathbf{n}_{x'} + \mathbf{n}_y \cdot \mathbf{n}_{y'} = \max \tag{7.1}$$

With the assumptions above as the fieldmapper rotates around the z' axis in the local cylindrical coordinate system one can observe the field components which depend on the rotation angle ϕ as follows:

$$\begin{aligned} B_{z'} &= B_z \cos\theta \simeq B_z \\ B_{r'} &= B_z \sin\theta \cdot \cos\left(\pi - \left(\phi - \phi_0\right)\right) = -B_z \sin\theta \cdot \left(\cos\phi \cos\phi_0 + \sin\phi \sin\phi_0\right) \\ B_{\phi'} &= B_z \sin\theta \cdot \sin\left(\pi - \left(\phi - \phi_0\right)\right) = \quad B_z \sin\theta \cdot \left(\sin\phi \cos\phi_0 - \cos\phi \sin\phi_0\right) \end{aligned} \tag{7.2}$$

The polar angle θ and the azimuthal angle ϕ_0 may be obtained from (7.2) by means of real Fourier transform. Let us introduce variables c_r, s_r, c_ϕ, s_ϕ, which are first Fourier expansion coefficients for the $B_{r'}$ and $B_{\phi'}$:

$$\begin{aligned} c_r &\equiv \frac{1}{\pi}\int_0^{2\pi} d\phi B_{r'}\cos\phi = -B_z \sin\theta\cos\phi_0 \qquad & s_r &\equiv \frac{1}{\pi}\int_0^{2\pi} d\phi B_{r'}\sin\phi = -B_z \sin\theta\sin\phi_0 \\ c_\phi &\equiv \frac{1}{\pi}\int_0^{2\pi} d\phi B_{\phi'}\cos\phi = -B_z \sin\theta\sin\phi_0 \qquad & s_\phi &\equiv \frac{1}{\pi}\int_0^{2\pi} d\phi B_{\phi'}\sin\phi = \ B_z \sin\theta\cos\phi_0 \end{aligned} \tag{7.3}$$

The polar angle θ and the azimuthal angle ϕ_0 may be expressed as:

$$\begin{aligned} \theta &= \arcsin\left(\frac{1}{B_z}\sqrt{c_r^2+s_r^2}\right) = \arcsin\left(\frac{1}{B_z}\sqrt{c_\phi^2+s_\phi^2}\right) \\ \phi_0 &= \mathrm{angle}\left(-c_r,-s_r\right) = \mathrm{angle}\left(s_\phi,-c_\phi\right) \end{aligned} \tag{7.4}$$

The function $\mathrm{angle}\left(x,y\right)$ returns the counter-clockwise angle between the $+x$ direction and the ray from coordinate origin which passes through the point $\left(x,y\right)$.

The set of variables θ,ϕ_0 is not the only possible parameterization of the misalignment problem. Let us consider variables (coordinates) defined as follows:

$$\begin{aligned} X &= -\frac{1}{B_z}c_r = \frac{1}{B_z}s_\phi \ = \sin\theta\cos\phi_0 \xrightarrow{\theta\ll 1} \theta\cos\phi_0 \\ Y &= -\frac{1}{B_z}s_r = -\frac{1}{B_z}c_\phi = \sin\theta\sin\phi_0 \xrightarrow{\theta\ll 1} \theta\sin\phi_0 \end{aligned} \tag{7.5}$$

The set of variables above allows a simple physical interpretation (assuming $\theta \ll 1$). If there is a laser mounted on the mapper axle, which produces a beam along the rotation axis the *X*,*Y* pair gives the position, in Cartesian coordinates, of the laser spot on a screen located at unit distance from the object.

The X,Y coordinate system (with $\theta \ll 1$) has several advantages against θ,ϕ_0 one:

- It is not cyclic. Different sets of X,Y corresponds to different misalignments
- It is linear. If there is an object **A** whose misalignment in global coordinate system is $\left(X_{\mathbf{A}},Y_{\mathbf{A}}\right)$ and an object **B** whose misalignment in the coordinate system aligned with

A is $\left(X_{\mathbf{B}}^{(\mathbf{A})}, Y_{\mathbf{B}}^{(\mathbf{A})}\right)$, then the misalignment of **B** in the global coordinate system is given by simple $\left(X_{\mathbf{B}}, Y_{\mathbf{B}}\right) = \left(X_{\mathbf{A}} + X_{\mathbf{B}}^{(\mathbf{A})}, Y_{\mathbf{A}} + Y_{\mathbf{B}}^{(\mathbf{A})}\right)$. Thus a mean for a set of misalignment point is well defined.

- It has a simple Euclidian metrics. Distance between $\left(X_{\mathbf{A}}, Y_{\mathbf{A}}\right)$ and $\left(X_{\mathbf{B}}, Y_{\mathbf{B}}\right)$ is given by $\left|\left(X_{\mathbf{A}}, Y_{\mathbf{A}}\right) - \left(X_{\mathbf{B}}, Y_{\mathbf{B}}\right)\right| = \sqrt{\left(X_{\mathbf{A}} - X_{\mathbf{B}}\right)^2 + \left(Y_{\mathbf{A}} - Y_{\mathbf{B}}\right)^2}$. Therefore a variance for a set of misalignment point is well defined.

7.2 Fieldmapper Tilt Correction (Results)

Before any data processing, it can be seen that the azimuthal misalignment angle ϕ_0 differs by approximately $\frac{1}{2}\pi$ for negative and positive arms of the fieldmapper, which means that the axis of rotation of the positive arm does not coincide with the axis of rotation of the negative arm. As an example, the dependence of B_ϕ on the azimuthal index and *z*-index for the N4 and P4 probes at nominal field value of 3.8 T is shown the Fig. 7.1 and Fig. 7.2 respectively.

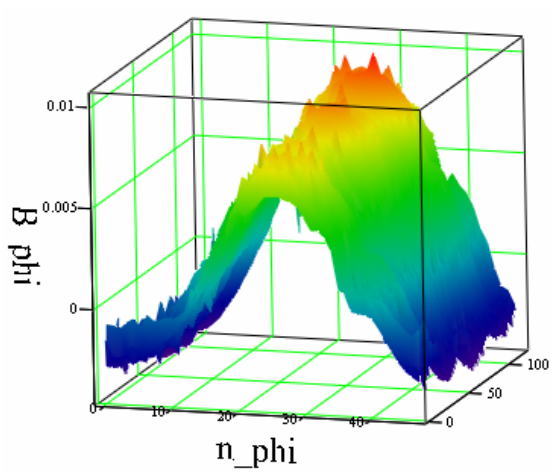


Fig. 7.1 B_ϕ dependence on angular and z-index for the N4 probe

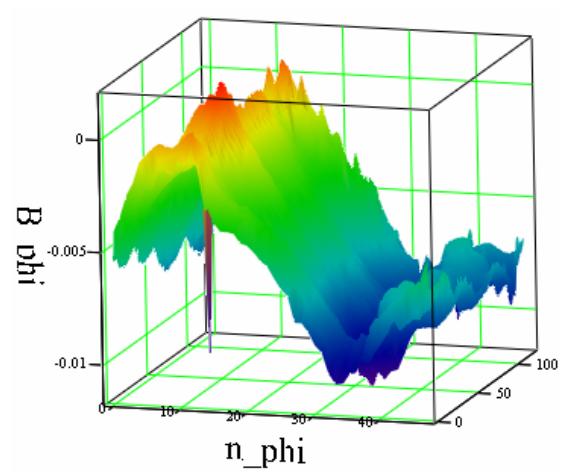


Fig. 7.2: B_ϕ dependence on angular and z-index for the P4 probe

In both Fig. 7.1 and Fig. 7.2 the vertical axis corresponds to the measured $B_{\phi'}$, the number on horizontal axis is the azimuthal index (0…48) and the number remaining axis

is z-index (0…121). Neglecting the non-physical offset of B_ϕ, which is probably caused by a misalignment of probes relatively to the rotation axis, it can be seen for all z-positions the $B_{\phi'}$ measured for the N-arm is roughly proportional to $-\cos\phi$, while for the P-arm it behaves close to $\sin\phi$. It cannot be if the rotation axis is common for negative and positive arms.

Angles θ and ϕ_0 for all probes were calculated using (7.3) and (7.4) for all nominal field values. The dependence of θ and ϕ_0 on z-index for all probes at a nominal field value of 3.8 T is shown in Fig. 7.3 - Fig. 7.6.

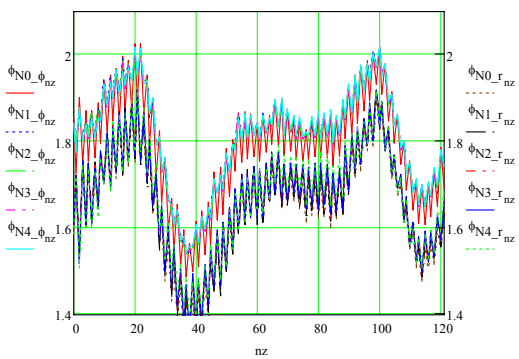


Fig. 7.3 Azimuthal angle ϕ_0 vs. z-index for the N-arm probes.

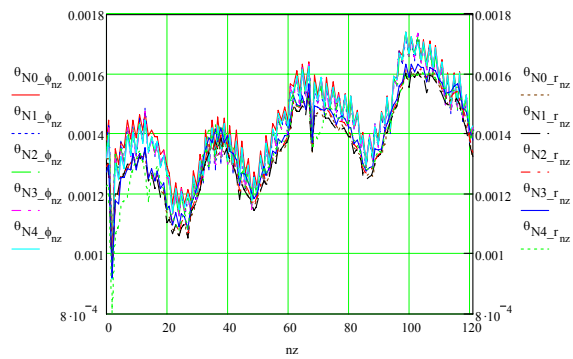


Fig. 7.5 Polar angle θ vs. z-index for the N-arm probes.

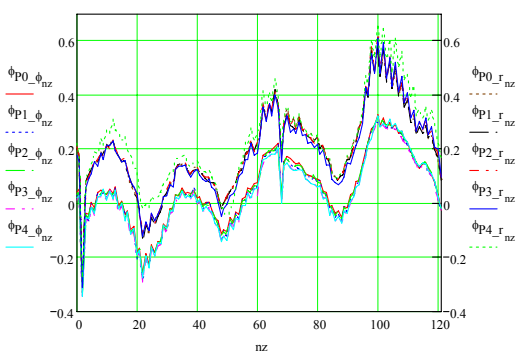


Fig. 7.4 Azimuthal angle ϕ_0 vs. z-index for the P-arm probes.

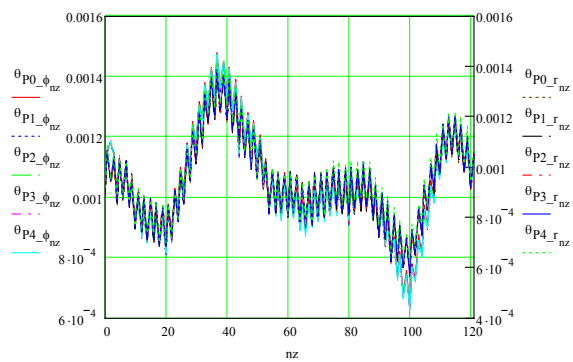


Fig. 7.6 Polar angle θ vs. z-index for the P-arm probes.

Both B_r and B_ϕ were used to calculate the misalignment angles. The notation is as follows: θ_{N1_ϕ} means θ calculated for the probe N1 using the B_ϕ data, ϕ_{P0_r} means ϕ calculated for the probe P0 using the B_r data, etc.

Looking at Fig. 7.3 - Fig. 7.6 it may noticed that there is an almost perfect correlation between $\theta_{N\boldsymbol{I}}$ and $\phi_{P\boldsymbol{I}}$ (here "***I***" stands for the probe index) and an anti-correlation between $\theta_{P\boldsymbol{I}}$ and $\phi_{N\boldsymbol{I}}$. However, from one-dimensional plots of θ, ϕ_0 *vs.* *z*-index it is hard to understand an underlying reason for such correlations. Therefore we switch to a 2D representation of the data.

The Fig. 7.7 and Fig. 7.8 represent the (ϕ_0, θ) data for the N-arm and a nominal field value of 3.8 T.

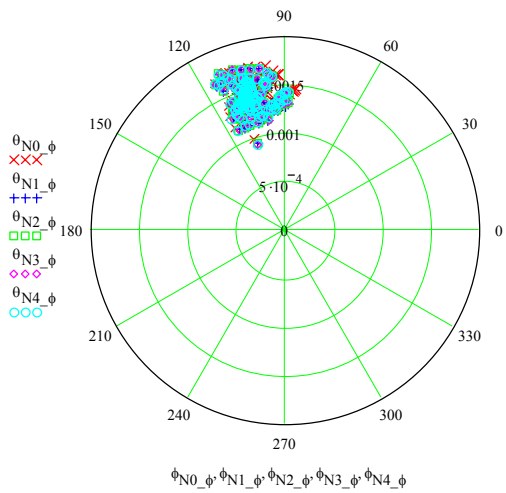


Fig. 7.7 θ–ϕ_0 **distribution for all z-indices for the N-arm obtained from the** B_ϕ

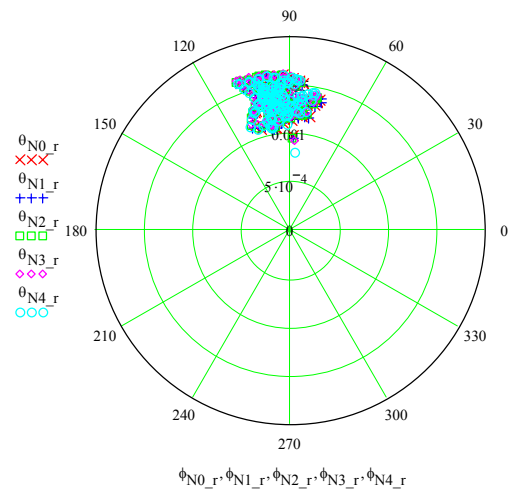


Fig. 7.8 θ–ϕ_0 **distribution for all z-indices for the N-arm obtained from the** B_r

The Fig. 7.9 and Fig. 7.10 represent the (ϕ_0, θ) data for the P-arm and a nominal field value of 3.8 T.

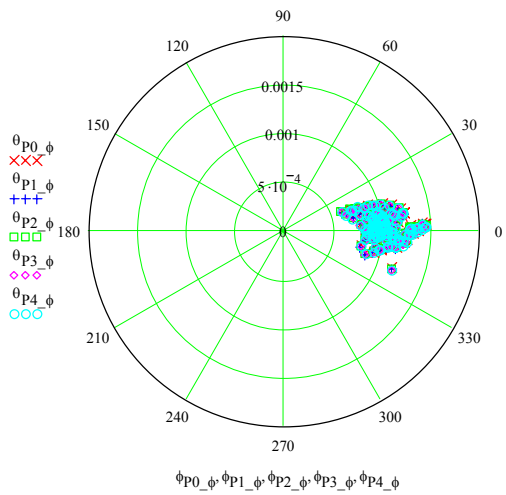


Fig. 7.9 θ–ϕ_0 distribution for all z-indices for the P-arm obtained from the B_ϕ

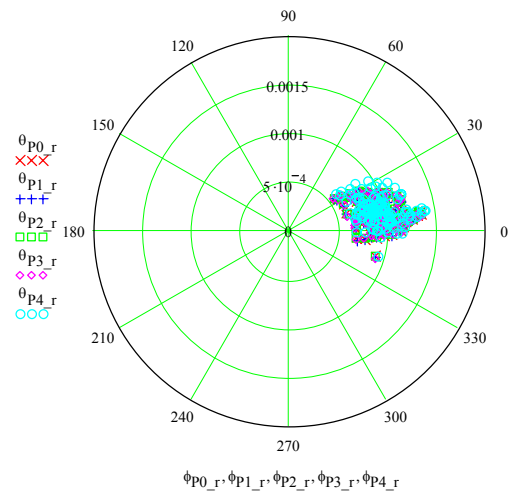


Fig. 7.10 θ–ϕ_0 distribution for all z-indices for the P-arm obtained from the B_r

As it can be seen in Fig. 7.7 - Fig. 7.10, although the "mass center" positions of "spots" on the θ–ϕ_0 plane are different for the negative and positive arm, the spots are similar in shape and their orientation is almost the same. In order to "zoom in" and see more details, we switch to *X*,*Y* coordinates. The notation is as follows: X_{N1_ϕ} means *X* calculated for the probe N1 using the B_ϕ data, Y_{P0_r} means *Y* calculated for the probe P0 using the B_r data, etc.

"Trajectories" (with running *z*-index) of calculated misalignment parameters in *X*,*Y* coordinated are plotted in Fig. 7.11 - Fig. 7.14.

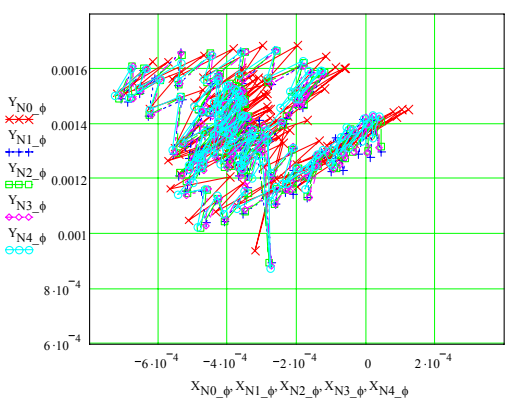


Fig. 7.11 X-Y trajectory for all z-indices for the N-arm obtained from the B_ϕ

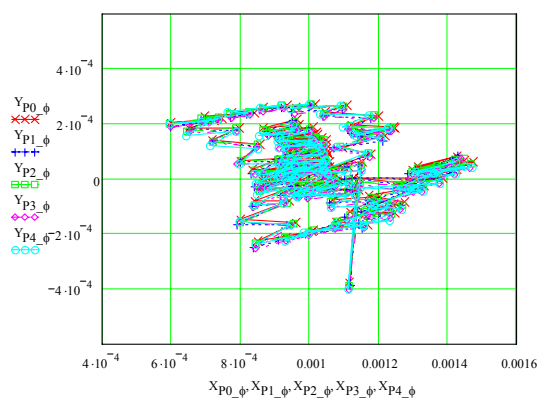


Fig. 7.12 X-Y trajectory for all z-indices for the P-arm obtained from the B_ϕ

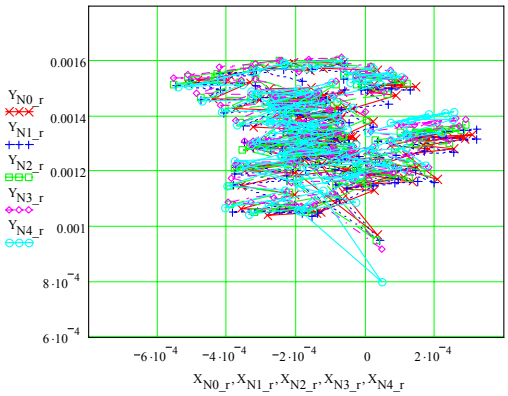


Fig. 7.13 X-Y trajectory for all z-indices for the N-arm obtained from the B_r

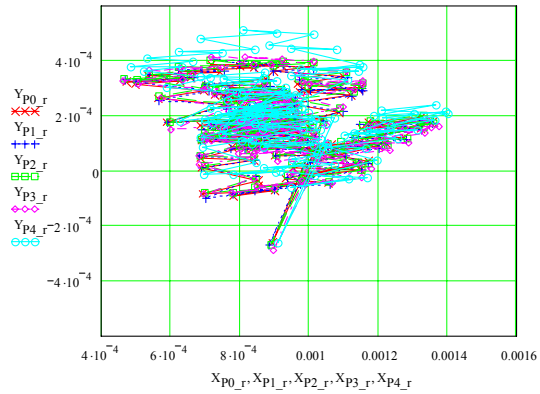


Fig. 7.14 X-Y trajectory for all z-indices for the P-arm obtained from the B_r

In all four cases above the "trajectory" in the X,Y plane has a similar "flying duck" shape and orientation, but is shifted in position.

Referring to the linearity of the X,Y coordinates, one may state that the observed phenomenon is compatible with the hypothesis, that for some reason the rotation axis of the P-arm is permanently tilted relatively the rotation axis of the N-arm.

Using the B_ϕ data it was obtained that the orientation of the N-arm rotation axis in the spherical coordinate system, which is aligned with the P-arm rotation axis is given by the polar angle $\theta \approx 1.9\,mrad$ and the azimuthal angle $\phi \approx 0.77\pi$. In order to verify whether such a misalignment of can appear, we consider a simple mechanical model: a rod (the axle) with the elastic modulus E, length L and radius r is simply supported at the ends and is loaded in the middle with a force F. The angle between rod's ends can be estimated as:

$$\theta \simeq \frac{2}{3\pi}\frac{L^2}{Er^4}F \tag{7.6}$$

With the substitution of $E = 7\cdot 10^{10}\,N\cdot m^{-2}$, $L = 0.56\,m$, $r = 0.015\,m$ it can be obtained that quite a moderate force $F \approx 106\,N$ is sufficient to yield $\theta \simeq 2\,mrad$.

Although the mechanical model considered is very coarse and we do not know where the axle bending force/torque appears (bearings, belt drive, arms' weight?) from the estimation one can see such a bending of the axle is quite feasible.

We calculated tilts of P-arm and N-arm, averaged over probes, Results in *X,Y* coordinates for a nominal field of 3.8 T are shown in Fig. 7.15 -- Fig. 7.18.

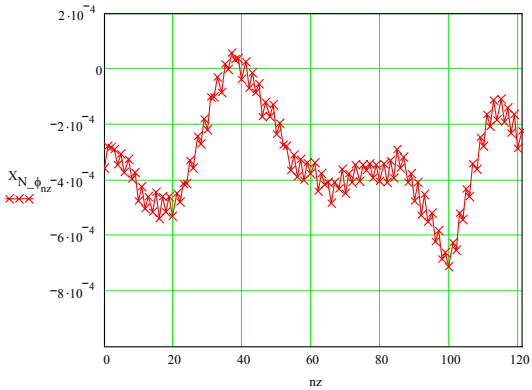


Fig. 7.15 *X*-misalignment for N-arm (from B_ϕ)

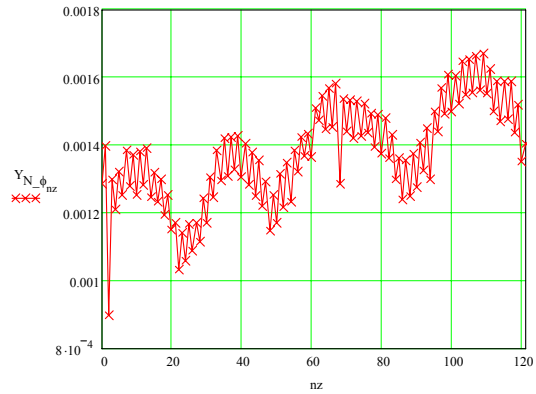


Fig. 7.17 *Y*-misalignment for N-arm (from B_ϕ)

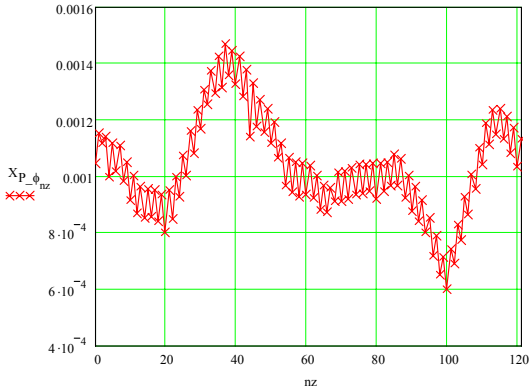


Fig. 7.16 *X*-misalignment for P-arm (from B_r)

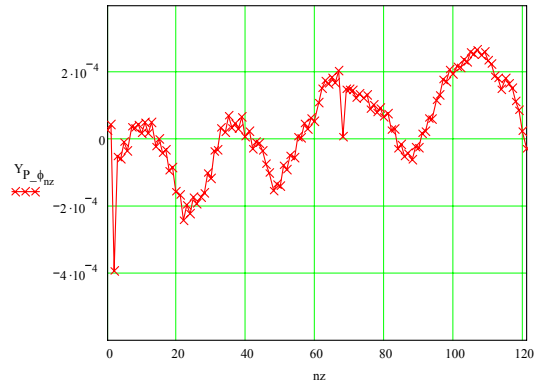


Fig. 7.18 *Y*-misalignment for P-arm (from B_r)

7.3 Fieldmapper Tilt Correction (Errors)

A straightforward way to get a feeling about the errors range of misalignment parameters *X,Y* obtained at each *z*-index, is to see how much these parameters vary for all 5 sensors of an arm. For each step we calculate the misalignment RMS error:

$$RMS = \sqrt{\frac{1}{4}\sum_{p=0}^{4}\left(\left(X_p - \bar{X}\right)^2 + \left(Y_p - \bar{Y}\right)^2\right)} \qquad \bar{X} = \frac{1}{5}\sum_{p=0}^{4} X_p \qquad \bar{Y} = \frac{1}{5}\sum_{p=0}^{4} Y_p \qquad (7.7)$$

Results for the N-arm are plotted in Fig. 7.19 - Fig. 7.20.

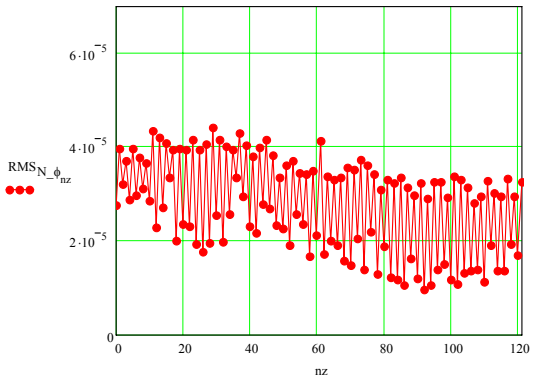


Fig. 7.19 RMS error of the misalignment vs. *z*-index for the N-arm (from B_ϕ)

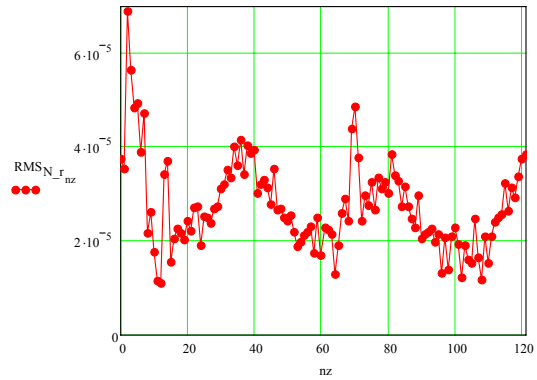


Fig. 7.20 RMS error of the misalignment vs. *z*-index for the N-arm (from B_r)

Average RMS error for the N-arm is 2.97×10^{-5} for the B_ϕ and 2.93×10^{-5} for the B_r.

Results for the P-arm are plotted in Fig. 7.21 - Fig. 7.22.

