Signatures of Cosmic Topology in the Polarised Cosmic Microwave Background

Laura J. Watson
Astrophysics Group

Department of Physics
Imperial College London

Thesis submitted for the Degree of Doctor of Philosophy to
Imperial College London

· 2014 ·

To my parents

Declaration and Copyright

This thesis is my own work, except where explicitly indicated in the text.

Laura Watson
March 2014

Publications

- *A Bayesian analysis of the 27 highest energy cosmic rays detected by the Pierre Auger Observatory*.
  Watson L. J., Mortlock D. J., Jaffe A. H. 2011, MNRAS, 418, 206

Abstract

Cosmic topology is difficult to constrain due to the lack of observable phenomena which are affected by this property of the Universe. The phenomenon with the most potential to reveal cosmic topology is the cosmic microwave background (CMB). The task of constraining topology with the CMB is challenging, and so the more data that is utilised the better. This thesis sets out a method that uses the full information available from the CMB, including polarisation, in the form of a Bayesian analysis of the full correlation matrix of the CMB. A catalogue of flat spaces is presented, of which four are analysed here; the remainder could be analysed with minimal modifications to the code developed in this work. With a little more modification, the code could also be utilised to investigate spherical and hyperbolic spaces. The four topologies analysed here are the flat torus, half turn space, Klein space and Klein space with vertical flip. More work needs to be done on the Bayesian analysis in order to achieve constraints on these four spaces; efforts in this work were concentrated on efficiently generating full correlation matrices. The code developed for this task is capable of generating at least 231 individual correlation matrices for a given topology (the parameters varied being the size and cosmology of the Universe, as well as the type of correlation, e.g. TT), for a CMB resolution of $\ell = 30$ and a spatial resolution of $k = 100/L$ (where k are Fourier modes and L is the size of the Universe), on an entry level server in less than one day.

Contents

List of Tables **10**

List of Figures **23**

Acknowledgments **24**

Prologue **25**

I Cosmic Topology **27**

1 Introduction to Topology **28**
1.1 Characterising Space 28
1.1.1 Intrinsic and Extrinsic Properties 29
1.2 Types of Topology 33
1.2.1 Connectedness 33
1.2.2 Orientability 34
1.2.3 Topologies that Support Constant Curvature 36
1.3 The Story So Far 38
1.3.1 The Theory 38
1.3.2 The Observations 41

2 Mathematical Descriptions **45**
2.1 Terminology 45
2.2 Eigenmodes of Different Spaces 46
2.2.1 Eigenmodes of the Universal Covering Space 46
2.2.2 Eigenmodes of a Quotient Space 46
2.2.3 Method for Finding the Eigenmodes of Flat Spaces 47
2.3 Catalogue of Flat Spaces 52

II The Cosmic Microwave Background (CMB) 59

3 The Impact of the CMB on Cosmology 60
3.1 Experiments . . . 62
3.1.1 How Well Does the Standard Model Fit the Data? . . . 66

4 Physics of the CMB 69
4.1 Temperature Anisotropies . . . 69
4.1.1 Origins of CMB Temperature Anisotropies . . . 69
4.2 Polarisation . . . 70
4.2.1 Origins of CMB Polarisation . . . 73
4.2.2 Relating E and B Modes to Stokes' Parameters . . . 76

5 Mapping the CMB 81
5.1 Harmonic Expansion . . . 81
5.2 Correlation Matrices and Angular Power Spectra . . . 83
5.2.1 Properties of the Correlation Matrix . . . 85
5.2.2 Properties of the Angular Power Spectrum . . . 85
5.3 Converting to Pixel Space . . . 86
5.4 Taking Measurements . . . 88
5.4.1 Uncertainties . . . 89
5.4.2 Challenges . . . 90

III Bayesian Analysis 95

6 The General Bayesian Approach 96
6.1 Bayes' Theorem . . . 96
6.1.1 Choosing Priors . . . 97
6.2 Model Comparison . . . 98
6.3 Bayesian Inference in Astrophysics and Cosmology . . . 99

7 Example Application of the Bayesian Approach: The Origin of UHECRs 100
7.1 Brief Introduction to UHECRs . . . 100
7.1.1 The GZK Effect . . . 101
7.2 Purpose . . . 103
7.3 Experiments and Data . . . 104
7.3.1 PAO Observations of UHECRs . . . 106

7.3.2 Local AGNs . . . 107
7.4 Method . . . 108
7.4.1 UHECR Model . . . 111
7.4.2 Simulations . . . 114
7.5 Results . . . 117
7.6 Discussion . . . 119

8 CMB Probability Density Functions 123
8.1 The CMB Likelihood . . . 124
8.2 Sampling Methods . . . 124
8.2.1 Gridding . . . 125
8.2.2 MCMC Sampling . . . 126
8.2.3 Choice of Sampling Method . . . 130

IV Constraining Topology with the Polarised CMB 131

9 Prescription 132
9.1 Data Used in this Thesis . . . 133
9.2 Cases Investigated in this Thesis . . . 133
9.3 Method for Constraining Topology using Polarisation . . . 135
9.4 Covariance Matrix Code . . . 139
9.4.1 Hardware and Software Specifications . . . 139
9.4.2 Calculations, Storage and Time . . . 139
9.4.3 Testing . . . 141

10 Results 143
10.1 Covariance Matrices . . . 143
10.2 Simulated Data . . . 156
10.2.1 Temperature Maps . . . 161
10.2.2 Polarisation Maps . . . 174
10.3 WMAP Data . . . 187

11 Discussion 190

Epilogue **192**

Bibliography **194**

Appendices **203**

A Code for Computing the CMB Correlation Matrix for E_1, E_3, E_7 and E_9 **204**

A.1 Usage 204

A.2 Hierachy 205

A.3 The Code 206

List of Tables

2.1 Table of flat spaces... 56

3.1 Best fit cosmological parameters for ΛCDM from WMAP 9-year data and Planck data (see Hinshaw et al. 2012 and Planck Collaboration et al. 2013b respectively). The uncertainties given represent the 68% confidence intervals of the values. In the case of the WMAP fit, the densities listed are three of six model parameters that have been optimised in order to fit the model to the data; H_0 has been derived from the other parameters once the best fit has been achieved. In the Planck case, only the physical densities are among the six fitted model parameters; Ω_Λ and H_0 are both derived parameters. As usual, h is the Hubble parameter (defined as $H_0 = 100h$). 66

7.1 The measured arrival directions of the 27 PAO UHECRs listed in Abraham et al. (2007b) along with their assessment of AGN correlation (PAO corr.) and values from the method presented in this chapter of the AGN progenitor probability (which is rounded to zero if less than 0.0005) for the three different smearing angles. The CRs marked with *1, *2 and *3 in the b column are those closest to Centaurus A, with angular separations of 0.9, 2.3 and 5.8 deg respectively. 120

9.1 Values of parameters used in this thesis. Unless stated otherwise, all combinations of parameters are investigated. 135

List of Figures

1.1 In this example, the square is the fundamental domain and the lattice of identical squares forms the manifold. The tiled universal covering space is the entire manifold. Credit: Levin 2002. 29
1.2 Classes of topology. The digits below the columns denote the genus of (number of handles in) a space. Spaces with the same genus are topologically equivalent, as one can be continuously deformed into another. Credit: Lachieze-Rey & Luminet 1995. 30
1.3 The properties of space fall into two categories: topology and geometry. Topology describes the connectedness of space, quantified by genus, g, (the number of handles or "holes" that a space has) and the face-pairing of the fundamental domain. Geometry combines the more intuitive properties of scale and curvature. The assumption of an homogeneous universe allows only constant curvatures, the three possibilities represented in 2d in the bottom right-hand panel of this figure. 31
1.4 How to put a twist is a rubber band. As outsiders living in a three dimensional world, we can see that the twisted band is topologically different to the original. An inhabitant of the surface of the rubber band, however, has no way of telling this. Credit: Weeks 2002. 32
1.5 To us, bending a sheet of paper changes it's geometry. To an inhabitant of its surface however, the shortest path between any two points remains the same, and, hence, the geometry appears unchanged. Credit: Weeks 2002. 32
1.6 Example of a simply-connected topology. All loops (closed paths) can be contracted to a point. Credit: Richard Morris. 33
1.7 Example of a multi-connected topology. Loop c can be contracted to a point, but loops a and b cannot. Credit: 2006 Encyclopaedia Britannica, Inc. 33

1.8 The inhabitant of the 2-torus sees a lattice of images of itself. Credit: Levin 2002. 34
1.9 A Mobius strip is similar to the case of the twisted rubber band in Fig. (1.4) except that, instead of a 360 twist, a 180 degree twist is made before reconnecting the ends. Now an inhabitant is aware of the difference in topology. As the two dimensional character "A Square" travels around a Mobius strip he comes back as his mirror image. Weeks 2002. 35
1.10 A Klein bottle is similar to the Mobius strip but in three dimensions. This is a two dimensional representation of the Klein bottle, which is created by glueing the edges of this square so that the arrows match up. An inhabitant of this surface will travel out the left side and come back through the right side as his mirror image. Credit: Weeks 2002. 35
1.11 A platonic solid is a regular, convex polyhedron with congruent faces of regular polygons. The same number of faces meet at each vertex. There are only five such solids, each named after the number of faces it has. Credit: Martin Chaplin 37
1.12 Fundamental domains of some spherical multi-connected manifolds. Left to right: the regular octahedron, the truncated cube and the regular dodecahedron. Credit: Niarchou & Jaffe 2006. 38
1.13 Intersecting images of the CMB sky. The observer is located at the centre of the middle sphere. The cross-section of the intersection is circular, hence the term "circles-in-the-sky". Credit: Riazuelo et al. 2006. 43

2.1 The difference between a non-trivial transformation (left) and trivial transformation (right). 46
2.2 The ten completely compact Euclidean manifolds. The shapes represent the fundamental domains of these manifolds. The doors indicate flips or turns that occur as an inhabitant moves out one face and back in through another, and similar for the windows. If no doors or windows are shown, there are no flips or turns between the faces. The exception is E6, where all pairs of faces have transformations like the single example shown. Note that E2 to E10 are all quotients of E1: the dimensions of the fundamental domains of E2 to E10 are given in terms of the dimensions of the fundamental domain of E1 (e.g., the length of E2 in the z-direction, L_z^{E2}, is equal to $L_z^{\mathrm{E1}}/2 = L_z/2$). 54

2.3 The four chimney spaces: Euclidean manifolds which only have two compact dimensions. The same rules apply here regarding the doors as in Fig. (2.2). 55
2.4 The two slab spaces: Euclidean manifolds which only have one compact dimension. The same rules apply here regarding the doors as in Fig. (2.2). 55

3.1 Intensity spectrum of the CMB obtained from the FIRAS instrument of the COBE satellite. The curve is a theoretical black body curve. The data is represented by points, which lie inside the curve, and error bars, which are smaller than the thickness of the curve. Credit: Fixsen et al. 1996 61
3.2 Temperature anisotropies simulated over the whole sky at COBE (left) and Planck (right) resolutions. Credit: Planck Collaboration 2005. 63
3.3 $5^\circ \times 5^\circ$ patch of sky simulated at WMAP (94 GHz, 15 FWHM) and Planck (217 GHz, 5 FWHM) resolutions. Left - WMAP 2 years. Centre - WMAP 8 years. Right - Planck 1 year. Credit: Planck Collaboration 2005. 64
3.4 Actual Planck one-year all-sky survey of the microwave background (without foregrounds subtracted). Image shows data spanning the full frequency range of Planck (30 to 857 GHz). Credit: ESA, HFI and LFI consortia. 64
3.5 Spectrum of the CMB and other sources of fluctuations in the microwave sky across the frequency channels (grey columns) of Planck. The total Galactic fluctuation levels depend on angular scale, and are shown for about 1°. Extra-galactic sources dominate smaller scales (EG, 10' shown on diagram). The highest frequencies are primarily sensitive to dust. Credit: Planck Collaboration 2005. 65
3.6 Planck temperature angular power spectrum. The red data points have error bars that include both noise and cosmic variance. The green line is the best-fit spectrum for a flat ΛCDM cosmology. The green region represents cosmic variance only (no noise). The fit is extremely tight above an ℓ of around 40. Below this value, data tend to sit underneath the model line; this deficit is between 5-10%. Credit: Planck collaboration et al. 2013c. 67

4.1 This is a theoretical CMB anisotropy power spectrum generated using the code CMBFAST (www.cmbfast.org). The multipole ℓ represents angular scale (higher multipoles for smaller angular scales). The vertical axis is the mean square temperature fluctuation at the scale represented by ℓ. Credit: Scott & Smoot 2008. 71
4.2 WMAP 9-year polarisation maps for different frequency bands. The coloured background shows the intensity, while the white lines indicate the angle. Credit: Bennett et al. 2012. 72
4.3 Simulated direction (left) and amplitude (right) of polarisation anisotropies at Planck resolution for a pure scalar fluctuation mode. Credit: Planck Collaboration 2005. 73
4.4 How a quadrupole in the photon fluid induces polarised scattered light. The green dot in the centre is an electron. Black lines are directions of propagation of photons (two incident and one scattered are shown, but photons will travel to and from the electron in all directions). The quadrupole lies in the plane of the page. The axes marked ϵ' are incident polarisation, while ϵ denotes the polarisation of photons scattered orthogonal to the plane of the quadrupole. Incident polarisation components orthogonal to the scattered direction are transferred to the scattered photons; incident components that are parallel to this direction are not transferred. Photons from the hot, overdense (blue) region are more abundant than those from the cool, underdense (red) region and so contribute more to the scattered polarisation. Hence non-polarised incident light can result in polarised scattered light. Credit: Hu & White 1997. 74
4.5 Different types of quadrupole on the LSS. The blue and red sheets represent the background of overdense (hot) and underdense (cold) regions respectively (scalar perturbations). Left: scalar perturbation - simple quadrupole in energy density. Centre: vector perturbation - opposing bulk motions in two adjacent isothermal regions (one hot, one cold) create an effective quadrupole. Right: tensor perturbations - gravity waves distort space in the plane of the perturbation, changing circles of particles into ellipses and, hence, isothermal regions into quadrupoles. Credit: Hu & White 1997. 75

4.6 Polarisation patterns for different types of modes. Left: positive E mode, centred on a hot spot. Centre left: negative E-mode, centred on a cold spot. Centre right: negative B mode, centred on a hot spot. Right: positive B-mode, centred on a cold spot. Credit: Kaplan et al. 2003. 75
4.7 Stokes' polarisation parameters. The x and y axes lie in the plane perpendicular to the direction of propagation of a wave. The magnitudes of Q, U and V indicate the amount of each type of polarisation present. Their sign is used to indicate one of two directions. Each of these plots depicts a different extreme case, where only one type of polarisation is present. Credit: Dan Moulton. 77
4.8 Positive Q (yellow lines) and U (purple lines) plotted on the celestial sphere (left) and map of the sky (right). Negative Q and U are not shown but are simply perpendicular to their positive counterparts. The local x- and y- axes for the Stokes' parameters have been chosen to be longitudinal and latitudinal with respect to the global z-axis (running from the bottom to the top of the sphere). This means than the net effect of the presence of Q polarisation is to create only E modes (and similar for U and B). Credit: Hu & White 1997 (note that this figure is used to explain a different situation in the source paper). 79

5.1 TT (labelled $\Theta\Theta$ on the plot), EE, and BB power spectra. The EE spectrum is out of phase with respect to the TT spectrum. The grey region covers a number of possible BB power spectra, depending on the strength of the gravity waves in the LSS. The line labelled g. lensing is a predicted B mode spectrum generated purely by gravitational lensing (see Section 5.4.2). Credit: Hu et al. 2003. 87
5.2 Reionisation bump in the low ℓ region of the TE power spectrum. Credit: Kogut et al. 2003. 92
5.3 ISW contribution to the TT power spectrum at low ℓ. The solid line is the theoretical prediction and the data points are estimated from 2MASS and NVSS galaxies with WMAP9 data. Credit: Rassat & Starck 2013. 93

5.4 Contribution of gravitational lensing to the temperature power spectrum, based on a typical ΛCDM model. Top: the lensed temperature power spectrum (blue, solid line), unlensed spectrum (red, dotted line) and power from lensing only (black, dashed line). Bottom: the fractional change in the power spectrum due to lensing. Credit: Lewis & Challinor 2006. 94

7.1 Plot of simulated received energy spectra from mono energetic sources at different distances from Earth. Each source emits cosmic rays at the energy of $E_{\rm emit} = 10^{21}$ eV. Curves from rightmost to leftmost represent sources at increasing distances. The received spectrum from a source at only 10 Mpc has a distinct spike containing cosmic rays that have reached us without losing energy to the GZK effect. By 100 Mpc, the fraction of cosmic rays whose energy remains at $E_{\rm emit} = 10^{21}$ eV is negligible. i.e., the probability of an $E_{\rm emit} = 10^{21}$ eV cosmic ray reaching us from a source more distant than 100 Mpc is negligible. Credit: Achterberg et al. 1999. 102

7.2 Cosmic ray energy spectrum containing events measured by various different experiments. Credit: William F. Hanlon, University of Utah. 105

7.3 Spectrum of PAO UHECRs, with HiRes detections for comparison. Credit: Abraham et al. 2010. 106

7.4 The arrival directions of the $N_{\rm c} = 27$ PAO UHECRs (black points) and the source-weighted exposure (greyscale: darker indicates greater exposure) for the background-only model (left) and the AGN-only model (right), in Galactic coordinates. The Galactic Centre (GC), South Celestial Pole (SCP) and PAO's field of view (FoV) are all indicated. Lines of constant Galactic latitude $|b| = 10^\circ$ are also shown. Credit: Watson et al. 2011. 107

7.5 The posterior probability of the UHECR rate from VCV AGNs, $\Gamma_{\rm src}$, and the uniform background rate, $R_{\rm bkg}$, implied from a simulated sample of 27 UHECRs, all of which were emitted by VCV AGNs. The contours enclose 68%, 95% and 99.7% of the posterior probability, and the line plots show the marginalised probability for each rate. 114

7.6 Same as Fig. (7.5), but for a simulated sample of 27 isotropically distributed UHECRs. 115

7.7 Same as Fig. (7.5), but for all 27 PAO UHECRs. 116

7.8 Same as Fig. (7.5), but for the 22 PAO UHECRs with arrival directions at least 10° from the Galactic plane. 116
7.9 Posterior distributions of the fraction of observed UHECRs that are from the population of VCV AGNs, $F_{\rm AGN}$, shown for simulated samples (both isotropic and AGN-only) and for the real PAO data. Curves for both the full sample of 27 UHECRs and the cut sample of 22 UHECRs (with arrival directions at least 10° from the Galactic plane) are shown in the same panel. Each panel represents a different smearing angle. 118

9.1 Flow diagram of method for constraining topology using the polarised CMB. The black boxes show code written as part of the work in this thesis. The Fourier mode and CMB covariance codes have been debugged but the likelihood and MCMC codes have not. Grey boxes depict freely available code from other sources. CAMB was modified to enable the Fourier modes to be read from file and the corresponding transfer functions to be written to file. A short routine was written in HEALPix to process the CMB data. Parameters, variables and data are shown in red, with the red arrows showing their progression through the code. 136

10.1 Maps showing the locations of pixels 500, 1000, 1500, 2000, 2500 and 3000. 144
10.2 Maps of the correlation matrices for the flat torus, E_1, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 144
10.3 Maps of the correlation matrices for the flat torus, E_1, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 145
10.4 Maps of the correlation matrices for half turn space, E_2, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 146

10.5 Maps of the correlation matrices for the flat torus, E_1, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 147
10.6 Maps of the correlation matrices for half turn space, E_2, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 148
10.7 Maps of the correlation matrices for half turn space, E_2, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 149
10.8 Maps of the correlation matrices for klein space, E_7, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 150
10.9 Maps of the correlation matrices for klein space, E_7, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 151
10.10 Maps of the correlation matrices for klein space, E_7, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 152
10.11 Maps of the correlation matrices for klein space with vertical flip, E_9, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 153

10.12Maps of the correlation matrices for klein space with vertical flip, E_9, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 154
10.13Maps of the correlation matrices for klein space with vertical flip, E_9, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE. 155
10.14Simulated maps for the flat torus, E_1, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters. 157
10.15Simulated maps for half turn space, E_2, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters. 158
10.16Simulated maps for klein space, E_7, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters. 159
10.17Simulated maps for klein space with vertical flip, E_9, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters. 160
10.18Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.2). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 162
10.19Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.3). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 163
10.20Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.4). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 164
10.21Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.5). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 165

10.22Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.6). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 166
10.23Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.7). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 167
10.24Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.8). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 168
10.25Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.9). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 169
10.26Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.10). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 170
10.27Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.11). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 171
10.28Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.12). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 172
10.29Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.13). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 173

10.30Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.2)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 175
10.31Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.3)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 176
10.32Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.4)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 177
10.33Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.5)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 178
10.34Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.6)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 179
10.35Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.7)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 180

10.36Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.8)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 181
10.37Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.9)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 182
10.38Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.10)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 183
10.39Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.11)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 184
10.40Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.12)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 185
10.41Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.13)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 186

10.42WMAP 9-year temperature maps. The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. 188
10.43WMAP 9-year Q and U polarisation maps (for E-mode polarisation only). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4. 189

A.1 Flow chart showing the alogorithm employed in c_topoterms.cpp, c_topoterms3.cpp, c_topoterms7.cpp and c_topoterms9.cpp. 207

Acknowledgements

I would like to extend my gratitude to Andrew Jaffe for the time he gave to supervising my work throughout my doctoral studies; his ideas and guidance were crucial. I would also like to offer special thanks to David Watson, who dedicated much time to helping me improve the efficiency of my code - a key part of this work.

There are many people that I wish to thank for their help with my research on the origins of ultra high energy cosmic rays. Daniel Mortlock and Andrew Jaffe were collaborators on the paper published on this work. Daniel Mortlock, especially, contributed essential guidance, information and ideas. In addition, this research would not have been possible without the help of several members of the Pierre Auger Collaboration, particularly Johannes Knapp, Angela Olinto, Benjamin Rouillé d'Orfeuil and Subir Sarkar. Abraham Achterberg's help was also invaluable, in particular by providing an unpublished manuscript. Roberto Trotta provided valuable input during the early stages of this project. The paper on this work was improved thanks to a number of thoughtful comments by the anonymous referee.

Finally, I wish to express my appreciation to my family and friends for their understanding and support throughout my studies. In particular, I wish to thank Jonathan Tottle for his encouragement, emotional support and patience, as well as many hours of proof-reading. I am also thankful to Brian Barham for his advice and inspiration, both of which aided my perseverance in my studies. Last, but not least, I could not have completed this work without the unwavering care and encouragement of my parents. They have supported me in too many ways to mention, but I am deeply grateful for everything that they have done.

Prologue

The shape, size and extent of the Universe comprise some of the most fundamental yet-unanswered questions in cosmology. Cosmic topology is concerned with the shape of the Universe and how its boundaries are connected; the concept of topology was developed in the 19th century and became an active field in the early 20th century. However, any attempt to constrain cosmic topology requires high-precision observations, even more so than attempts to constrain cosmology. This is because there are fewer observations that we can make that depend on topology than depend on cosmology. The Cosmic Microwave Background (CMB), being the most powerful probe of the early Universe, provides us with the most promising opportunity to constrain topology. In the last few decades, observations of the CMB have started to reach a precision that allows us to take advantage of this opportunity.

Surprisingly few attempts have been made to utilise all the information provided by the full correlation matrix of the CMB, due to the process being computationally expensive. Many analyses have been performed using the CMB power spectrum (a reduced form of the correlation matrix), but this ignores valuable information. In addition, very few have made use of the information contained in the polarisation data of the CMB. This is because it was only in 2003 that the Wilkinson Microwave Anisotropy Probe (WMAP) provided the first glimpse of the polarised CMB. These are rather noisy, but the most recent CMB space observatory, Planck (launched in 2009), promises much improved measurements. This data is due to be released in 2014, making the case for utilising CMB polarisation even stronger, as any methods that are developed for WMAP data could quickly be applied to Planck data upon release. In this thesis, a method is adopted that harnesses the full CMB correlation matrix for both temperature and polarisation.

This thesis consists of four parts. The first three explore the three main ingredients used in this work to try to constrain cosmic topology; properties of different topologies, CMB data and Bayesian analysis. Part I introduces cosmic topology and describes how we define different topologies. At the end of the last chapter of Part I,

a catalogue, or look-up table, of mathematical properties of topologies is given; this will be useful when trying to capture these properties in computational models. The second part explores the significance of the CMB and its properties before finishing with a chapter on mapping the CMB; this chapter details processes that will be needed in order to utilise data from observations of the CMB. Part III explores how Bayesian analysis is used in cosmology and astrophysics and ends with a chapter detailing a how to calculate probabilities for different topologies, given CMB data.

Part IV brings the ingredients from the first three parts together to provide a prescription for constraining topology using CMB data. The results from following this prescription are then presented and discussed.

Part I

Cosmic Topology

Chapter 1

Introduction to Topology

1.1 Characterising Space

The shape, size and extent of the universe comprise some of the most fundamental yet-unanswered questions in cosmology. The characteristics of space can be divided into two types, geometry and topology. Topology focuses on spatial properties which are preserved under continuous deformation. In order to understand these properties, it is first helpful to identify some basic concepts:

- Universal covering space: a simply connected space (see Section 1.2.1) with the same geometry as the space in question.
- Fundamental domain: the most basic element (a polygon or polyhedron) from which a manifold can be constructed (see Fig. (1.1)). The fundamental domain contains the entire universe. Tiling a universal covering space with the fundamental domain is a convenient way of visualising how one can move "out" of the universe through one boundary while simultaneously moving "back in" through another. A universe connected in this way is technically unbounded, as one could never come across a point where the universe terminates. The term "boundary" is used to describe the shape and extent of the fundamental domain.

The key topological properties of a space are:

- Holonomy group: a set of transformations within the universal covering space that describe how the faces of the fundamental domain are paired (connected).
- Genus: the number of handles in a space. These are "holes" which are created depending on the way in which the faces of the fundamental domain are

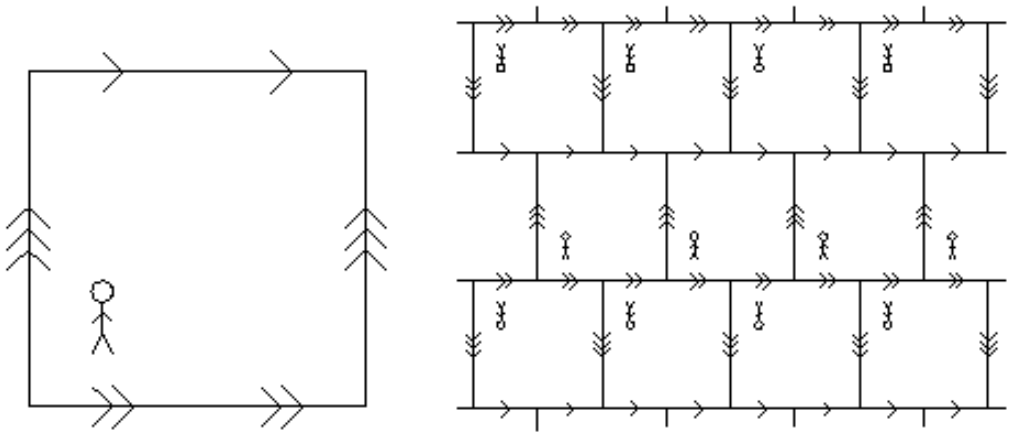


Figure 1.1: In this example, the square is the fundamental domain and the lattice of identical squares forms the manifold. The tiled universal covering space is the entire manifold. Credit: Levin 2002.

connected, not by tearing the manifold. All surfaces with the same number of handles are topologically equivalent, such as the doughnut and the teapot shown in the central panel of Fig. (1.2). These are both manifestations of the 2-d torus, which has finite area yet has no boundaries.

Combining these topological properties with the geometrical properties of scale and curvature, we can completely define a homogeneous space, see Fig. (1.3).

The geometry of space relates to the matter and energy density of the universe, e.g.: an overdense universe has positive curvature; an underdense universe has negative curvature; a critically dense universe is flat. This means that geometry can be inferred from Einstein's theory of general relativity (GR). However, each geometry can be supported by different topologies, and GR alone can not determine which of these topologies exists. Although it is currently impossible to theoretically predict cosmic topology, there is growing interest in the search through observation due to the improving precision of experiments in cosmology.

1.1.1 Intrinsic and Extrinsic Properties

Only certain aspects of topology and geometry have the potential to be detected through observation. There are of course practical issues such as the precision of instruments and size of the observable universe. But let's put these aside for a moment and assume a universe in which these considerations are not a problem. No matter how ideal the situation, there will always be properties that an inhab-

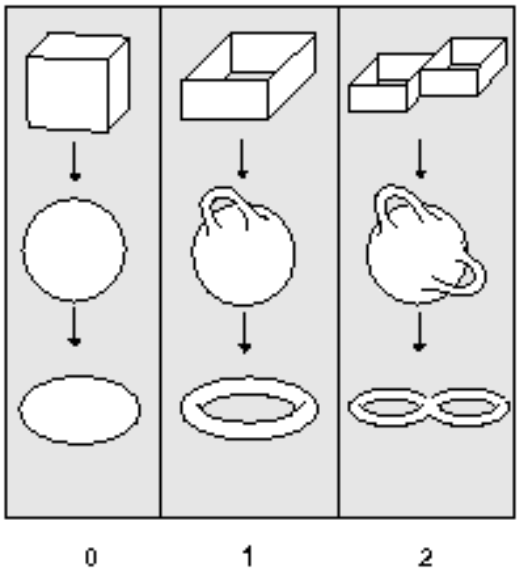


Figure 1.2: Classes of topology. The digits below the columns denote the genus of (number of handles in) a space. Spaces with the same genus are topologically equivalent, as one can be continuously deformed into another. Credit: Lachieze-Rey & Luminet 1995.

itant cannot determine. We call these *extrinsic* properties, while those which are potentially detectable are known as *intrinsic*. A two dimensional example of the distinction between intrinsic and extrinsic topology is given in Fig. (1.4), and similarly for geometry in Fig. (1.5).

Extrinsic properties are a result of embedding a space in another of higher dimensions, for example, embedding three dimensional space in four dimensions (or a two dimensional surface in three dimensions) in order to help visualise the situation. While we can describe such properties mathematically, they may not exist in physical reality. Whether or not we do in fact inhabit a three dimensional "sub-universe" embedded within some "higher-dimensional universe" and, hence, whether these extrinsic properties exist, is of little consequence, as we cannot hope to experience or detect their effects.

The matter of extrinsic and intrinsic properties is discussed in more detail in Weeks 2002.

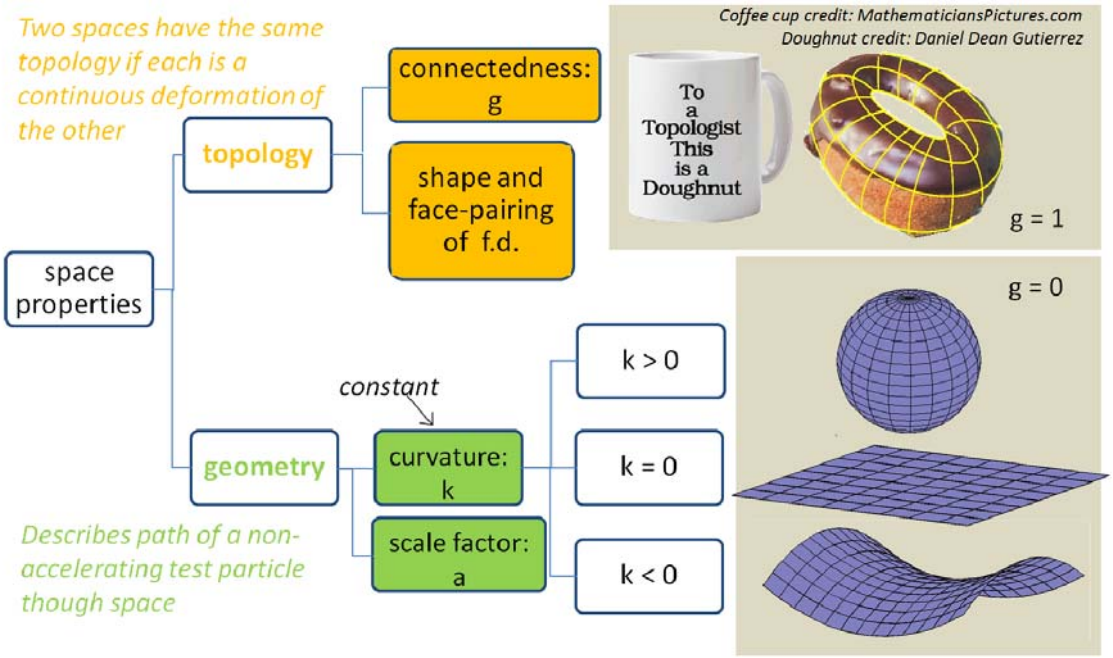


Figure 1.3: The properties of space fall into two categories: topology and geometry. Topology describes the connectedness of space, quantified by genus, g, (the number of handles or "holes" that a space has) and the face-pairing of the fundamental domain. Geometry combines the more intuitive properties of scale and curvature. The assumption of an homogeneous universe allows only constant curvatures, the three possibilities represented in 2d in the bottom right-hand panel of this figure.

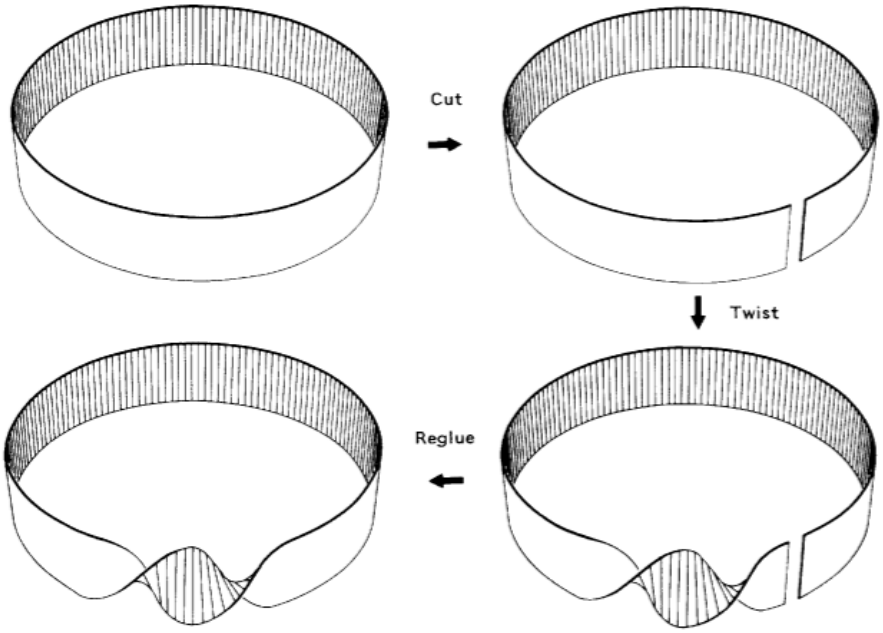


Figure 1.4: How to put a twist is a rubber band. As outsiders living in a three dimensional world, we can see that the twisted band is topologically different to the original. An inhabitant of the surface of the rubber band, however, has no way of telling this. Credit: Weeks 2002.

Figure 1.5: To us, bending a sheet of paper changes it's geometry. To an inhabitant of its surface however, the shortest path between any two points remains the same, and, hence, the geometry appears unchanged. Credit: Weeks 2002.

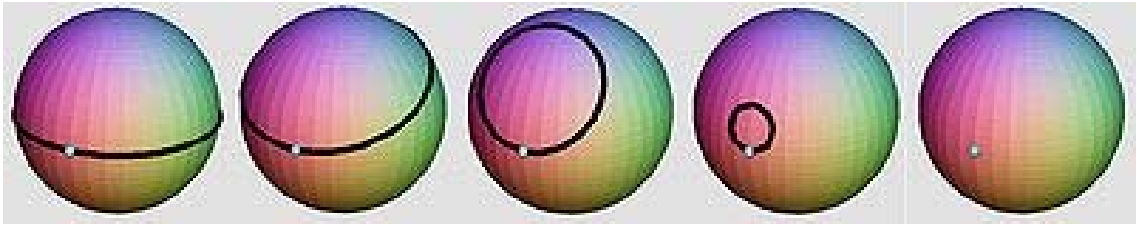


Figure 1.6: Example of a simply-connected topology. All loops (closed paths) can be contracted to a point. Credit: Richard Morris.

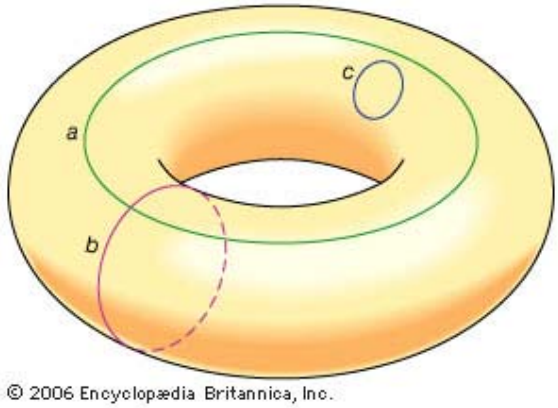


Figure 1.7: Example of a multi-connected topology. Loop c can be contracted to a point, but loops a and b cannot. Credit: 2006 Encyclopaedia Britannica, Inc.

1.2 Types of Topology

1.2.1 Connectedness

Let us distinguish between simply and multi connected spaces (or trivial and non-trivial topologies).

A trivial topology is essentially a simply connected space; one in which any loop describing a closed path through space can be contracted to a point. e.g. a space existing on the surface of a sphere, see Fig. (1.6). The universal covering space is simply connected.

Multi-connected space is one in which there exists a loop which cannot be contracted to a point. Any space containing a topological "hole", or handle, is multi-connected, e.g. a space existing on the surface of a doughnut (Fig. (1.7)). This occurs whenever the fundamental domain is not equal to the universal covering space.

The universal covering spaces that can support the geometries depicted in Fig. (1.3) are generally assumed as the topologies in the analyses of cosmological data. Usually ignored, is the possibility of a non-trivial (multi-connected) topol-

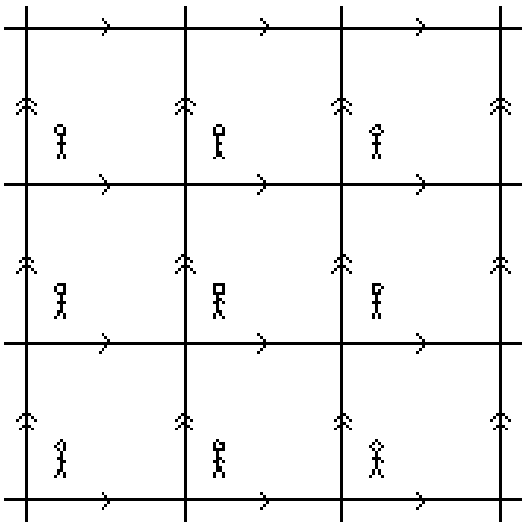


Figure 1.8: The inhabitant of the 2-torus sees a lattice of images of itself. Credit: Levin 2002.

ogy. This could have an effect on observations in astronomy: light from an observer wraps around the space so that they see themselves many times. This illusion, where an observer sees many copies of themselves, and other objects, is demonstrated by looking at the stick man observer in Fig. (1.8). It is the often neglected non-trivial topologies that draw the attention of this thesis.

1.2.2 Orientability

It is possible for a traveller in a multi-connected space to leave home, traverse space and arrive back home to find that his left and right are now the opposite way around to that of the people who stayed behind. This happens if a flip occurs as he passes from one boundary to another of the fundamental domain. The two dimensional case, the Mobius strip, is given in Fig. (1.9). The Klein Bottle, Fig. (1.10), is a three-dimensional example.

A space in which this phenomenon occurs is known as non-orientable, as no amount of rotation within the dimensions available to the inhabitant can return him to the correct "handedness". Note that, if the traveller returns upside down, with no change in handedness, he can simply rotate back to his original orientation. This, therefore, is an orientable universe. Again, a more detailed discussion can be found in Weeks 2002.

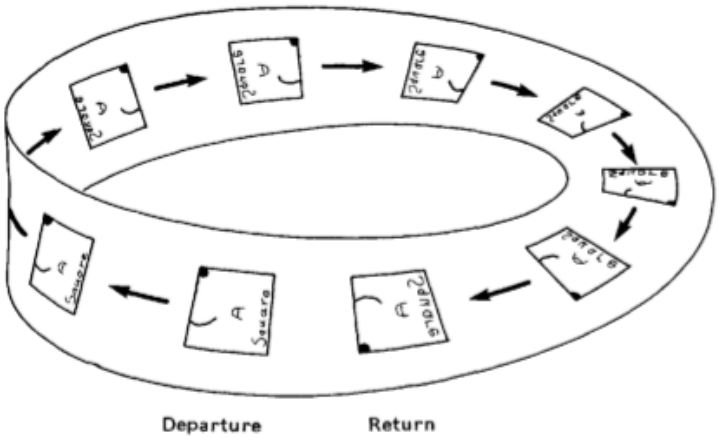


Figure 1.9: A Mobius strip is similar to the case of the twisted rubber band in Fig. (1.4) except that, instead of a 360 twist, a 180 degree twist is made before reconnecting the ends. Now an inhabitant is aware of the difference in topology. As the two dimensional character "A Square" travels around a Mobius strip he comes back as his mirror image. Weeks 2002.

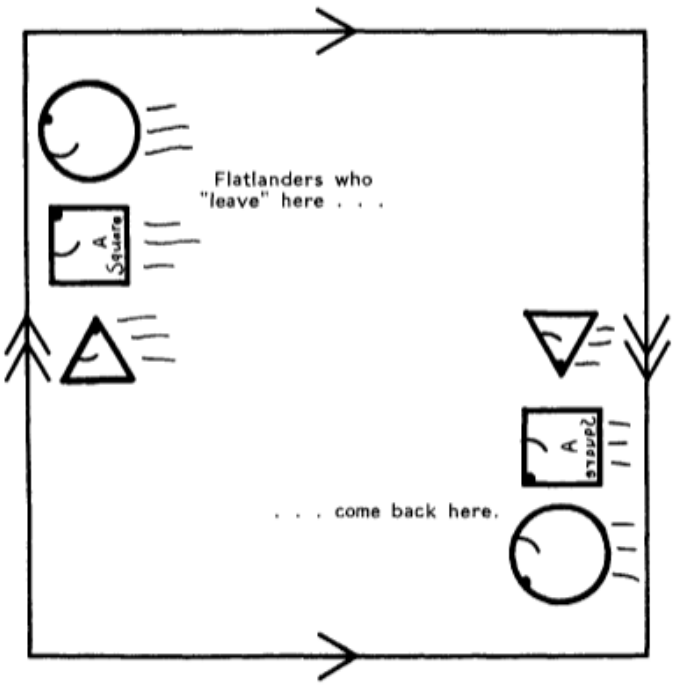


Figure 1.10: A Klein bottle is similar to the Mobius strip but in three dimensions. This is a two dimensional representation of the Klein bottle, which is created by glueing the edges of this square so that the arrows match up. An inhabitant of this surface will travel out the left side and come back through the right side as his mirror image. Credit: Weeks 2002.