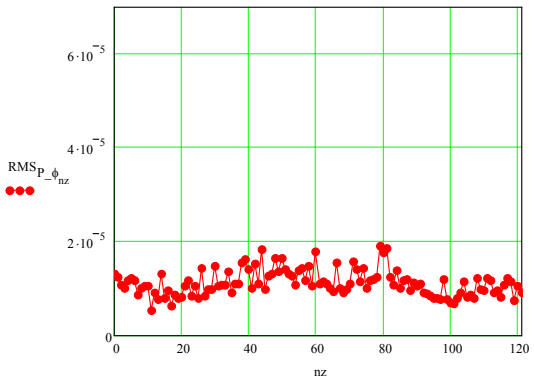


Fig. 7.21 RMS error of the misalignment vs. *z*-index for the P-arm (from B_ϕ)

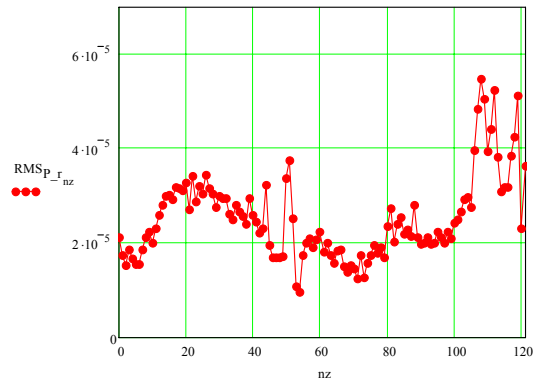


Fig. 7.22 RMS error of the misalignment vs. *z*-index for the P-arm (from B_r)

Average RMS error for the P-arm is 1.16×10^{-5} for the B_ϕ and 2.69×10^{-5} for the B_r.

For the N-arm, average errors are approximately the same, but for the P-arm, use of the B_ϕ for tilt correction yields substantially lower error, compared to the use of the B_r. I have chosen to use the B_ϕ data for the misalignment correction.

7.4 Fieldmapper Tilt Correction (Verification)

An optical survey of the plane rotation of the N-arm was performed immediately after opening the detector after the fieldmapping program was completed, in 8 z-positions

along the rail. In Fig. 7.23 and Fig. 7.24 the result of this survey is compared to the tilt predicted from the magnetic field data of the 4 T measurements. An excellent agreement is has been observed in 7 of 8 survey points. In the 8th survey point, the fit of the plane to the measurements was unreliable because only 4 measurements were usable compared to 8 measurements at other *z*-positions. These results confirm the misalignment of fieldmapper arms can be fitted from the magnetic field measurement data with a precision of about 0.1 mrad.

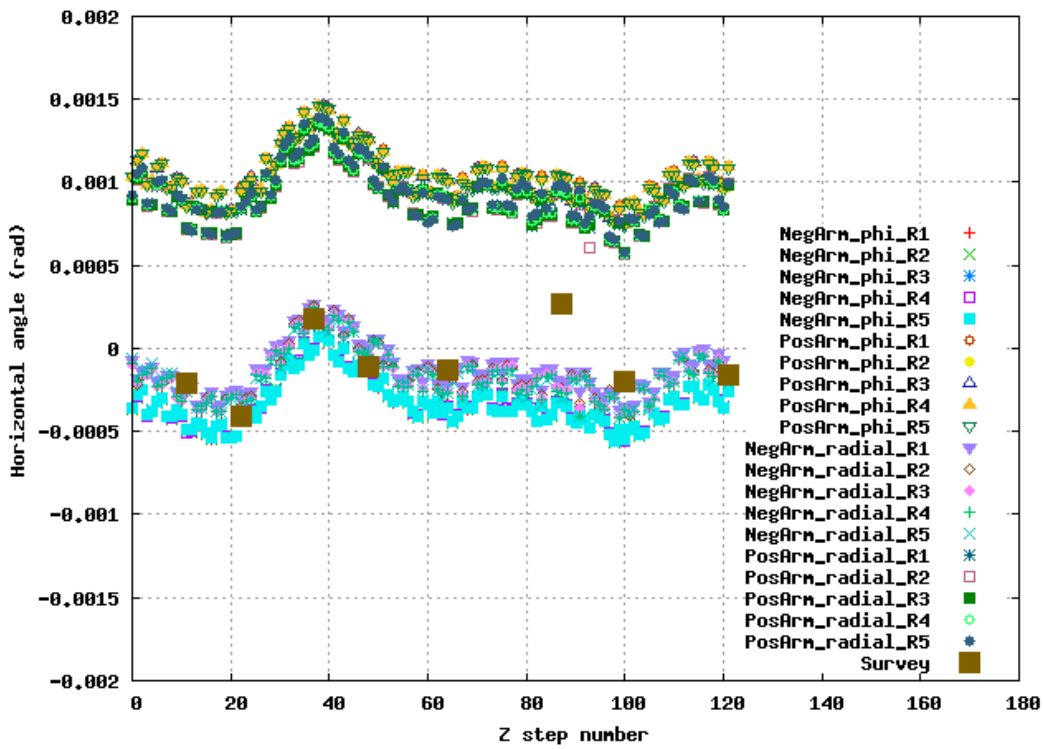


Fig. 7.23 Comparison of the optical survey results to the X-tilt predicted from the magnetic field data from the 4 T measurements (published by permission of Martijn Mulders, CERN).

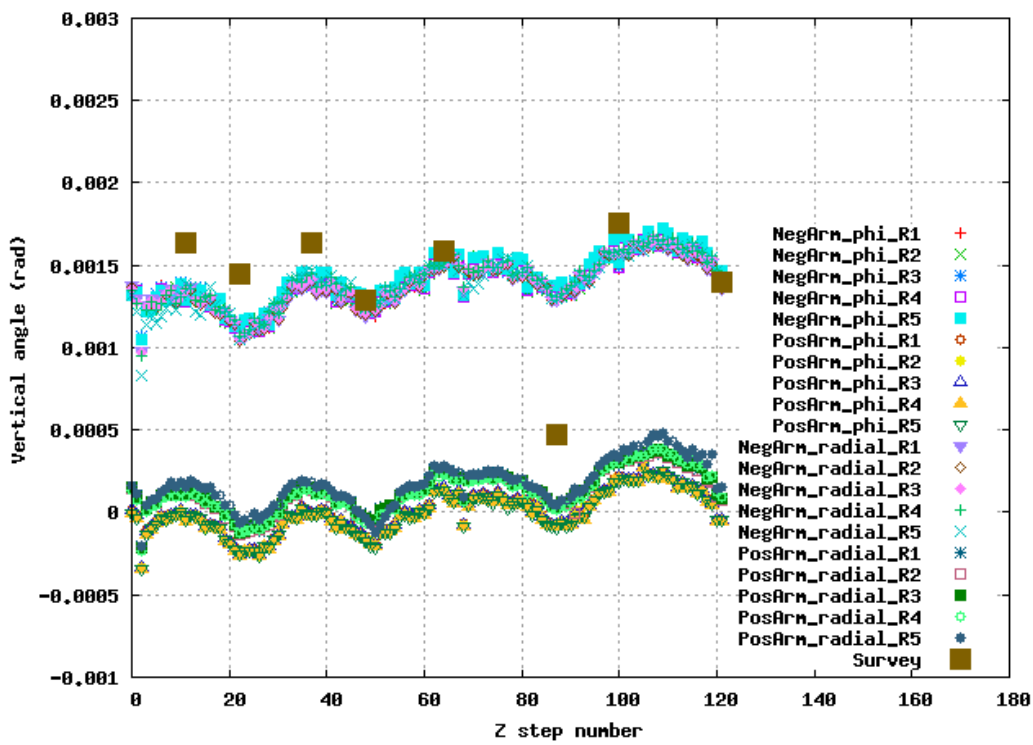


Fig. 7.24 Comparison of the optical survey results to the Y-tilt predicted from the magnetic field data from the 4 T measurements (published by permission of Martijn Mulders, CERN).

Concluding this chapter, we notice the plots in Fig. 7.23 and Fig. 7.24 are in full agreement with the plots in Fig. 7.15 - Fig. 7.18.

7.5 Data Smoothing and Elimination of Outliers

Keeping in mind the equation matrix dimension for 3D fit will be huge, we decided to perform a data preconditioning: to smooth the data and carefully remove outliers. The purpose of such preconditioning is to sharpen the minimum in the least-square fit, thus to improve the precision of the system solution. We tried several types of filters and finally chosen the "sliding polynomial fit" filter. It works in the following way: for each probe at fixed angular position we consider the linear data set along z. For this data set, we perform the fit with a polynomial of degree N in a "sliding" window of a fixed length L. For each position of the window we mark outliers, the points in which measured value is beyond the $\pm A\sigma$ interval, here A is a fixed coefficient and σ is the RMS error of the fit. If outliers are found after a pass of the window along all the data set, the procedure is repeated again (marked points are not used in repeated passes) until no outliers are found

in a pass. After that the value at each point is replaced with the average of fits in the last pass for all positions of the sliding window at which the point was included.

After a number of tries it was found that a good choice for the filter parameters is: $L = 16$, $N = 4$, $A = 2.8$. Plots below are samples of the filter work. Red arrows mark outliers.

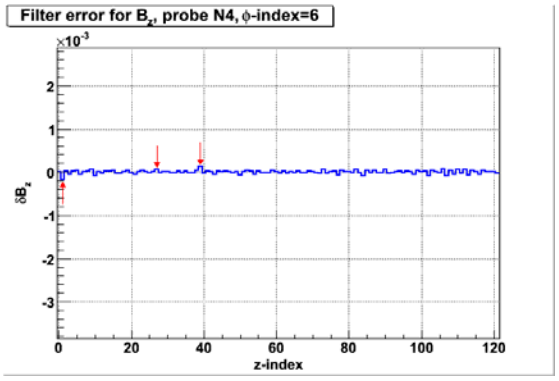


Fig. 7.25 Filter error for the N4 probe at $n_\phi = 6$

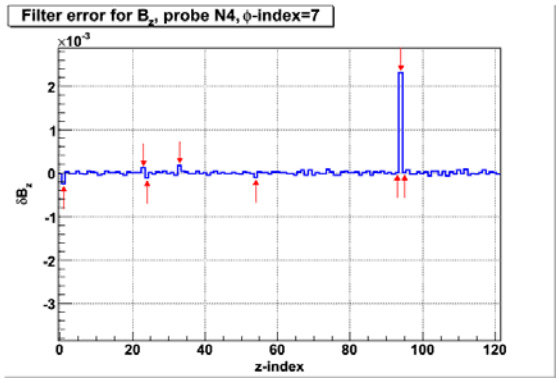


Fig. 7.26 Filter error for the N4 probe at $n_\phi = 7$

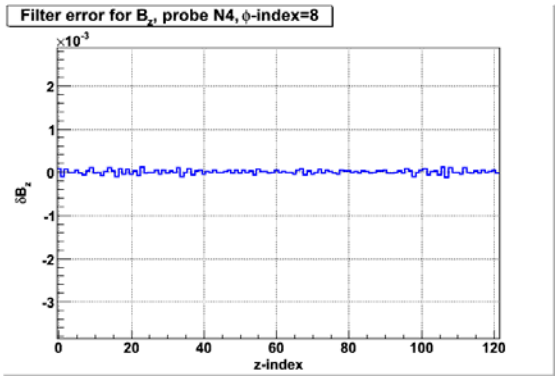


Fig. 7.27 Filter error for the N4 probe at $n_\phi = 8$

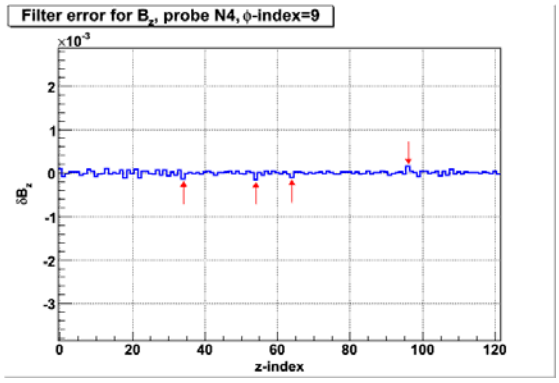


Fig. 7.28 Filter error for the N4 probe at $n_\phi = 9$

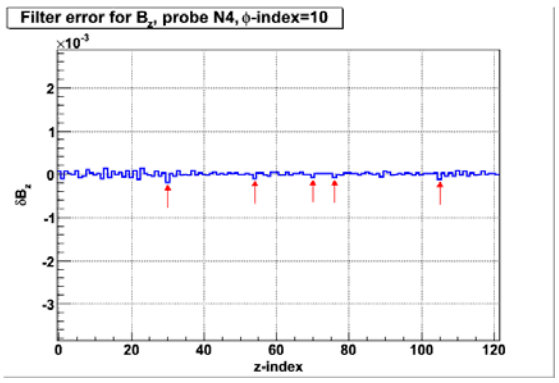


Fig. 7.29 Filter error for the N4 probe at $n_\phi = 0$

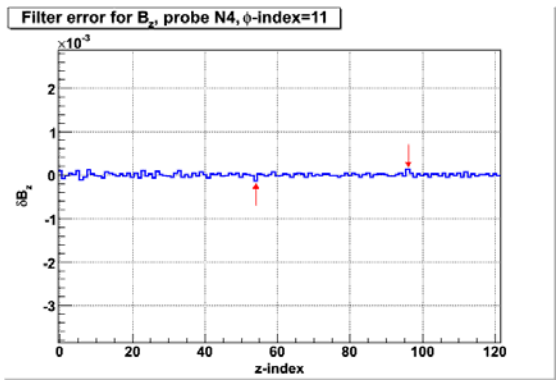


Fig. 7.30 Filter error for the N4 probe at $n_\phi = 11$

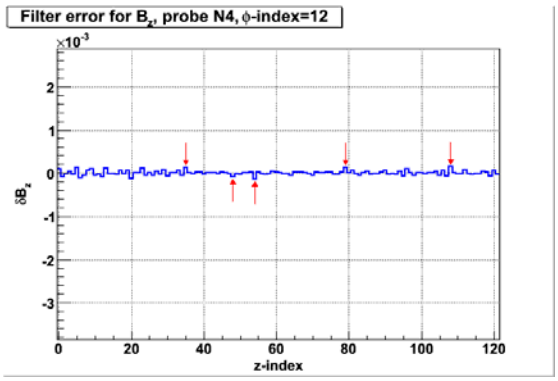


Fig. 7.31 Filter error for the N4 probe at n_ϕ= 12

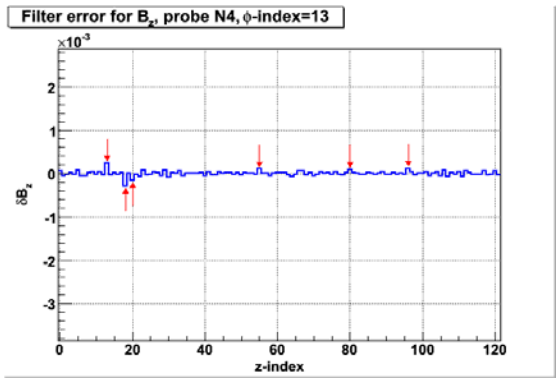


Fig. 7.32 Filter error for the N4 probe at n_ϕ= 13

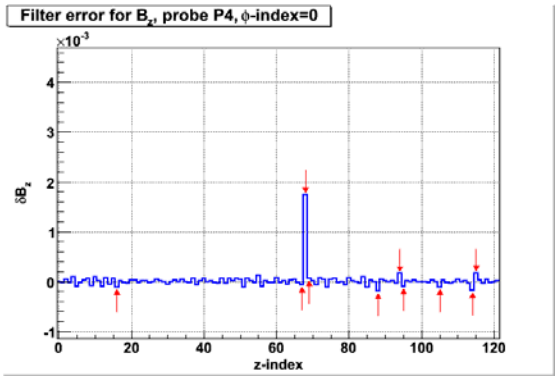


Fig. 7.33 Filter error for the P4 probe at n_ϕ= 0

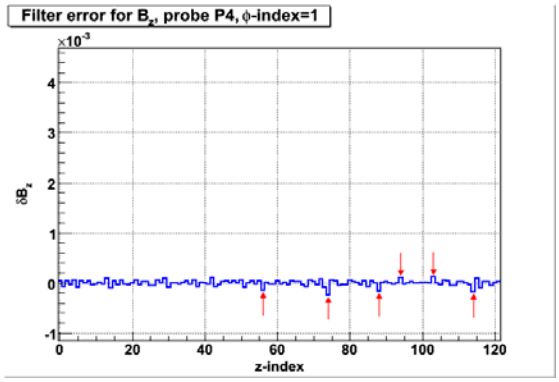


Fig. 7.34 Filter error for the P4 probe at n_ϕ= 1

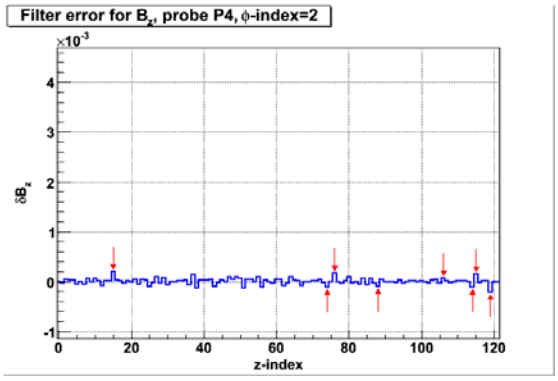


Fig. 7.35 Filter error for the P4 probe at n_ϕ= 2

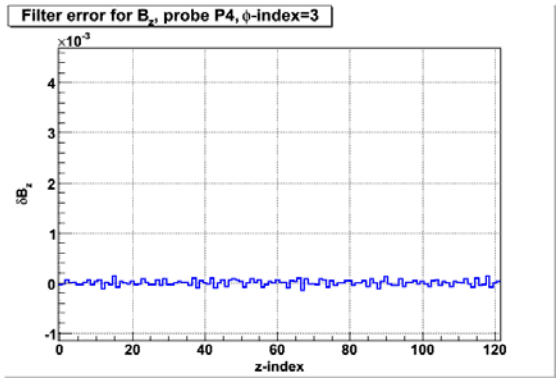


Fig. 7.36 Filter error for the P4 probe at n_ϕ= 3

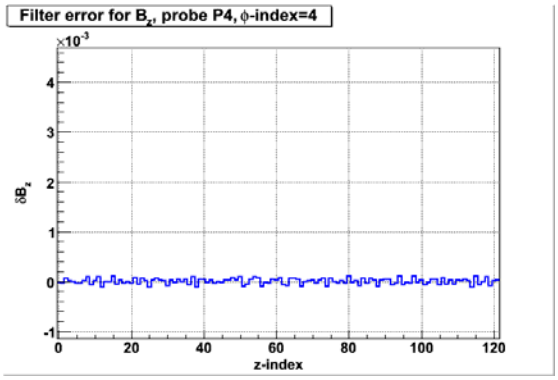


Fig. 7.37 Filter error for the P4 probe at n_ϕ= 4

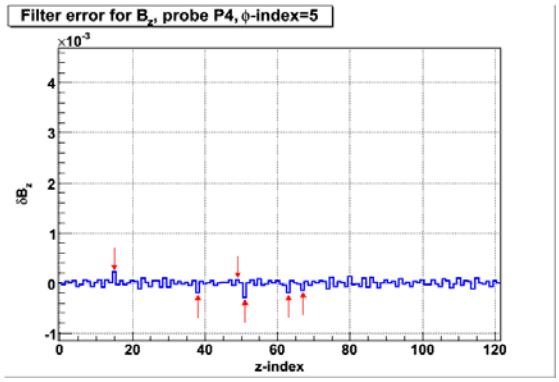


Fig. 7.38 Filter error for the P4 probe at n_ϕ= 5

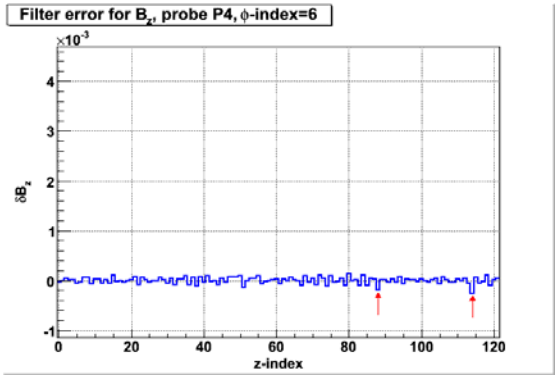


Fig. 7.39 Filter error for the P4 probe at n_ϕ= 6

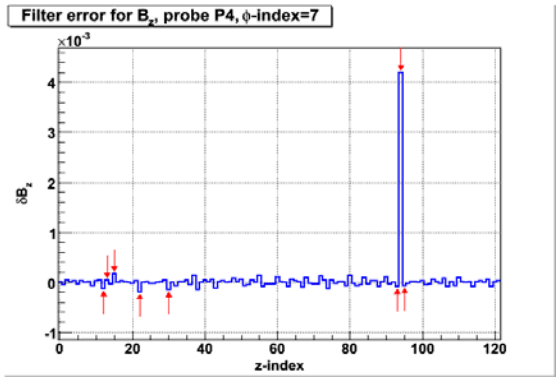


Fig. 7.40 Filter error for the P4 probe at n_ϕ= 7

For the N1--N4 and P1--P4 probes the filter works very well (all result were checked visually). The situation differs for the outermost probes N5 and P5.

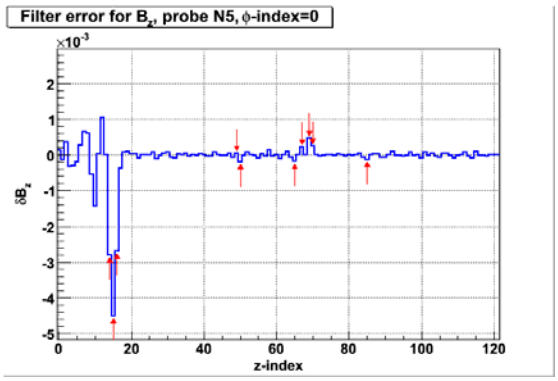


Fig. 7.41 Filter error for the N5 probe at n_ϕ= 0

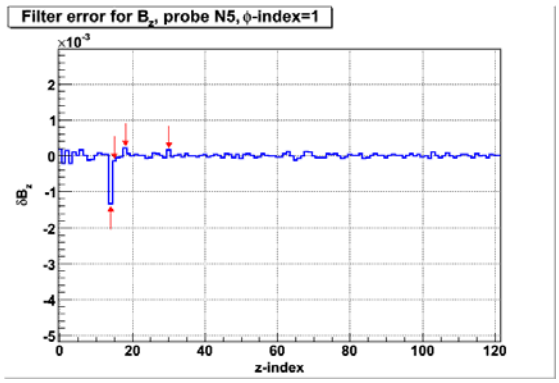


Fig. 7.42 Filter error for the N5 probe at n_ϕ= 1

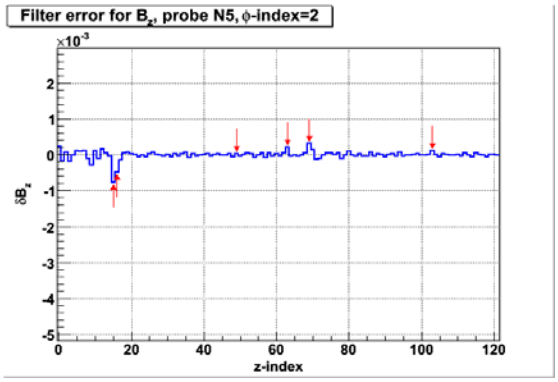


Fig. 7.43 Filter error for the N5 probe at $n_\phi = 2$

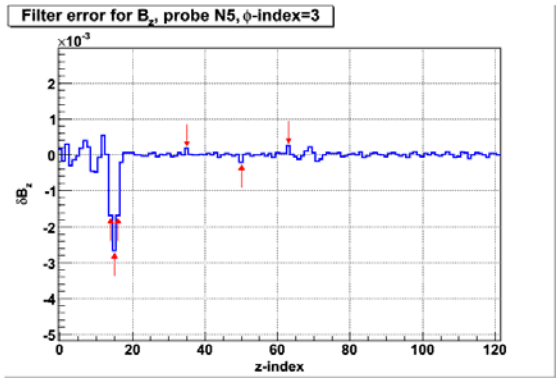


Fig. 7.44 Filter error for the N5 probe at $n_\phi = 3$

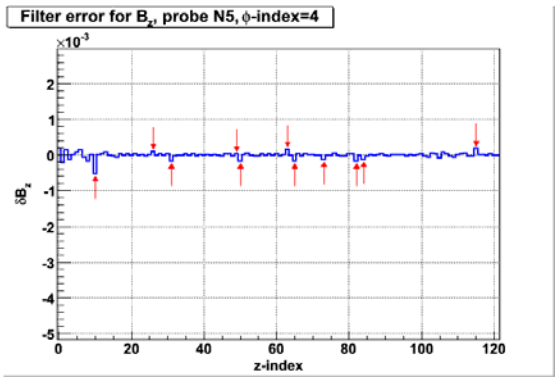


Fig. 7.45 Filter error for the N5 probe at $n_\phi = 4$

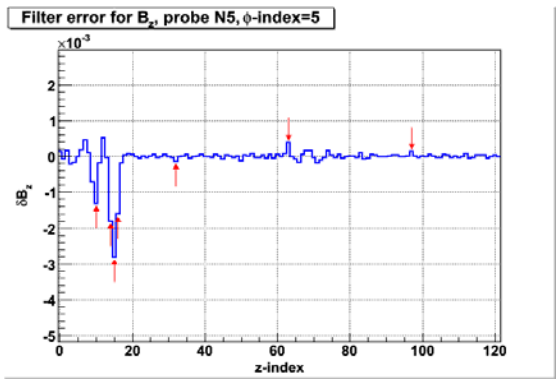


Fig. 7.46 Filter error for the N5 probe at $n_\phi = 5$

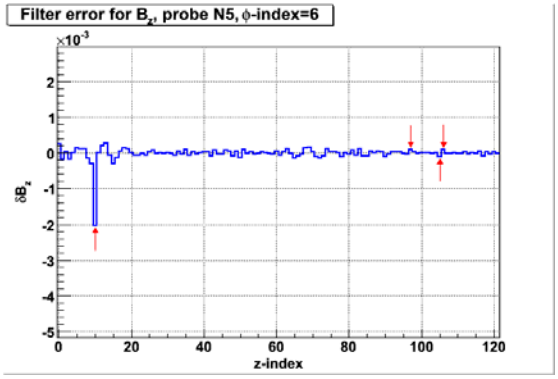


Fig. 7.47 Filter error for the N5 probe at $n_\phi = 6$

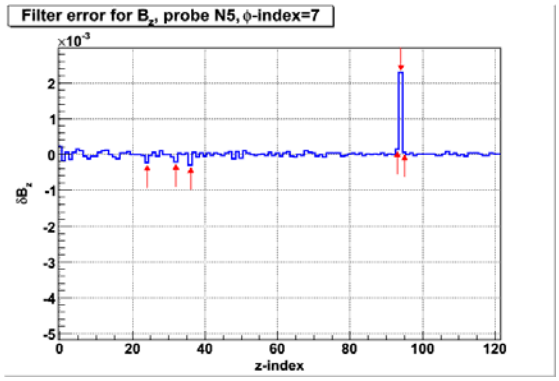


Fig. 7.48 Filter error for the N5 probe at $n_\phi = 7$

In the plots for N5 probe one may notice peaks which appear in the interval of *z*-positions 0…20. However, there is not a single point with a value that is far away from adjacent values. Instead, some persistent structure is observed there; one may compare Fig. 7.44 and Fig. 7.46, for example. We consider now the P5 probe.