Non-orientable Universes pose a problem for the standard model of particle physics. This model requires CPT symmetry (an overall symmetry in charge, parity and time in the Universe). The change in handedness, or parity, that occurs in a non-orientable universe breaks this symmetry. Arguably, this is not an issue if the Universe is large enough that no particle can travel far enough to make the round trip required to break P-symmetry. So, while this does not rule out non-orientable spaces, it perhaps makes them less favourable.

1.2.3 Topologies that Support Constant Curvature

Assuming that the Universe is homogeneous and isotropic, only topologies that support constant curvature are allowed. As explained in preceding parts of this chapter, we are interested in non-trivial topologies. We will also focus on completely compact spaces (in which all dimensions of the fundamental domain are finite); justification for this is given in Section 1.3.1. This section describes which topologies fit these criteria for each of the possible constant curvatures. Much of the time, comparisons will be made with two-dimensional spaces as they are easier to visualise.

1.2.3.1 Flat Spaces

The universal covering space of zero curvature is three-dimensional Euclidean space, $\mathbb{E}^3$.

First, consider two-dimensional Euclidean space ($\mathbb{E}^2$), the infinite plane. i.e., $\mathbb{E}^2$ is our universal covering space. There are only three possible shapes for the fundamental domain: the triangle, the quadrilateral, and the hexagon. It is not possible to "tile" a plane with any other type of polygon without leaving gaps. There is also a finite number of ways in which the edges of these shapes can be "glued" together. Thus, there is a finite number of potential topologies.

Similarly, there is a limited number of possible three-dimensional flat topologies. It has long been established that there are 18 such topologies (see Varshalovich et al. 1988) and these are described in mathematical detail in Section 2.3. However, only six satisfy the criteria of being non-trivial, completely compact and orientable and thus investigated in this thesis.

Note that there is degeneracy between some of the fundamental domains. For example, in two dimensions, a flat torus can be constructed from either a square (or other quadrilateral) or a hexagon.

Flat spaces are a sensible starting point for investigating topologies. Not only is there a finite number of them, but they are conceptually and mathematically easiest

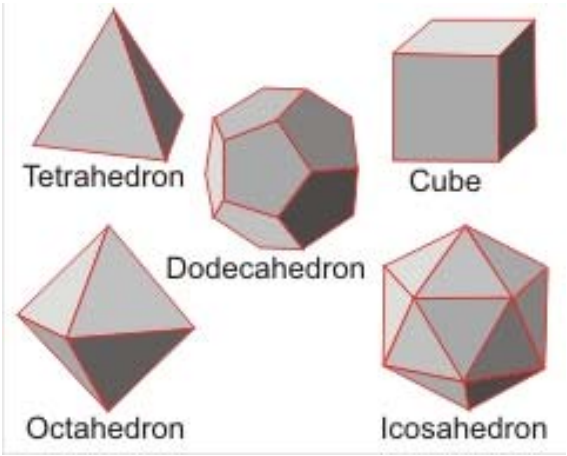


Figure 1.11: A platonic solid is a regular, convex polyhedron with congruent faces of regular polygons. The same number of faces meet at each vertex. There are only five such solids, each named after the number of faces it has. Credit: Martin Chaplin

to deal with.

1.2.3.2 Spherical Spaces

The universal covering space of positive curvature is three-dimensional Spherical space, $\mathbb{S}^3$.

The analogous two-dimensional space, the surface of the sphere ($\mathbb{S}^2$), can only be tiled in five ways. These are related to the five Platonic solids (see Fig. (1.11)). Again, with only a finite number of ways to glue the faces, we are left with a finite number of topologies.

In contrast, there is an infinite number of spherical spaces, but they are countable. The simplest manifolds are listed below and some of their fundamental domains are depicted in Fig. (1.12).

- Quaternionic space: the fundamental domain is a 4-sided prism.
- Octahedral space: the fundamental domain is a regular octahedron.
- Truncated cube space: the fundamental domain is a truncated cube.
- Poincare space: the fundamental domain is a regular dodecahedron.

1.2.3.3 Hyperbolic Spaces

The universal covering space of constant negative curvature is three-dimensional hyperbolic space, $\mathbb{H}^3$. Hyperbolic spaces are the most difficult to work with both

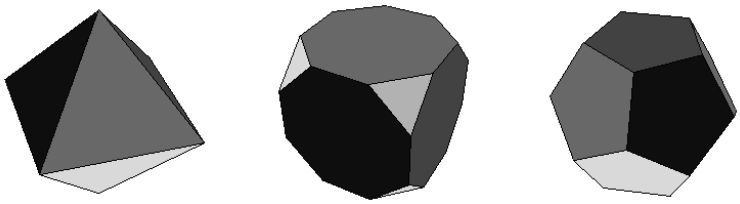


Figure 1.12: Fundamental domains of some spherical multi-connected manifolds. Left to right: the regular octahedron, the truncated cube and the regular dodecahedron. Credit: Niarchou & Jaffe 2006.

mathematically and conceptually. In two dimensions, spaces with genus zero are spheres, spaces with genus one are flat tori and spaces with genus two or higher are all hyperbolic; so there are only three possible geometries and constant, negative curvature is associated with by far the most topologies. The situation is similar in three dimensions, although there are eight possible geometries, but only three have constant curvature; while there is a finite number of flat spaces, and a countably infinite number of spherical spaces, there is an uncountably infinite number of hyperbolic spaces. Important models of hyperbolic spaces are the Klein model, the hyperboloid model and the Poincaré ball and Poincaré half-plane models.

1.3 The Story So Far

1.3.1 The Theory

In the fourth century BC, Aristotle proposed that the Universe is finite, as he believed Earth to be at its centre; something with a centre needs boundaries in order to define the central point. This was not the view of all his contemporaries, however. Archytas famously argued the opposite by highlighting the paradoxical nature of such a boundary; "If I arrived at the outermost edge of the heaven, could I extend my hand or staff into what is outside or not?", as translated in Huffman 2005.

There are some theories that *may* require a finite universe. For example, there is the (non-standard) theory of quantum cosmology, in which it is impossible to generate enough energy to create an infinite universe. Examples of work in this field include Linde (1984) and Linde (1986); Andrei Linde pioneered theories of cosmic inflation, due to quantum effects in the early universe, along with Alan Guth and Alexei Starobinski (Guth 1981, Starobinski 1982, Linde 1983). However, it generally seems to boil down to the human struggle with the concept of infinity. Plato rejected the notion of infinity. Aristotle, his student, claimed to accept the notion of infinity, where a number can be incremented endlessly, but ultimately rejected the concept

of infinity as an actual entity. Based on the strength of our often willingly-adopted arguments for a finite universe, we still struggle with it today.

It wasn't until the 19th century AD that a finite-yet-boundless model for the universe, the hypersphere, was proposed by Georg Riemann. The new concept of embedding the three-dimensional space in four dimensions in order to assign it a "shape", or *topology*, led to the discovery of many more boundless finite spaces. This was an active field in the early 20th century, which, in 1917, saw Albert Einstein adopt the hypersphere in his first relativistic model of the Universe. Einstein preferred the elegance of the simply connected hypersphere, although Willem de Sitter pointed out that the multi connected, closed space known as "projective space" was also a viable solution. Alexander Freidmann extended the solutions further to include hyperbolic spaces.

Friedmann derived two equations (describing the expansion and evolution of the Universe) from Einstein's field equations (describing the gravitational effects produced by a mass, part of the theory of General Relativity). Crucially, these equations allow all three constant curvatures and are completely independent of which topology is chosen to support a given curvature: i.e. they do not distinguish between simply- and multi-connected solutions.

Below is an overview of some of the work done in topology since the beginning of the 20^{th} century:

- 1900 - At a meeting of the German Astronomical Society in Heidelberg, Karl Schwarzschild discussed the possibility that space was non-Euclidean and suggested two kinds of possible curvatures: elliptic and hyperbolic. In the same year, he published a paper (Schwarzschild 1900) giving a lower limit for the radius of curvature of space as 2500 light years. He considered non-trivial topologies and pointed out that multiple images of the same object could be seen.

- 1913 - William Frankland pointed out limitations in the multiple image technique (Frankland 1913), mainly that different images capture an object at different times in its history (as it takes light different amounts of time to traverse different paths).

- 1913 - Duncan Sommerville published his book *The Elements of Non-Euclidean Geometry* (Sommerville 1913), which became the standard text in the field almost immediately. It was renowned for its lucid yet meticulous exposition

and discussed topics such as hyperbolic and elliptic geometries, analytic non-Euclidean geometry and representations of non-Euclidean geometry in Euclidean space, as well as philosophical implications of non-Euclidean geometry.

- 1917 - Albert Einstein presented the first cosmological solution of general relativity, a static model with three-dimensional spheres $\mathbb{S}^3$ as spatial sections (Einstein 1917).

- 1917 - Soon after Einstein published his solution, Willem de Sitter (1917) pointed out that the solution could just as well be applied to $\mathbb{P}^3$.

- 1922 - Alexander Friedmann published non-static, homogenous and isotropic cosmological solutions to Einstein's field equations (Friedmann 1922) and, soon after, realised that, while spherical manifolds are intrinsically compact, flat and hyperbolic manifolds could also be compact (Friedmann 1924).

- 1927 - Georges Lemaitre also found non-static, homogenous and isotropic comological solutions to Einstein's field equations, Lemaître (1927), and in Lemaître (1958) recognised the possibility that hyperbolic spaces could be compact.

- 1962 - Otto Heckmann and Engelbert Schucking (1962) explained how an expanding universe might rotate and experience shear.

- 1971 - George Ellis published an important article reviewing recent developments concerning the classification of spaces and their possible application to cosmology (Ellis 1971). This was followed by a revival of interest in multi-connected cosmologies, lead by theorists such as Dmitri Sokolov and Alexei Starobinski (Sokolov & Starobinski 1975).

- 1974 - Dmitri Sokolov and Victor Shvartsman estimate the size of the universe from a topological point of view (Sokolov & Shvartsman 1974).

- 1978 - A set of university lecture notes by William Thurston began to circulate around the world; in them, Thurston introduced new ideas in geometric topology as well as a number of new quotient $\mathbb{H}^3$ manifolds. These ideas made their way into a book, *Three-dimesional Geometry and Topology Volume 1*, two decades later (Thurston 1997).

- 1980 - J. Richard Gott pointed out the possibility that the universe is not only negatively curved but compact, in Gott (1980). He determined that hyperbolic and Euclidean space could admit thermalisation in a chaotic early universe.

- 1983 - Helio Fagundes also acknowledged that the universe could be negatively curved and compact (Fagundes 1983). He discusses the subsitution of a compact hyperbolic three-manifold for the infinite space $\mathbb{H}^3$, and analyses some consequences of this substitution.

- 1984 - Yakov Zel'dovich and Starobinski collaborated in Zeldovich & Starobinskii (1984) to discuss quantum creation of a universe with non-trivial topology.

- 1984 - Thurston and Jeffery Weeks publish an article to aid the visualision and study of two- and three-dimensional manifolds (Thurston & Weeks 1984).

- 1987 - Charles Dyer attended the Vatican Observatory conference and presented the idea that the universe could have a multiply-connected topology, describing several attractive features of multi-connected spaces from a theoretical stand-point and establishing some restrictions on which topologies are plausible on physical grounds. This work was published in the conference proceedings, Dyer (1987).

Despite the wealth of knowledge of different possible spaces, simply connected spaces have often been assumed by default in cosmological models. While attempts have been made to find observational evidence for multi-connected spaces since the idea was introduced, research into cosmic topology over the last few decades has increasingly incorporated analyses of data from observations; this is discussed in the next section.

1.3.2 The Observations

Another resurgence of interest in cosmic topology over the last couple of decades has resulted from developments in technology and scientific techniques that have allowed us to probe the distant universe with increasing precision. Methods of investigating topology involve identifying patterns in the distribution of astronomical objects or cosmic microwave background (CMB) anisotropies. The near-isotropy of the CMB implies a near-constant curvature universe and so the topologies considered for investigation are those that support these curvatures. It is conceivable that the universe is not homogeneous, and hence does not have constant curvature, on scales much larger than the observable universe. However, this hypothesis is not presently testable. Similarly, it is not possible to test spaces that are larger than the observable universe and we are only interested in spaces small enough to observe.

A non-trivial topology would have a number of observational effects. These include the appearance of multiple images of the same object in the sky, the suppression of power at large scales due to the finiteness of the fundamental domain, and patterns in the CMB. Some helpful reviews on topology and cosmology are Levin (2002) and Lachieze-Rey & Luminet (1995), which provide detailed discussions on different types of topology and methods for constraining topology, as well as Starkman (1998) and Luminet (1998), who describe the history of and developments in cosmic topology.

1.3.2.1 Crystallographic Method

The first method to be implemented was searching for multiple images of the same object (e.g. Sokolov & Shvartsman 1974; Fagundes & Wichoski 1987; Demianski & Lapucha 1987; Fagundes 1989). This known as the crystallographic method and is based on the idea that it is possible to observe multiple images of the same object in a multi connected universe.

In a closed, topologically connected universe, light from distant objects can reach us by multiple paths. In order to observe multiple images, we require light from a given object to have sufficient time to reach us along some of these multiple paths. This requires that the fundamental domain is sufficiently small, i.e. the distance to the observable horizon must be greater than the injectivity radius (a characteristic topological length scale of the fundamental domain).

It has been shown that it is possible to reconstruct the topology of a closed or flat universe based on the observation of a very small number of multiple images (Gomero 2003). But, as pointed out by Frankland in 1913, the identification of these images is, in practice, incredibly challenging. There are a number of reasons for this: different images represent an object at different stages of its life, making it difficult to identify them as images of the same object; different images would be seen from different directions, and therefore, different perspectives, again making it difficult to recognise them as identical; some images may be masked or hidden by other objects/high obscuration regions. Clearly, in reality, this method is not very promising (at least, with current observations).

1.3.2.2 Circles in the sky

This is a phenomenon that occurs in the CMB in the presence of a non-trivial topology if at least one dimension of the fundamental domain is smaller that the diameter

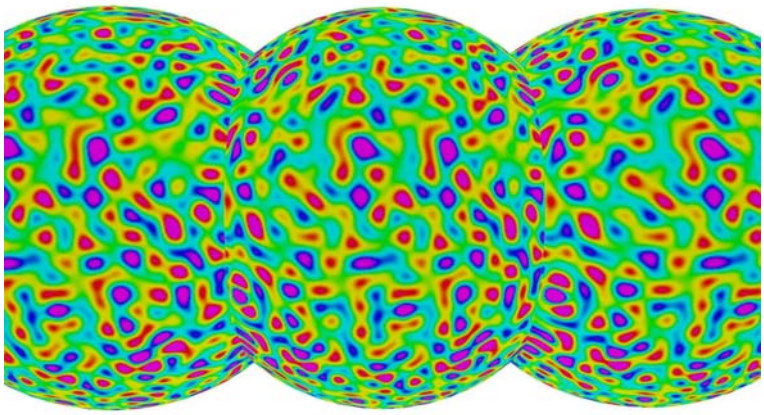


Figure 1.13: Intersecting images of the CMB sky. The observer is located at the centre of the middle sphere. The cross-section of the intersection is circular, hence the term "circles-in-the-sky". Credit: Riazuelo et al. 2006.

of the last scattering surface (LSS). Images of the LSS would intersect one another as shown in Fig. (1.13), leading to pairs of circles, of equal radii, centred at different locations in the CMB sky (Cornish et al. 1998). The size of the circles indicate the size of the dimensions of the fundamental domain; the larger the circles, the larger the dimensions of the fundamental domain that they relate to. This means that there is a limit to the size of universe that can be detected; the circles become points when the fundamental domain is the same size as the observable universe, and vanish altogether when the domain becomes larger. The angular separation of pairs of circles on the sky depend on the connectedess of the topology. Results of matching-circles analyses of CMB data have ruled out topologies much smaller than the scale of the observable universe (e.g.: Key et al. 2007; Bielewicz & Banday 2011; Planck Collaboration et al. 2013d) but have not yet identified a leading candidate for the topology itself.

1.3.2.3 Full Analysis of the CMB

The multiple-images and circles-in-the-sky methods both focus on conceptually clear observational consequences of topology. But there are other phenomena that should not be overlooked: different modes in the CMB anisotropies can be suppressed or correlated with other modes in ways that depend on the finiteness, orientability and particular symmetries of the fundamental domain. A more thorough, but more computationally expensive, method is to analyse the entire correlation matrices of CMB anisotropies (e.g.: Bond et al. 2000b,a; Niarchou 2006; Planck Collaboration

et al. 2013d). As with the circles-in-the-sky method, constraints on the lower limit of the size of the Universe have been obtained, but a dominant candidate topology has not. Due to the expensiveness of this method, the CMB power spectrum is often used rather that the full correlation matrix. The main objective of this thesis is to utilse as much information as possible from the correlation matrix, which also means including polarisation. The details required to perform this analysis are accumulated in subsequent chapters, culminating in the exact prescription in Ch. 9 and results in Ch. 10.

Chapter 2

Mathematical Descriptions

2.1 Terminology

Here, we define a few terms that will be used in this chapter.

A **fundamental domain** is a polygon or polyhedron from which a manifold, or topological space, may be constructed.

X is a **covering space** of Y if it maps onto X in a locally homeomorphic way (i.e. if a local region of X, that is no bigger than Y, can be continuously deformed into Y and if this deformation is invertable). A **universal covering space** is a simply connected covering space. For spaces Y of constant curvature, the universal covering space will have the same curvature as Y.

A multi-connected space Y can be expressed as a **quotient space**, $Y = X/\Gamma$, of covering space X, where Γ is a symmetry group of the quotient space (see below).

A **group generator**, g, of a quotient space is an operation, or spatial transformation, that maps points in the quotient space to equivalent points in the universal covering space. These transformations should be distance-preserving and non-trivial, as demonstrated in Fig. (2.1). Euclidean space, for example, has no such generators; any transformations returning the point on which they act are trivial (right diagram in Fig. (2.1)).

A **symmetry group**, Γ, of a quotient space Y contains a set of generators that can be used to completely map Y onto its entire universal covering space.

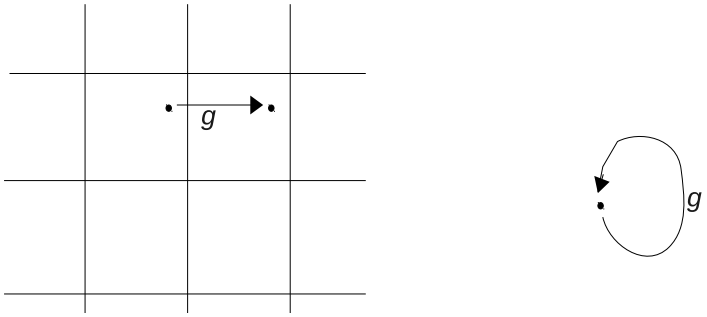


Figure 2.1: The difference between a non-trivial transformation (left) and trivial transformation (right).

2.2 Eigenmodes of Different Spaces

The eigenmodes of a space are solutions, or an orthogonal subset of the solutions, of the generalised Helmholtz equation.

2.2.1 Eigenmodes of the Universal Covering Space

The more familiar form of the Helmholtz equation is a time-independent partial differential equation which involves a scalar field in a flat space, Eq. (2.1).

$$\nabla^2 \Upsilon_k (\mathbf{x}) = -k^2 \Upsilon_k (\mathbf{x}) \tag{2.1}$$

where ∇^2 is the Laplacian, $\mathbf{x}$ describes position in space, and $\Upsilon_k (\mathbf{x})$ is one particular solution to the equation associated with wavenumber k. In this case, $\Upsilon_k (\mathbf{x})$ is an eigenmode of Euclidean space, and $-k^2$ the corresponding eigenvalue.

The generalised Helmholtz equation extends Eq. (2.1) to any space with constant curvature:

$$\nabla^2 \Upsilon_k (\mathbf{x}) = -(k^2 - K) \Upsilon_k (\mathbf{x}) \tag{2.2}$$

where the eigenvalue, $-(k^2 - K)$, now includes the curvature K.

2.2.2 Eigenmodes of a Quotient Space

The eigenmodes of a multi-connected space are a subset of those of its universal covering space. A simple example of this is the flat torus. Only wavenumbers which

correspond to a wavelength that fits a whole number of times into the fundamental domain are allowed. One can halve an allowed wavelength (or divide by any integer) to find another allowed wavelength ad infinitum. While there is an infinite number of possible wavenumbers, they comprise a discrete set of values. In contrast, Euclidean space admits wavenumbers corresponding to an infinite, *continuous* set of real-valued wavelengths; there are no boundaries to place any constraints. So some, but not all, of the wavenumbers (and corresponding eigenmodes) allowed in Euclidean space are allowed in the case of the torus. That is, the eigenmodes of the flat torus are a subset of those of Euclidean space.

The boundaries of a fundamental domain are defined by the symmetry group of the space. So, the group generators can be used to identify the subset of eigenmodes of the universal covering space that also belong to the multi-connected space in question. The method for doing so for flat spaces is described in Section 2.2.3.

2.2.3 Method for Finding the Eigenmodes of Flat Spaces

An eigenmode of a multi-connected space is one which satisfies

$$\Upsilon_{\mathbf{k}}\left(g\mathbf{x}\right) = \Upsilon_{\mathbf{k}}\left(\mathbf{x}\right) \tag{2.3}$$

for all generators, g, in the symmetry group Γ. A generator is simply a spatial transformation and so Eq. (2.3) can be re-cast as Eq. (2.4):

$$\Upsilon_{\mathbf{k}}\left(\mathcal{M}_g\mathbf{x} + \mathbf{T}_g\right) = \Upsilon_{\mathbf{k}}\left(\mathbf{x}\right), \tag{2.4}$$

where $\mathcal{M}_g$ is a transformation matrix and $\mathbf{T}_g$ is a translation vector.

These eigenmodes are eigenmodes of the universal covering space that have a periodicity in $\mathbf{x}$ of $g^n\mathbf{x}$, i.e. performing the transformation g any integer n number of times returns a point $g^n\mathbf{x}$ which has the same value of $\Upsilon_{\mathbf{k}}$ as the original point $\mathbf{x}$. These eigenmodes satisfy Eq. (2.5).

$$\Upsilon_{\mathbf{k}}\left(\mathcal{M}_g^n\mathbf{x} + \sum_{m=0}^{n-1}\mathcal{M}_g^m\mathbf{T}_g\right) = \Upsilon_{\mathbf{k}}\left(\mathbf{x}\right). \tag{2.5}$$

The flat universal covering space is Euclidean space, $\mathbb{E}^3$. One possible basis for Euclidean space is planar waves, which means the eigenmodes can be expressed as

$$\Upsilon_{\mathbf{k}}\left(\mathbf{x}\right) = e^{i\mathbf{k}\cdot\mathbf{x}}. \tag{2.6}$$

Eq. (2.6) substituted into Eq. (2.5), yields Eq. (2.7):

$$\Upsilon_{\mathbf{k}}(\mathbf{x}) = e^{i\mathbf{k}\cdot\left(\mathcal{M}_g^n\mathbf{x}+\sum_{m=0}^{n-1}\mathcal{M}_g^m\mathbf{T}_g\right)} = e^{i\mathbf{k}\cdot\sum_{m=0}^{n-1}\mathcal{M}_g^m\mathbf{T}_g}\Upsilon_{\mathbf{k}\mathcal{M}_g^n}(\mathbf{x}). \tag{2.7}$$

Note that $\Upsilon_{\mathbf{k}\mathcal{M}_g^n}(\mathbf{x})$ is only a different eigenmode to $\Upsilon_{\mathbf{k}}(\mathbf{x})$ for $\mathbf{k}\mathcal{M}_g^n \neq \mathbf{k}$. Letting N be the smallest integer that satisfies

$$\mathbf{k} = \mathbf{k}\mathcal{M}_g^N, \tag{2.8}$$

and considering the case of $n = N$ in Eq. (2.7), it is evident that only N eigenmodes are related by generator g.

Consider a linear combination of these eigenmodes,

$$\sum_{n=0}^{N-1} a_n \Upsilon_{\mathbf{k}\mathcal{M}_g^n} = \sum_{n=0}^{N-1} a_n e^{i\mathbf{k}\mathcal{M}_g^n\mathbf{T}_g}\Upsilon_{\mathbf{k}\mathcal{M}_g^{n+1}}, \tag{2.9}$$

where the right hand side of the equation is simply the left hand side acted upon once by g. A quick inspection of this equation reveals that the coefficients a_n satisfy the following relations:

$$a_{n+1} = a_n e^{i\mathbf{k}\mathcal{M}_g^n\mathbf{T}_g}, \tag{2.10}$$

$$a_0 = a_{N-1}e^{i\mathbf{k}\mathcal{M}_g^{N-1}\mathbf{T}_g}. \tag{2.11}$$

Eq. (2.9), with Eq. (2.10) and Eq. (2.11), can be solved for $\mathbf{k}$ (see the examples given in Section 2.2.3.2 and Section 2.2.3.3). And once we know the allowed wavenumbers, we also know the allowed eigenmodes, $\Upsilon_{\mathbf{k}}$.

In summary, to identify the allowed eigenmodes of a space, perform the following tasks for each generator of the space:

1. Solve Eq. (2.8) for N.

2. Solve Eq. (2.9), with Eq. (2.10) and Eq. (2.11), for $\mathbf{k}$.

3. Select only the eigenmodes $\Upsilon_{\mathbf{k}}$ corresponding to the allowed $\mathbf{k}$.

2.2.3.1 Choice of Basis

So far, we have only considered eigenmodes as planar waves, which are convenient for exploring multi-connected flat spaces. However, astronomical observations, especially those of the CMB, lend themselves to spherical waves. So, when it comes

to constraining topology with observations, it may be more convenient to express the eigenmodes of as sums of spherical waves:

$$\Upsilon_{\mathbf{k}}(\mathbf{x}) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} \xi^{\hat{\mathbf{k}}}_{k\ell m} \mathcal{Y}_{k\ell m}(\mathbf{x}), \tag{2.12}$$

where $\mathcal{Y}_{k\ell m}$ is a spherical wave, ℓ and m are the spherical harmonic multipole moments, and $\xi^{\hat{\mathbf{k}}}_{k\ell m}$ are coefficients which, as a set, capture all the properties of a topology. For Euclidean universal covering space $\mathbb{E}^3$,

$$\xi^{\hat{\mathbf{k}}}_{k\ell m} = i^{\ell} Y^{*}_{\ell m}(\hat{\mathbf{k}}) \tag{2.13}$$

where Y is the spherical harmonic function.

It is in fact the $\xi^{\hat{\mathbf{k}}E_i}_{k\ell m}$ that will be useful for constraining topology with observational data. These are listed for all possible flat spaces, E_i where i indexes the space, in Section 2.3 and were found, as part of the work presented in this thesis, by extending the prescription for finding the eigenmodes as follows:

4. Find the eigenbasis for the space. The eigenbasis modes are the set of linear combinations of covering-space eigenmodes given by the set of generators of the multi-connected space. These are easily found by substituting the solutions for N and the a_n (from steps 1. and 2.) into Eq. (2.9), and normalising the sum.

5. The $\xi^{\hat{\mathbf{k}}E_i}_{k\ell m}$ are then simply equal to the eigenbasis modes multiplied by i^{ℓ}, which is an artifact of converting from planar waves to spherical waves.

2.2.3.2 Worked Example for the 3-Torus (E_1)

Three generators of the form $g_i(\mathbf{x}) = \mathcal{M}\mathbf{x} + \mathbf{T}_i$ are needed for E_1 where

$$\mathcal{M} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \mathbb{I} \tag{2.14}$$

$$\begin{aligned} \mathbf{T}_1 &= (L_x, 0, 0) \\ \mathbf{T}_2 &= (0, L_y, 0) \\ \mathbf{T}_3 &= (0, 0, L_z) \end{aligned} \tag{2.15}$$

1. Solving Eq. (2.8) for N:

$$\mathbf{k}\mathcal{M} = \mathbf{k} \quad \Longrightarrow \quad N = 1 \tag{2.16}$$

2. Solving Eq. (2.9), with Eq. (2.10) and Eq. (2.11), for $\mathbf{k}$:

$$\begin{aligned} a_0\Upsilon_{\mathbf{k}} = a_0 e^{i\mathbf{k}\mathcal{M}\mathbf{T}_i}\Upsilon_{\mathbf{k}\mathcal{M}} &= a_0 e^{i\mathbf{k}\mathbf{T}_i}\Upsilon_{\mathbf{k}} \\ \implies \quad e^{i\mathbf{k}\mathbf{T}_i} &= 1 \\ \implies \quad \mathbf{k}\mathbf{T}_i &\in 2\pi\mathbb{Z} \end{aligned} \tag{2.17}$$

Substituting the $\mathbf{T}_i$ into Eq. (2.17) yields restrictions on $\mathbf{k}$,

$$\begin{aligned} \mathbf{T}_1 \;\text{ gives }\; k_x &= 2\pi n_x/L_x \;\text{ where }\; n_x \in \mathbb{Z} \\ \mathbf{T}_2 \;\text{ gives }\; k_y &= 2\pi n_y/L_y \;\text{ where }\; n_y \in \mathbb{Z} \\ \mathbf{T}_3 \;\text{ gives }\; k_z &= 2\pi n_z/L_z \;\text{ where }\; n_z \in \mathbb{Z} \end{aligned} \tag{2.18}$$

3. Select only the eigenmodes $\Upsilon_{\mathbf{k}}$ of $\mathbb{E}^3$ corresponding to the allowed $\mathbf{k}$ given by Eq. (2.18).

4. In this case, since $N = 1$, Eq. (2.9) simply yields the eigenbasis for the space as the set of universal-covering-space eigenmodes with the restriction $n_x, n_y, n_z \in \mathbb{Z}$.

5. This means that the coefficients $\xi^{\hat{\mathbf{k}}E_1}_{k\ell m}$ are simply the universal-covering-space coefficients, $\xi^{\hat{\mathbf{k}}}_{k\ell m} = i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$, with the restriction $n_x, n_y, n_z \in \mathbb{Z}$.

2.2.3.3 Worked Example for Quarter Turn Space (E_3)

Four generators of the form $g_{ji}(\mathbf{x}) = \mathcal{M}_j\mathbf{x} + \mathbf{T}_{ji}$ are needed for E_3 where

$$\mathcal{M}_A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \mathbb{I}, \quad \mathcal{M}_B = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.19}$$

$$\begin{aligned} \mathbf{T}_{A1} &= (L_x, 0, 0) \\ \mathbf{T}_{A2} &= (0, L_y, 0) \\ \mathbf{T}_{A3} &= (0, 0, L_z) \\ \mathbf{T}_{B1} &= (0, 0, L_z/4) \end{aligned} \tag{2.20}$$

Note that $\mathcal{M}_A$ and $\mathbf{T}_{Ai}$ form the generators of E_1 as E_3 is a quotient space of E_1. This means that the eigenmodes of E_3 are linear combinations of the eigenmodes of E_1 that we found in Section 2.2.3.2.

1. Solving Eq. (2.8) for N:

$$\begin{aligned}
&\mathbf{k}\mathcal{M}_B = (k_y, -k_x, k_z) = \mathbf{k} \quad \text{if} \quad (k_x, k_y) = (0,0) \text{ and } N = 1 \\
&\mathbf{k}\mathcal{M}_B^2 = (-k_x, -k_y, k_z) = \mathbf{k} \quad \text{if} \quad (k_x, k_y) = (0,0) \text{ and } N = 2 \\
&\mathbf{k}\mathcal{M}_B^3 = (-k_y, k_x, k_z) = \mathbf{k} \quad \text{if} \quad (k_x, k_y) = (0,0) \text{ and } N = 3 \\
&\mathbf{k}\mathcal{M}_B^4 = (k_x, k_y, k_z) = \mathbf{k} \quad \text{if} \quad (k_x, k_y) \neq (0,0) \text{ and } N = 4
\end{aligned} \tag{2.21}$$

Note that the conditions $(k_x, k_y) = (0,0)$ apply for N =1, 2 and 3 but not $N = 4$. So we only need $N = 1$ and $N = 4$ for the next step.

2. Solving Eq. (2.9), with Eq. (2.10) and Eq. (2.11), for $\mathbf{k}$:

$$\begin{aligned}
N = 1 \implies & a_0 \Upsilon_{\mathbf{k}} = a_0 e^{i\mathbf{k}\mathcal{M}_B \mathbf{T}_{B1}} \Upsilon_{\mathbf{k}\mathcal{M}_B} = a_0 e^{i\mathbf{k}\mathbf{T}_{B1}} \Upsilon_{\mathbf{k}} \\
\implies & e^{i\mathbf{k}\mathbf{T}_{B1}} = 1 \\
\implies & \mathbf{k}\mathbf{T}_{B1} \in 2\pi\mathbb{Z}
\end{aligned} \tag{2.22}$$

$$\begin{aligned}
N = 4 \implies & a_0 \Upsilon_{\mathbf{k}} + a_1 \Upsilon_{\mathbf{k}\mathcal{M}_B} + a_2 \Upsilon_{\mathbf{k}\mathcal{M}_B^2} + a_3 \Upsilon_{\mathbf{k}\mathcal{M}_B^3} \\
& = a_0 e^{i\mathbf{k}\mathcal{M}_B \mathbf{T}_{B1}} \Upsilon_{\mathbf{k}\mathcal{M}_B} + a_1 e^{i\mathbf{k}\mathcal{M}_B \mathbf{T}_{B1}} \Upsilon_{\mathbf{k}\mathcal{M}_B^2} \\
& \quad + a_2 e^{i\mathbf{k}\mathcal{M}_B \mathbf{T}_{B1}} \Upsilon_{\mathbf{k}\mathcal{M}_B^3} + a_3 e^{i\mathbf{k}\mathcal{M}_B \mathbf{T}_{B1}} \Upsilon_{\mathbf{k}} \\
\implies & a_0 = a_3 e^{i\mathbf{k}\mathcal{M}_B^3 \mathbf{T}_{B1}}, \;\; a_1 = a_0 e^{i\mathbf{k}\mathbf{T}_{B1}}, \\
& \quad a_2 = a_1 e^{i\mathbf{k}\mathcal{M}_B \mathbf{T}_{B1}}, \;\; a_3 = a_2 e^{i\mathbf{k}\mathcal{M}_B^2 \mathbf{T}_{B1}} \\
\implies & e^{i\mathbf{k}\mathcal{M}_B \mathbf{T}_{B1}} e^{i\mathbf{k}\mathcal{M}_B^2 \mathbf{T}_{B1}} e^{i\mathbf{k}\mathcal{M}_B^3 \mathbf{T}_{B1}} e^{i\mathbf{k}\mathbf{T}_{B1}} = 1 \\
\implies & \mathbf{k}\left(\mathcal{M}_B \mathbf{T}_{B1} + \mathcal{M}_B^2 \mathbf{T}_{B1} + \mathcal{M}_B^3 \mathbf{T}_{B1} + \mathbf{T}_{B1}\right) \in 2\pi\mathbb{Z}
\end{aligned} \tag{2.23}$$

Substituting $\mathbf{T}_{B1}$ into Eq. (2.22) and Eq. (2.23) yields restrictions on $\mathbf{k}$,

$$\begin{aligned}
N = 1 \implies & k_z L_z/4 \in 2\pi\mathbb{Z} \\
\implies & n_z \in 4\mathbb{Z}
\end{aligned} \tag{2.24}$$

$$\begin{aligned}
N = 4 \implies & k_z L_z \in 2\pi\mathbb{Z} \\
\implies & n_z \in \mathbb{Z}
\end{aligned} \tag{2.25}$$

3. Select only the eigenmodes $\Upsilon_{\mathbf{k}}$ of $\mathbb{E}^3$ corresponding to the allowed $\mathbf{k}$ given by Eq. (2.24) and Eq. (2.25) to use in the next step. There are additional restrictions that we need to place on $\mathbf{k}$ to ensure that the eigenmodes don't degener-

ate. As shown in the first line of working for Eq. (2.23), the four covering-space eigenmodes that we combine to find eigenmodes of E_3 are $\Upsilon_{\mathbf{k}}$, $\Upsilon_{\mathbf{k}\mathcal{M}_B}$, $\Upsilon_{\mathbf{k}\mathcal{M}_B^2}$ and $\Upsilon_{\mathbf{k}\mathcal{M}_B^3}$. We need to ensure that all combinations of $\mathbf{k}$, $\mathbf{k}\mathcal{M}_B$, $\mathbf{k}\mathcal{M}_B^2$ and $\mathbf{k}\mathcal{M}_B^3$ are unique. This requires the conditions $n_x \in \mathbb{Z}^+$ and $n_y \in \mathbb{Z}^+ \cup \{0\}$.

4. For $N = 1$, Eq. (2.9) simply yields the eigenmodes equivalent to those of the universal-covering-space eigenmodes with the restriction $n_x, n_y = 0$ and $n_z \in 4\mathbb{Z}$. $N = 4$, however, does give rise to eigenmodes that are linear combinations of the eigenmodes of the covering space:

$$
\begin{aligned}
&\Upsilon_{\mathbf{k}}^{E_3} \\
&=\frac{1}{2}\left(a_0\Upsilon_{\mathbf{k}} + a_1\Upsilon_{\mathbf{k}\mathcal{M}_B} + a_2\Upsilon_{\mathbf{k}\mathcal{M}_B^2} + a_3\Upsilon_{\mathbf{k}\mathcal{M}_B^3}\right) \\
&=\frac{1}{2}\Big(\Upsilon_{\mathbf{k}} + e^{i\mathbf{k}\mathbf{T}_{B1}}\Upsilon_{\mathbf{k}\mathcal{M}_B} + e^{i\mathbf{k}\mathbf{T}_{B1}}e^{i\mathbf{k}\mathcal{M}_B\mathbf{T}_{B1}}\Upsilon_{\mathbf{k}\mathcal{M}_B^2} \\
&\qquad\qquad + e^{i\mathbf{k}\mathbf{T}_{B1}}e^{i\mathbf{k}\mathcal{M}_B\mathbf{T}_{B1}}e^{i\mathbf{k}\mathcal{M}_B^2\mathbf{T}_{B1}}\Upsilon_{\mathbf{k}\mathcal{M}_B^3}\Big) \\
&=\frac{1}{2}\Big(\Upsilon_{\mathbf{k}} + e^{ik_zL_z/4}\Upsilon_{\mathbf{k}\mathcal{M}_B} + e^{ik_zL_z/4}e^{ik_zL_z/4}\Upsilon_{\mathbf{k}\mathcal{M}_B^2} \\
&\qquad\qquad + e^{ik_zL_z/4}e^{ik_zL_z/4}e^{ik_zL_z/4}\Upsilon_{\mathbf{k}\mathcal{M}_B^3}\Big) \\
&=\frac{1}{2}\left(\Upsilon_{\mathbf{k}} + i^{n_z}\Upsilon_{\mathbf{k}\mathcal{M}_B} + i^{2n_z}\Upsilon_{\mathbf{k}\mathcal{M}_B^2} + i^{3n_z}\Upsilon_{\mathbf{k}\mathcal{M}_B^3}\right) \\
&=\frac{1}{2}\sum_{j=0}^{3} i^{jn_z}\Upsilon_{\mathbf{k}\mathcal{M}_B^j} \\
&=\frac{1}{2}\sum_{j=0}^{3} i^{jn_z}i^{\ell}Y_{\ell m}^*(\hat{\mathbf{k}}\mathcal{M}_B^j)\mathcal{Y}_{k\ell m} \qquad \text{for } n_x \in \mathbb{Z}^+; n_y \in \mathbb{Z}^+ \cup \{0\}; n_z \in \mathbb{Z}
\end{aligned}
\tag{2.26}
$$

where the factor of $1/2$ is a normalisation constant.

5. This means that the coefficients $\xi_{k\ell m}^{\hat{\mathbf{k}}E_3}$ are either simply the universal covering space coefficients, $\xi_{k\ell m}^{\hat{\mathbf{k}}} = i^{\ell}Y_{\ell m}^*(\hat{\mathbf{k}})$, with the restriction $n_x, n_y = 0$ and $n_z \in 4\mathbb{Z}$ or given by

$$
\xi_{k\ell m}^{\hat{\mathbf{k}}E_3} = \frac{i^{\ell}}{2}\sum_{j=0}^{3} i^{jn_z}Y_{\ell m}^*(\hat{\mathbf{k}}\mathcal{M}_B^j) \qquad \text{for } n_x \in \mathbb{Z}^+; n_y \in \mathbb{Z}^+ \cup \{0\}; n_z \in \mathbb{Z} \tag{2.27}
$$

2.3 Catalogue of Flat Spaces

There are only 18 possible flat spaces (Feodoroff 1885; Bierberbach 1911; Novacki 1934), all of which are catalogued in this section. Visual representations of the

spaces are given in Fig. (2.2), Fig. (2.3) and Fig. (2.4), followed by a table of their mathematical descriptions, Table 2.1.

Of the 18 spaces, only ten are compact in all three dimensions (E_1 to E_{10}) and, of these, only six are orientable (E_1 to E_6). Just completely compact spaces are investigated in this thesis. The reasons for choosing them are given in Ch. 1, but the arguments in favour of these spaces act only to select them as a sensible/convenient starting point for investigation. Of these spaces, E_1, E_2, E_7 and E_9 are selected, as they are the easiest to implement (this is discussed in Ch. 9). The other spaces are by no means ruled out and would be worth exploration.