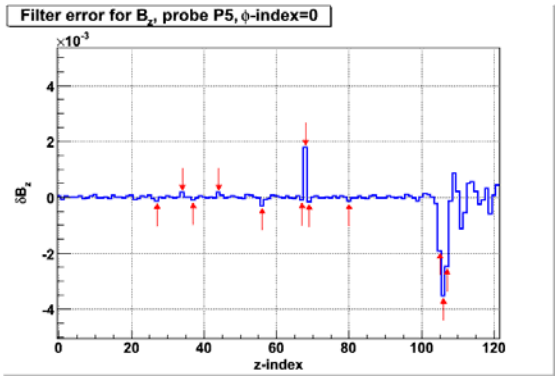


Fig. 7.49 Filter error for the P5 probe at n_ϕ= 0

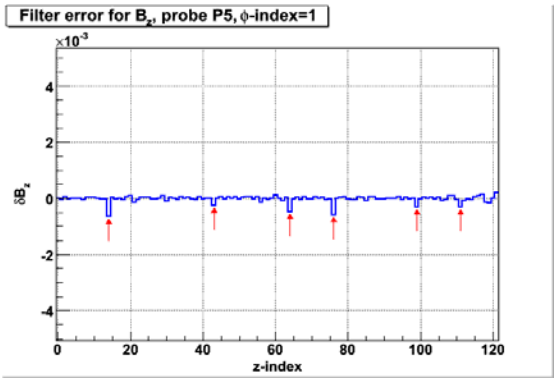


Fig. 7.50 Filter error for the P5 probe at n_ϕ= 1

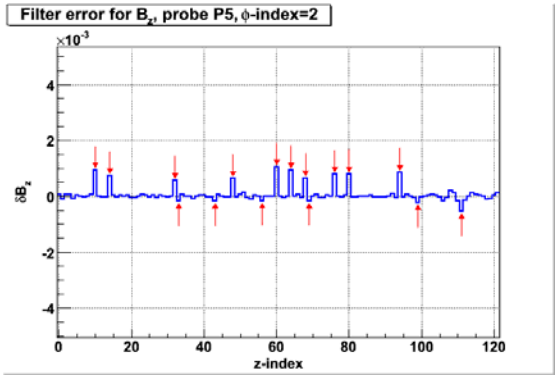


Fig. 7.51 Filter error for the P5 probe at n_ϕ= 2

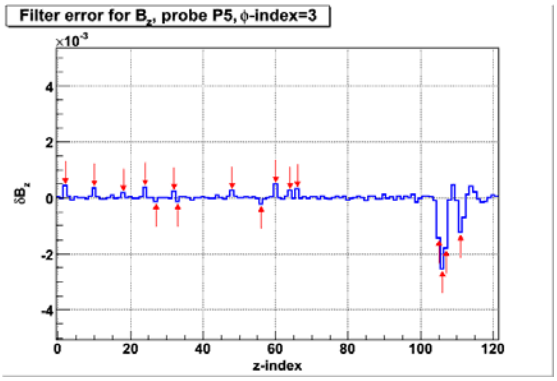


Fig. 7.52 Filter error for the P5 probe at n_ϕ= 3

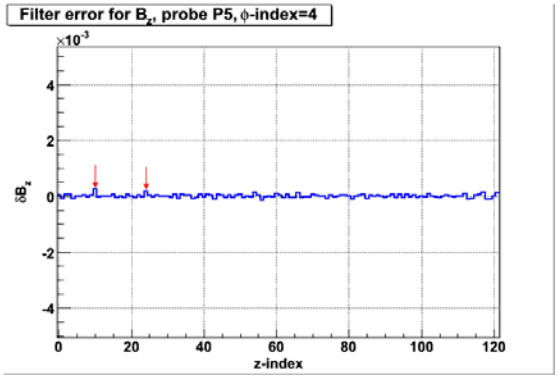


Fig. 7.53 Filter error for the P5 probe at n_ϕ= 4

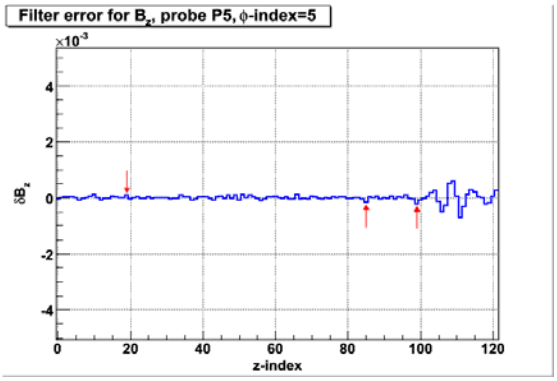


Fig. 7.54 Filter error for the P5 probe at n_ϕ= 5

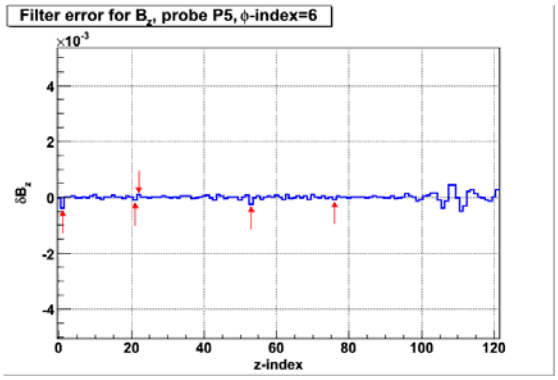


Fig. 7.55 Filter error for the P5 probe at n_ϕ= 6

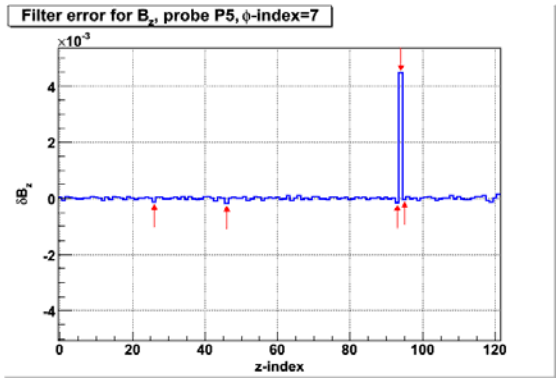


Fig. 7.56 Filter error for the P5 probe at n_ϕ= 7

The filter for the probe P5 works well except for the range of outermost *z*-positions 101…121. Similar to the probe N5, there is a persistent structure, which appears periodically as the ϕ-index runs. See, for instance Fig. 7.49 and Fig. 7.52. Again, it cannot be said that outliers are observed, because there is not a single point which jumps far away from neighboring points. The filter in some cases leaves this structure unchanged (Fig. 7.54, Fig. 7.55), but in other cases it treats most of the points in the outermost position as outliers (Fig. 7.49, Fig. 7.52).

It is not understood yet what the reason is for fast spatial field oscillations in the outermost *z*-positions of the N5 and P5 probes. After careful consideration it was decided to make the data preconditioning in the following way: to filter all data for the N1÷N4 and P1÷P4 probes, but leave the data in the outermost positions for N5 and P5 unfiltered and restrict the use of this unfiltered data in fits until a reason of the fast spatial field oscillations is understood.

8 FIT IN 3 DIMENSIONS (RESULTS AND DISCUSSION)

The measured data had been corrected for the fieldmapper tilt, gain shifts, z-offsets, $B_z \rightarrow B_r$ mixing and then filtered as described in the previous chapter.

8.1 Normal Equation Matrix Evaluation

The least square fit of the corrected data was made over the 3D basis with the maximum $l = 18$. Some precautions had been taken in preparation of the matrix **A** before the normal equation matrix $\mathbf{A}^{\mathbf{T}}\mathbf{A}$ (3.12) was calculated:

- Equations for B_r were multiplied by the weight coefficient 0.3, so the fit enchanced the B_z measurement.
- Further the equations for the outermost z-positions of the N5 (z-index = 0…20) and P5 (z-index = 101…121) were multiplied by the weight coefficients 0.1. This way we suppressed the influence of strange and not yet understood field oscillation in these z-ranges.

The $\mathbf{A}^{\mathbf{T}}\mathbf{A}$ matrix calculation evaluation of each matrix element had been made with preceding sort of terms by absolute values and the sum accumulation started with smaller terms. This precaution had to be taken because with the chosen basis the dimension of matrix **A** is 119560×360.

8.2 Results and Discussion

The fit was performed at all nominal field values. The fit field maps for B_z, B_r, B_ϕ at radii of probes and the residual maps (measured field minus the fit) for B_z, B_r[3] at 2 T and 4 T, are presented in Fig. 9.1 through Fig. 9.120. Overall errors for the B_z fit are presented in Table 8.1 through Table 8.4.

[3] It makes no sense to present the B_ϕ error map, because no corrections for the $B_z \rightarrow B_\phi$ mixing was made.

Table 8.1 The average and RMS of the B_z fit residuals for the N-arm probes at 2 T

Probe	N1	N2	N3	N4	N5
Mean(δB_z)	2.04×10^{-6}T	-3.69×10^{-6}T	-3.1×10^{-6}T	3.1×10^{-6}T	3.66×10^{-5}T
RMS(δB_z)	2.95×10^{-5}T	5.46×10^{-5}T	3.51×10^{-5}T	5.94×10^{-5}T	2.76×10^{-4}T

Table 8.2 B_z The average and RMS of the fit residuals for the P-arm probes at 2 T

Probe	P1	P2	P3	P4	P5
Mean(δB_z)	-2.64×10^{-6}T	7.08×10^{-7}T	-3.31×10^{-6}T	2.85×10^{-7}T	3.89×10^{-5}T
RMS(δB_z)	3.36×10^{-5}T	4.34×10^{-5}T	4.23×10^{-5}T	6.12×10^{-5}T	3.12×10^{-4}T

Table 8.3 The average and RMS of the B_z fit residuals for the N-arm probes at 4 T

Probe	N1	N2	N3	N4	N5
Mean(δB_z)	-1.01×10^{-6}T	5.8×10^{-5}T	1.44×10^{-6}T	-6.08×10^{-5}T	3.89×10^{-5}T
RMS(δB_z)	4.87×10^{-5}T	7.26×10^{-5}T	6.08×10^{-5}T	8.25×10^{-5}T	3.3×10^{-4}T

Table 8.4 B_z The average and RMS of the fit residuals for the P-arm probes at 4 T

Probe	P1	P2	P3	P4	P5
Mean(δB_z)	5.89×10^{-6}T	5.7×10^{-6}T	2.29×10^{-5}T	1.2×10^{-5}T	-1.16×10^{-5}T
RMS(δB_z)	5.49×10^{-5}T	6.42×10^{-5}T	5.5×10^{-5}T	7.92×10^{-5}T	3.5×10^{-4}T

Both the plots and tables show an excellent B_z fit for N1 through N4 and P1 through P4 probes; the precision is within $\pm1\times10^{-4}$ T range.

For the N5 and P5 probes the fit has larger errors (locally: up to $\pm2\times10^{-3}$ T at 2 T and $\pm4\times10^{-3}$ T at 4 T) in the outermost z-positions, but I had neither the intention nor the possibility to reproduce fast spatial oscillations of the field in this areas. It can be counted that there are 18 periods of oscillation along ϕ at fixed z (see Fig. 9.125, for instance) and 2 periods along z at fixed ϕ (see Fig. 7.46 and Fig. 7.52 , for example). In order to reproduce these oscillations a basis with $l \geq 18\times2 = 36$ must be used. The total number of the basis elements (= number of unknowns in the normal equation) in that case is $l(l+2) = 1368$.

The field fit maps and residuals for N5, P5 probes at a shortened z-range are presented in Fig. 9.121 through Fig. 9.124, Fig. 9.127 through Fig. 9.130, Fig. 9.133 through Fig. 9.136 and Fig. 9.139 through Fig. 9.142. Overall B_z fit errors are presented in Table 8.5.

Table 8.5 The average and RMS of the B_z fit residuals for the N5 and P5 probes at 4 T at shortened z-range

Field	2 T		4 T	
Probe	N5	P5	N5	P5
Mean(δB_z)	-1.43×10^{-5}T	-1.37×10^{-5}T	-2.39×10^{-5}T	-5.38×10^{-5}T
RMS(δB_z)	1.16×10^{-4}T	1.38×10^{-4}T	1.43×10^{-4}T	1.68×10^{-4}T

For a shortened z-range, residuals of the fit for N5 and P5 probes are approximately by factor 2 larger than residuals for inner probes. This increase of the RMS δB_z due to a shoulder on the right slope of the main peak in Fig. 9.122, Fig. 9.128, Fig. 9.134 and Fig. 9.140. This shoulder corresponds to a structure in δB_z plots for the outermost probes which can be described as a "double ridge" along z near $\phi = -\frac{1}{2}\pi$ (that corresponds to ϕ-index = 24 for the N-arm and ϕ-index = 0 for the P-arm); see Fig. 9.126 and Fig. 9.132. An appearance of another structure in the δB_r plots, which may be described as a "ridge transforming to a canyon" along z at the same ϕ can be noticed. We do not know what has caused this local field variation, but it appears in the position of an arm aligned with the fieldmapper "leg". It looks very likely the field perturbation was somehow caused by the fieldmapper itself. Anyway, our fit cannot reproduce such a fast local variation of the field.

9 CONCLUSIONS AND PLANS

The analytical field fit, both for the axially symmetric and complete 3D field models, had been obtained which is within $\pm 4 \cdot 10^{-4}$ T agreement with the measured field values in whole volume spanned in measurements except the most remote off center part of it, which makes only 6% of total volume. The axial field model, implemented in the *BFit* C++ code, shows both high precision and speed. At the time of writing it is being integrated into the CMSSW software for the offline particle tracking. That is the most important practical outcome of this work.

The representation of spherical harmonics in cylindrical coordinate system that has been derived during this work is a result of essential practical importance too. The fit could not been made with no use of the basis derived.

The 3D field model will be used to find out whether neglect of the field deviation from the axial symmetry essentially worsens the momentum resolution. In the case if it does, a further work aimed to improve the 3D model computing speed will be carried out.