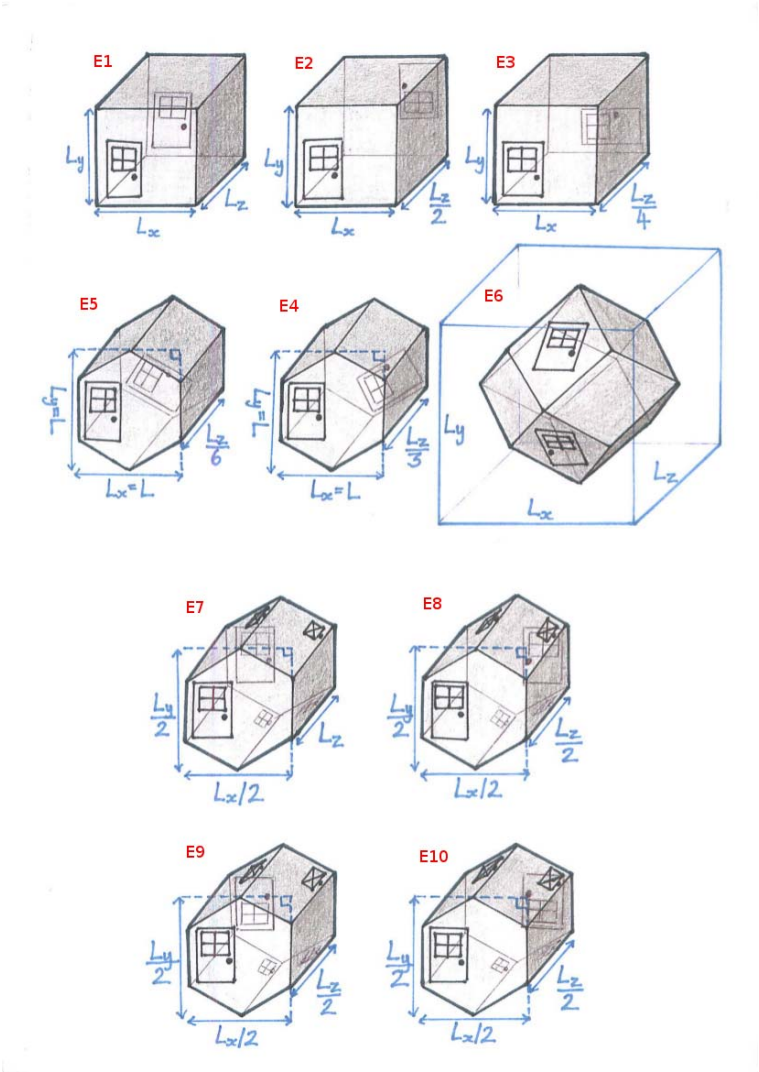


Figure 2.2: The ten completely compact Euclidean manifolds. The shapes represent the fundamental domains of these manifolds. The doors indicate flips or turns that occur as an inhabitant moves out one face and back in through another, and similar for the windows. If no doors or windows are shown, there are no flips or turns between the faces. The exception is E6, where all pairs of faces have transformations like the single example shown. Note that E2 to E10 are all quotients of E1: the dimensions of the fundamental domains of E2 to E10 are given in terms of the dimensions of the fundamental domain of E1 (e.g., the length of E2 in the z-direction, L_z^{E2}, is equal to $L_z^{\mathrm{E1}}/2 = L_z/2$).

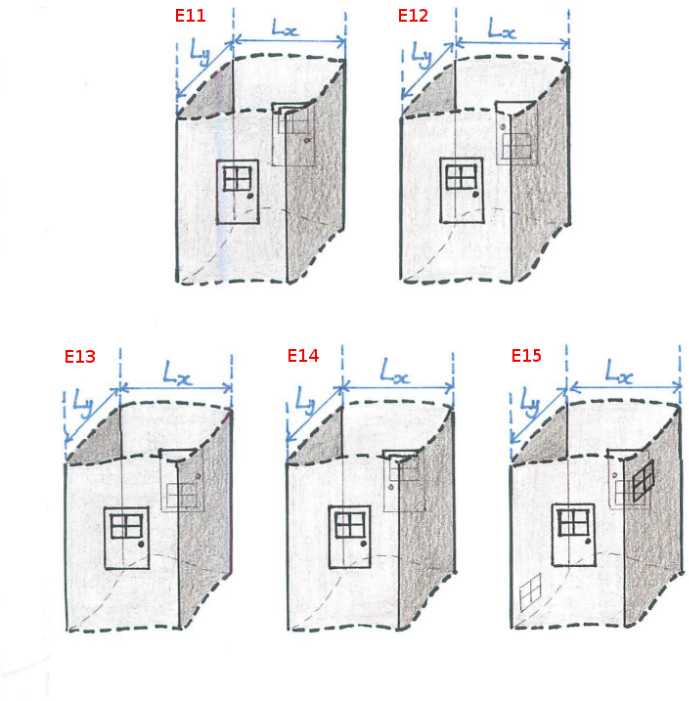


Figure 2.3: The four chimney spaces: Euclidean manifolds which only have two compact dimensions. The same rules apply here regarding the doors as in Fig. (2.2).

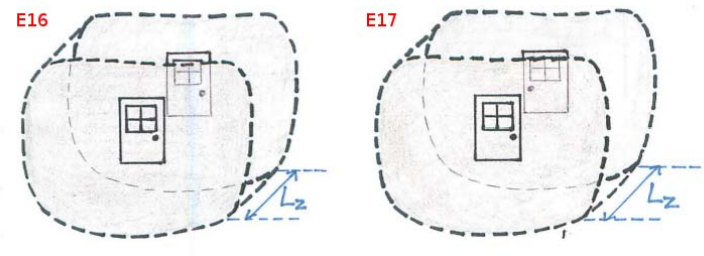


Figure 2.4: The two slab spaces: Euclidean manifolds which only have one compact dimension. The same rules apply here regarding the doors as in Fig. (2.2).

Table 2.1: Table of flat spaces...

Space	$N_{\rm cd}$	**Orientable?**	**Generators**		$\xi^{\hat{k}}_{k\ell m}$
E_1: 3-torus ($\mathbb{T}^3$)	3	Yes	$\mathbf{T}_1 = (L_x, 0, 0)$ $\mathbf{T}_2 = (0, L_y, 0)$ $\mathbf{T}_3 = (0, 0, L_z)$	$\mathcal{M} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \mathbb{I}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x, n_y, n_z \in \mathbb{Z}$
E_2: Half turn space	3	Yes	$\mathbf{T}_{A1} = (L_x, 0, 0)$ $\mathbf{T}_{A2} = (0, L_y, 0)$ $\mathbf{T}_{A3} = (0, 0, L_z)$	$\mathcal{M}_A = \mathbb{I}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x, n_y = 0; n_z \in 2\mathbb{Z}$
			$\mathbf{T}_B = (0, 0, L_z/2)$	$\mathcal{M}_B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\frac{i^\ell}{\sqrt{2}} \sum_{j=0}^{1} (-1)^{jn_z} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B^j)$ for $n_x \in \mathbb{Z}^+; n_y, n_z \in \mathbb{Z}$ or $n_x = 0; n_y \in \mathbb{Z}^+; n_z \in \mathbb{Z}$
E_3: Quarter turn space	3	Yes	$\mathbf{T}_{A1} = (L_x, 0, 0)$ $\mathbf{T}_{A2} = (0, L_y, 0)$ $\mathbf{T}_{A3} = (0, 0, L_z)$	$\mathcal{M}_A = \mathbb{I}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x, n_y = 0; n_z \in 4\mathbb{Z}$
			$\mathbf{T}_B = (0, 0, L_z/4)$	$\mathcal{M}_B = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\frac{i^\ell}{2} \sum_{j=0}^{3} i^{jn_z} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B^j)$ for $n_x \in \mathbb{Z}^+; n_y \in \mathbb{Z}^+ \cup \{0\}; n_z \in \mathbb{Z}$
E_4: Third turn space	3	Yes	$\mathbf{T}_{A1} = (L, 0, 0)$ $\mathbf{T}_{A2} = (-L/2, \sqrt{3}L/2, 0)$ $\mathbf{T}_{A3} = (-L/2, \sqrt{3}L/2, 0)$ $\mathbf{T}_{A4} = (0, 0, L_z)$	$\mathcal{M}_A = \mathbb{I}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_1, n_2 = 0; n_3 \in 3\mathbb{Z}$
			$\mathbf{T}_B = (0, 0, L_z/3)$	$\mathcal{M}_B = \begin{pmatrix} -1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & -1/2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\frac{i^\ell}{\sqrt{3}} \sum_{j=0}^{2} \left(e^{i2\pi/3}\right)^{jn_z} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B^j)$ for $n_1 \in \mathbb{Z}^+; n_2 \in \mathbb{Z}^+ \cup \{0\}; n_3 \in \mathbb{Z}$
E_5: Sixth turn space	3	Yes	$\mathbf{T}_{A1} = (L, 0, 0)$ $\mathbf{T}_{A2} = (-L/2, \sqrt{3}L/2, 0)$ $\mathbf{T}_{A3} = (-L/2, \sqrt{3}L/2, 0)$ $\mathbf{T}_{A4} = (0, 0, L_z)$	$\mathcal{M}_A = \mathbb{I}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_1, n_2 = 0; n_3 \in 6\mathbb{Z}$
			$\mathbf{T}_B = (0, 0, L_z/6)$	$\mathcal{M}_B = \begin{pmatrix} 1/2 & -\sqrt{3}/2 & 0 \\ \sqrt{3}/2 & 1/2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$\frac{i^\ell}{\sqrt{6}} \sum_{j=0}^{5} \left(e^{i2\pi/6}\right)^{jn_z} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B^j)$ for $n_1 \in \mathbb{Z}^+; n_2 \in \mathbb{Z}^+ \cup \{0\} < n_1; n_3 \in \mathbb{Z}$
E_6: Hantzsche-Wendt space	3	Yes	$\mathbf{T}_A = (L_x/2, L_y/2, 0)$	$\mathcal{M}_A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$	$\frac{i^\ell}{\sqrt{2}} \left[Y^*_{\ell m}(\hat{\mathbf{k}}) + Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_X)\right]$ where $X \in \{A, B, C\}$ for $X = A; n_x \in \mathbb{Z}^+; n_y, n_z = 0;$ or $X = B; n_y \in \mathbb{Z}^+; n_x, n_z = 0;$ or $X = C; n_z \in \mathbb{Z}^+; n_x, n_y = 0;$
			$\mathbf{T}_B = (0, L_y/2, L_z/2)$	$\mathcal{M}_B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$	$\frac{i^\ell}{2} \left[Y^*_{\ell m}(\hat{\mathbf{k}}) + \sum_X (-1)^{n_X} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_X)\right]$ where $X \in \{A, B, C\}$; $n_A = n_x - n_y; n_B = n_y - n_z;$ $n_C = n_z - n_x$ for $X = A; n_x, n_y \in \mathbb{Z}^+; n_z \in \mathbb{Z}$ or $X = B; n_y, n_z \in \mathbb{Z}^+; n_x = 0$ or $X = C; n_x, n_z \in \mathbb{Z}^+; n_y = 0$
			$\mathbf{T}_C = (L_x/2, 0, L_z/2)$	$\mathcal{M}_C = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	
E_7: Klein space	3	No	$\mathbf{T}_{A1} = (L_x/2, L_y/2, 0)$ $\mathbf{T}_{A2} = (L_x/2, -L_y/2, 0)$	$\mathcal{M}_A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x \in 2\mathbb{Z}; n_y = 0; n_z \in \mathbb{Z}$
			$\mathbf{T}_B = (0, 0, L_z)$	$\mathcal{M}_B = \mathbb{I}$	$\frac{i^\ell}{\sqrt{2}} \sum_{j=0}^{1} (-1)^{j(n_x+n_y)} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_A^j)$ for $n_x, n_z \in \mathbb{Z}; n_y \in \mathbb{Z}^+$

Continued on next page.

Table 2.1 – continued from previous page.

Space	$N_{\rm cd}$	Orientable?	Generators		$\xi^{\vec{k}}_{k\ell m}$
E_8: Klein space w/ horizontal flip	3	No	$\mathbf{T}_{A1} = (L_x/2, L_y/2, 0)$ $\mathbf{T}_{A2} = (L_x/2, -L_y/2, 0)$	$\mathcal{M}_A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x \in 2\mathbb{Z}; n_y = 0; n_z \in \mathbb{Z}$ $\frac{i^\ell}{\sqrt{2}}\left[Y^*_{\ell m}(\hat{\mathbf{k}}) + (-1)^{n_X} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_X)\right]$ where $X \in \{A, B\}; n_A = n_y; n_B = n_x$ for $X = A; n_x = 0; n_y \in \mathbb{Z}^+; n_z \in 2\mathbb{Z}$ or $X = B; n_x \in 2\mathbb{Z}^+; n_y = 0; , n_z \in \mathbb{Z}$ $\frac{i^\ell}{2}[\, Y^*_{\ell m}(\hat{\mathbf{k}}) + (-1)^{n_x+n_y} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_A) + (-1)^{n_z} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B) + (-1)^{n_x+n_y+n_z} Y^*_{\ell m}(\hat{\mathbf{k}}(\mathcal{M}_A\mathcal{M}_B))\,]$ for $n_x, n_y \in \mathbb{Z}^+; n_z \in \mathbb{Z}$
E_9: Klein space w/ vertical flip	3	No	$\mathbf{T}_1 = (L_x/2, L_y/2, 0)$ $\mathbf{T}_2 = (L_x/2, -L_y/2, 0)$ $\mathbf{T}_3 = (0, 0, L_z/2)$	$\mathcal{M} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x \in 2\mathbb{Z}; n_y = 0; n_z \in 2\mathbb{Z}$ $\frac{i^\ell}{\sqrt{2}}\sum_{j=0}^{1}(-1)^{j(n_x+n_y)} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}^j)$ for $n_x, n_z \in \mathbb{Z}; n_y \in \mathbb{Z}^+;$ $(n_x + n_y)\mathrm{mod}2 \equiv n_z\mathrm{mod}2$
E_{10}: Klein space w/ half turn	3	No	$\mathbf{T}_{A1} = (L_x/2, L_y/2, 0)$ $\mathbf{T}_{A2} = (L_x/2, -L_y/2, 0)$ $\mathbf{T}_B = (0, 0, L_z/2)$	$\mathcal{M}_A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ $\mathcal{M}_B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x, n_y = 0; n_z \in 2\mathbb{Z}$ $\frac{i^\ell}{\sqrt{2}}\left[Y^*_{\ell m}(\hat{\mathbf{k}}) + (-1)^{n_X} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_X)\right]$ where $X \in \{A, B\}; n_A = n_y; n_B = n_x$ for $X = A; n_x = 0; n_y \in \mathbb{Z}^+; n_z \in \mathbb{Z}; n_y\mathrm{mod}2 \equiv n_z\mathrm{mod}2$ or $X = B; n_x \in 2\mathbb{Z}^+; n_y = 0; , n_z \in \mathbb{Z}$ $\frac{i^\ell}{2}[\, Y^*_{\ell m}(\hat{\mathbf{k}}) + (-1)^{n_x+n_y} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_A) + (-1)^{n_z} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B) + (-1)^{n_x+n_y+n_z} Y^*_{\ell m}(\hat{\mathbf{k}}(\mathcal{M}_A\mathcal{M}_B))\,]$ for $n_x = 0; n_y \in \mathbb{Z}^+; n_z \in \mathbb{Z}$
E_{11}: Chimney space	2	Yes	$\mathbf{T}_1 = (L_x, 0, 0)$ $\mathbf{T}_2 = (0, L_y, 0)$	$\mathcal{M} = \mathbb{I}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x, n_y \in \mathbb{Z}; r_z \in \mathbb{R}$
E_{12}: Chimney space w/ half turn	2	Yes	$\mathbf{T}_A = (L_x, 0, 0)$ $\mathbf{T}_B = (0, 0, L_y/2)$	$\mathcal{M}_A = \mathbb{I}$ $\mathcal{M}_B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x, r_z = 0; n_y \in 2\mathbb{Z}$ $\frac{i^\ell}{\sqrt{2}}\sum_{j=0}^{1}(-1)^{jn_y} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B^j)$ for $n_x \in \mathbb{Z}^+; n_y \in \mathbb{Z}; r_z \in \mathbb{R}$ or $n_x = 0; n_y \in \mathbb{Z}; r_z \in \mathbb{R}^+$
E_{13}: Chimney space w/ vertical flip	2	No	$\mathbf{T}_A = (L_x, 0, 0)$ $\mathbf{T}_B = (0, L_y/2, 0)$	$\mathcal{M}_A = \mathbb{I}$ $\mathcal{M}_B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x \in \mathbb{Z}; n_y \in 2\mathbb{Z}; r_z = 0$ $\frac{i^\ell}{2}\sum_{j=0}^{1}(-1)^{jn_y} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B^j)$ for $n_x, n_y \in \mathbb{Z}; r_z \in \mathbb{R}^+$
E_{14}: Chimney space w/ horizontal flip	2	No	$\mathbf{T}_A = (L_x, 0, 0)$ $\mathbf{T}_B = (0, L_y/2, 0)$	$\mathcal{M}_A = \mathbb{I}$ $\mathcal{M}_B = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x = 0; n_y \in 2\mathbb{Z}; r_z = \in \mathbb{R}$ $\frac{i^\ell}{\sqrt{2}}\sum_{j=0}^{1}(-1)^{jn_y} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B^j)$ for $n_x \in \mathbb{Z}^+; n_y \in \mathbb{Z}; r_z \in \mathbb{R}$
E_{15}: Chimney space w/ half turn & flip	2	No	$\mathbf{T}_A = (0, L_y/2, 0)$	$\mathcal{M}_A = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $n_x, r_z = 0; n_y \in 2\mathbb{Z}$ $\frac{i^\ell}{\sqrt{2}}\left[Y^*_{\ell m}(\hat{\mathbf{k}}) + (-1)^{n_y} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_A)\right]$ for $n_x \in 2\mathbb{Z}^+; n_y \in \mathbb{Z}; r_z = 0$

Continued on next page.

Table 2.1 – continued from previous page.

Space	$N_{\rm cd}$	Orientable?	Generators		$\xi^k_{k\ell m}$
			$\mathbf{T}_B = (L_x/2, 0, 0)$	$\mathcal{M}_B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$	$\frac{i^\ell}{\sqrt{2}}\left[Y^*_{\ell m}(\hat{\mathbf{k}}) + Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B)\right]$ for $n_x = 0; n_y \in 2\mathbb{Z}; r_z\mathbb{R}^+$ $\frac{i^\ell}{2}[\ Y^*_{\ell m}(\hat{\mathbf{k}}) + (-1)^{n_y} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_A) + (-1)^{n_x} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}_B) + (-1)^{n_x+n_y} Y^*_{\ell m}(\hat{\mathbf{k}}(\mathcal{M}_A\mathcal{M}_B))\ ]$ for $n_x \in \mathbb{Z}^+; n_y \in \mathbb{Z}; r_z \in \mathbb{R}^+$
E_{16}: Slab space	1	Yes	$\mathbf{T} = (0, 0, L_z)$	$\mathcal{M} = \mathbb{I}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $r_x, r_y \in \mathbb{R}; n_z \in \mathbb{Z}$
E_{17}: Slab space w/ flip	1	No	$\mathbf{T} = (0, 0, L_z/2)$	$\mathcal{M} = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $r_x = 0; r_y \in \mathbb{R}; n_z = \in \mathbb{Z}$ $i^\ell \sum_{j=0}^{1} (-1)^{j n_z} Y^*_{\ell m}(\hat{\mathbf{k}}\mathcal{M}^j)$ for $r_x \in \mathbb{R}^+; r_y \in \mathbb{R}; n_z \in \mathbb{Z}$
E_{18}: Euclidean space ($\mathbb{E}^3$)	0	Yes	N/A		$i^\ell Y^*_{\ell m}(\hat{\mathbf{k}})$ for $r_x, r_y, r_z \in \mathbb{R}$

Part II

The Cosmic Microwave Background (CMB)

Chapter 3

The Impact of the CMB on Cosmology

The Cosmic Microwave Background (CMB) consists of the oldest observable photons in existence, rendering it the most powerful probe of the early Universe. The primordial Universe was so hot and dense that photons experienced multiple scatterings off electrons. As the Universe expanded and cooled, the rate of these interactions decreased until, eventually, the last scatterings occurred; we refer to the region where CMB photons last scattered as the last scattering surface (LSS). There are two effects involved here; decoupling of photons from electrons, and (re)combination of electrons with hydrogen and helium nuclei. Decoupling occurs as a result of the drop in the free electron density due recombination (photon scattering from bound electrons is less efficient than scattering involving free electrons) and the increasing mean-free-path of photons due to the expansion of the Universe. Full recombination occurs when the number of photons with energies greater than or equal to the ionising energy of hydrogen (13.6 eV) is negligible compared to the number of electrons. This takes place when the Universe has cooled to around 3000 K, when the typical photon energy becomes about 0.3 eV. It is thought that this happened about 380,000 years after the Big Bang, equivalent to a cosmological redshift of $z \sim 1100$. The photons were then free to traverse space and ultimately reach us today. Note that recombination was not instantaneous and so the LSS has a finite thickness.

Observations of the CMB have had a dramatic impact on our understanding of the cosmology of our universe:

- The observation of the near-perfect blackbody spectrum of the CMB (Fig. 1.1) is one of the main pillars in *support of the hot Big Bang model*: the coupling between photons and electrons before last scattering ensured the

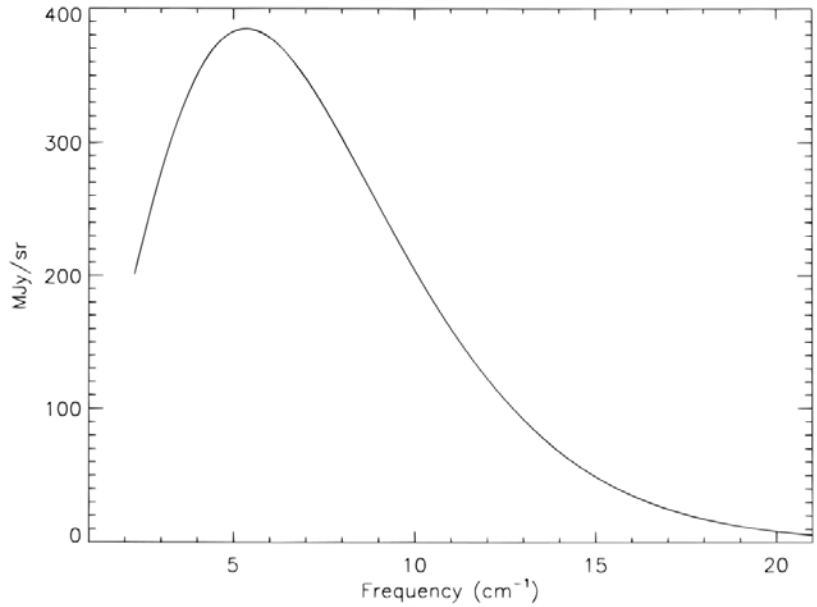


Figure 3.1: Intensity spectrum of the CMB obtained from the FIRAS instrument of the COBE satellite. The curve is a theoretical black body curve. The data is represented by points, which lie inside the curve, and error bars, which are smaller than the thickness of the curve. Credit: Fixsen et al. 1996

thermal equilibrium necessary for a blackbody.