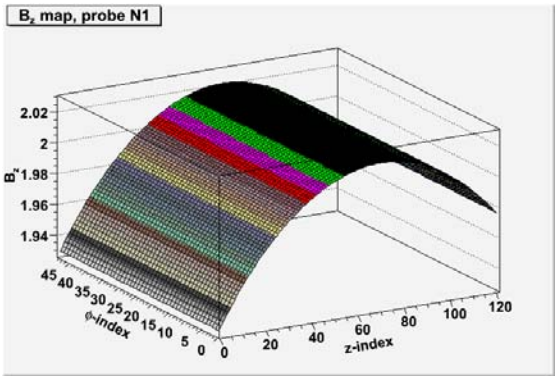


Fig. 9.1 B_z **map for the N1 probe at 2 T**

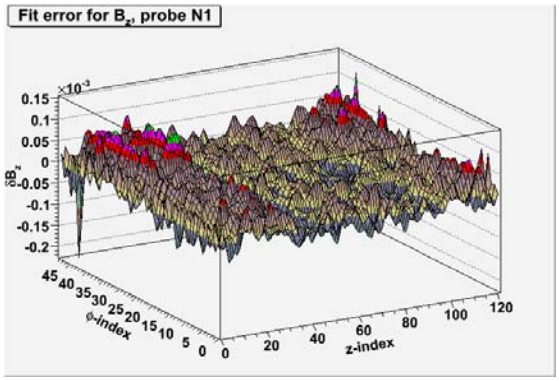


Fig. 9.2 B_z **residuals for the N1 probe at 2 T**

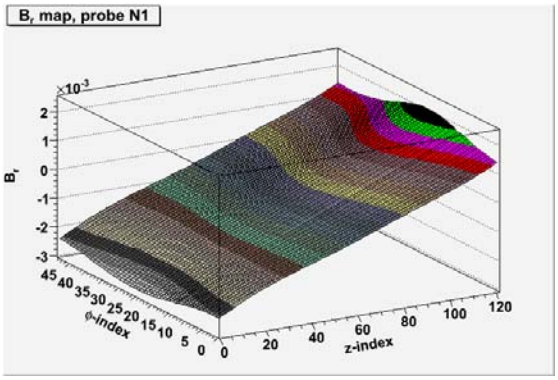


Fig. 9.3 B_r **map for the N1 probe at 2 T**

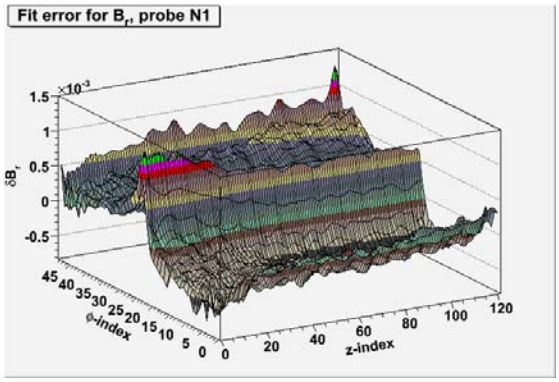


Fig. 9.4 B_r **residuals for the N1 probe at 2 T**

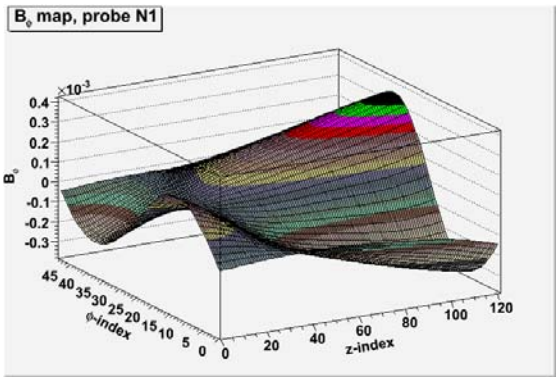


Fig. 9.5 B_ϕ **map for the N1 probe at 2 T**

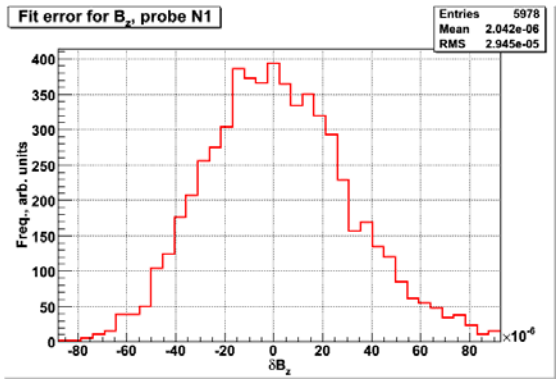


Fig. 9.6 δB_z **spread for the N1 probe at 2 T**

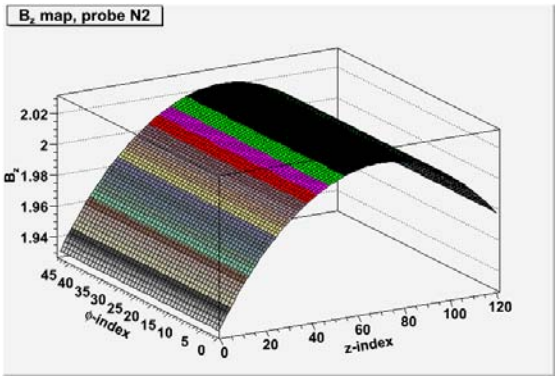


Fig. 9.7 B_z **map for the N2 probe at 2 T**

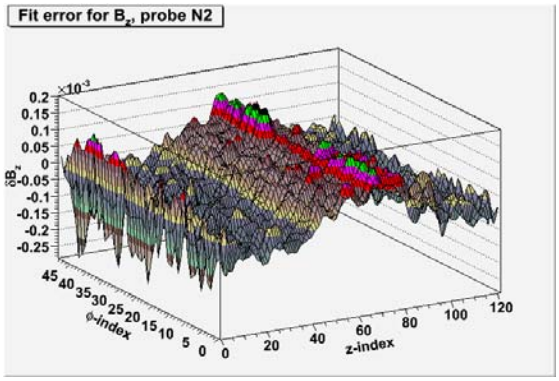


Fig. 9.8 B_z **residuals for the N2 probe at 2 T**

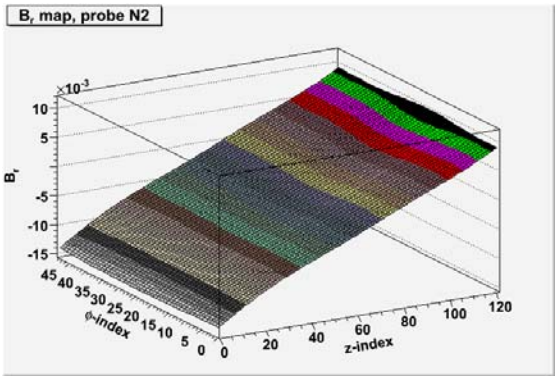


Fig. 9.9 B_r **map for the N2 probe at 2 T**

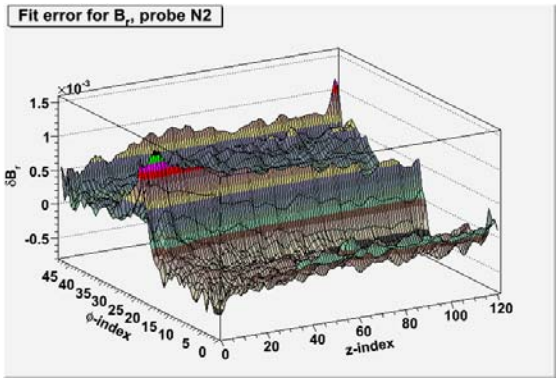


Fig. 9.10 B_r **residuals for the N2 probe at 2 T**

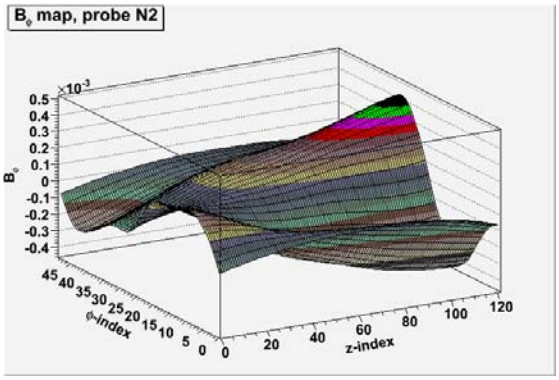


Fig. 9.11 B_ϕ **map for the N2 probe at 2 T**

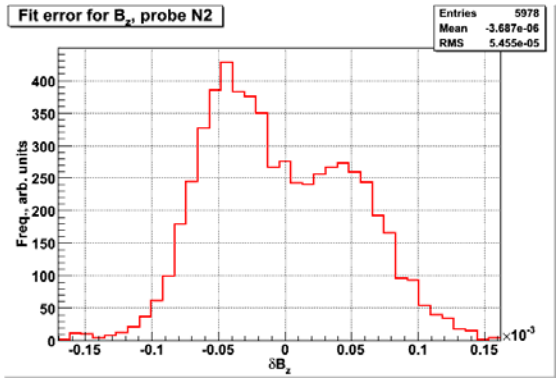


Fig. 9.12 δB_z **spread for the N2 probe at 2 T**

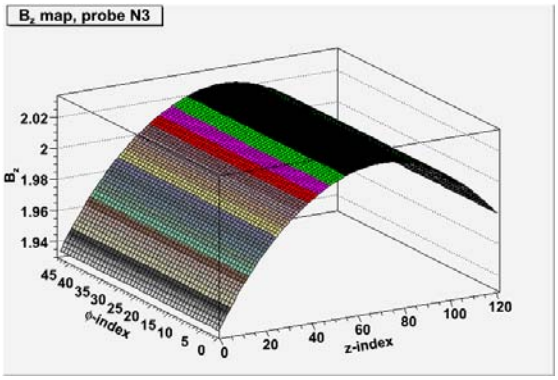


Fig. 9.13 B_z **map for the N3 probe at 2 T**

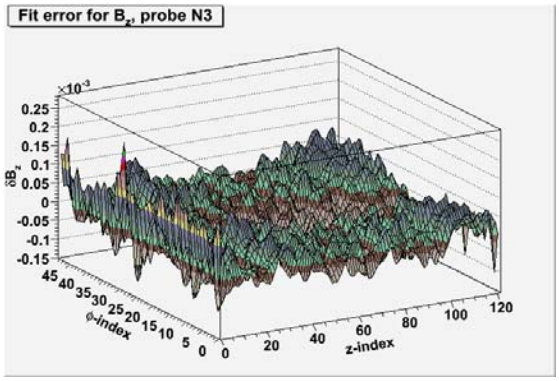


Fig. 9.14 B_z **residuals for the N3 probe at 2 T**

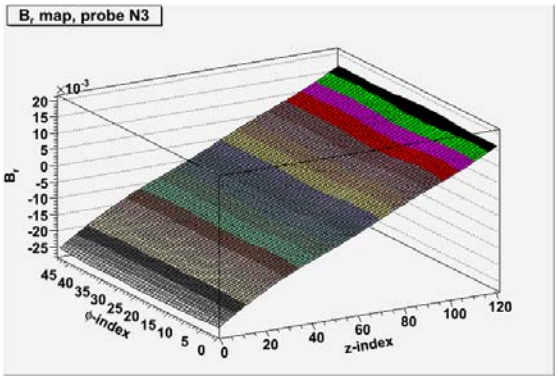


Fig. 9.15 B_r **map for the N3 probe at 2 T**

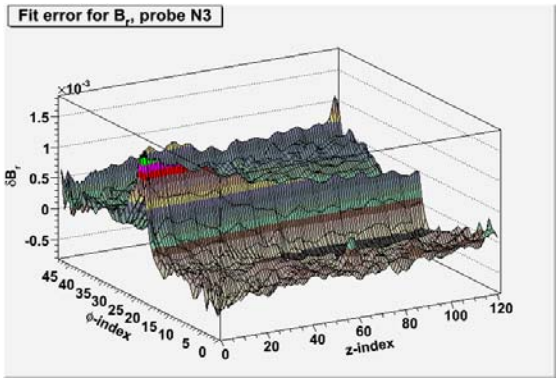


Fig. 9.16 B_r **residuals for the N3 probe at 2 T**

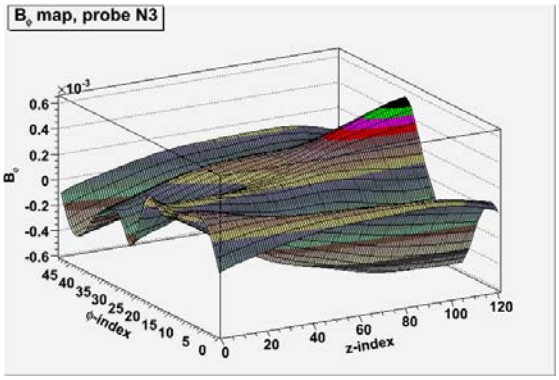


Fig. 9.17 B_ϕ **map for the N3 probe at 2 T**

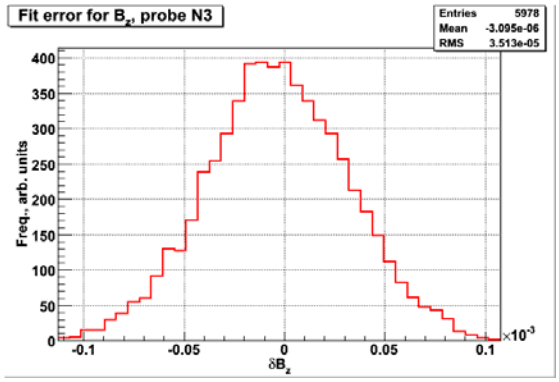


Fig. 9.18 δB_z **spread for the N3 probe at 2 T**

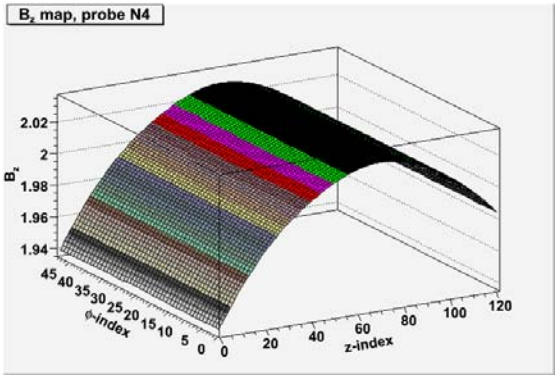


Fig. 9.19 B_z **map for the N4 probe at 2 T**

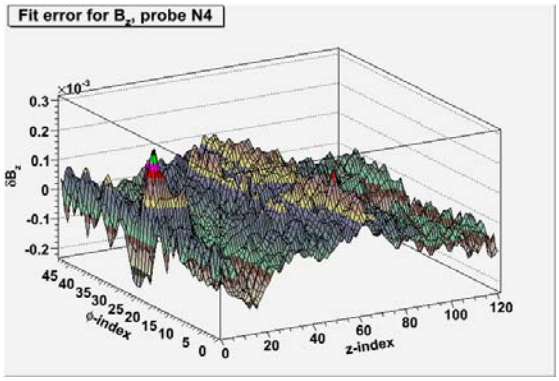


Fig. 9.20 B_z **residuals for the N4 probe at 2 T**

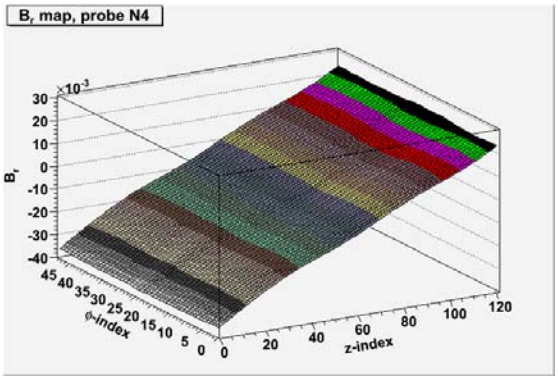


Fig. 9.21 B_r **map for the N4 probe at 2 T**

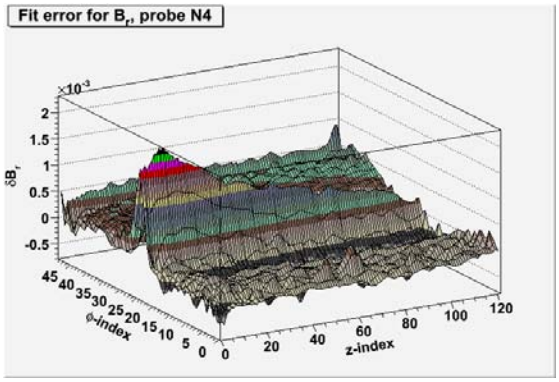


Fig. 9.22 B_r **residuals for the N4 probe at 2 T**

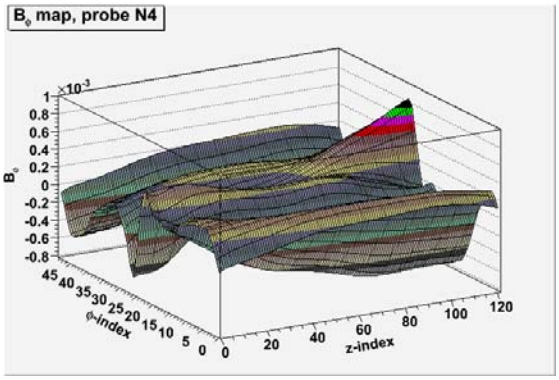


Fig. 9.23 B_ϕ **map for the N4 probe at 2 T**

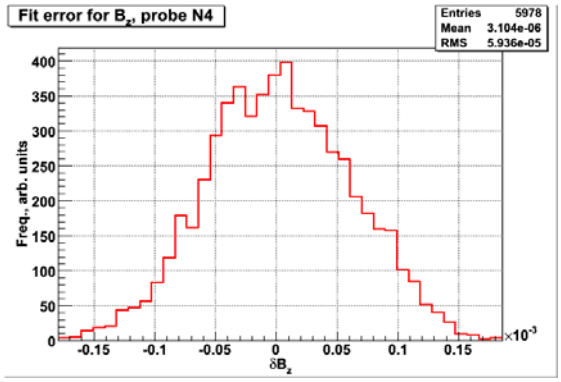


Fig. 9.24 δB_z **spread for the N4 probe at 2 T**

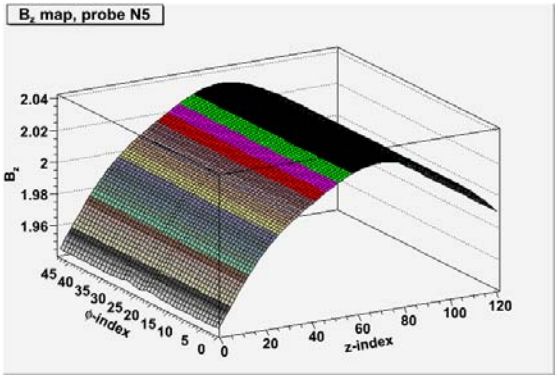


Fig. 9.25 B_z **map for the N5 probe at 2 T**

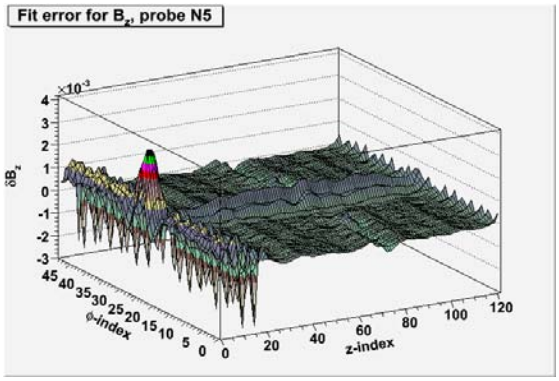


Fig. 9.26 B_z **residuals for the N5 probe at 2 T**

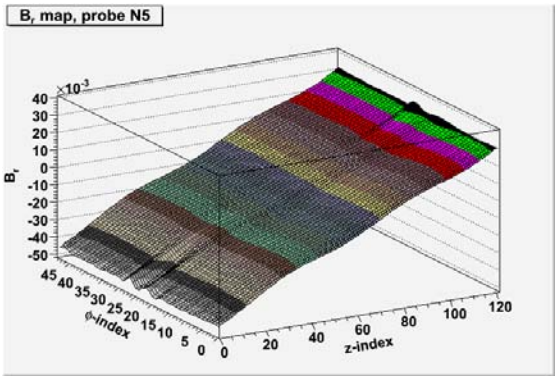


Fig. 9.27 B_r **map for the N5 probe at 2 T**

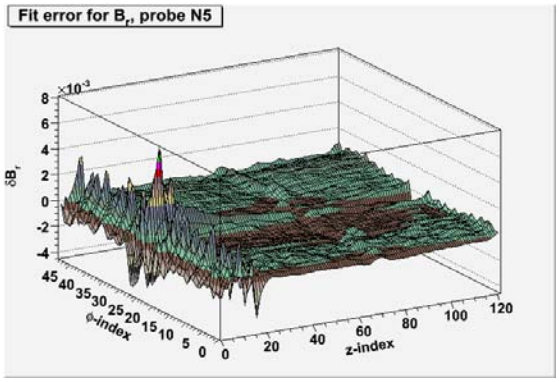


Fig. 9.28 B_r **residuals for the N5 probe at 2 T**

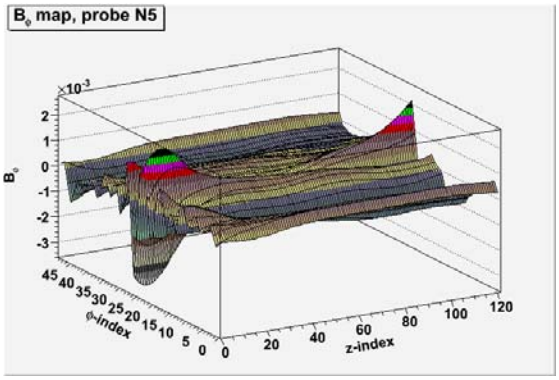


Fig. 9.29 B_ϕ **map for the N5 probe at 2 T**

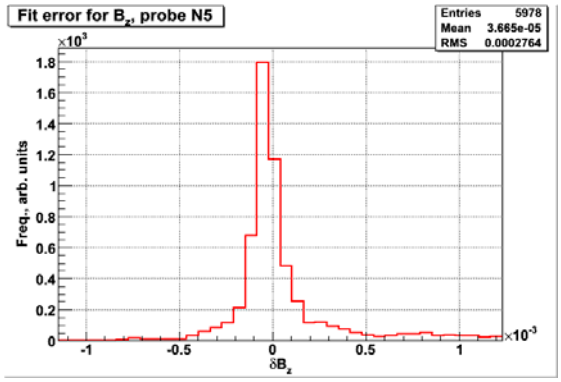


Fig. 9.30 δB_z **spread for the N5 probe at 2 T**

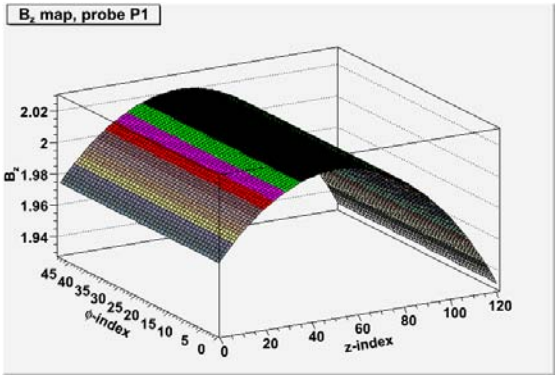


Fig. 9.31 B_z **map for the P1 probe at 2 T**

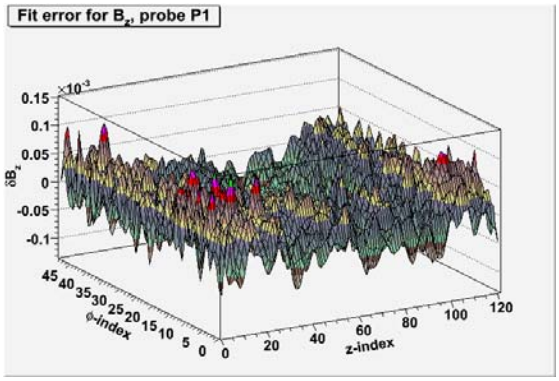


Fig. 9.32 B_z **residuals for the P1 probe at 2 T**

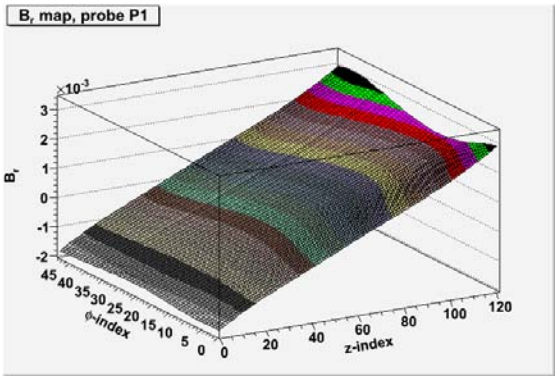


Fig. 9.33 B_r **map for the P1 probe at 2 T**

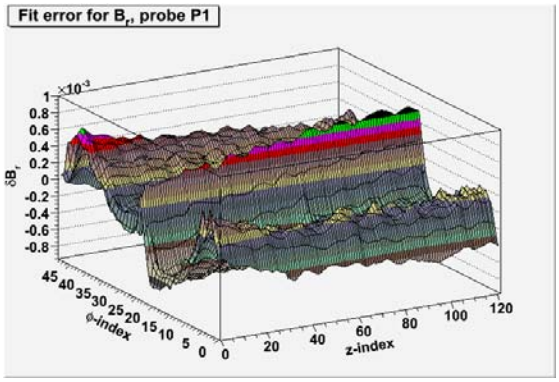


Fig. 9.34 B_r **residuals for the P1 probe at 2 T**

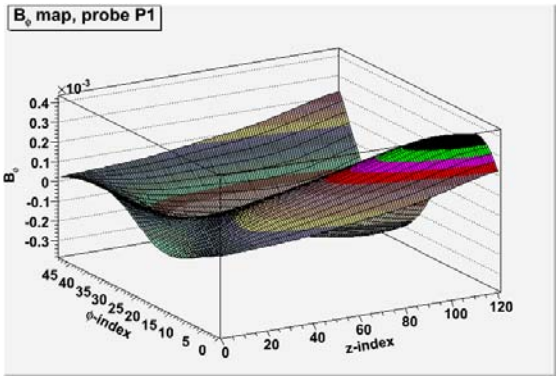


Fig. 9.35 B_ϕ **map for the P1 probe at 2 T**

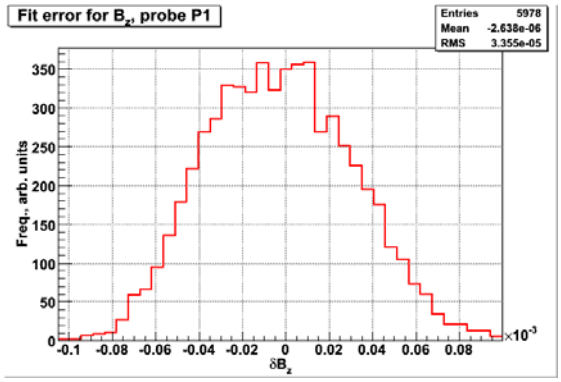


Fig. 9.36 δB_z **spread for the P1 probe at 2 T**

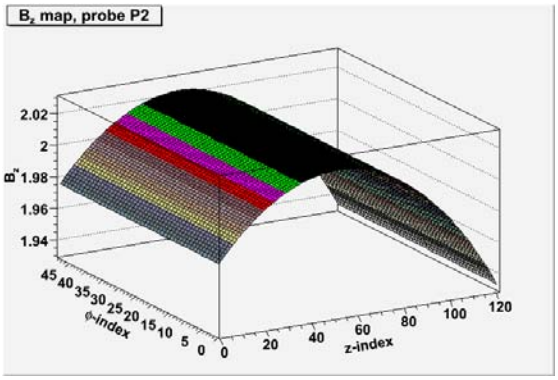


Fig. 9.37 B_z **map for the P2 probe at 2 T**

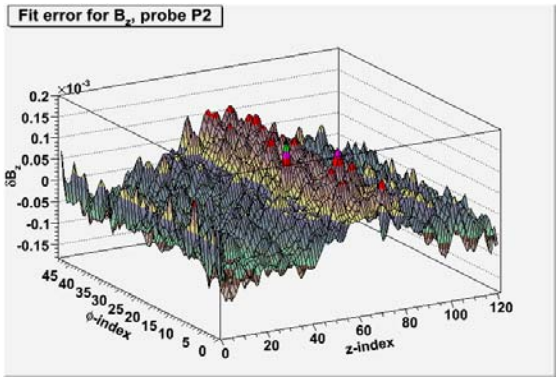


Fig. 9.38 B_z **residuals for the P2 probe at 2 T**

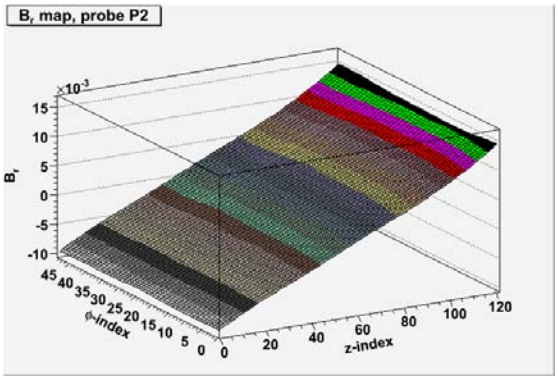


Fig. 9.39 B_r **map for the P2 probe at 2 T**

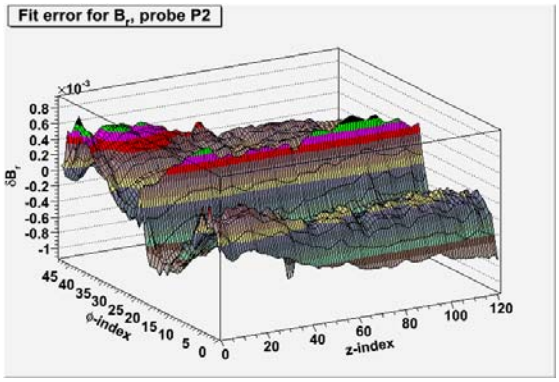


Fig. 9.40 B_r **residuals for the P2 probe at 2 T**

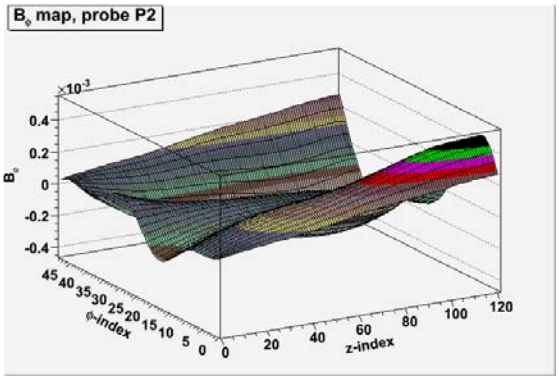


Fig. 9.41 B_ϕ **map for the P2 probe at 2 T**

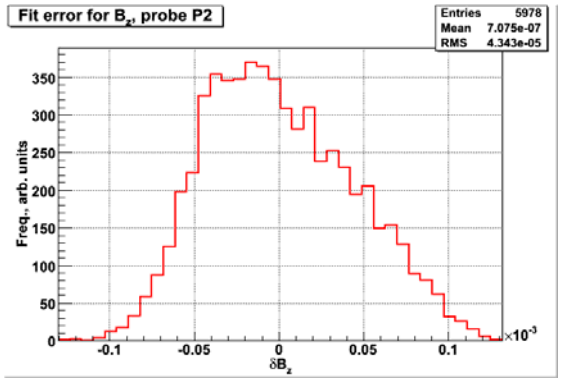


Fig. 9.42 δB_z **spread for the P2 probe at 2 T**

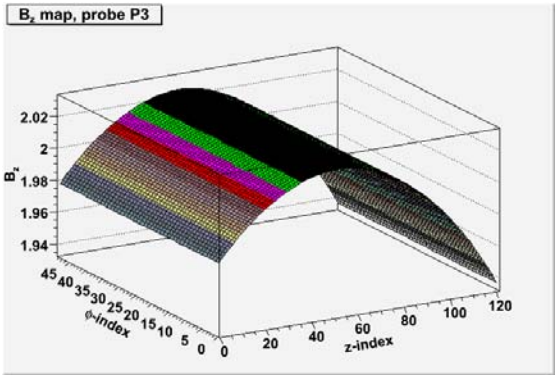


Fig. 9.43 B_z **map for the P3 probe at 2 T**

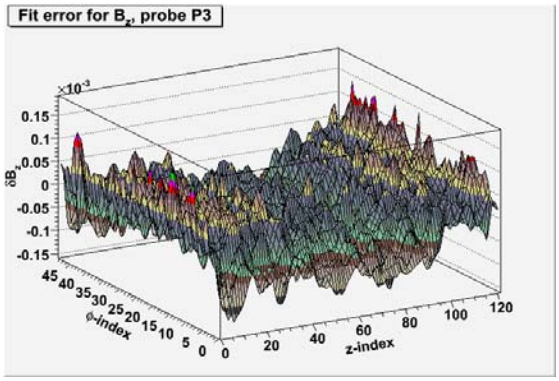


Fig. 9.44 B_z **residuals for the P3 probe at 2 T**

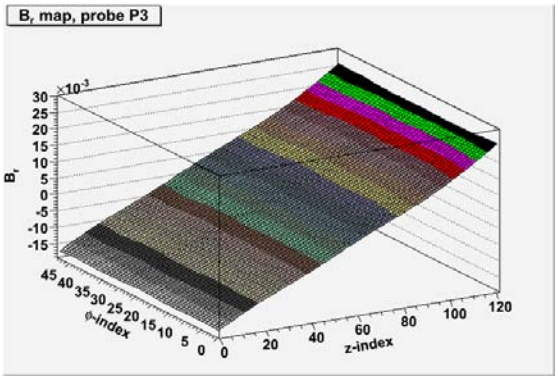


Fig. 9.45 B_r **map for the P3 probe at 2 T**

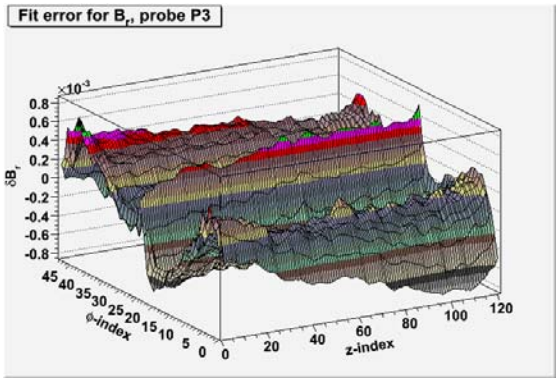


Fig. 9.46 B_r **residuals for the P3 probe at 2 T**

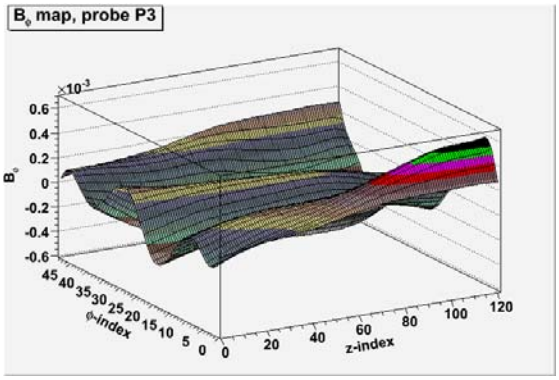


Fig. 9.47 B_ϕ **map for the P3 probe at 2 T**

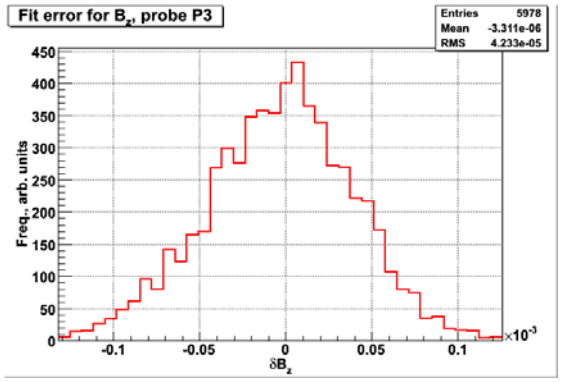


Fig. 9.48 δB_z **spread for the P3 probe at 2 T**

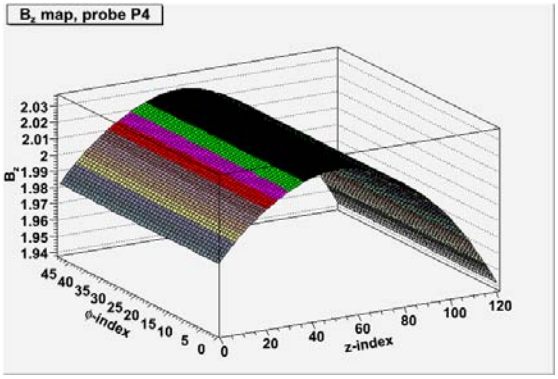


Fig. 9.49 B_z **map for the P4 probe at 2 T**

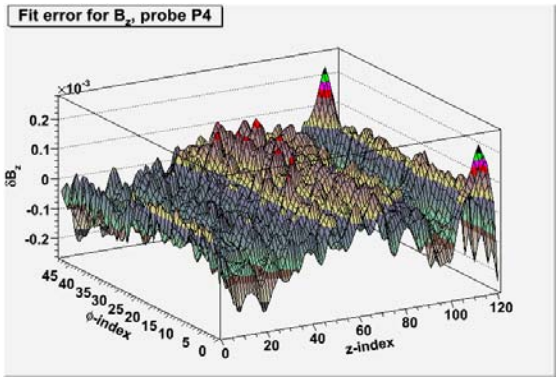


Fig. 9.50 B_z **residuals for the P4 probe at 2 T**

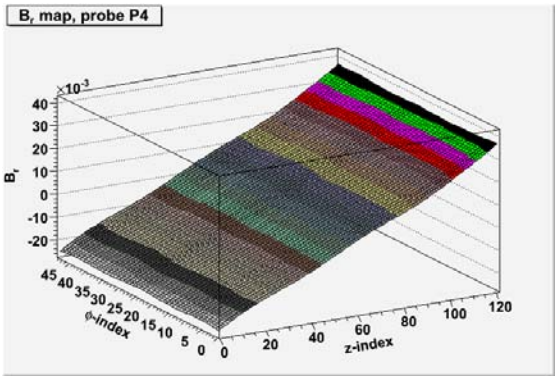


Fig. 9.51 B_r **map for the P4 probe at 2 T**

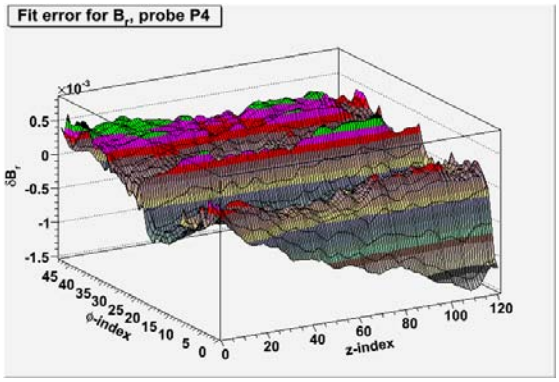


Fig. 9.52 B_r **residuals for the P4 probe at 2 T**

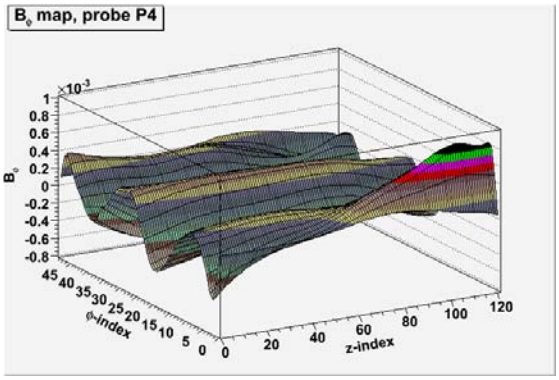


Fig. 9.53 B_ϕ **map for the P4 probe at 2 T**

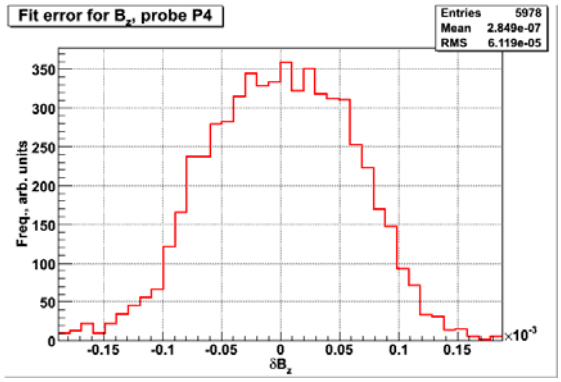


Fig. 9.54 δB_z **spread for the P4 probe at 2 T**

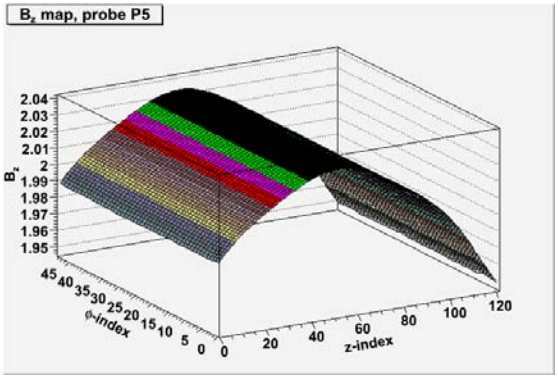


Fig. 9.55 B_z **map for the P5 probe at 2 T**

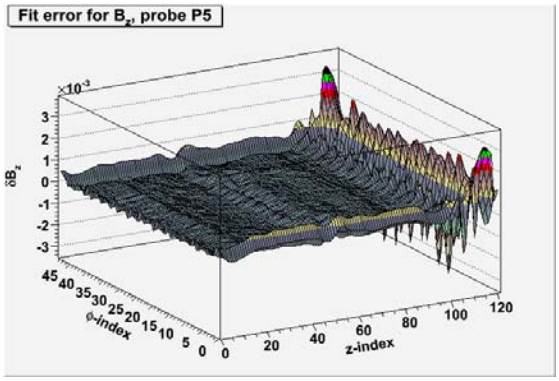


Fig. 9.56 B_z **residuals for the P5 probe at 2 T**

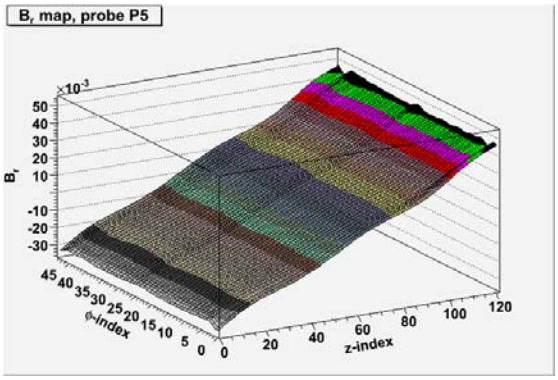


Fig. 9.57 B_r **map for the P5 probe at 2 T**

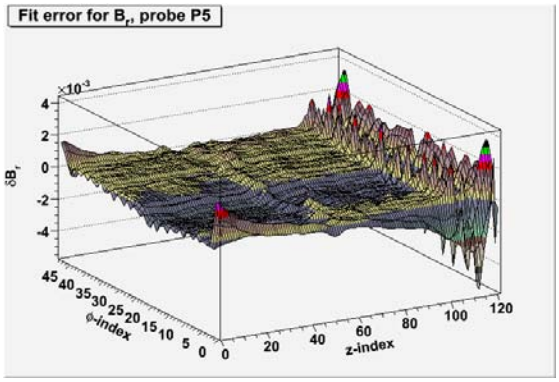


Fig. 9.58 B_r **residuals for the P5 probe at 2 T**

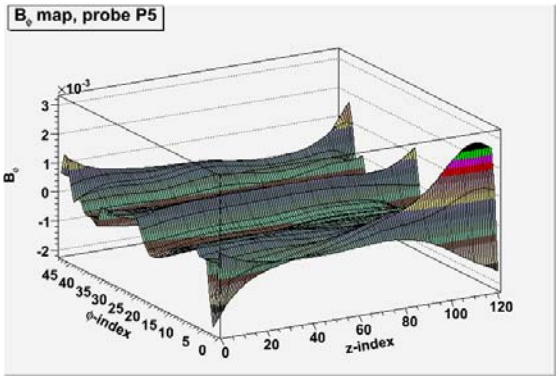


Fig. 9.59 B_ϕ **map for the P5 probe at 2 T**

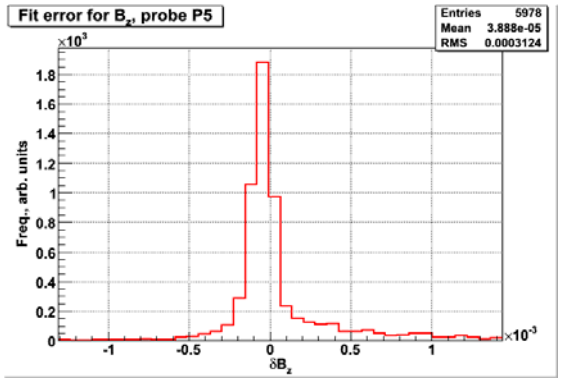


Fig. 9.60 δB_z **spread for the P5 probe at 2 T**

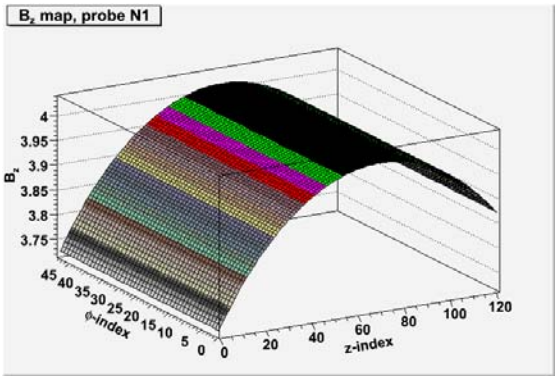


Fig. 9.61 B_z **map for the N1 probe at 4 T**

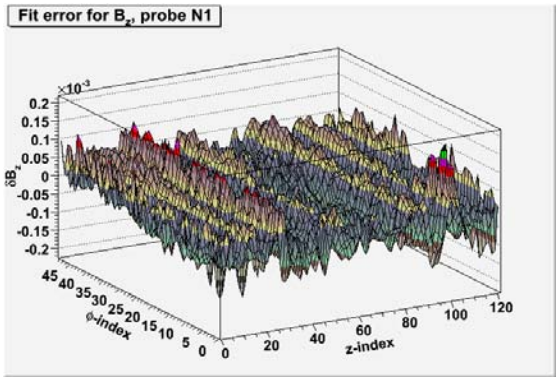


Fig. 9.62 B_z **residuals for the N1 probe at 4 T**

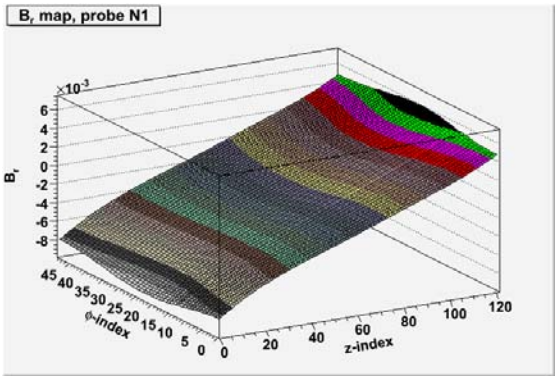


Fig. 9.63 B_r **map for the N1 probe at 4 T**

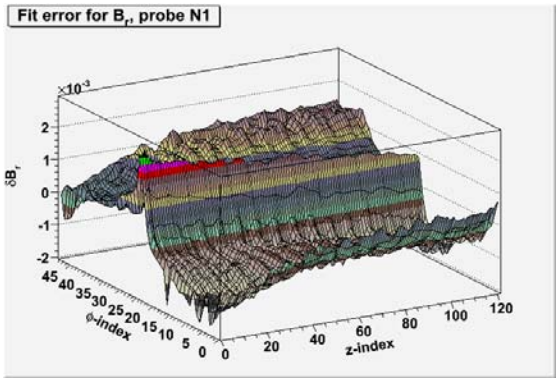


Fig. 9.64 B_r **residuals for the N1 probe at 4 T**

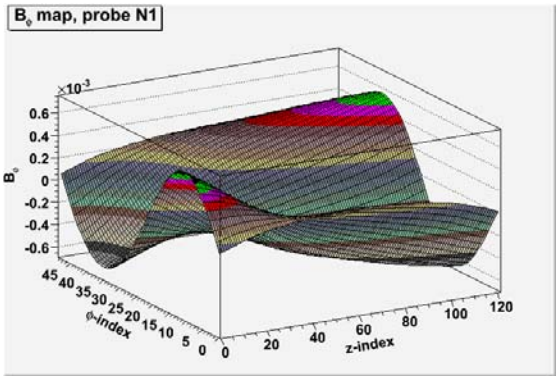


Fig. 9.65 B_ϕ **map for the N1 probe at 4 T**

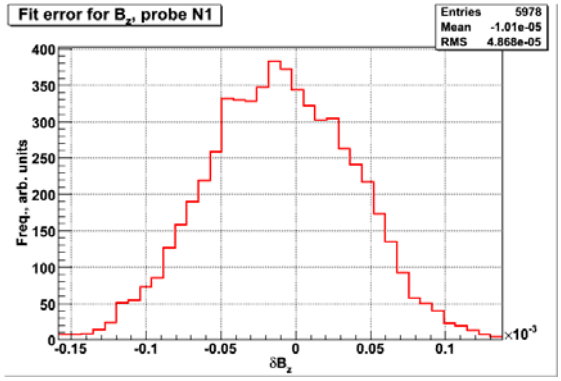


Fig. 9.66 δB_z **spread for the N1 probe at 4 T**

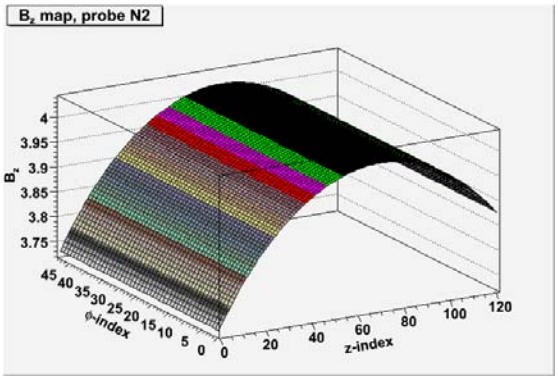


Fig. 9.67 B_z **map for the N2 probe at 4 T**

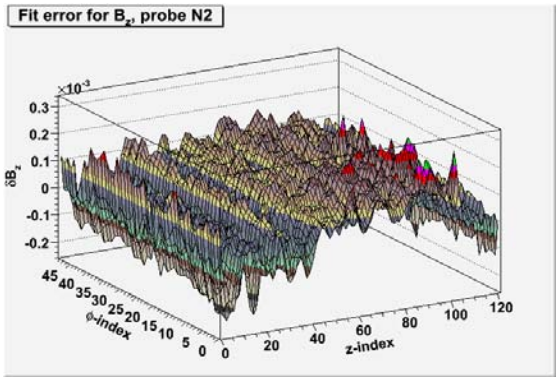


Fig. 9.68 B_z **residuals for the N2 probe at 4 T**

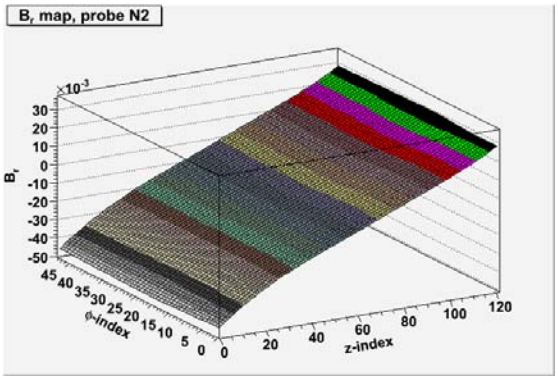


Fig. 9.69 B_r **map for the N2 probe at 4 T**

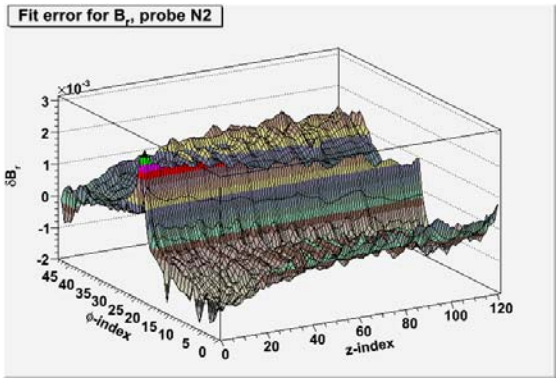


Fig. 9.70 B_r **residuals for the N2 probe at 4 T**

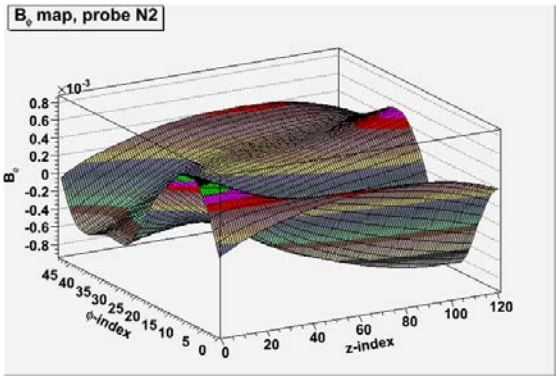


Fig. 9.71 B_ϕ **map for the N2 probe at 4 T**

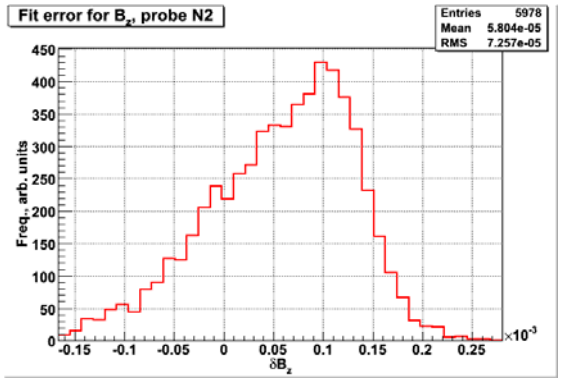


Fig. 9.72 δB_z **spread for the N2 probe at 4 T**

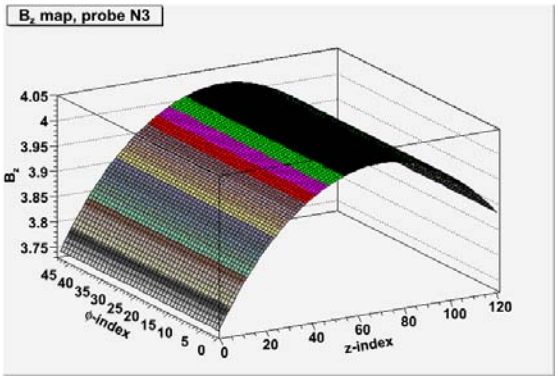


Fig. 9.73 B_z **map for the N3 probe at 4 T**

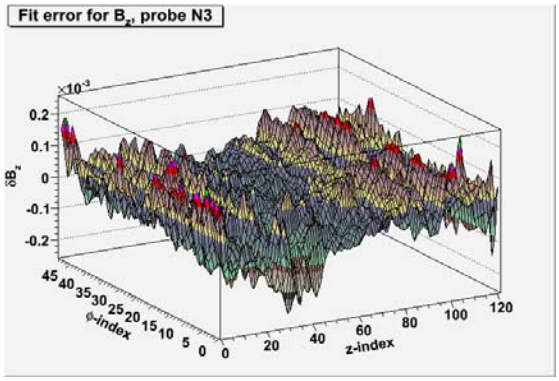


Fig. 9.74 B_z **residuals for the N3 probe at 4 T**

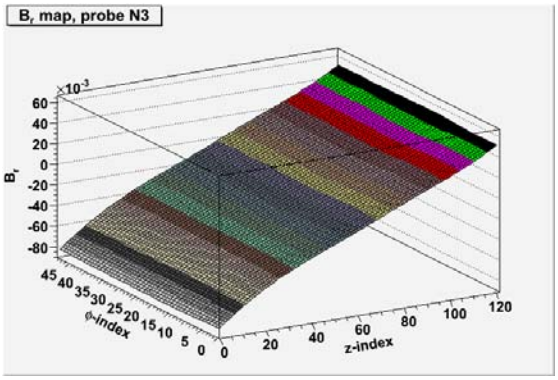


Fig. 9.75 B_r **map for the N3 probe at 4 T**

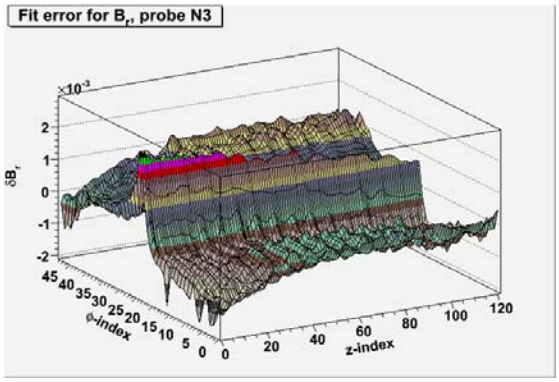


Fig. 9.76 B_r **residuals for the N3 probe at 4 T**

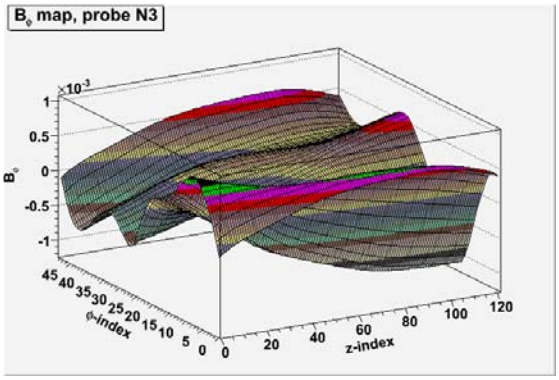


Fig. 9.77 B_ϕ **map for the N3 probe at 4 T**

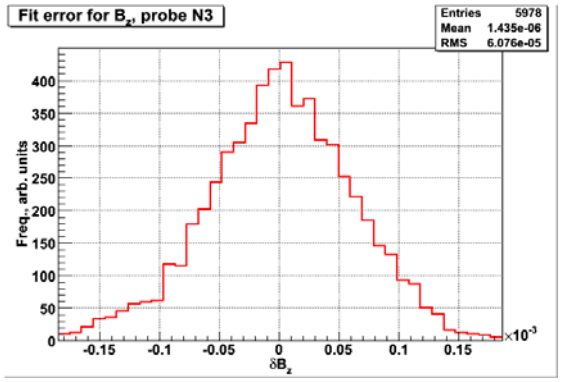


Fig. 9.78 δB_z **spread for the N3 probe at 4 T**

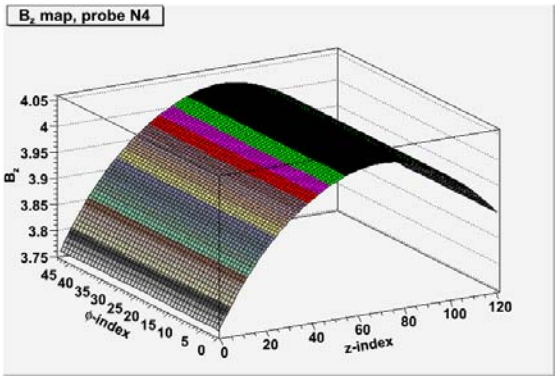


Fig. 9.79 B_z map for the N4 probe at 4 T

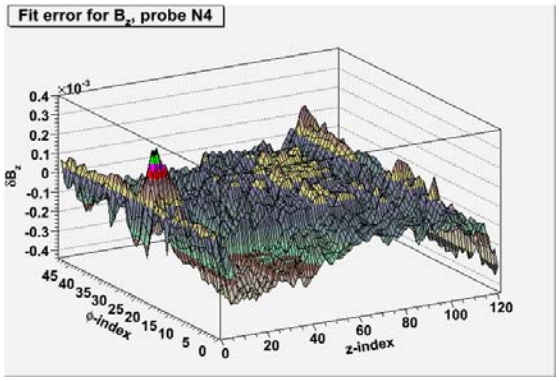


Fig. 9.80 B_z residuals for the N4 probe at 4 T

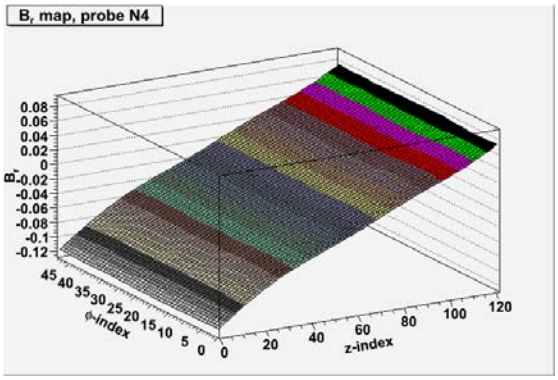


Fig. 9.81 B_r map for the N4 probe at 4 T

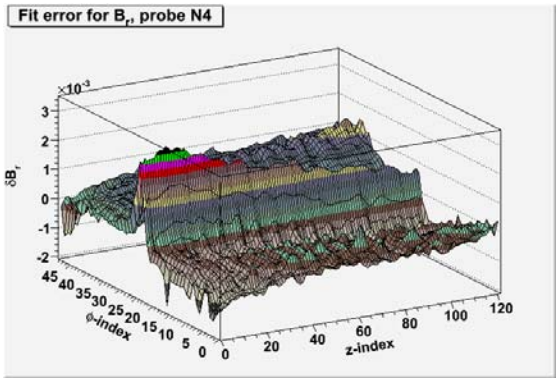


Fig. 9.82 B_r residuals for the N4 probe at 4 T

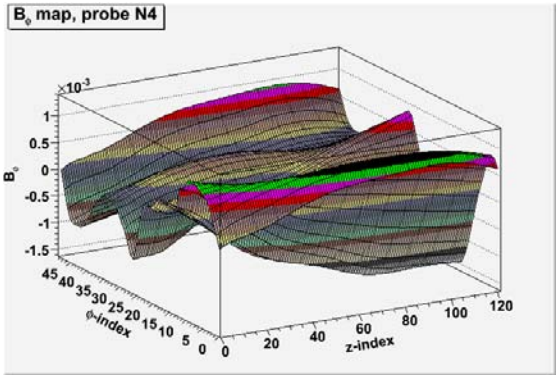


Fig. 9.83 B_ϕ map for the N4 probe at 4 T

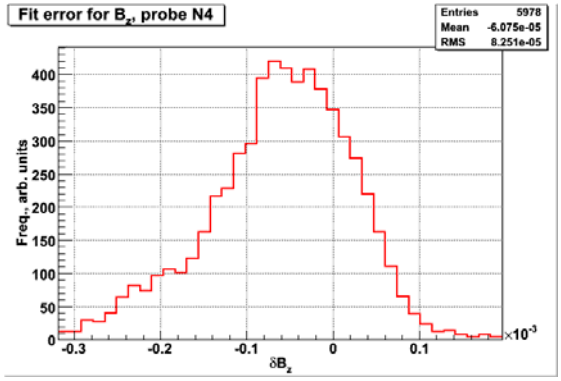


Fig. 9.84 δB_z spread for the N4 probe at 4 T

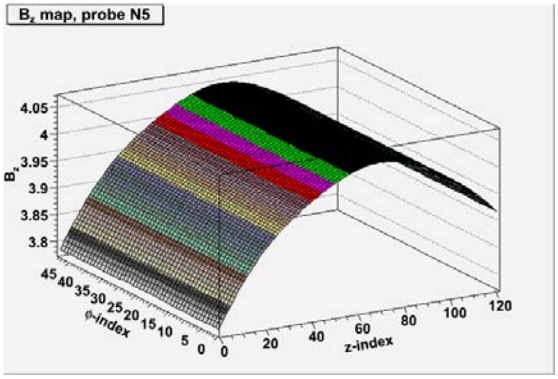


Fig. 9.85 B_z **map for the N5 probe at 4 T**

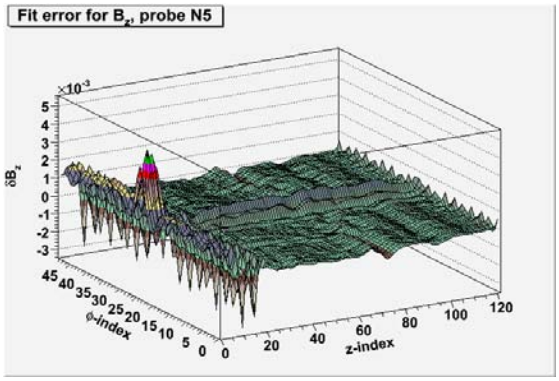


Fig. 9.86 B_z **residuals for the N5 probe at 4 T**

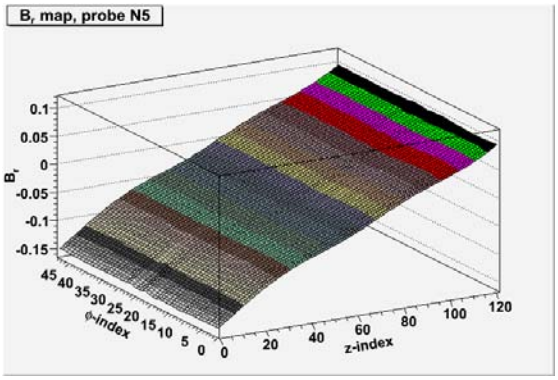


Fig. 9.87 B_r **map for the N5 probe at 4 T**

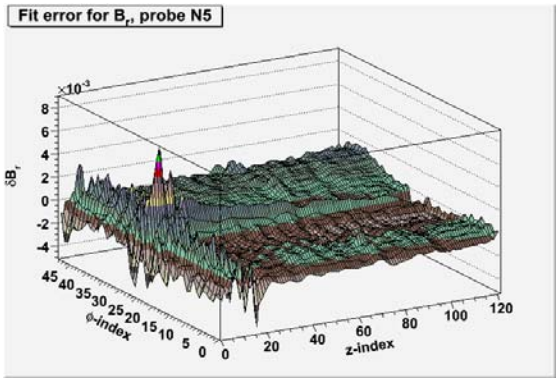


Fig. 9.88 B_r **residuals for the N5 probe at 4 T**

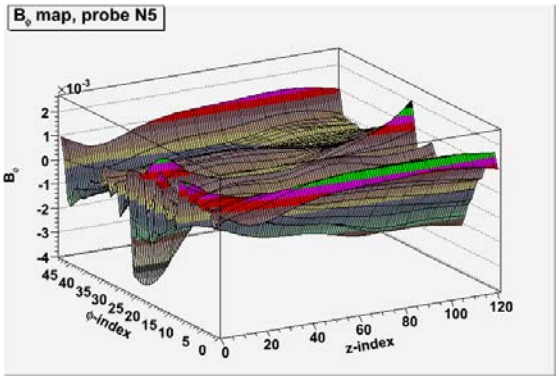