- The isotropic nature of the CMB (as well as other cosmological phenomena) seems to *require an early period of inflation*, during which the expansion rate of the universe increased exponentially, to allow causal contact between diametrically opposing points on the LSS.

- Anisotropies over a wide range of angular scales are evident as fluctuations of order 10^{-5} about a mean temperature of 2.725 ± 0.001 K. Most of the information that we get from the CMB is found by analysing the power spectrum of these anisotropies (see Section 3.1.1 and Section 5.2). Anisotropy measurements are in agreement with the Standard Model (SM) of cosmology and have led to quite precise *constraints on many of the fundamental cosmological parameters*.

The potential wealth of knowledge encoded by the CMB has spurred many experiments, some of which are described in Section 3.1.

3.1 Experiments

The CMB was first predicted in 1948 by Ralph Alpher, Robert Herman, and George Gamow as a consequence of the Big Bang model (see: Alpher & Herman 1948; Gamow 1948). However, it was not observed until 1964 when Arno Penzias and Robert Wilson encountered a mysterious excess noise (Penzias & Wilson 1965) while testing a radio antenna at Holmdel, New Jersey, built to receive signals from the first communication satellites. These Echo balloon satellites were not transceivers but simply reflected microwave signals. This meant that the signals were faint and all interference had to be eliminated; Penzias and Wilson tried to identify all sources of interference, but could not explain the signal that we now know to be the CMB. At that time, Robert Dicke, James Peebles, Peter Roll, and Dave Wilkinson had been devising an experiment to detect the CMB and recognised the cosmological significance of Penzias and Wilson's discovery (Dicke et al. 1965). The discovery gained Penzias and Wilson the 1978 Nobel Prize in Physics. There have since been many balloon and ground based experiments but some of the major contributors to advances in measurements of the CMB are:

(i) COBE (the COsmic microwave Background Explorer) was a satellite launched in 1989 and operated for 4 years. It confirmed the CMB to be blackbody radiation and provided the first measurements of primordial temperature anisotropies. (See Smoot 1999.)

(ii) BOOMERANG (Balloon Observations Of Millimetric Extragalactic Radiation ANd Geophysics) was a balloon experiment that circumnavigated the South Pole for ten and a half days in 1998 and fourteen days in 2003, along with a six-hour test flight in 1997. It allowed the determination of fundametal cosmic parameters to within a few percent (MacTavish et al. 2006). During the 2003 flight, it made measurement of polarisation (Piacentini et al. 2006; Montroy et al. 2006) using bolometers identical to those planned for Planck's HFi instrument.

(iii) DASI (Degree Angular Scale Interferometer) is a ground-based telescope that was set up in 1999/2000 in Antarctica. In 2001, it made the first measurements of CMB polarisation (Kovac et al. 2002), beating BOOMERANG by almost two years.

(iv) WMAP (the Wilkinson Microwave Anisotropy Probe) was a satellite launched in 2001 and finished its survey of the CMB in 2010. It achieved levels of sensi-

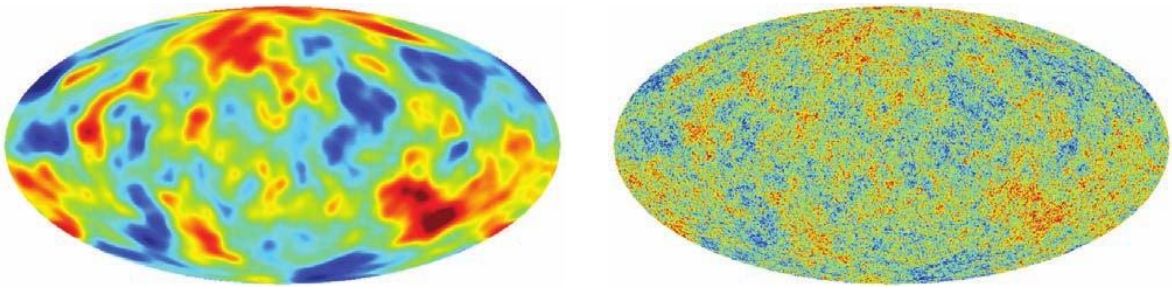


Figure 3.2: Temperature anisotropies simulated over the whole sky at COBE (left) and Planck (right) resolutions. Credit: Planck Collaboration 2005.

tivity and angular resolution that allowed constraints to be placed on cosmological parameters at a precision of a few per cent. It also made the first measurements of polarisation at large angular scales (DASI and BOOMERANG did not survey the entire sky and so did not gather information on large angular scales). (See, e.g., Komatsu et al. 2009, Bennett et al. 2012, Hinshaw et al. 2012.)

(v) Planck is a satellite that was launched in 2009 and first released cosmological data in March 2013. It has been designed not only to achieve the highest levels of sensitivity and resolution to date (with the aim of determining cosmological parameters to the unprecedented precision of better than a percent) but to measure the polarisation of the CMB in detail. As of yet, no polarisation data have been released, but cosmological parameters have been determined to percent-level precision. (See, e.g., Planck Collaboration 2005, Planck Collaboration et al. 2013a, Planck Collaboration et al. 2013b.)

The marked improvement in resolution from COBE to Planck is shown in the temperature anisotropy maps of Fig. (3.2). It is harder to see the difference in resolution between Planck and WMAP without zooming in on the maps, as in Fig. (3.3). Fig. (3.4) shows an anisotropy map without foregrounds subtracted and is the first all-sky image from Planck. The removal of foregrounds poses something of a challenge and WMAP and Planck go some way to resolving this problem by observing over a large range of frequencies: the knowledge that different foreground sources are present in different frequency bands can be utilised to eliminate their signals (see Fig. (3.5)). Since we are still awaiting the release of Planck polarisation data, this thesis will use data from WMAP.

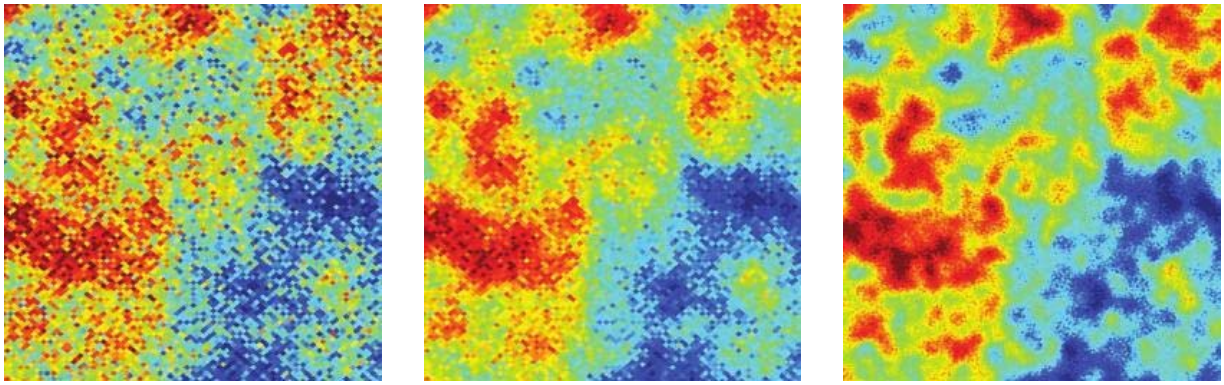


Figure 3.3: $5° \times 5°$ patch of sky simulated at WMAP (94 GHz, 15 FWHM) and Planck (217 GHz, 5 FWHM) resolutions. Left - WMAP 2 years. Centre - WMAP 8 years. Right - Planck 1 year. Credit: Planck Collaboration 2005.

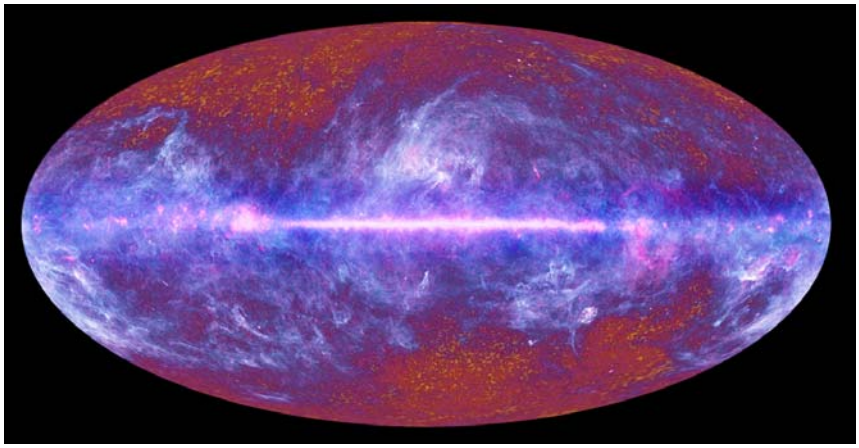


Figure 3.4: Actual Planck one-year all-sky survey of the microwave background (without foregrounds subtracted). Image shows data spanning the full frequency range of Planck (30 to 857 GHz). Credit: ESA, HFI and LFI consortia.

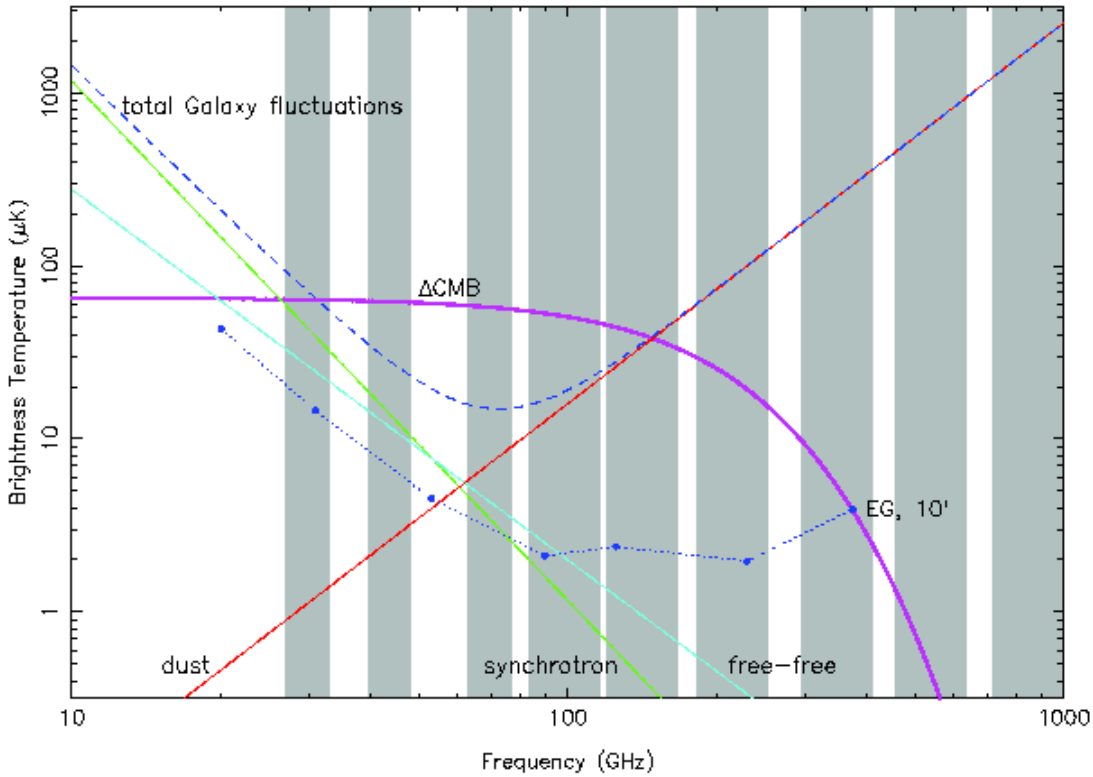


Figure 3.5: Spectrum of the CMB and other sources of fluctuations in the microwave sky across the frequency channels (grey columns) of Planck. The total Galactic fluctuation levels depend on angular scale, and are shown for about 1°. Extra-galactic sources dominate smaller scales (EG, 10' shown on diagram). The highest frequencies are primarily sensitive to dust. Credit: Planck Collaboration 2005.

Table 3.1: Best fit cosmological parameters for ΛCDM from WMAP 9-year data and Planck data (see Hinshaw et al. 2012 and Planck Collaboration et al. 2013b respectively). The uncertainties given represent the 68% confidence intervals of the values. In the case of the WMAP fit, the densities listed are three of six model parameters that have been optimised in order to fit the model to the data; H_0 has been derived from the other parameters once the best fit has been achieved. In the Planck case, only the physical densities are among the six fitted model parameters; Ω_Λ and H_0 are both derived parameters. As usual, h is the Hubble parameter (defined as $H_0 = 100h$).

Parameter	**Symbol**	**WMAP 9-year**	**Planck**
Physical baryon density	$\Omega_b h^2$	0.0222 ± 0.0003	0.0221 ± 0.0002
Physical dark matter density	$\Omega_c h^2$	0.1153 ± 0.0019	0.1187 ± 0.0017
Dark energy density	Ω_Λ	0.714 ± 0.010	0.692 ± 0.010
Hubble const. ($\mathrm{km.s^{-1}.Mpc^{-1}}$)	H_0	69.32 ± 0.80	67.80 ± 0.77

3.1.1 How Well Does the Standard Model Fit the Data?

The SM of cosmology is the Lambda Cold Dark Matter (ΛCDM) model. This incorporates vacuum energy (the "Λ") and cold dark matter (which, unlike hot or warm dark matter, does not interact in any way, other than gravitationally, with radiation) into a Big Bang scenario. There are various ways of parameterising the model, although there can be no fewer than six parameters. Some of the parameters involved are displayed in Table 3.1, where the best fit values are given for both WMAP 9-year and Planck data.

Spatial curvature, k, depends on H_0 and total density through the Friedmann equation. The best fit curvature density for WMAP 9-year is $\Omega_k = -0.0027^{+0.0039}_{-0.0038}$ (68% confidence level, Hinshaw et al. 2012,), while for Planck $\Omega_k = -0.0010^{+0.0062}_{-0.0065}$ (95% confidence level, Planck Collaboration et al. 2013b). Since the data are consistent with a flat (Ω_k=0), or nearly flat, universe, the analysis in this thesis will concentrate on flat spaces. While open spaces are not investigated in this thesis, they would provide an interesting extension.

Both the WMAP and Planck data agree well with a flat ΛCDM model on small angular scales (large multipoles, ℓ), but not quite so well at large angular scales (small ℓ). This is demonstrated in Fig. (3.6). One possible explanation for this discrepancy is that the model implicitly uses a simply-connected topology; incorporating a multi-connected topology may solve the problem. However, to date, such attempts have not been particularly successful (e.g. Niarchou 2006, Uzan et al. 2004).

The level of deficit between the data and model depends on the method of analysis used. For example, it diminished between the WMAP 1-year results and

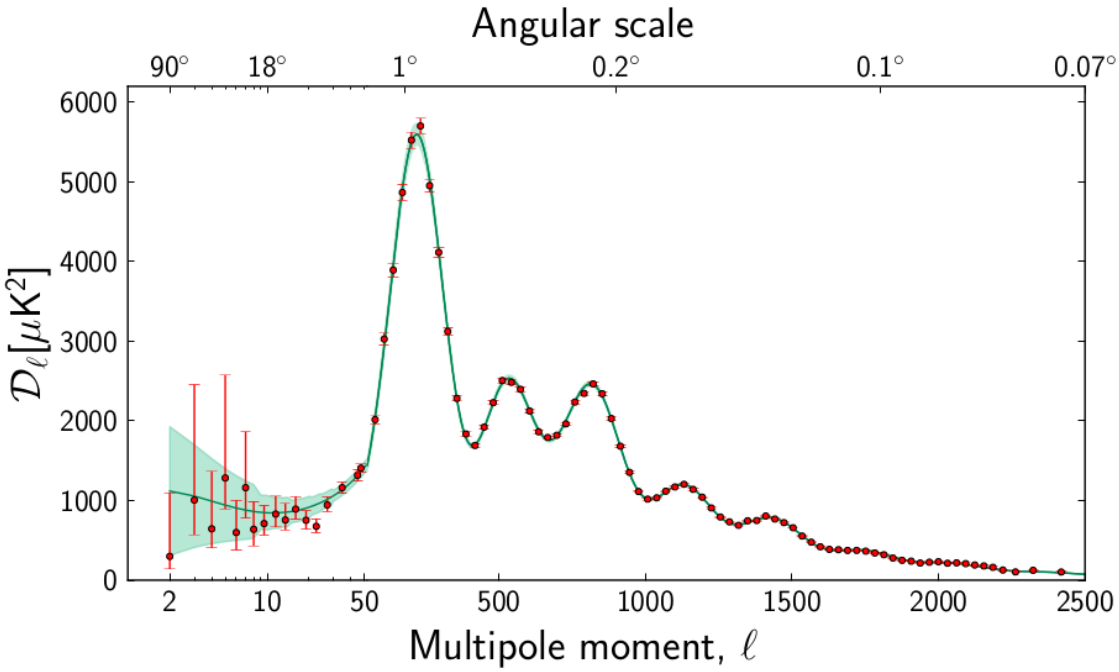


Figure 3.6: Planck temperature angular power spectrum. The red data points have error bars that include both noise and cosmic variance. The green line is the best-fit spectrum for a flat ΛCDM cosmology. The green region represents cosmic variance only (no noise). The fit is extremely tight above an ℓ of around 40. Below this value, data tend to sit underneath the model line; this deficit is between 5-10%. Credit: Planck collaboration et al. 2013c.

WMAP 3-year results, not because of the additional data, but because a new method proposed by George Efstathiou (Efstathiou 2004) was applied to the 3-year data; Efstathiou argued that the low power could arise from uncertainties due to masking foregrounds, or biases inherent in the frequentist statistics used. Some now argue that the significance of the deficit is not strong enough for it to be forced into the model (Crittenden 2004). However, Planck collaboration et al. (2013c) reported a deficit of $5-10\%$ below $\ell \sim 40$ and work continues to to try explain this deficit (e.g. Hearin et al. 2011; Bunn & Bourdon 2008; Shankaranarayanan & Sriramkumar 2005), including whether it could be a result of a multi-connected topology (e.g. Aurich et al. 2006).

Chapter 4

Physics of the CMB

The CMB is produced by one of the cleanest astrophysical systems known: interactions between photons and electrons (bound in hydrogen and helium) in weak gravitational fields. This means that predictions of CMB properties can be calculated accurately and reliably. If effects that have altered CMB radiation since it left the LSS, as well as foregrounds, can be successfully removed, CMB anisotropies can provide valuable information about the early universe, large-scale structure formation, and the cosmological parameters (e.g.: Peebles & Yu 1970; Peebles 1981; Jungman et al. 1996b,a).

4.1 Temperature Anisotropies

Foregrounds are more easily removed from temperature data than from polarisation data and the vast majority of the information acquired to date has been deduced from temperature anisotropies.

4.1.1 Origins of CMB Temperature Anisotropies

The evolution of CMB perturbations (Dodelson 2003, Scott & Smoot 2008) can be divided into two phases: before and after the epoch of recombination (when electrons combined with nucleons to form atoms and ceased scattering photons). Before this time, the tightly coupled photons and electrons could be described as a single ("baryon-photon") fluid. Effects that contribute to perturbations before and on the LSS (which all arise from primordial perturbations and so are correlated) include:

- Fluctuations in density across the LSS: these provide fluctuations in temperature - more photons corresponds to more energy.

- Sachs-Wolfe (SW) effect: the gravitational potential at the LSS affects photon energies. Variations in temperature due to the SW effect are given by $\Delta T = \phi/3c^2$, where ϕ is gravitational potential and c is the speed of light.

- Doppler effect: variations in the line-of-sight velocity of the baryon fluid (and, hence, electrons) across the LSS lead to variations in the observed energy of the photons scattered by the electrons.

- Damping at small angular scales due to the finite thickness of the LSS: the finite period of time (as opposed to instantaneous) over which last scattering occurs leads to deconstructive interference between photons emerging at different times, weakening the signal of anisotropies at smaller angular scales.

Effects that occur as photons propagate from the LSS to Earth include:

- Reionisation: an epoch of reionisation occurred after recombination when the universe was still sufficiently dense for a significant amount (about 10%) of CMB photons to be rescattered by free electrons. This event is believed to have occurred at a redshift between 20 and 5 (1.5 Myr to 1 Gyr after the Big Bang), but the exact time is unknown and currently a hot topic for research.

- Integrated Sachs-Wolfe (ISW) effect: the effect of variations in gravitational potential integrated between the LSS and Earth. Variations in temperature due to the ISW effect are given by $\Delta T = 2\Delta\phi/c^2$, where $\Delta\phi$ is the change in gravitational potential while a photon traverses the potential well.

Fig. (4.1) demonstrates how these effects appear in the angular power spectrum.

The former set of effects characterise the primordial power spectrum $P(k)$, where k denotes wavenumber, or Fourier mode. The latter effects contribute to the radiation transfer functions $\Delta_\ell^T(k)$, where ℓ is a multipole mode (see Bond & Efstathiou 1984). As we shall see, polarisation is susceptible to different effects to temperature, and has its own set of transfer functions, $\Delta_\ell^E(k)$ and $\Delta_\ell^B(k)$. Both temperature and polarisation transfer functions can be found using the publicly available code such as CAMB (Lewis & Bridle 2002) or CMBFAST (Zaldarriaga & Seljak 2000).