Fig. 9.89 B_ϕ **map for the N5 probe at 4 T**

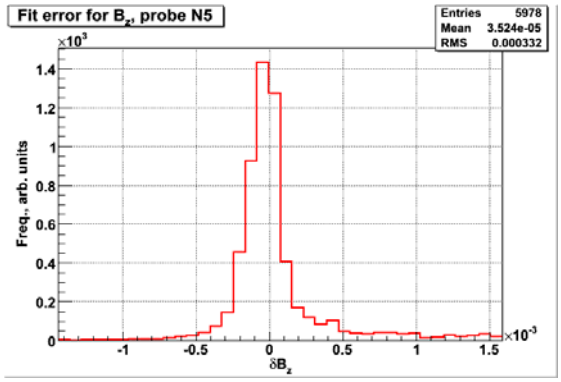


Fig. 9.90 δB_z **spread for the N5 probe at 4 T**

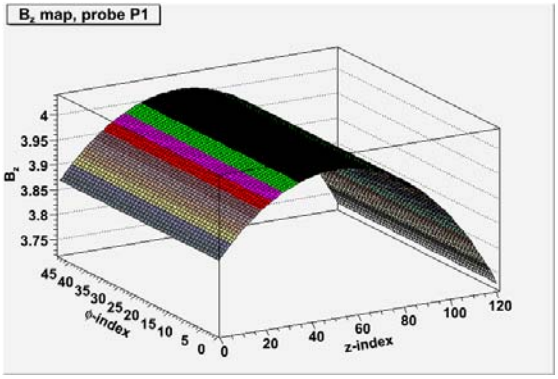


Fig. 9.91 B_z **map for the P1 probe at 4 T**

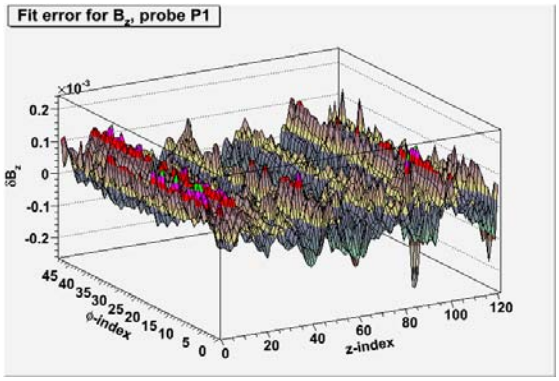


Fig. 9.92 B_z **residuals for the P1 probe at 4 T**

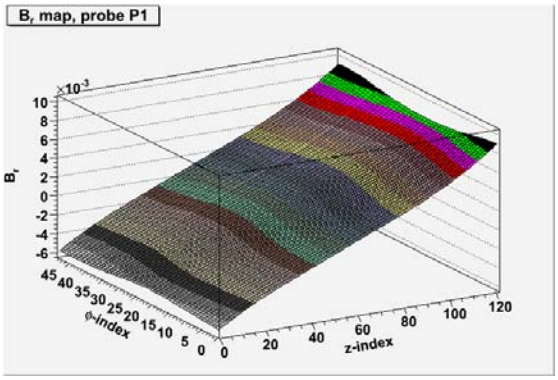


Fig. 9.93 B_r **map for the P1 probe at 4 T**

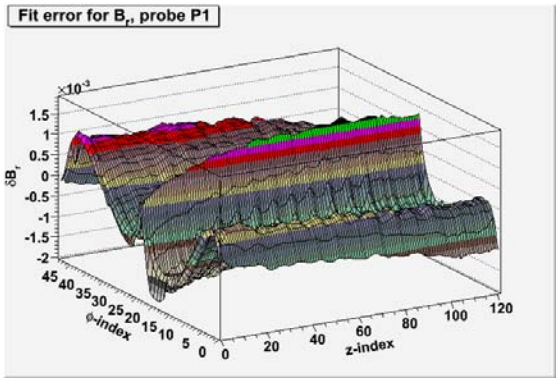


Fig. 9.94 B_r **residuals for the P1 probe at 4 T**

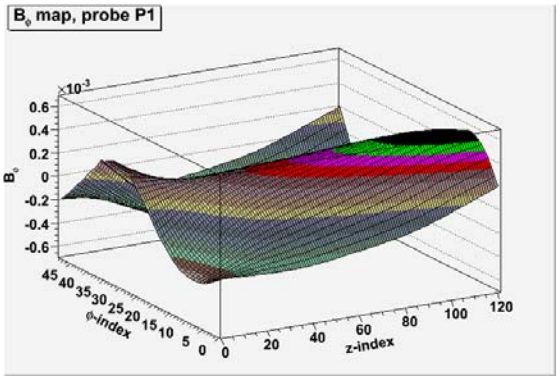


Fig. 9.95 B_ϕ **map for the P1 probe at 4 T**

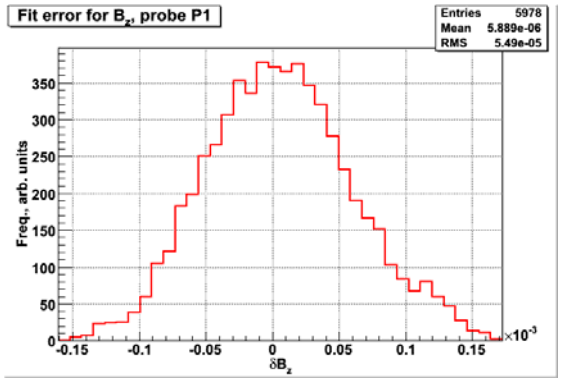


Fig. 9.96 δB_z **spread for the P1 probe at 4 T**

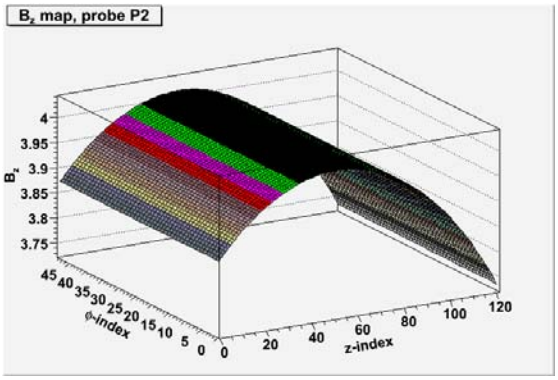


Fig. 9.97 B_z **map for the P2 probe at 4 T**

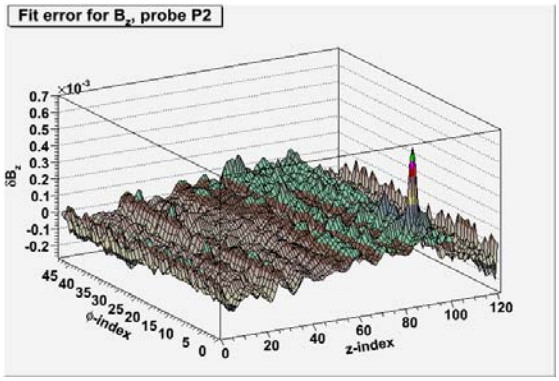


Fig. 9.98 B_z **residuals for the P2 probe at 4 T**

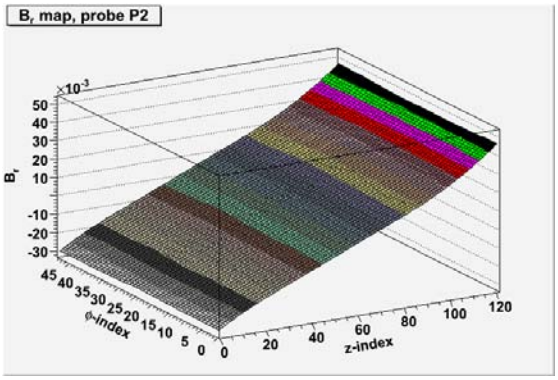


Fig. 9.99 B_r **map for the P2 probe at 4 T**

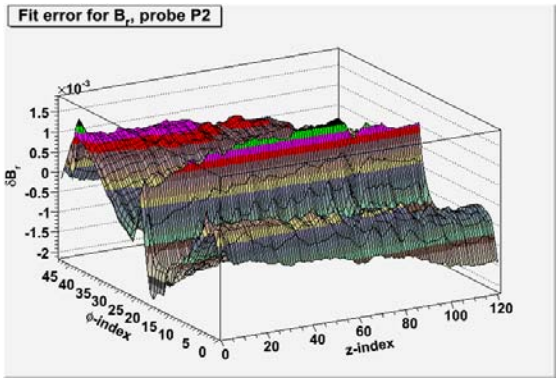


Fig. 9.100 B_r **residuals for the P2 probe at 4 T**

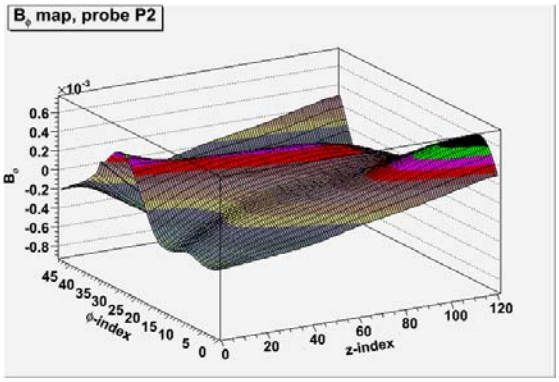


Fig. 9.101 B_ϕ **map for the P2 probe at 4 T**

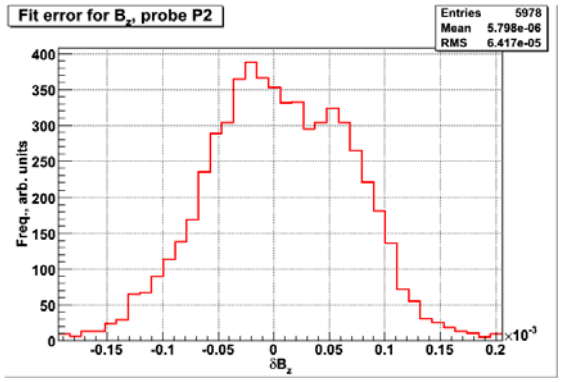


Fig. 9.102 δB_z **spread for the P2 probe at 4 T**

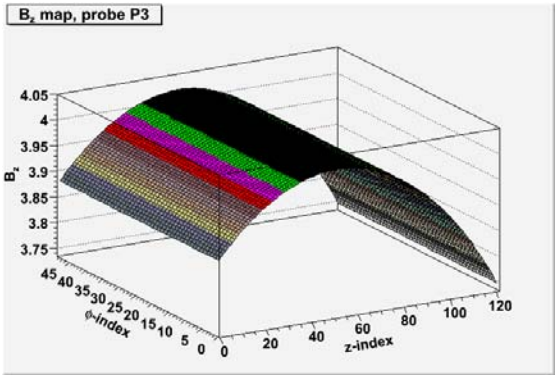


Fig. 9.103 B_z **map for the P3 probe at 4 T**

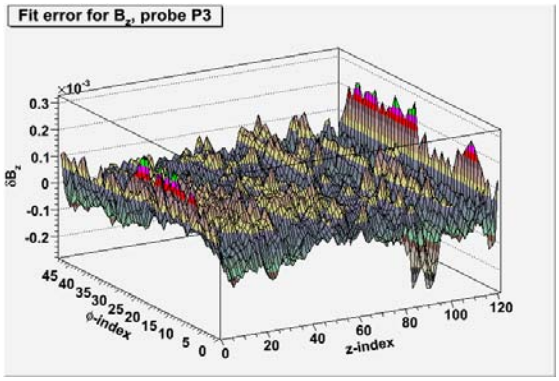


Fig. 9.104 B_z **residuals for the P3 probe at 4 T**

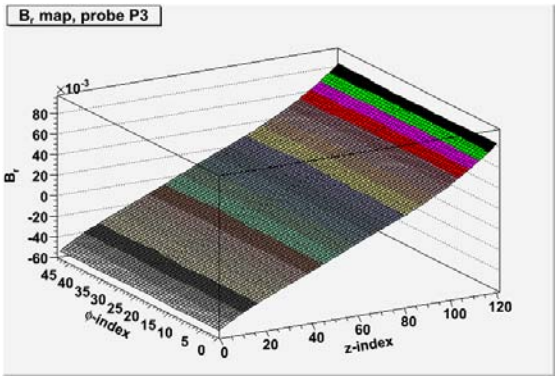


Fig. 9.105 B_r **map for the P3 probe at 4 T**

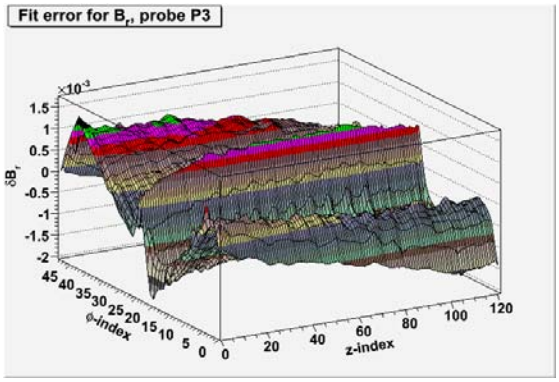


Fig. 9.106 B_r **residuals for the P3 probe at 4 T**

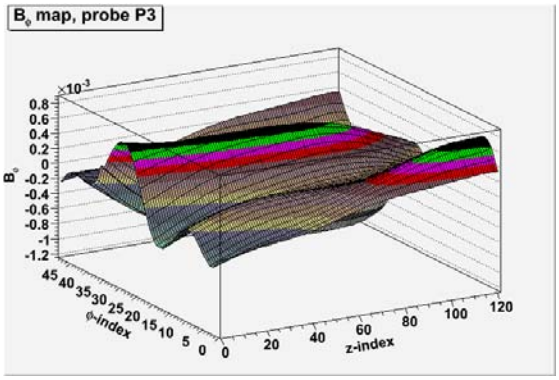


Fig. 9.107 B_ϕ **map for the P3 probe at 4 T**

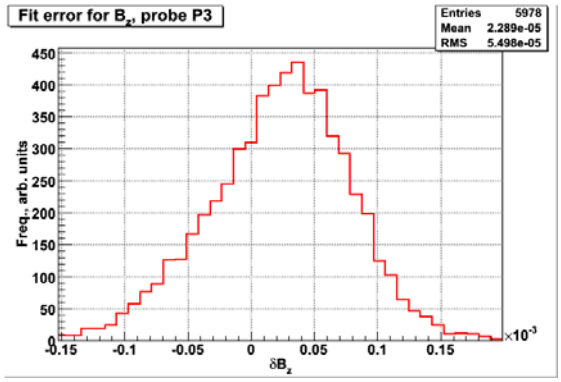


Fig. 9.108 B_ϕ **residuals for the P3 probe at 4 T**

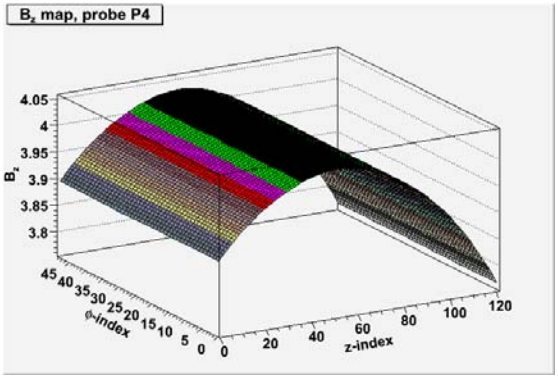


Fig. 9.109 B_z map for the P4 probe at 4 T

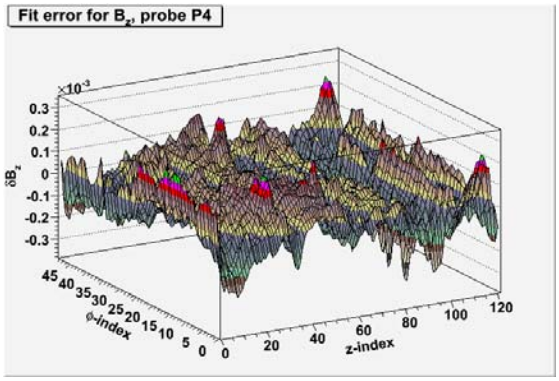


Fig. 9.110 B_z residuals for the P4 probe at 4 T

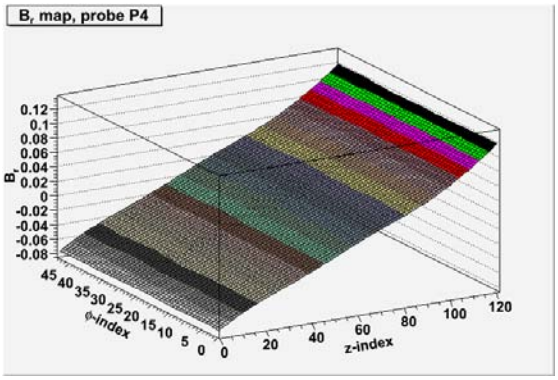


Fig. 9.111 B_r map for the P4 probe at 4 T

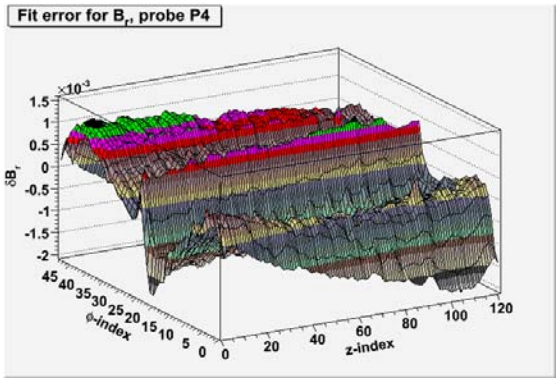


Fig. 9.112 B_r residuals for the P4 probe at 4 T

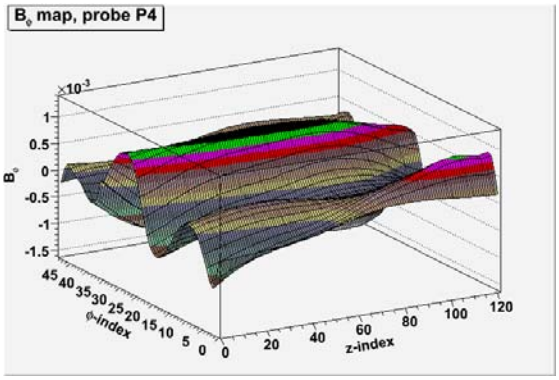


Fig. 9.113 B_ϕ map for the P4 probe at 4 T

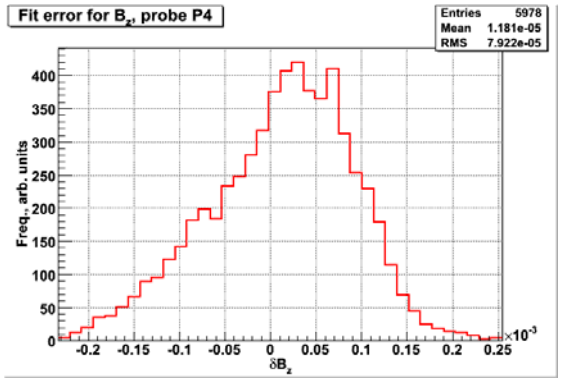


Fig. 9.114 δB_z spread for the P4 probe at 4 T

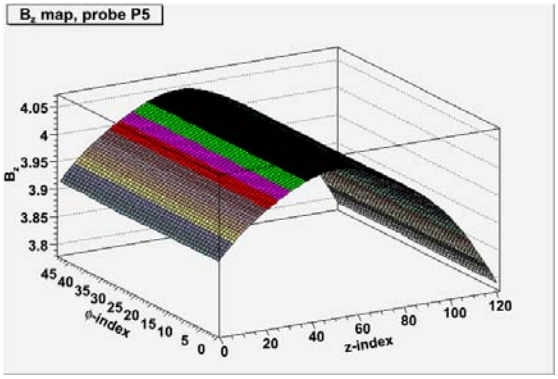


Fig. 9.115 B_z **map for the P5 probe at 4 T**

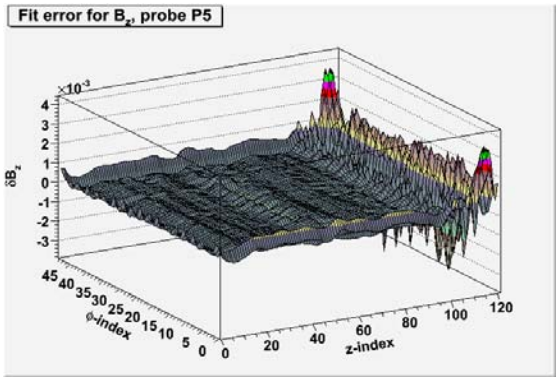


Fig. 9.116 B_z **residuals for the P5 probe at 4 T**

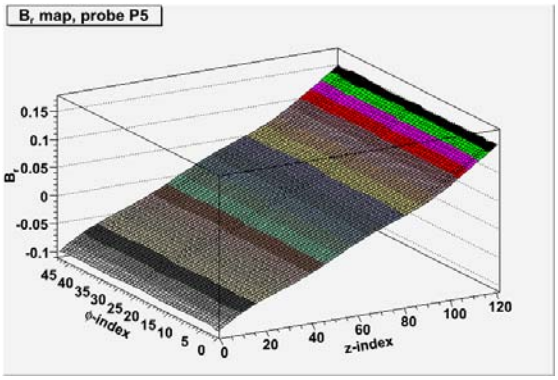


Fig. 9.117 B_r **map for the P5 probe at 4 T**

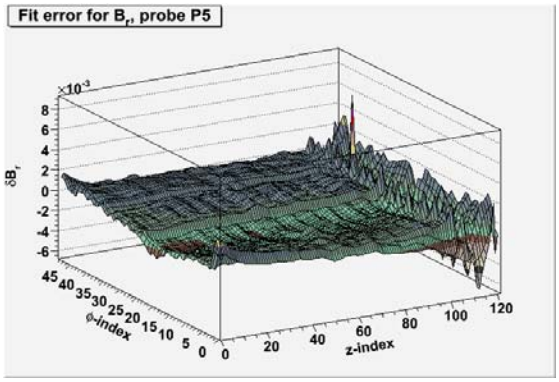


Fig. 9.118 B_r **residuals for the P5 probe at 4 T**

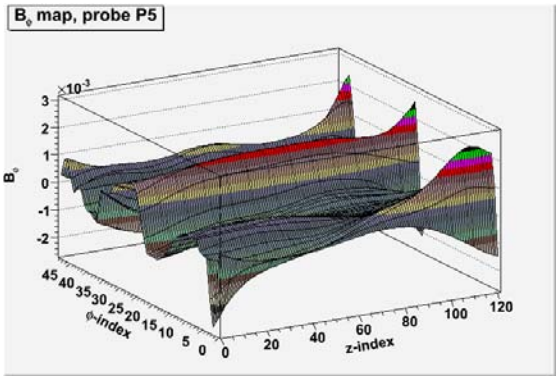


Fig. 9.119 B_ϕ **map for the P5 probe at 4 T**

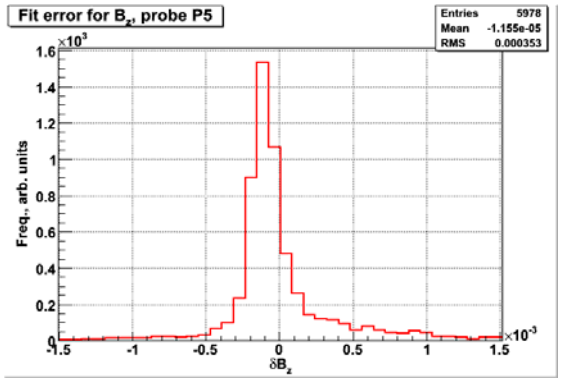


Fig. 9.120 δB_z **spread for the P5 probe at 4 T**

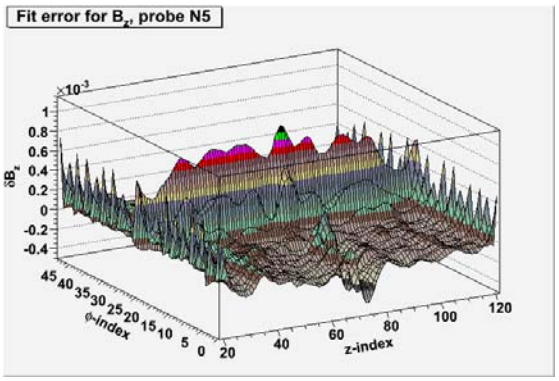


Fig. 9.121 B_z **residuals for the N5 probe at 2 T, the outermost** z**-positions are excluded**

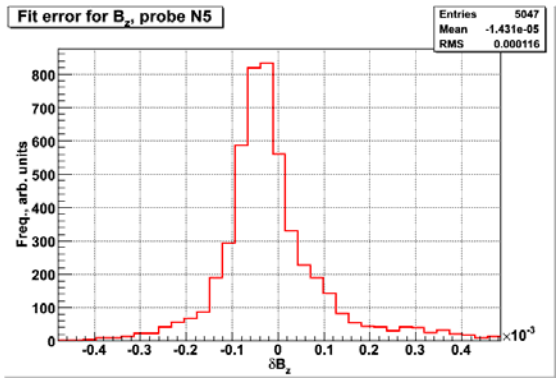


Fig. 9.122 δB_z **spread for the N5 probe at 2 T, the outermost** z**-positions are excluded**

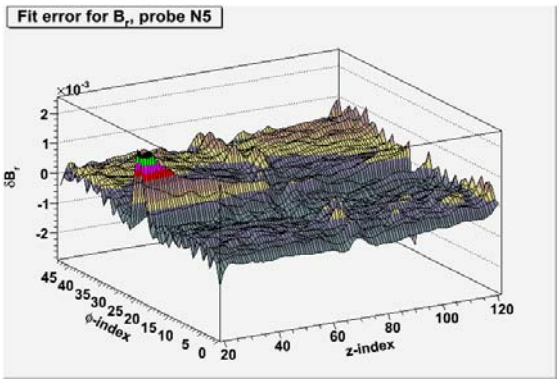


Fig. 9.123 B_r **residuals for the N5 probe at 2 T, the outermost** z**-positions are excluded**

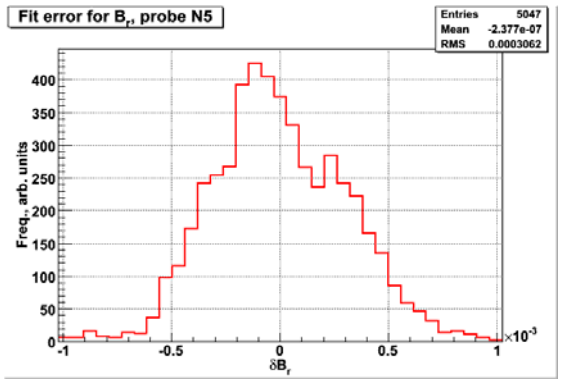


Fig. 9.124 δB_r **spread for the N5 probe at 2 T, the outermost** z**-positions are excluded**

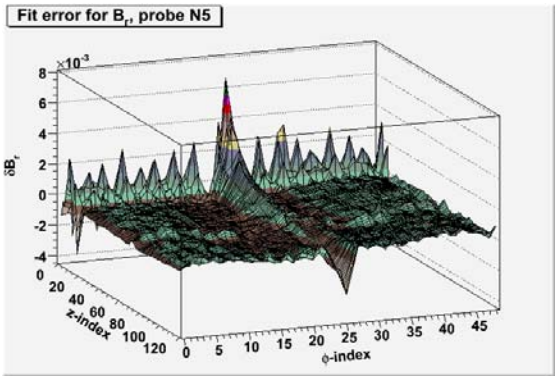


Fig. 9.125 The "ridge-to-canyon" structure of the B_r **residuals around** ϕ**–index = 24 for the N5 probe at 2 T**

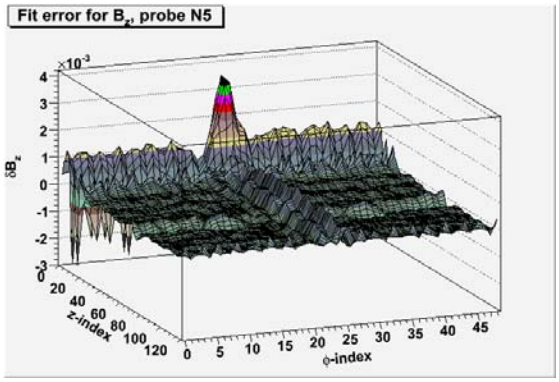


Fig. 9.126 The "double ridge" structure of the B_z **residuals around** ϕ**–index = 24 for the N5 probe at 2 T**

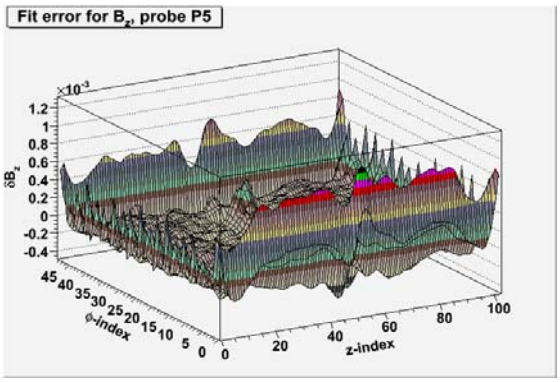


Fig. 9.127 B_z **residuals for the P5 probe at 2 T, the outermost** z**-positions are excluded**

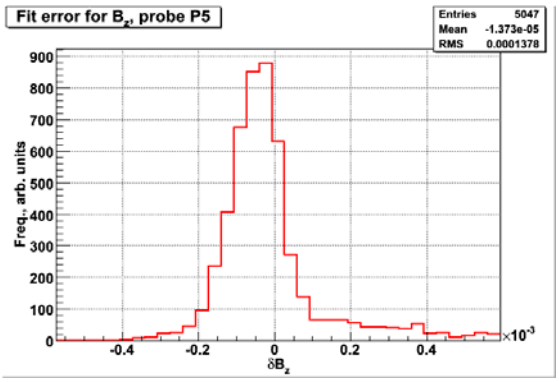


Fig. 9.128 δB_z **spread for the P5 probe at 2 T, the outermost** z**-positions are excluded**