4.2 Polarisation

Polarisation data are not only complimentary to temperature data in anisotropy studies but, as they provide information that cannot be obtained from temperature,

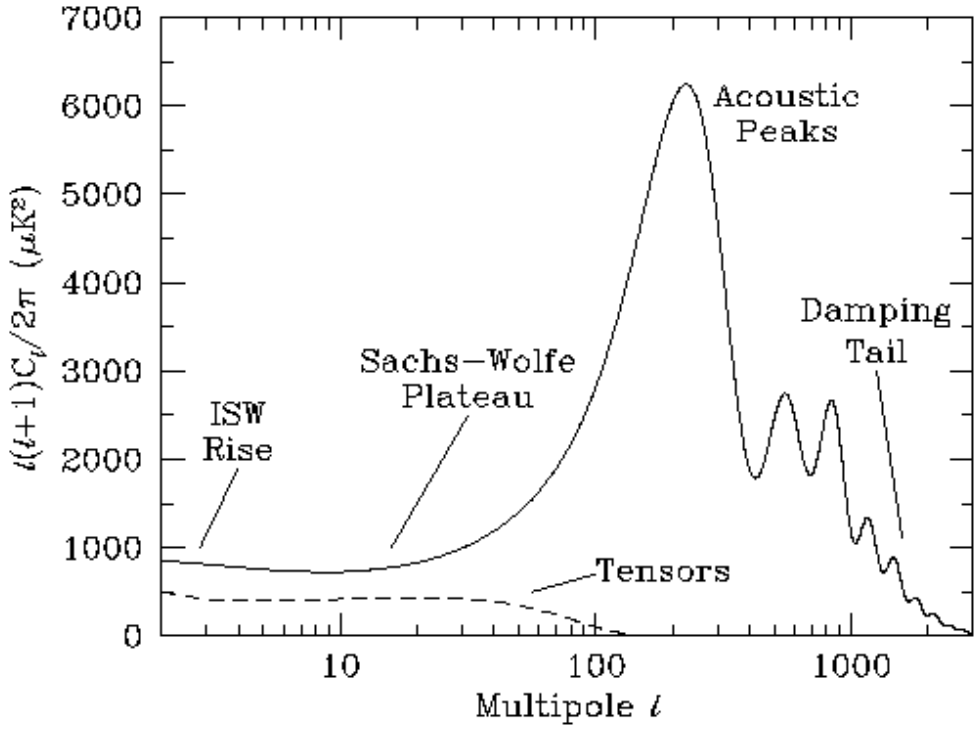


Figure 4.1: This is a theoretical CMB anisotropy power spectrum generated using the code CMBFAST (www.cmbfast.org). The multipole ℓ represents angular scale (higher multipoles for smaller angular scales). The vertical axis is the mean square temperature fluctuation at the scale represented by ℓ. Credit: Scott & Smoot 2008.

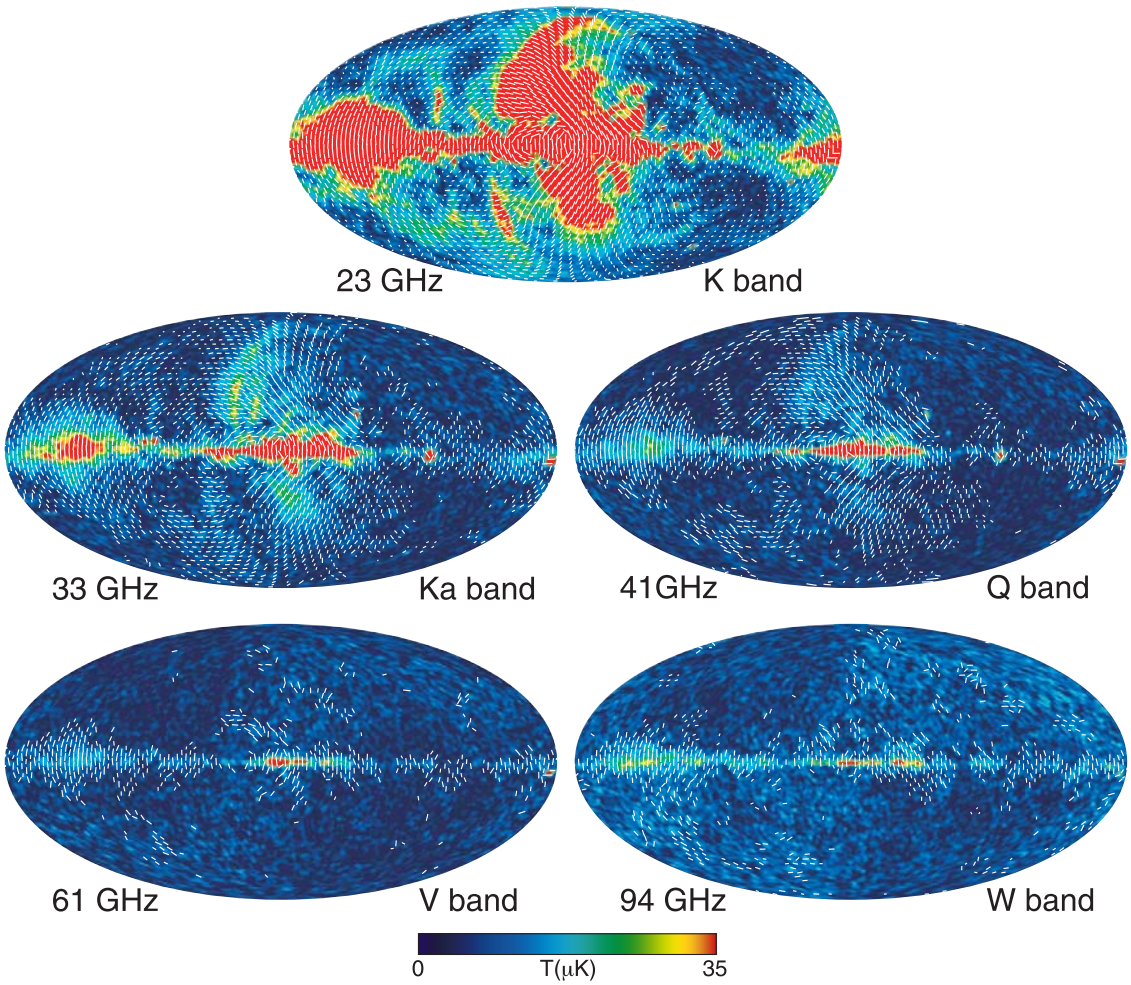


Figure 4.2: WMAP 9-year polarisation maps for different frequency bands. The coloured background shows the intensity, while the white lines indicate the angle. Credit: Bennett et al. 2012.

are necessary to achieve precision cosmology (Zaldarriaga et al. 1997; Eisenstein et al. 1998; Zaldarriaga & Harari 1995; Kaplan et al. 2003). WMAP provided the first glimpse of the full-sky polarised CMB (Kogut et al. 2003), as shown in Fig. (4.2), and Planck promises more detailed observations (see Fig. (4.3)).

This section describes the mechanisms for producing a polarisation signal in the CMB and how to characterise this signal mathematically. The next chapter, Ch. 5, will develop the mathematical description further and show how to find polarisation correlation matrices. Good introductions to CMB polarisation are given by Hu & White (1997) and Kosowsky (1999), while detailed discussions on how to find the transfer functions and correlation matrices for polarisation can be found in Kosowsky (1996), Kamionkowski et al. (1997) and Zaldarriaga & Seljak (1997).

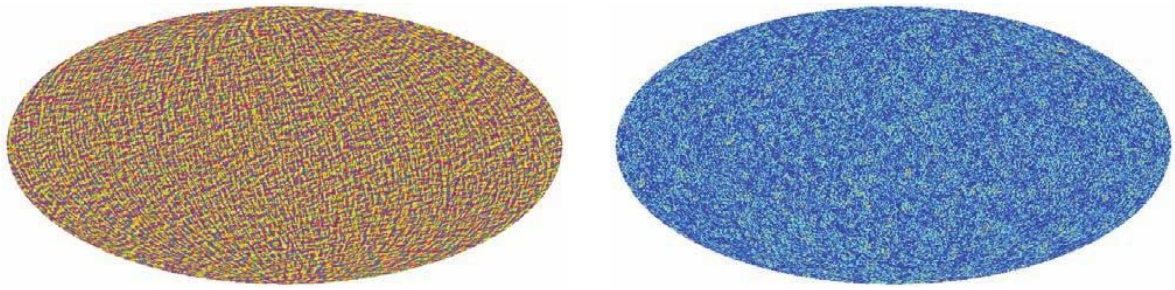


Figure 4.3: Simulated direction (left) and amplitude (right) of polarisation anisotropies at Planck resolution for a pure scalar fluctuation mode. Credit: Planck Collaboration 2005.

4.2.1 Origins of CMB Polarisation

CMB polarisation is linear (i.e. the preferred axis of oscillation in the electromagnetic field of a CMB photon does not alter direction, or rotate, with time). Fig. (4.4) shows how this polarisation emerges from quadrupoles in the energy density of the photon fluid at the LSS. Fig. (4.5) demonstrates how quadrupoles are formed in the CMB.

More photons travel to a given point from hot, overdense regions than from cool, underdense regions. In a quadrupole induced by scalar perturbations, photons from the hottest regions travel to a central point in a direction perpendicular to that of photons from the coldest regions. The contribution to the polarisation of photons that are Thomson-scattered at this point is greater from the hotter regions than the colder regions. Therefore the polarisation of the scattered photons has a greater amplitude in the direction aligned with the hottest regions. The amount of polarisation measured depends on the orientation of the observer to the quadrupole. This process simply requires the energy density to vary over the LSS i.e. a scalar mode of perturbation, the leftmost case of Fig. (4.5). This can give rise to two types of polarisation pattern, both of which have even parity (are unchanged under reflection) and are known as E modes (see left two images of Fig. (4.6)).

Polarisation can also arise due to vector perturbations. In this case, the bulk motion in a region of constant energy density produces a Doppler shift in the energies of the photons, creating an effective dipole. Therefore, a dipole in the energy density of the photon fluid can become an effective quadrupole when there are bulk motions in the fluid which follow opposite directions in under- and over- dense regions (see central image of Fig. (4.5)). The right two images in Fig. (4.6) show B mode polarisation, which has odd parity (sign changes upon reflection) and is the dominant mode produced by vector perturbations.

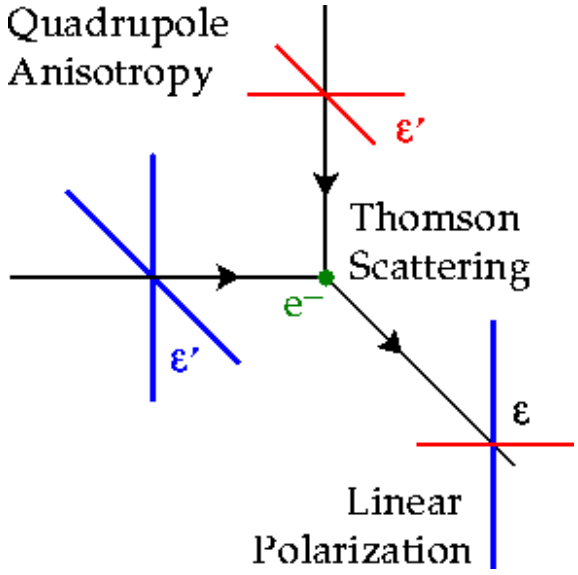


Figure 4.4: How a quadrupole in the photon fluid induces polarised scattered light. The green dot in the centre is an electron. Black lines are directions of propagation of photons (two incident and one scattered are shown, but photons will travel to and from the electron in all directions). The quadrupole lies in the plane of the page. The axes marked ϵ' are incident polarisation, while ϵ denotes the polarisation of photons scattered orthogonal to the plane of the quadrupole. Incident polarisation components orthogonal to the scattered direction are transferred to the scattered photons; incident components that are parallel to this direction are not transferred. Photons from the hot, overdense (blue) region are more abundant than those from the cool, underdense (red) region and so contribute more to the scattered polarisation. Hence non-polarised incident light can result in polarised scattered light. Credit: Hu & White 1997.

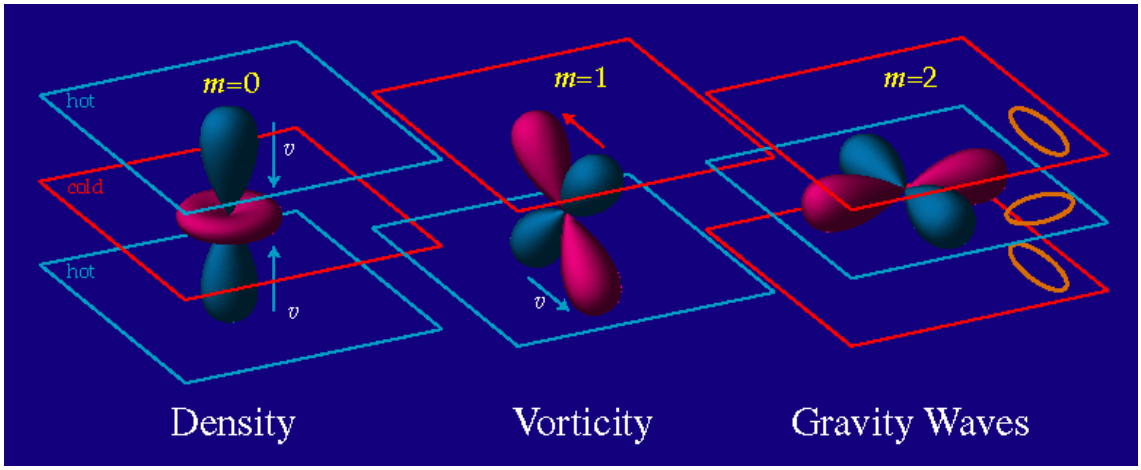


Figure 4.5: Different types of quadrupole on the LSS. The blue and red sheets represent the background of overdense (hot) and underdense (cold) regions respectively (scalar perturbations). Left: scalar perturbation - simple quadrupole in energy density. Centre: vector perturbation - opposing bulk motions in two adjacent isothermal regions (one hot, one cold) create an effective quadrupole. Right: tensor perturbations - gravity waves distort space in the plane of the perturbation, changing circles of particles into ellipses and, hence, isothermal regions into quadrupoles. Credit: Hu & White 1997.

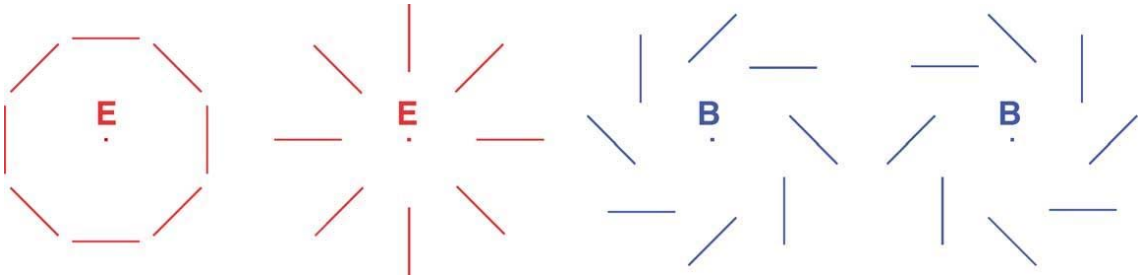


Figure 4.6: Polarisation patterns for different types of modes. Left: positive E mode, centred on a hot spot. Centre left: negative E-mode, centred on a cold spot. Centre right: negative B mode, centred on a hot spot. Right: positive B-mode, centred on a cold spot. Credit: Kaplan et al. 2003.

Finally, tensor perturbations can also produce polarisation. These perturbations are produced by gravity waves (see rightmost image of Fig. (4.5)). Gravity waves distort space in the plane of the perturbation such that a circle of particles becomes an ellipse. The density is increased along the minor axis of the ellipse and decreased along the major axis. In an isothermal (constant energy density) region, this induces a quadrupole. Different polarisation modes result from different gravitational wave formations; tensor perturbations produce comparable amounts of E modes and B modes.

To see why the different types of perturbation lead to different modes of polarisation, refer to Fig. (4.5). In a scalar perturbation, photons move parallel to the major axis of the hot lobe towards the cold lobe, the polarisation direction is therefore perpendicular to this axis (and parallel to the diametral axis of the cold lobe). In a vector perturbation, however, photons move from hot lobes to cold lobes at 45° to the major axes of the lobes, the polarisation direction also 45° to these axes. Tensor perturbations can produce polarisation both perpendicular (or parallel) and 45° to the major axes of the quadrupoles, depending on the shape of the quadrupole induced.

Which type of mode dominates, if any, depends on which type of perturbation dominates: scalar, vector or tensor. Vector perturbations are expected to be insignificant as they arise from velocities in the baryon fluid; the greater the physical size of the quadrupole, the larger the velocity required, meaning there is a limit to the size of such a quadrupole. The expansion of the Universe since the formation of the LSS is expected to have rendered these quadrupoles too small to measure the effects of in the CMB. This means that the B mode contribution should arise almost exclusively from tensor perturbations. The magnitude of tensor perturbations is limited by the energy scale of inflation and expected to be much smaller than the magnitude of scalar perturbations. Since E modes are produced by both scalar and tensor perturbations, they should dominate over B modes.

4.2.2 Relating E and B Modes to Stokes' Parameters

E and B are related to Stokes' parameters Q and U. Q and U describe polarisation with respect to a local coordinate system, usually aligned with the observer's line of sight, which changes according to where in the sky the observer is looking. E and B however, are independent of choice of coordinate system and are more convenient for studying global properties of CMB polarisation.

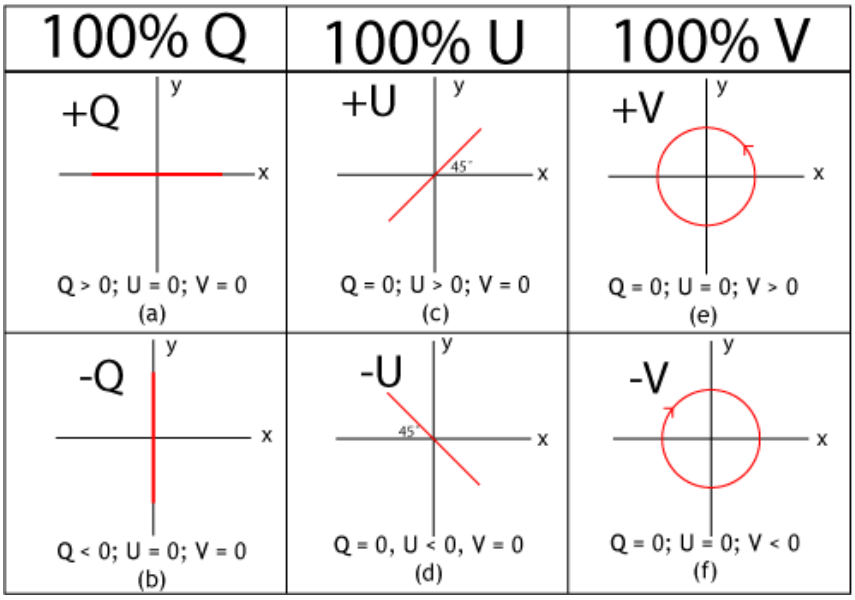


Figure 4.7: Stokes' polarisation parameters. The x and y axes lie in the plane perpendicular to the direction of propagation of a wave. The magnitudes of Q, U and V indicate the amount of each type of polarisation present. Their sign is used to indicate one of two directions. Each of these plots depicts a different extreme case, where only one type of polarisation is present. Credit: Dan Moulton.

Stokes' parameters completely describe the polarisation characteristics of light (e.g. Chandrasekhar 1950). I represents intensity, Q linear polarisation in two orthogonal directions, U linear polarisation in two orthogonal directions rotated 45 degrees with respect to Q, V circular polarisation (see Fig. (4.7)). Eq. (4.1) is Stokes' theorem.

$$\boxed{I^2 \geq Q^2 + U^2 + V^2 \quad \textbf{Stokes' theorem}} \tag{4.1}$$

A monochromatic wave can be described as an electric field with the components

$$\begin{aligned} E_x &= a_x(t)cos[\nu t - \theta_x] \\ E_y &= a_y(t)cos[\nu t - \theta_y] \end{aligned} \tag{4.2}$$

where a_x and a_y are amplitudes in the x- and y- directions, ν is the frequency of the light, and θ_x and θ_y the phase angles. Light is polarised if the two components E_x and E_y are correlated in some way. Stokes' parameters are defined with respect to these parameters as

$$I \equiv \langle a_x^2 \rangle + \langle a_y^2 \rangle \tag{4.3}$$

$$Q \equiv \langle a_x^2 \rangle - \langle a_y^2 \rangle \tag{4.4}$$

$$U \equiv \langle 2a_x a_y cos(\theta_x - \theta_y) \rangle \tag{4.5}$$

$$V \equiv \langle 2a_x a_y sin(\theta_x - \theta_y) \rangle \tag{4.6}$$

where the angle brackets represent time averages. Note that circularly polarised light in the CMB is often considered negligible and so V vanishes. There are mechanisms for producing it, in the presence of magnetic fields for example, and attempts to detect circular polarisation in the CMB were made in the 1980s (Stark 1981; Lubin et al. 1983; Tolman & Matzner 1984). An upper limit to the degree of circular polarisation in the CMB at large angular scales was determined to be about 10^{-3}, although it is expected to be about 10^{-9}. Comparing this to temperature anisotropies, which are of order 10^{-5} and E-mode polarisation, at 10^{-7}, circular polarisation would be much harder to detect. However, as instrumental sensitivies improve, interest in circular polarisation has started to revive (Zarei et al. 2010; Mainini et al. 2013); some argue that the expected amplitude is not very different to that expected for B-mode polarisation and, since experiments for detecting B-modes are currently under way, we should be considering circular polarsiation too. B-modes will be considered, at least initially, in this thesis as there are experiments that may detect them in the near future. With less activity surrounding circular polarisation, it will not be dealt with here.