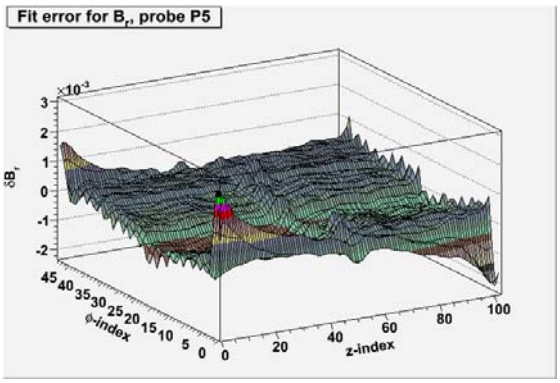


Fig. 9.129 B_r **residuals the P5 probe at 2 T, the outermost** z**-positions are excluded**

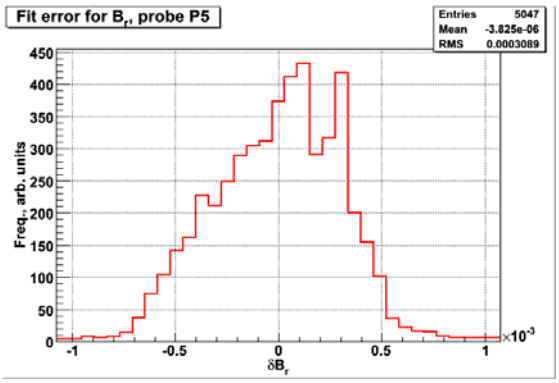


Fig. 9.130 δB_r **spread for the P5 probe at 2 T, the outermost** z**-positions are excluded**

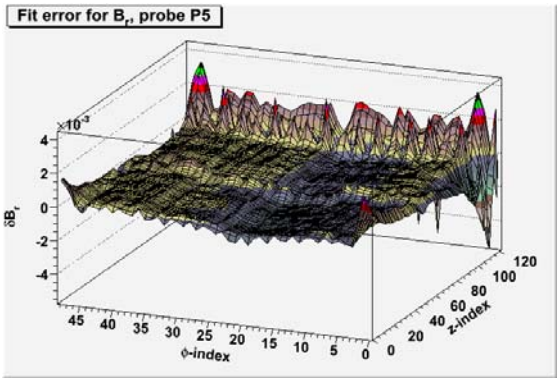


Fig. 9.131 The "ridge-to-canyon" structure of the B_r **residuals around** ϕ**–index = 0 for the P5 probe at 2 T**

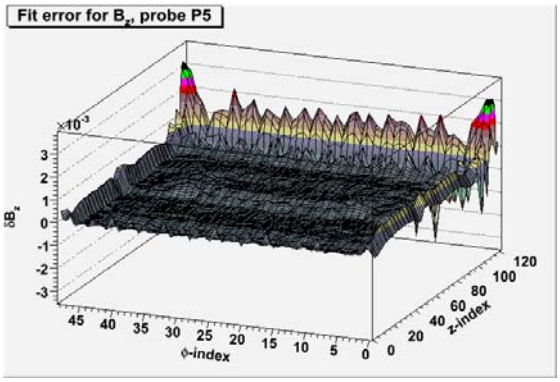


Fig. 9.132 The "double ridge" structure of the B_z **residuals around** ϕ**–index = 0 for the P5 probe at 2 T**

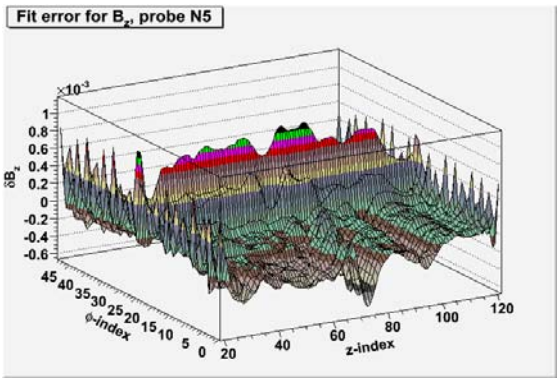


Fig. 9.133 B_z **residuals for the N5 probe at 4 T, the outermost** z**-positions are excluded**

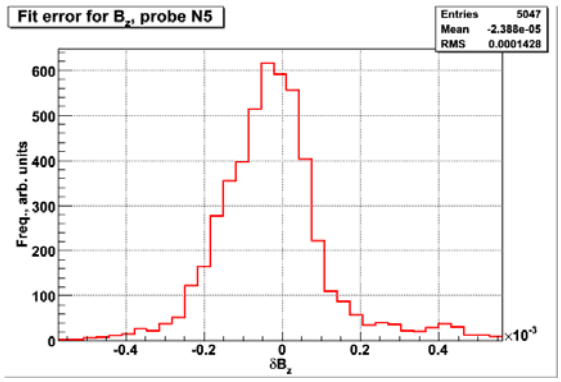


Fig. 9.134 δB_z **spread for the N5 probe at 4 T, the outermost** z**-positions are excluded**

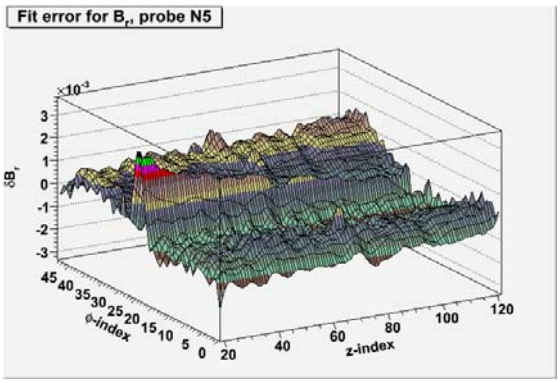


Fig. 9.135 B_r **residuals for the N5 probe at 4 T, the outermost** z**-positions are excluded**

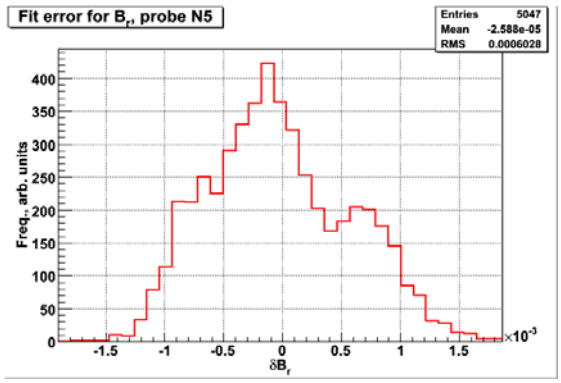


Fig. 9.136 δB_r **spread for the N5 probe at 4 T, the outermost** z**-positions are excluded**

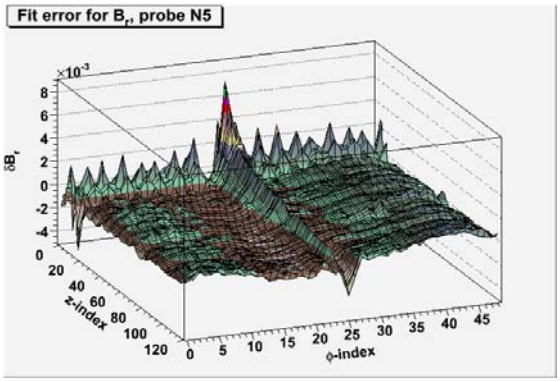


Fig. 9.137 The "ridge-to-canyon" structure of the B_r **residuals around** ϕ**–index = 24 for the N5 probe at 4 T**

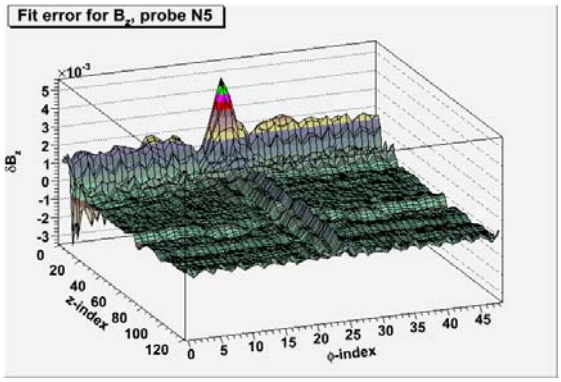


Fig. 9.138 The "double ridge" structure of the B_z **residuals around** ϕ**–index = 24 for the N5 probe at 4 T**

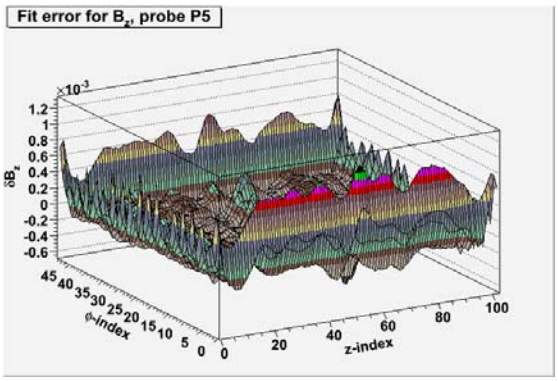


Fig. 9.139 B_z **residuals for the P5 probe at 4 T, the outermost** z**-positions are excluded**

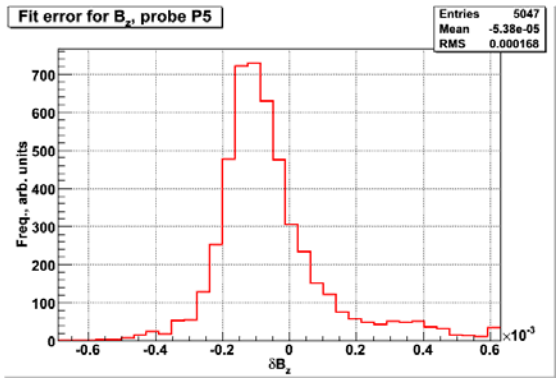


Fig. 9.140 δB_z **spread for the P5 probe at 4 T, the outermost** z**-positions are excluded**

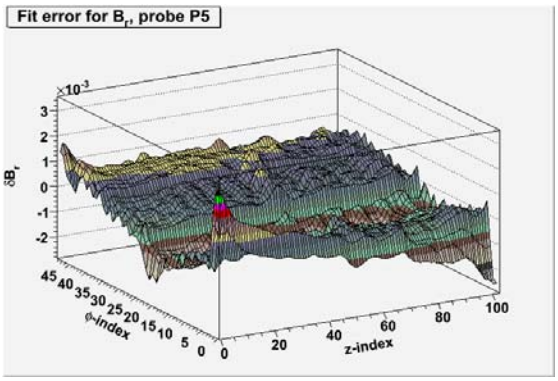


Fig. 9.141 B_r **residuals for the P5 probe at 4 T, the outermost** z**-positions are excluded**

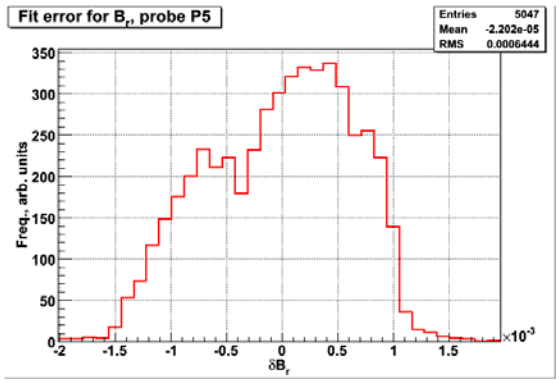


Fig. 9.142 δB_r **spread for the P5 probe at 4 T, the outermost** z**-positions are excluded**

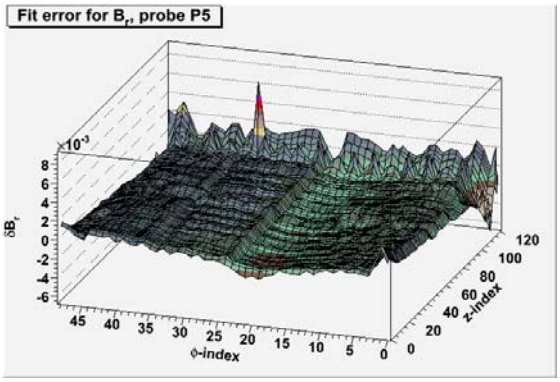


Fig. 9.143 The "ridge-to-canyon" structure of the B_r **residuals around** ϕ**–index = 0 for the P5 probe at 4 T**

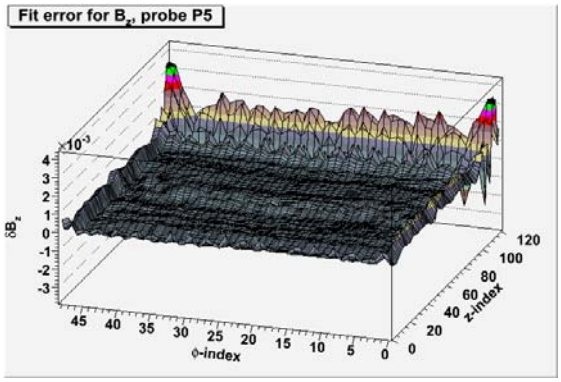


Fig. 9.144 The "double ridge" structure of the B_z **residuals around** ϕ**–index = 0 for the P5 probe at 4 T**

LIST OF REFERENCES

LIST OF REFERENCES

[1] CERN. LHC Project Home Page. http://lhc.web.cern.ch/lhc/ (accessed March 2008).

[2] CERN. LHC Design Report. http://ab-div.web.cern.ch/ab-div/Publications/LHC-DesignReport.html (accessed March 2008).

[3] CERN. LEP: the Z Factory. http://public.web.cern.ch/public/en/Research/LEP-en.html (accessed March 2008).

[4] CERN User's pages. http://user.web.cern.ch/user/Welcome.asp (accessed March 2008).

[5] The ATLAS Experiment. http://atlas.ch/ (accessed March 2008).

[6] The CMS Collaboration. The Compact Muon Solenoid Experiment. http://cms.cern.ch/ (accessed March 2008).

[7] The CMS Collaboration. CMS Physics Design Report, vol. I: Detector Performance and Software. http://doc.cern.ch//archive/electronic/cern/preprints/lhcc/public/lhcc-2006-001.pdf (accessed March 2008).

[8] The CMS Collaboration. CMS Physics Design Report, vol. II: Physics Performance. http://doc.cern.ch/archive/electronic/cern/preprints/lhcc/public/lhcc-2006-021.pdf (accessed March 2008).

[9] The CMS Collaboration. CMS Outreach – The CMS Tracker. http://cmsinfo.cern.ch/outreach/CMSdetectorInfo/CMStracker.html (accessed March 2008).

[10] The CMS Collaboration. CMS Outreach – The CMS Electromagnetic Calorimeter. http://cmsinfo.cern.ch/outreach/CMSdetectorInfo/CMSecal.html (accessed March 2008).

[11] The CMS Collaboration. CMS Outreach – The CMS Hadronic Calorimeter. http://cmsinfo.cern.ch/outreach/CMSdetectorInfo/CMShcal.html (accessed March 2008).

[12] The CMS Collaboration. CMS Outreach – the CMS Superconducting Solenoidal Magnet. http://cmsinfo.cern.ch/outreach/CMSdetectorInfo/CMSmagnet.html (accessed March 2008).

[13] The CMS Collaboration. CMS Outreach – the CMS Muon Detector System. http://cmsinfo.cern.ch/outreach/CMSdetectorInfo/CMSmuon.html (accessed March 2008).

[14] Klyukhin, V. I, Magnetic Field Mapping at CMS. http://edms.cern.ch/file/879493/1/CMS_Fieldmapping_071009.pdf (accessed March 2008).

[15] Maxwell, James Clerk. *A Dynamical Theory of the Electromagnetic Field.* Scottish Academic Press, Edinburg EH7 5JX, UK, 1982[4].

[16] Jackson, D. J. Classical Electrodynamics, 3rd edition. John Wiley & Sons, New York, USA, 1998.

[17] Gauss, C. F. "Theoria combinationis obsevationum erroribus minimis obnoxiae." Werke, Vol. 4. Göttingen, Germany, 1823.

[18] Bock, R. K. and Krischer, W. *The Data Analysis BriefBook (Internet Version).* http://rkb.home.cern.ch/rkb/AN16pp/node189.html (accessed March 2008).

[19] The Minuit Home Page. http://www.cern.ch/minuit (accessed March 2008).

[20] The ROOT System Home Page. http://root.cern.ch/ (accessed March 2008).

[21] CMS Software Page. http://cmsdoc.cern.ch/cms/cpt/Software/html/General/ (accessed March 2008).

[22] Laplace, Pierre-Simon. "Théorie des Attractions des Sphéroïdes et de la Figure des Planètes.", *Mémoires de l'Académie des Sciences*, 1782[5].

[23] Weisstein, Eric W. "Laplace's Equation." *MathWorld -- A Wolfram Web Resource*. http://mathworld.wolfram.com/LaplacesEquation.html (accessed March 2008).

[24] Legendre, A. M. "Sur l'attraction des Sphéroïdes." *Mém. Math. et Phys. présentés à l'Ac. r. des. sc. par divers savants* 10, 1785.

[25] Byerly, W. E. *An Elementary Treatise on Fourier's Series and Spherical, Cylindrical, and Ellipsoidal Harmonics, with Applications to Problems in Mathematical Physics*. Ginn & Company, Publishers, Boston, USA, 1895[6].

[26] Weisstein, Eric W. "Legendre Polynomial." *MathWorld--A Wolfram Web Resource*. http://mathworld.wolfram.com/LegendrePolynomial.html (accessed March 2008).

[27] Landau, L. D. and Lifshitz, E. M. *Quantum Mechanics, Non-Relativistic Theory, 3rd edition*. Translated from the Russian by Sykes, J. B and Bell, J. S. Elsevier Butterworth-Heinemann, Oxford, UK, 2003.

[28] Sansone, G. *Orthogonal functions*. Translated from the Italian by Diamond, A. H. Interscience Publishers, 1959.

[29] Masaru Takeuchi. "Modern Spherical Functions". Translated from the Japanese by Toshinobu Nagura. *Translations of Mathematical Monographs*, Vol. 135. American Mathematical Society, 1994.

---

[4] Reprint of the original Maxwell's essay, that was presented to the Royal Society in London on 27th October, 1864, and published in volume CLV of *Philosophical Transactions* in the following year.

[5] This article, although bearing an earlier date than that of Legendre, was really inspired by it. It is here that "Laplace's equation" first appears, occurring, however, only in polar coordinates.

[6] This book, published in 19th century covers subjects listed in its' title in such a clear and comprehensive way that it still can be recommended as one of the best handbooks for students and engineers of 21st century.

APPENDIX

APPENDIX

The C++ *HarmBasis3DCyl* object is derived from two other objects in sequence: *rz_harm_poly* and *poly2d_base* . Here we present the code for all of them, from the bottom (*poly2d_base*) to the top (*HarmBasis3DCyl*).

poly2d_base

File: Poly2dbase.h

```
#ifndef poly2d_base_h
#define poly2d_base_h

#include <cmath>
#include <iostream>
#include <fstream>
#include <iomanip>
#include <vector>
#include <set>

#include <float.h> //in order to use DBL_EPSILON (1+DBL_EPSILON > 1)

using namespace std;

////////////////////////////////////////////////////////////////////////////////
//                                                                            //
//  The "poly2d_term" represent a term of a polynomial of 2 variables.        //
//                                                                            //
////////////////////////////////////////////////////////////////////////////////

//____________________________________________________________________________
struct poly2d_term {
   double   coeff;  //Coefficient of the term
   unsigned np[2];  //Powers of variables

   poly2d_term() {memset(this, 0, sizeof(*this));}
   poly2d_term(double C, unsigned nr, unsigned nz)
   {
      coeff = C; np[0] = nr; np[1] = nz;
   }
   void Print(ostream &out = cout, bool first_term = true);
};

////////////////////////////////////////////////////////////////////////////////
//                                                                            //
//  Base class that represent a polynomial of 2 variables. It isn't supposed  //
//  to be used directly and provides no way of setting coefficients directly. //
//  Such methods must be defined in derived classes.                          //
//                                                                            //
////////////////////////////////////////////////////////////////////////////////
```

```
//_____________________________________________________________________________
class poly2d_base {   // a general polynomial of 2 variables

protected:
   //Group of static members for the class memory management
   //-----------------------------------------------------------------------------
   static double     rval;   //last r-value used in calculation
   static double     zval;   //last z-value used in calculation

   static double   **rz_pow; //table with calculated r^n*z^m values
   static unsigned   NTab;   //rz_pow table size
   static unsigned   NPwr;   //max power in use by CLASS
   static bool       rz_set;

   static const double MIN_COEFF; //Threshold for assigning a coeff. to 0

   static set<poly2d_base*> poly2d_base_set;  //Set of all poly2d_base objects

   static void SetTabSize(const unsigned N); //Set rz-table size
   static void FillTable (const double r, const double z);

   static void AdjustTab();
   //-----------------------------------------------------------------------------

   vector<poly2d_term> data; //polynomial terms
   unsigned max_pwr;         //max power in use by INSTANCE

public:
   static void     IncNPwr(const unsigned N) {if (N > NPwr) NPwr = N;}
   static int      GetMaxPow();
   static unsigned Count() { return poly2d_base_set.size();}
   static void     PrintTab(ostream &out = cout);

   static void SetPoint(const double r, const double z);

   poly2d_base() {
      max_pwr = 0;
      poly2d_base_set.insert(this);
   }
   poly2d_base(const poly2d_base &S) {
      data    = S.data;
      max_pwr = S.max_pwr;
      poly2d_base_set.insert(this);
   }

   virtual ~poly2d_base();

   bool IsOn()    { return bool(data.size());}
   bool IsRZSet() { return rz_set;}

   void Collect(); //Collect terms and remove zero terms
   void Compress() { Collect();}

   void Diff  (int nvar); //differentiate the polynomial by variable# nvar
   void Int   (int nvar); //Integrate the polynomial by variable# nvar
   void IncPow(int nvar); //Multiply the polynomial by variable# nvar
   void DecPow(int nvar); //Divide the polynomial by variable# nvar

   void Scale(const double C);
//   poly2d_base& operator*=(const double C) { Scale(C); return *this;}

   double Eval(); //Evaluation with no check that rz_pow table exist
   double GetVal() {if (rz_set) return Eval(); else return 0.;}
   double GetVal(const double r, const double z) { SetPoint(r,z); return Eval();}

   void Print(ostream &out = cout);

}; //Class poly2d_base
#endif
```

File: Poly2dbase.cxx

```
#include "poly2d_base.h"

////////////////////////////////////////////////////////////////////////////////
//                                                                            //
//  The "poly2d_term" represent a term of a polynomial of 2 variables.        //
//                                                                            //
////////////////////////////////////////////////////////////////////////////////

//____________________________________________________________________________
void poly2d_term::Print(ostream &out, bool first_term)
{
   if (first_term) out << coeff;
   else if (coeff > 0.) out << " + " << coeff;
        else out << " - " << -coeff;
   if (np[0] != 0) {
      out << "*r";
      if (np[0] != 1) out << "^" << np[0];
   }
   if (np[1] != 0) {
      out << "*z";
      if (np[1] != 1) out << "^" << np[1];
   }
}

////////////////////////////////////////////////////////////////////////////////
//                                                                            //
//  Base class that represent a polynomial of 2 variables. It isn't supposed  //
//  to be used directly and provides no way of setting coefficients directly. //
//  Such methods must be defined in derived classes.                          //
//                                                                            //
////////////////////////////////////////////////////////////////////////////////

//____________________________________________________________________________
double     poly2d_base::rval   = 0.; //Last values of r and z used
double     poly2d_base::zval   = 0.;

double   **poly2d_base::rz_pow = 0;  //table with calculated r^n*z^m values
unsigned   poly2d_base::NTab   = 0;  //rz_pow table size
unsigned   poly2d_base::NPwr   = 0;  //max power in use by CLASS

bool       poly2d_base::rz_set = false;

const double poly2d_base::MIN_COEFF = DBL_EPSILON; //Thresh. for coeff == 0
set<poly2d_base*> poly2d_base::poly2d_base_set;

//____________________________________________________________________________
poly2d_base::~poly2d_base()
{
   poly2d_base_set.erase(poly2d_base_set.find(this));
   if(poly2d_base_set.size()) { //some objects left
      if (max_pwr >= NPwr) NPwr = GetMaxPow();
   } else {
      if (rz_pow) {
         delete [] rz_pow[0]; //deleting the last instance -> memory cleanup
         delete [] rz_pow;
      }
      rz_pow = 0;
      rval = zval = 0.;
      NPwr = 0;
      NTab = 0;
      rz_set = false;
//      poly2d_base_set.resize(0);
   }
}
```

```
//_____________________________________________________________________
void poly2d_base::SetTabSize(const unsigned N)
{
   if (N <= NTab) return;
   if (rz_pow) {
      delete [] rz_pow[0];
      delete [] rz_pow;
   }
   rz_pow    = new double* [N];
   unsigned jr, dN = N*(N+1)/2;
   rz_pow[0] = new double  [dN];
   memset(rz_pow[0], 0, dN*sizeof(double));
   rz_pow[0][0] = 1.;
   for (jr = 1, dN = N; jr < N; ++jr, --dN) {
      rz_pow[jr] = rz_pow[jr-1] + dN;
   }
   rval = zval = 0.;
   NTab = N;
}

//_____________________________________________________________________
void poly2d_base::FillTable(const double r, const double z)
{
   if (!rz_pow) return;
   unsigned jr, jz;
   for (jz = 1; jz <= NPwr; ++jz) rz_pow[0][jz] = z*rz_pow[0][jz-1];
   for (jr = 1; jr <= NPwr; ++jr) {
      for (jz = 0; jz <= (NPwr - jr); ++jz) {
         rz_pow[jr][jz] = r*rz_pow[jr-1][jz];
      }
   }
}

//_____________________________________________________________________
int poly2d_base::GetMaxPow()
{
   int curp, maxp = 0;
   set<poly2d_base*>::iterator it;

   for (it = poly2d_base_set.begin(); it != poly2d_base_set.end(); ++it) {
      curp = (*it)->max_pwr;
      if (curp > maxp) maxp = curp;
   }
   return maxp;
}

//_____________________________________________________________________
void poly2d_base::AdjustTab()
{
   NPwr = GetMaxPow();
   if (NPwr >= NTab) SetTabSize(NPwr+1);
}

//_____________________________________________________________________
void poly2d_base::PrintTab(ostream &out)
{
   out << "poly2d_base table size NTab = " << NTab
       << "\tmax. power NPwr = " << NPwr << endl;
   if (rz_pow) {
      if (NPwr < NTab) {
         unsigned jr, jz, old_prec = out.precision();
         out.precision(5);
         out << "Table content:" << endl;
         for (jr = 0; jr <= NPwr; ++jr) {
            for (jz = 0; jz <= (NPwr-jr); ++jz) {
               out << setw(12) << left << rz_pow[jr][jz];
            }
            out << "|" << endl;
         }
```

```
         out.precision(old_prec);
      } else {
         out << "\tTable size is not adjusted." << endl;
      }
   } else {
      out << "\tTable is not allocated." << endl;
   }
}

//_____________________________________________________________________
void poly2d_base::SetPoint(const double r, const double z)
{
   if (!Count()) return;
   if (NPwr >= NTab) { SetTabSize(NPwr+1); FillTable(r, z);}
   else if ((r != rval) || (z != zval)) FillTable(r, z);
   rz_set = true;
}

//_____________________________________________________________________
double poly2d_base::Eval()
{
   double S = 0.;
   for (unsigned j = 0; j < data.size(); ++j)
      S += data[j].coeff*rz_pow[data[j].np[0]][data[j].np[1]];
   return S;
}

//_____________________________________________________________________
void poly2d_base::Collect()
{
   if (!(data.size())) return;

   unsigned j1, j2, rpow, zpow, noff = 0, jend = data.size();
   double C;
   vector<bool> mask(jend, false);
   max_pwr = 0;

   for (j1 = 0; j1 < jend; ++j1) {
      if (mask[j1]) continue;
      C = data[j1].coeff;
      rpow = data[j1].np[0];
      zpow = data[j1].np[1];
      for (j2 = j1+1; j2 < jend; ++j2) {
         if (mask[j2]) continue;
         if ((rpow == data[j2].np[0]) && (zpow == data[j2].np[1])) {
            C += data[j2].coeff;
            mask[j2] = true;
            ++noff;
         }
      }
      if (fabs(C) > MIN_COEFF) {
         data[j1].coeff = C;
         if ((rpow = rpow+zpow) > max_pwr) max_pwr = rpow;
      } else {
         mask[j1] = true;
         ++noff;
      }
   }
   vector<poly2d_term> newdata; newdata.reserve(jend - noff);
   for (j1 = 0; j1 < jend; ++j1) {
      if (!(mask[j1])) newdata.push_back(data[j1]);
   }
   data.swap(newdata);
}
```

```
//_____________________________________________________________________
void poly2d_base::Print(ostream &out)
{
   if (!data.size()) {
      out << "this polynomial object contains no terms." << endl;
      return;
   }
   out << data.size() << " terms; max. degree = " << max_pwr << ":" << endl;
   unsigned old_prec = out.precision();
   out.precision(5);
   data[0].Print(out);
   for (unsigned it = 1; it < data.size(); ++it) {
         data[it].Print(out, false);
   }
   out << endl;
   out.precision(old_prec);
}

//_____________________________________________________________________
void poly2d_base::Diff(int nvar)
{
//differentiate the polynomial by variable nvar.
//
   poly2d_term  v3;
   vector<poly2d_term> newdata;
   newdata.reserve(data.size());
   unsigned cur_pwr = 0, maxp = 0, oldp = max_pwr;
   for (unsigned it = 0; it < data.size(); ++it) {
      v3 = data[it];
      v3.coeff *= v3.np[nvar];
      if (v3.coeff != 0.) {
         --v3.np[nvar];
         newdata.push_back(v3);
         if ((cur_pwr = v3.np[0] + v3.np[1]) > maxp) maxp = cur_pwr;
      }
   }
   newdata.resize(newdata.size());
   max_pwr = maxp;
   data.swap(newdata);
   if (oldp >= NPwr) NPwr = GetMaxPow();
}

//_____________________________________________________________________
void poly2d_base::Int(int nvar)
{
//Integrate the polynomial by variable# nvar. Doesn't remove terms
//with zero coefficients; if you suspect they can appear, use Compress()
//after the integration.
//
   for (unsigned it = 0; it < data.size(); ++it) {
      data[it].coeff /= ++data[it].np[nvar];
   }
   ++max_pwr;
   if (max_pwr > NPwr) NPwr = GetMaxPow();
}

//_____________________________________________________________________
void poly2d_base::IncPow(int nvar)
{
//Multiply the polynomial by variable# nvar
//
   for (unsigned it = 0; it < data.size(); ++it) {
      ++data[it].np[nvar];
   }
   ++max_pwr;
   if (max_pwr > NPwr) NPwr = GetMaxPow();
}
```

```
//_____________________________________________________________________
void poly2d_base::DecPow(int nvar)
{
//Divide the polynomial by variable# nvar. Remove terms with zero coefficients
//and also terms where the initial power of nvar is equal zero
//
   poly2d_term  v3;
   vector<poly2d_term> newdata;
   newdata.reserve(data.size());
   unsigned cur_pwr = 0, maxp = 0, oldp = max_pwr;
   for (unsigned it = 0; it < data.size(); ++it) {
      v3 = data[it];
      if ((v3.coeff != 0.) && (v3.np[nvar] > 0)) {
         --v3.np[nvar];
         newdata.push_back(v3);
         if ((cur_pwr = v3.np[0] + v3.np[1]) > maxp) maxp = cur_pwr;
      }
   }
   newdata.resize(newdata.size());
   max_pwr = maxp;
   data.swap(newdata);
   if (oldp >= NPwr) NPwr = GetMaxPow();
}

//_____________________________________________________________________
void poly2d_base::Scale(const double C)
{
//Multiply the polynomial by a constant.
//
   if (C != 0.) {
      for (unsigned it = 0; it < data.size(); ++it) {
         data[it].coeff *= C;
      }
   } else data.resize(0);
}
```

rz_harm_poly

File: rz_harm_poly.h

```
#ifndef rz_harm_poly_h
#define rz_harm_poly_h

#include "poly2d_base.h"

//////////////////////////////////////////////////////////////////////////////
//                                                                          //
//  Pair (Cos(phi),Sin(Phi)). Intended for internal use by rz_harm_poly.    //
//                                                                          //
//////////////////////////////////////////////////////////////////////////////

//_____________________________________________________________________
struct trig_pair {
   double CosPhi;
   double SinPhi;

   trig_pair() : CosPhi(1.), SinPhi(0.) {}
   trig_pair(const trig_pair &tp) : CosPhi(tp.CosPhi), SinPhi(tp.SinPhi) {}
   trig_pair(const double C, const double S) : CosPhi(C), SinPhi(S) {}
   trig_pair(const double phi) : CosPhi(cos(phi)), SinPhi(sin(phi)) {}

   //Return trig_pair fo angle increased by angle of tp.
   trig_pair Add(const trig_pair &tp) {
      return trig_pair(this->CosPhi*tp.CosPhi - this->SinPhi*tp.SinPhi,
                       this->SinPhi*tp.CosPhi + this->CosPhi*tp.SinPhi);
   }
```

```
};

////////////////////////////////////////////////////////////////////////////////
//                                                                            //
//  Harmonic homogeneous polynomial in cylindrical system.                    //
//                                                                            //
////////////////////////////////////////////////////////////////////////////////

//____________________________________________________________________________
class rz_harm_poly : public poly2d_base {

private:
   unsigned L;
   int      M;

   static unsigned   Cnt;      //Number of the "rz_harm_poly" objects
   static double     phival;   //Last phi value used
   static bool       phi_set;  //TRUE if phi value is set
   static unsigned   MaxM;     //Max. M among "rz_harm_poly" objects

   static unsigned   TASize;   //TrigArr size
   static trig_pair *TrigArr;  //Array with angular data

   static void SetTrigArrSize(const unsigned N);
   static void FillTrigArr   (const double phi);

public:

   static int      GetMaxM(); //return Max. M for the class
   static unsigned ParentCount() { return poly2d_base::Count();}
   static unsigned Count() { return Cnt;}
   static void     SetPhi(const double phi);
   static void     SetPoint(const double r, const double z, const double phi)
   {
      poly2d_base::SetPoint(r, z); SetPhi(phi);
   }

   rz_harm_poly() : poly2d_base(), L(0), M(0) {++Cnt;}
   rz_harm_poly(const poly2d_base &S) : poly2d_base(S), L(0), M(0) {++Cnt;}
   rz_harm_poly(const rz_harm_poly &S) : poly2d_base(S), L(S.L), M(S.M) {++Cnt;}
   rz_harm_poly(const unsigned N);
   ~rz_harm_poly();

   bool IsPhiSet() { return phi_set;}

   rz_harm_poly GetDiff  (int nvar) { rz_harm_poly R(*this); R.Diff  (nvar); return R;}
   rz_harm_poly GetInt   (int nvar) { rz_harm_poly R(*this); R.Int   (nvar); return R;}
   rz_harm_poly GetIncPow(int nvar) { rz_harm_poly R(*this); R.IncPow(nvar); return R;}
   rz_harm_poly GetDecPow(int nvar) { rz_harm_poly R(*this); R.DecPow(nvar); return R;}

   rz_harm_poly LadderUp();
   rz_harm_poly LadderDwn();

   unsigned GetL() { return L;}
   int      GetM() { return M;}

   //Next functions return value of angular terms.
   //No check is made, wheither the TrigArr is initialized.
   //User can check if IsPhiSet() == true
   double GetCos() { return TrigArr[M].CosPhi;}
   double GetSin() { return TrigArr[M].SinPhi;}

   void CheatL(const unsigned newL) { L = newL;}
   void Print(ostream &out = cout);

   static void PrintTrigArr(ostream &out = cout);

}; //class rz_harm_poly
#endif
```

File: rz_harm_poly.cxx

```
#include <typeinfo>
#include "rz_harm_poly.h"

////////////////////////////////////////////////////////////////////////////////
//                                                                            //
//  Harmonic homogeneous polynomials in cylindrical system.                   //
//                                                                            //
////////////////////////////////////////////////////////////////////////////////

//____________________________________________________________________________
unsigned rz_harm_poly::Cnt    = 0;      //Number of the "rz_harm_poly" objects
double   rz_harm_poly::phival = 0.;     //Last phi value used
bool     rz_harm_poly::phi_set = false; //TRUE if phi value is set
unsigned rz_harm_poly::MaxM   = 0;      //Max. M among "rz_harm_poly" objects

unsigned   rz_harm_poly::TASize  = 0;   //TrigArr size
trig_pair *rz_harm_poly::TrigArr = 0;   //Array with angular data

//____________________________________________________________________________
rz_harm_poly::rz_harm_poly(const unsigned N)
{
//Constructor for rz_harm_poly of length N. The polynomial P(r,z) is normalized
//in such a way that dP/dz(r=0,z)=z^(N-1)
//
   unsigned nz = N, nr = 0, nv = 0;
   poly2d_term v3(1./N, nr, nz);

   data = vector<poly2d_term>((N + 2) / 2, v3);

   while (nz >= 2) {
      nz -= 2;
      nr += 2;
      nv += 1;
      data[nv].coeff = -data[nv-1].coeff*(nz+1)*(nz+2)/(nr*nr);
      data[nv].np[0] = nr;
      data[nv].np[1] = nz;
   }
   max_pwr = N;
   if (max_pwr > NPwr) {
      NPwr = max_pwr;
      rz_set = false;
      phi_set = false;
   }
   L = N;
   M = 0;
   poly2d_base_set.insert(this);
   ++Cnt;
}

//____________________________________________________________________________
rz_harm_poly::~rz_harm_poly()
{
   if (--Cnt) {
      if (abs(M) >= MaxM) { //a number of objects still left
         M = 0;
         MaxM = GetMaxM();
      }
   } else { //last instance -> memory cleanup
      if (TrigArr) delete [] TrigArr;
      TrigArr = 0;
      TASize = 0;
      MaxM = 0;
      phival = 0.;
      phi_set = false;
   }
}
```

```
//_____________________________________________________________________________
int rz_harm_poly::GetMaxM()
{
//Return max abs(M) for all rz_harm_poly objects created
//
   int M_cur, M_max = 0;
   set<poly2d_base*>::iterator it;
   for (it = poly2d_base_set.begin(); it != poly2d_base_set.end(); ++it) {
      if (typeid(**it) == typeid(rz_harm_poly)) {
         M_cur = abs(((rz_harm_poly*)(*it))->M);
         if (M_cur > M_max) M_max = M_cur;
      }
   }
   return M_max;
}

//_____________________________________________________________________________
void rz_harm_poly::SetPhi(const double phi)
{
//Set value of the angle argument, adjust the TrigArr size if neccessary
//and fill TrigArr if the phi value is changed
//
   if (MaxM >= TASize) { SetTrigArrSize(MaxM+1); FillTrigArr(phi);}
   else if (phi != phival) FillTrigArr(phi);
   phi_set = true;
}

//_____________________________________________________________________________
void rz_harm_poly::SetTrigArrSize(const unsigned N)
{
//Increase TrigArr size if neccessary
//
   if (N <= TASize) return;
   if (TrigArr) delete [] TrigArr;
   TrigArr = new trig_pair [N];
   (*TrigArr) = trig_pair(1., 0.);
   TASize = N;
   phi_set = false;
}

//_____________________________________________________________________________
void rz_harm_poly::FillTrigArr(const double phi)
{
//Fill TrigArr with trig_pair(jp*phi)
   if (!TrigArr) return;
   trig_pair tp(phi);
   TrigArr[1] = tp;
   for (unsigned jp = 2; jp <= MaxM; ++jp) TrigArr[jp] = TrigArr[jp-1].Add(tp);
}

//_____________________________________________________________________________
void rz_harm_poly::PrintTrigArr(ostream &out)
{
   out << "TrigArr: TASize = " << TASize
       << "\tMaxM = " << MaxM << endl;
   if (TrigArr) {
      if (MaxM < TASize) {
         unsigned jm, old_prec = out.precision();
         out.precision(5);
         out << "M:     ";
         for (jm = 0; jm <= MaxM; ++jm) {
            out << setw(12) << left << jm;
         }
         out << "|\nCos_M: ";
         for (jm = 0; jm <= MaxM; ++jm) {
            out << setw(12) << left << TrigArr[jm].CosPhi;
         }
         out << "|\nSin_M: ";
         for (jm = 0; jm <= MaxM; ++jm) {
```

```
            out << setw(12) << left << TrigArr[jm].SinPhi;
         }
         out << "|" << endl;
         out.precision(old_prec);
      } else {
         out << "\tTrigArr size is not adjusted." << endl;
      }
   } else {
      out << "\tTrigArr is not allocated." << endl;
   }
}

//_____________________________________________________________________________
rz_harm_poly rz_harm_poly::LadderUp()
{
//Return a polynomial with increased M
//
   rz_harm_poly p_out; p_out.data.reserve(2*L);
   unsigned it;
   poly2d_term term;

   //In 2 passes (for-cycles) to get terms in z-descending order
   for(it = 0; it < data.size(); ++it) {
      term = data[it];
      if (term.np[0]) {
         term.coeff *= int(term.np[0]) - M;
         --term.np[0];
         ++term.np[1];
         p_out.data.push_back(term);
      }
   }
   for(it = 0; it < data.size(); ++it) {
      term = data[it];
      if (term.np[1]) {
         term.coeff *= -(int)term.np[1];
         --term.np[1];
         ++term.np[0];
         p_out.data.push_back(term);
      }
   }
   p_out.Collect();
   if (p_out.data.size()) {
      p_out.L = L;
      p_out.M = M+1;
      if (abs(p_out.M) > MaxM) MaxM = abs(p_out.M);
      p_out.Scale(1./sqrt(double((L-M)*(L+M+1))));
   }
   return p_out;
}

//_____________________________________________________________________________
rz_harm_poly rz_harm_poly::LadderDwn()
{
//Return a polynomial with decreased M
//
   rz_harm_poly p_out; p_out.data.reserve(2*L);
   unsigned it;
   poly2d_term term;

   //In 2 passes (for-cycles) to get terms in z-descending order
   for(it = 0; it < data.size(); ++it) {
      term = data[it];
      if (term.np[0]) {
         term.coeff *= -int(term.np[0]) - M;
         --term.np[0];
         ++term.np[1];
         p_out.data.push_back(term);
      }
   }
```

```
   for(it = 0; it < data.size(); ++it) {
      term = data[it];
      if (term.np[1]) {
         term.coeff *= term.np[1];
         --term.np[1];
         ++term.np[0];
         p_out.data.push_back(term);
      }
   }
   p_out.Collect();
   if (p_out.data.size()) {
      p_out.L = L;
      p_out.M = M-1;
      if (abs(p_out.M) > MaxM) MaxM = abs(p_out.M);
      p_out.Scale(1./sqrt(double((L+M)*(L-M+1))));
   }
   return p_out;
}

//_____________________________________________________________________________
void rz_harm_poly::Print(ostream &out)
{
   out << "L=" << L << ", M=" << M << "; ";
   poly2d_base::Print(out);
}
```

HarmBasis3DCyl

File: HarmBasis3DCyl.h

```
#ifndef HarmBasis3DCyl_h
#define HarmBasis3DCyl_h

#include "rz_harm_poly.h"

typedef vector<rz_harm_poly>  harm_poly_vec;
typedef vector<harm_poly_vec> harm_poly_arr;

//////////////////////////////////////////////////////////////////////////////
//                                                                          //
//  HarmBasis3DCyl: set of basis harmonic polynomials in cylindrical CS     //
//                                                                          //
//////////////////////////////////////////////////////////////////////////////

class HarmBasis3DCyl {

private:
   unsigned Dim;   //Dimension of the basis
   unsigned Len;   //Length of the basis, accounting negative M's

   int *L_k, *M_k; //Translation arrays from linear to (L,M) address;
   double *P_k, *Br_k, *Bz_k, *Bphi_k; //Calculated values for (r,z) terms

   harm_poly_arr PtB;   //Potential basis
   harm_poly_arr BrB;   //Br basis
   harm_poly_arr BzB;   //Bz basis
   harm_poly_arr BphiB; //phi basis

   void   EvalRZ(harm_poly_arr &B, double *val);
   double GetVal(double *coeff, double *basis);

   void Print(harm_poly_arr &B, ostream &out = cout);

public:
   HarmBasis3DCyl(const unsigned N); //The only legal constructor
   virtual ~HarmBasis3DCyl();
```

```
   unsigned GetDim() { return Dim;}
   unsigned GetLen() { return Len;}
   void     GetLM(const unsigned j, int &Lj, int &Mj) { Lj = L_k[j]; Mj = M_k[j];}

   //Sets point for the basis components evaluation
   void SetPoint(const double r, const double z, const double phi)
   { rz_harm_poly::SetPoint(r, z, phi);}

   //Fill tables with the basis component values. SetPoint(r,z,phi)
   //must be called before EvalXXX() calls.
   void EvalPtn() { EvalRZ(PtB, P_k);}
   void EvalBr()  { EvalRZ(BrB, Br_k);}
   void EvalBz()  { EvalRZ(BzB, Bz_k);}
   void EvalBphi();

   //Return the basis component value for the linear address k.
   //EvalXXX() must be called before GetXXX_k() call
   double GetPtn_k (const unsigned k) { return P_k[k];}
   double GetBr_k  (const unsigned k) { return Br_k[k];}
   double GetBz_k  (const unsigned k) { return Bz_k[k];}
   double GetBphi_k(const unsigned k) { return Bphi_k[k];}

   //Return the the potential and the field component values
   //resulted by the basis expansion with coefficients in <coeff>
   //EvalXXX() must be called before GetXXX() call
   double GetPtn (double *coeff) { return GetVal(coeff, P_k);}
   double GetBr  (double *coeff) { return GetVal(coeff, Br_k);}
   double GetBz  (double *coeff) { return GetVal(coeff, Bz_k);}
   double GetBphi(double *coeff) { return GetVal(coeff, Bphi_k);}

   void PrintPtB  (ostream &out = cout) { Print(PtB,   out);}
   void PrintBrB  (ostream &out = cout) { Print(BrB,   out);}
   void PrintBzB  (ostream &out = cout) { Print(BzB,   out);}
   void PrintBphiB(ostream &out = cout) { Print(BphiB, out);}
   void Print     (ostream &out = cout);

}; //class HarmBasis3DCyl

#endif
```

File: HarmBasis3DCyl.cxx

```
////////////////////////////////////////////////////////////////////////////
//                                                                        //
//  HarmBasis3DCyl: set of basis harmonic polynomials in cylindrical CS   //
//                                                                        //
////////////////////////////////////////////////////////////////////////////

#include "HarmBasis3DCyl.h"
//_________________________________________________________________________
HarmBasis3DCyl::HarmBasis3DCyl(const unsigned N)
{
//Construct a basis of dimension N
//
   Dim = N;
   Len = N*(N+2);

   L_k = new int [Len];
   M_k = new int [Len];

   P_k    = new double [Len];
   Br_k   = new double [Len];
   Bz_k   = new double [Len];
   Bphi_k = new double [Len];

   PtB.reserve(N);
   BrB.reserve(N);
```

```
   BzB.reserve(N);
   BphiB.reserve(N);

   rz_harm_poly::IncNPwr(N); //In order to prevent GetMaxPow() calls
   unsigned M, vLen, k = 0;
   for (unsigned L = 1; L <= N; ++L) {
      vLen = L+1;
      harm_poly_vec Pt_vec;   Pt_vec.reserve(vLen);
      harm_poly_vec Br_vec;   Br_vec.reserve(vLen);
      harm_poly_vec Bz_vec;   Bz_vec.reserve(vLen);
      harm_poly_vec Bphi_vec; Bphi_vec.reserve(vLen);

      Pt_vec.push_back  (rz_harm_poly(L));
      Br_vec.push_back  (Pt_vec[0].GetDiff(0));
      Bz_vec.push_back  (Pt_vec[0].GetDiff(1));
      Bphi_vec.push_back(rz_harm_poly());
      Bphi_vec[0].CheatL(L);

      L_k[k] = L; M_k[k] = 0; ++k;

      for (M = 1; M <= L; ++M) {
         Pt_vec.push_back  (Pt_vec[M-1].LadderUp());
         Br_vec.push_back  (Pt_vec[M].GetDiff(0));
         Bz_vec.push_back  (Pt_vec[M].GetDiff(1));
         Bphi_vec.push_back(Pt_vec[M].GetDecPow(0));
         Bphi_vec[M].Scale(M);
         L_k[k] = L; M_k[k] =  M; ++k;
         L_k[k] = L; M_k[k] = -M; ++k;
      }
      PtB.push_back  (Pt_vec);
      BrB.push_back  (Br_vec);
      BzB.push_back  (Bz_vec);
      BphiB.push_back(Bphi_vec);
   }
}

//_____________________________________________________________________________
HarmBasis3DCyl::~HarmBasis3DCyl()
{
   delete [] Bphi_k;
   delete [] Bz_k;
   delete [] Br_k;
   delete [] P_k;
   delete [] M_k;
   delete [] L_k;
}

//_____________________________________________________________________________
void HarmBasis3DCyl::EvalRZ(harm_poly_arr &B, double *val)
{
//Fills the linear array val[Len] with values of basis polynomials.
//Private function, intended for internal use only.
//
   unsigned M;
   double   V;
   rz_harm_poly *P;
   for (unsigned L = 1, k = 0; L <= Dim; ++L, ++k) {
      (*val) = B[k][0].Eval(); ++val;
      for (M = 1; M <= L; ++M) {
         P = &(B[k][M]);
         V = P->Eval();
         (*val) = V*P->GetCos(); ++val;
         (*val) = V*P->GetSin(); ++val;
      }
   }
}
```

```
//_____________________________________________________________________
void HarmBasis3DCyl::EvalBphi()
{
//Fills the array Bphi_k[Len] with values of phi-basis polynomials.
//
   unsigned M;
   double   V;
   double  *val = Bphi_k;
   rz_harm_poly *P;
   for (unsigned L = 1, k = 0; L <= Dim; ++L, ++k) {
      (*val) = 0.; ++val;
      for (M = 1; M <= L; ++M) {
         P = &(BphiB[k][M]);
         V = P->Eval();
         (*val) = -V*P->GetSin(); ++val;
         (*val) =  V*P->GetCos(); ++val;
      }
   }
}

//_____________________________________________________________________
double HarmBasis3DCyl::GetVal(double *coeff, double *basis)
{
//return value of the expansion with coefficients coeff[Len] for the basis
//Private function, intended for internal use only.
//
   double S = 0.;
   for (unsigned k = 0; k < Len; ++k) S += coeff[k]*basis[k];
   return S;
}

//_____________________________________________________________________
void HarmBasis3DCyl::Print(harm_poly_arr &B, ostream &out)
{
   unsigned jL, jM, wdt = 60;
   char fc1 = '-', fc0 = out.fill(fc1);
   for (jL = 0; jL < B.size(); ++jL) {
      out << setw(wdt) << fc1 << endl;
      out << "Basis subset " << jL+1 << endl;
      out << setw(wdt) << fc1 << endl;
      for (jM = 0; jM < B[jL].size(); ++jM) {
         B[jL][jM].Print(out);
      }
   }
   out.fill(fc0);
}

//_____________________________________________________________________
void HarmBasis3DCyl::Print(ostream &out)
{
   out << "BASIS POLYNOMIALS FOR THE POTENTIAL:\n" << endl;
   PrintPtB(out);
   out << "\nBASIS POLYNOMIALS FOR R-COMPONENT   OF THE FIELD:\n" << endl;
   PrintBrB(out);
   out << "\nBASIS POLYNOMIALS FOR Z-COMPONENT   OF THE FIELD:\n" << endl;
   PrintBzB(out);
   out << "\nBASIS POLYNOMIALS FOR PHI-COMPONENT OF THE FIELD:\n" << endl;
   PrintBphiB(out);
}
```

VITA

VITA

Vassili Maroussov was born in Irkutsk, Russia on August 1, 1961. He attended secondary School №7 in Irkutsk in 1968-1976 and Mathematical School №165 at Novosibirsk State University in 1976-1978. He won the All-Russian school competition in physics 1978. From 1978-1983 he attended Novosibirsk State University and graduated with a Russian equivalent of the MS degree in Physics in 1983. In 1983-1986 he worked in the Applied Physics Institute, Novosibirsk in the field of non-destructive analysis (neutron imaging). In 1986-1995 he worked in the Budker Institute for Nuclear Physics, Novosibirsk in the field of accelerator physics and magnet design. In 1996-2001 he worked at CERN Geneva, Switzerland in the MTA (Magnet Test and Analysis) group of the LHC division and on ATLAS, COMPASS and CMS projects. He started his Doctoral of Science in Physics at Purdue University in August 2001. He completed the thesis research in spring of 2008 and received his PhD in May 2008.

PUBLICATIONS

PUBLICATIONS

1. V.Maroussov, S.Sanfilippo, A.Siemko. "*Temperature profiles during quenches in LHC superconducting dipole magnets protected by quench heaters*", 16th International Conference on Magnet Technology (MT-16, 1999), Tallahassee, FL, USA, 26 Sept.-2 Oct. 1999. , In: IEEE Transactions on Applied Superconductivity (USA), vol.10, no.1, pp. 661-664, March 2000
2. V.Maroussov, A.Siemko. "*A method to evaluate the temperature profile in a superconducting magnet during a quench*", 1998 Applied Superconductivity Conference - ASC '98, Desert Springs Resort, CA, USA, 13-18 Sep 1998 / Ed. by Adams, M - IEEE Trans. Appl. Superconduct. : 9 (1999) No.2.
3. V.Maroussov, M.Benedikt. "*A new method for the calibration of magnetic field measuring devices*", CERN internal note MTA-IN-98-025, February 18, 1998, CERN, Geneva, Switzerland.
4. V.Maroussov. "*Estimation of the longitudinal force on a conducting tube inside the magnet aperture during a quench*", CERN internal note MTA-IN-97-15, November 28, 1997, CERN, Geneva, Switzerland.
5. L.Bottura, V.Maroussov. "*Influence of additive noise on magnetic measurements by rotating coil*", CERN internal note MTA-IN-97-14, November 28, 1997, CERN, Geneva, Switzerland.
6. A.D.Chernyakin, V.N.Eschenko, V.N.Marusov, G.I.Silvestrov "*Fast raster scanning system for active dose field formation*" ", pp. 573-578 NIM-A 370 (1996).
7. V.P.Belov, V.N.Karasjuk, G.M.Kazakevich, V.N.Marusov and G.I.Silvestrov. "*Microtron of Variable Energy*", pp. 253-256, Proceedings of the II Asian Symposium on Free Electron Lasers, Novosibirsk, June 13-16 1995, Russia.
8. G.M. Kazakevich, V.N. Marusov and G.I. Silvestrov. *"8 MeV Microtron - the injector for an Electron Synchrotron*" - the same Proceedings, pp. 257-261.
9. I.I.Averbukh, A.D.Cherniakin, L.L.Danilov, V.N.Karusyuk, M.M.Karliner, V.N.Marusov, G.I.Silvestrov, V.G.Volokhov, T.A.Vsevolozhskaja and G.S.Willewald. "*Project of small-dimentional 200MeV proton synchrotron*", p.413, EPAC-88, Rome, June 7-11.