Q and U depend on the orientation of the local coordinate system. The local z-axis, by convention, lies along the axis of propagation of the wave and, hence, along the line of sight of the observer. (Here, the z-axis is taken to be directed out from the observer, but some texts take it to be towards the observer.) The x-y plane is therefore tangential to the celestial sphere, but the orientation of the x-y plane is not constrained. A wave that can be described by Q and U in one particular coordinate system, can also be described by Q' and U' in a system where the x-y plan has been rotated by ϕ degrees:

$$\begin{aligned} Q' &= Qcos(2\phi) + Usin(2\phi) \\ U' &= -Qsin(2\phi) + Ucos(2\phi) \end{aligned} \tag{4.7}$$

If ϕ is chosen to be 45 degrees, Q' is in fact equivalent to U, and U' equivalent to $-Q$.

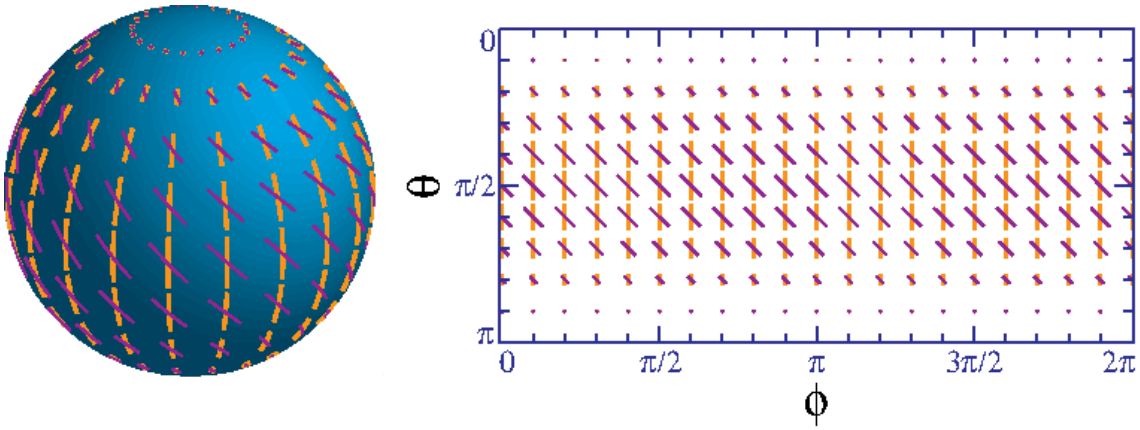


Figure 4.8: Positive Q (yellow lines) and U (purple lines) plotted on the celestial sphere (left) and map of the sky (right). Negative Q and U are not shown but are simply perpendicular to their positive counterparts. The local x- and y- axes for the Stokes' parameters have been chosen to be longitudinal and latitudinal with respect to the global z-axis (running from the bottom to the top of the sphere). This means than the net effect of the presence of Q polarisation is to create only E modes (and similar for U and B). Credit: Hu & White 1997 (note that this figure is used to explain a different situation in the source paper).

Unlike Q and U, E and B cannot be transformed into one another (this should be apparent upon inspection of Fig. (4.6)). This is because they are a global, and not a local, property. In order to properly understand the connection between these two ways of describing polarisation, we need to identify the differences between them. These differences are not often emphasised, but are quite fundamental:

- A wave emerging from a single point in the sky can have polarisations Q and/or U. It cannot, however, have E or B mode polarisation.
- To identify an E or B mode, we must compare multiple points in the sky, searching for patterns such as those in Fig. (4.6).

E and B can be defined in terms of spherical harmonics (see Section 5.1), a global coordinate system. It is convenient to choose the local (Q and U) x- and y-axes to be longitudinal and latitudinal, respectively, with regard to the global (E and B), spherical harmonic, z-axis. This choice of coordinate system results in E being dependent only on Q, and B only on U, as in Fig. (4.8).

In general, the polarisation tensor is defined as

$$\boxed{\frac{\mathcal{P}_{ab}(\hat{\mathbf{n}})}{T_0} = \frac{\mathcal{P}^E_{ab}(\hat{\mathbf{n}})}{T_0} + \frac{\mathcal{P}^B_{ab}(\hat{\mathbf{n}})}{T_0} = \frac{1}{2T_0}\begin{pmatrix} Q(\hat{\mathbf{n}}) & -U(\hat{\mathbf{n}})\,sin(\theta) \\ -U(\hat{\mathbf{n}})\,sin(\theta) & -Q(\hat{\mathbf{n}})\,sin^2(\theta) \end{pmatrix} \quad \textbf{polarisation tensor}} \tag{4.8}$$

where T_0 is the mean CMB temperature. With the coordinate systems chosen as above, $\mathcal{P}^E_{ab}(\hat{\mathbf{n}})$ is a diagonal tensor, while $\mathcal{P}^B_{ab}(\hat{\mathbf{n}})$ is a symmetric, off-diagonal tensor, i.e.

$$\frac{\mathcal{P}^E_{ab}(\hat{\mathbf{n}})}{T_0} = \frac{1}{2T_0}\begin{pmatrix} Q(\hat{\mathbf{n}}) & 0 \\ 0 & -Q(\hat{\mathbf{n}})\,sin^2(\theta) \end{pmatrix}. \tag{4.9}$$

$$\frac{\mathcal{P}^B_{ab}(\hat{\mathbf{n}})}{T_0} = \frac{1}{2T_0}\begin{pmatrix} 0 & -U(\hat{\mathbf{n}})\,sin(\theta) \\ -U(\hat{\mathbf{n}})\,sin(\theta) & 0 \end{pmatrix}. \tag{4.10}$$

Note that $\mathcal{P}^E_{ab}(\hat{\mathbf{n}})$ and $\mathcal{P}^B_{ab}(\hat{\mathbf{n}})$ are **not** the E and B modes themselves; they are the components of the polarisation tensor that we use to test for E and B modes. E modes are present on a certain scale, ℓ, if there is a net correlation between all $\mathcal{P}^E_{ab}(\hat{\mathbf{n}})$ and $\mathcal{P}^E_{ab}\left(\hat{\mathbf{n}}'\right)$ on that scale (i.e., if C^{EE}_ℓ is non-zero - see Eq. (5.17)), and similar for B.

Chapter 5

Mapping the CMB

This chapter explores how we can extract information about the CMB from temperature and polarisation maps, $T(\hat{\mathbf{n}})$ and $\mathcal{P}_{ab}(\hat{\mathbf{n}})$ respectively (where $\hat{\mathbf{n}}$ is the unit vector denoting position on the sky and a, b are matrix indices), as well as the challenges involved in collecting the data that form these maps. $T(\hat{\mathbf{n}})$ is a scalar field and $\mathcal{P}_{ab}(\hat{\mathbf{n}})$ is a tensor field. So, while the treatment of polarisation is analogous to temperature, it is somewhat more complex (e.g. Kamionkowski et al. 1997).

5.1 Harmonic Expansion

$T(\hat{\mathbf{n}})$ and $\mathcal{P}_{ab}(\hat{\mathbf{n}})$ can both be expanded in terms of complete sets of orthonormal basis functions, as in Eq. (5.1) and Eq. (5.2).

$$\frac{T(\hat{\mathbf{n}})}{T_0} = 1 + \sum_{\ell=1}^{\infty} \sum_{m=-\ell}^{\ell} a^T_{\ell m} Y_{\ell m}(\hat{\mathbf{n}}) \tag{5.1}$$

$$\frac{\mathcal{P}_{ab}(\hat{\mathbf{n}})}{T_0} = \sum_{\ell=2}^{\infty} \sum_{m=-\ell}^{\ell} \left[a^E_{\ell m} Y^E_{(\ell m)ab}(\hat{\mathbf{n}}) + a^B_{\ell m} Y^B_{(\ell m)ab}(\hat{\mathbf{n}}) \right] \tag{5.2}$$

where

- T_0 is the mean CMB temperature and, as well as expressing T in units T_0, it is convention to express $\mathcal{P}_{ab}$ in units of T_0 as the polarisation is correlated to the temperature anisotropies.
- $Y_{\ell m}$ are the spherical harmonics and the basis functions $Y^E_{(\ell m)ab}$ and $Y^B_{(\ell m)ab}$ can be expressed in terms of the spherical harmonics by Eq. (5.3) and Eq. (5.4). Note that the harmonics of $\mathcal{P}_{ab}$ do not exist for $\ell = 1$ (because this provides only a local x- or local y- axis, but both are needed to describe polarisation).

While often ignored due to being contaminated by the kinematic dipole (an apparent dipole in the CMB caused by Earth's motion and the Doppler effect), the $\ell = 1$ term does exist for T.

$$Y^E_{(\ell m)ab} = N_\ell \left(Y_{(\ell m):ab} - \frac{1}{2} g_{ab} Y_{(\ell m):c}{}^{c} \right) \tag{5.3}$$

$$Y^B_{(\ell m)ab} = \frac{N_\ell}{2} \left(Y_{(\ell m):ac}\, \epsilon^c{}_b + Y_{(\ell m):bc}\, \epsilon^c{}_a \right) \tag{5.4}$$

where : denotes covariant differentiation on the two-sphere, g_{ab} is the metric tensor on the two-sphere, ϵ_{ab} is the completely antisymmetric unit tensor, defined in Eq. (5.5), and N_ℓ is a normalisation factor given by Eq. (5.6).

$$\epsilon_{ab} = \sqrt{g} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, \quad \epsilon^a{}_b = \frac{1}{\sqrt{g}} \begin{pmatrix} g_{12} & g_{22} \\ -g_{11} & -g_{21} \end{pmatrix}, \quad \epsilon^{ab} = \frac{1}{\sqrt{g}} \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \tag{5.5}$$

where $g \equiv ||g_{ab}||$.

$$N_\ell \equiv \sqrt{\frac{2\,(\ell - 2)!}{(\ell + 2)!}} \tag{5.6}$$

Multipoles, ℓ, express the angular scale, with smaller scales represented by larger multipoles. However, there is no one-to-one conversion between multipole and angular scale, although a single spherical harmonic $Y_{\ell m}$ corresponds to angular variations of $\theta \sim \pi/\ell$.

- $a^X_{\ell m}$ are the *multipole moments*: $a^T_{\ell m}$ are the temperature multipole coefficients; $a^E_{\ell m}$ and $a^B_{\ell m}$ are the polarisation expansion coefficients. Rearranging Eq. (5.1) and Eq. (5.2) leads to the following expressions for the multipole moments:

$$\boxed{a^T_{\ell m} = \frac{1}{T_0} \int d\hat{\mathbf{n}}\, \Delta T(\hat{\mathbf{n}})\, Y^*_{\ell m}(\hat{\mathbf{n}}) \qquad T \textbf{ multipole moments}} \tag{5.7}$$

$$\boxed{a^E_{\ell m} = \frac{1}{T_0} \int d\hat{\mathbf{n}}\, \mathcal{P}_{ab}(\hat{\mathbf{n}})\, Y^{E\,ab\,*}_{\ell m}(\hat{\mathbf{n}}) \qquad E \textbf{ multipole moments}} \tag{5.8}$$

$$\boxed{a^B_{\ell m} = \frac{1}{T_0} \int d\hat{\mathbf{n}}\, \mathcal{P}_{ab}(\hat{\mathbf{n}})\, Y^{B\,ab\,*}_{\ell m}(\hat{\mathbf{n}}) \qquad B \textbf{ multipole moments}} \tag{5.9}$$

where $\Delta T = T - T_0$. For example, to get Eq. (5.7):

$$\frac{\Delta T}{T_0} Y^*_{\ell' m'}(\hat{\mathbf{n}}) = \sum_{\ell=1}^{\infty} \sum_{m=-\ell}^{\ell} a^T_{\ell m} Y_{\ell m}(\hat{\mathbf{n}}) Y^*_{\ell' m'}(\hat{\mathbf{n}}) \tag{5.10}$$

$$\begin{aligned} \Rightarrow \int d\hat{\mathbf{n}} \frac{\Delta T}{T_0} Y^*_{\ell' m'}(\hat{\mathbf{n}}) &= \sum_{\ell=1}^{\infty} \sum_{m=-\ell}^{\ell} a^T_{\ell m} \int d\hat{\mathbf{n}} Y_{\ell m}(\hat{\mathbf{n}}) Y^*_{\ell' m'}(\hat{\mathbf{n}}) \\ &= a^T_{\ell' m'} \end{aligned} \tag{5.11}$$

due to the orthonormality of the spherical harmonics, which dictates that

$$\int d\hat{\mathbf{n}} Y_{\ell m}(\hat{\mathbf{n}}) Y^*_{\ell' m'}(\hat{\mathbf{n}}) = \delta_{\ell \ell'} \delta_{m m'}. \tag{5.12}$$

Thus, if we have measurements for T and $\mathcal{P}_{ab}$ (from observations of the CMB) we can calculate $a^T_{\ell m}$, $a^E_{\ell m}$ and $a^B_{\ell m}$. These three multipole moments fully describe the temperature-polarisation map of the sky.

5.2 Correlation Matrices and Angular Power Spectra

The covariance of $a^X_{\ell m}$ and $a^Y_{\ell m}$ for a particular ℓ and m is given by Eq. (5.13),

$$\text{Covar}_{\ell m \ell' m'}(X, Y) = \left\langle \left(a^{X*}_{\ell m} - \left\langle a^{X*}_{\ell m} \right\rangle\right) \left(a^Y_{\ell' m'} - \left\langle a^Y_{\ell' m'} \right\rangle\right) \right\rangle = \left\langle a^{X*}_{\ell m} a^Y_{\ell' m'} \right\rangle, \tag{5.13}$$

where $X, Y = T, E, B$, the angle brackets represent the average over all realisations (note that we can only observe one realisation, or universe), and $\left\langle a^{X*}_{\ell m} \right\rangle = \left\langle a^Y_{\ell m} \right\rangle = 0$ (because the sum of the deviations from the mean temperature and the net polarisation are both zero).

We use $C^{XY}_{\ell m \ell' m'}$ to denote the *correlation matrix*. Technically, to convert from covariance to correlation, we must divide by the product of the appropriate standard deviations:

$$C^{XY}_{\ell m \ell' m'} = \frac{\left\langle a^{X*}_{\ell m} a^Y_{\ell' m'} \right\rangle}{\sigma_{a^X_{\ell m}} \sigma_{a^Y_{\ell' m'}}}. \tag{5.14}$$

However, it is convention to assume that the multipole moments have been normalised such that $\sigma_{a^X_{\ell m}} = \sigma_{a^Y_{\ell' m'}} = 1$, yielding Eq. (5.15).

$$\boxed{C^{XY}_{\ell m \ell' m'} = \left\langle a^{X*}_{\ell m} a^{Y}_{\ell' m'} \right\rangle \qquad XY \textbf{ correlation matrix}} \tag{5.15}$$

Note that Eq. (5.7), Eq. (5.8) and Eq. (5.9) have **not** been normalised in this way as, in reality, it is more convenient to perform the normalisation on the correlation matrix itself. It also may not be necessary to normalise, depending on the properties being investigated.

For Gaussian perturbation theories, the correlation matrices fully describe the statistical properties of the temperature-polarisation map.

The diagonal of the correlation matrix is given in Eq. (5.16),

$$C^{XY}_{\ell m} \equiv \delta_{\ell \ell'} \delta_{m m'} C^{XY}_{\ell m \ell' m'}. \tag{5.16}$$

Averaging over m, we get the correlation for given ℓ, the *angular power spectrum*, Eq. (5.17).

$$\boxed{C^{XY}_{\ell} \equiv \frac{1}{2\ell+1} \sum_m C^{XY}_{\ell m} \equiv \frac{1}{2\ell+1} \sum_m \left\langle a^{X*}_{\ell m} a^{Y}_{\ell m} \right\rangle \qquad XY \textbf{ angular power spectrum}} \tag{5.17}$$

Note that Eq. (5.17) is often expressed as $C^{XY}_{\ell} \equiv \left\langle a^{X*}_{\ell m} a^{Y}_{\ell m} \right\rangle$, where the angle brackets denote the average over m, as well as all realisations. For the sake of consistency, this notation will not be used in this thesis. Formalisms exist for computing any C_ℓ for any FRW (Friedmann-Robertson-Walker) space-time and any structure formation model (Callin 2006). Many cases can be computed using codes such as CAMB (http://camb.info, Lewis & Bridle 2002) and CMBFAST (http://www.cmbfast.org, Zaldarriaga & Seljak 2000), which are publicly available.

An important operation performed by these codes is finding the radiation transfer functions, $\Delta^X_\ell(k)$, where X denotes either temperature or type of polarisation (T, Q or U), ℓ is the multipole mode and k is wavenumber. The transfer functions describe the evolution of fluctuations from primordial to present-day, at a particular ℓ and k. Kosowsky (1996) provides a detailed discussion on how the transfer functions can be calculated, along with how they can be used to find the CMB correlation matrix. Eq. (5.18) gives the correlation matrix in terms of the transfer functions using a notation very similar to that adopted by Phillips & Kogut (2006) (whose algorithm for calculating the correlation matrix is also similar to that used

in this thesis).

$$\mathbf{C}^{XY,L}_{\ell m,\ell' m'} = (4\pi)^2 \sum_n \Delta^{X*}_\ell(k_n)\Delta^Y_{\ell'}(k_n)\,\frac{\mathcal{P}(k)}{k^3}\,A^L_{\ell m,\ell' m'}(n) \tag{5.18}$$

where n and k_n satisfy $k = 2\pi n/L$, $\mathbf{n}$ represents the integer triplets (n_x, n_y, n_z), $\mathcal{P}(k)$ is the primoridal power spectrum, $A^L_{\ell m,\ell' m'}(n)$ is given by Eq. (5.19) and L is the topology scale. For simplicity and readability, we assume the dimensions of the fundametal domain to be equal, $L_x = L_y = L_z = L$, but the equations can be modified to allow for different lengths.

$$A^L_{\ell m,\ell' m'}(n) = \sum_{\{|\mathbf{n}|=n\}} \xi^{\mathbf{n}*}_{k\ell m}\xi^{\mathbf{n}}_{k\ell' m'} \tag{5.19}$$

where $\xi^{\mathbf{n}}_{k\ell m}$ are the coefficients that characterise topology discussed in Section 2.2.3. Note that $A^L_{\ell m,\ell' m'}(n)$ depends on topology, not cosmology, and will be referred to as a 'topoterm' for the remainder of this thesis.

5.2.1 Properties of the Correlation Matrix

If the CMB is statistically isotropic, the correlation matrix must be rotationally invariant (i.e. diagonal) and individual correlations must be independent of m (which relates to the orientation/location of the associated ℓ modes on the sky). Since the power spectrum is by definition diagonal and independent of m, the correlation matrix could then be expressed as

$$C^{XY}_{\ell m\ell' m'} = C^{XY}_\ell \delta_{\ell\ell'}\delta_{mm'}. \tag{5.20}$$

Therefore, for the isotropic case, we lose no information by using the power spectrum, Eq. (5.17), for statistical analyses instead of the complete correlation matrix, Eq. (5.15). This is an advantage as it is computationally less expensive to calculate.

In a simply connected universe, the CMB should be isotropic. In a multiconnected universe, however, it is generally not and, in order to retain all information, we need to perform statistical analyses on the correlation matrix itself.

5.2.2 Properties of the Angular Power Spectrum

Fig. (5.1) shows the predicted forms of the CMB power spectra. The cross power spectra of B with T and E are expected to vanish because of the odd parity of B,

which should not correlate with even parity T and E (Bartlett 2006). While not displayed in this figure, the TE power spectrum is non-vanishing.

The EE power spectrum is out of phase with the TT power spectrum. This is because oscillatory velocities (*baryonic acoustic oscillations*) in the plasma induce Doppler shifts and, hence, the scalar perturbations responsible for the majority of the E modes. The maximum temperature variations (or Doppler shift) occurs at peaks and troughs in the oscillations, where the plasma has minimum velocity. Polarisaton, on the other hand, requires bulk motions in the plasma; maximum polarisation occurs at maximum plasma velocities. So, while peaks in the T mode spectrum correspond to density modes that are at their minimum velocities, peaks in the E mode correspond to density modes that are at their maximum velocities. Minimum velocity density modes are 90° out of phase with maximum velocity density modes. Hence, the T and E modes, and the corresponding power spectra, are out of phase.

B modes are assumed to be produced solely by tensor perturbations (i.e. gravity waves) since vector perturbations are not expected to be present in the LSS and scalar perturbations only produce E modes (see Section 4.2.1). Gravity waves in the LSS are a prediction of inflation and, therefore, B modes can be used to constrain the strength of inflation. The strength of inflation, r, is defined as

$$r = \frac{\text{amplitude of tensor perturbations}}{\text{amplitude of scalar perturbations}}. \tag{5.21}$$

Fig. (5.1) shows predicted CMB BB power spectra due to gravity waves ranging from 3.2×10^{15} GeV (the minimum detectable, see Knox & Song 2002) to 2.6×10^{16} GeV (the maximum allowable, see Wang et al. 2002) .

5.3 Converting to Pixel Space

Spherical harmonics can be used to transform between harmonic space and pixel space. The harmonic to pixel transformation for the correlation matrix is

$$C_{pp'} = \sum_{\ell m \ell' m'} Y^*_{\ell m}(\mathbf{p}) Y_{\ell' m'}(\mathbf{p}') C_{\ell m \ell' m'} \tag{5.22}$$

where p, p' are the pixel numbers and $\mathbf{p}$, $\mathbf{p}'$ are the respective pixel directions. This is an idealistic pixel space, where each pixel contains information only from a single point in the sky at any one time. To convert to a realistic detector pixel space, which has some pixel beam function (describing the area, as opposed to point, in

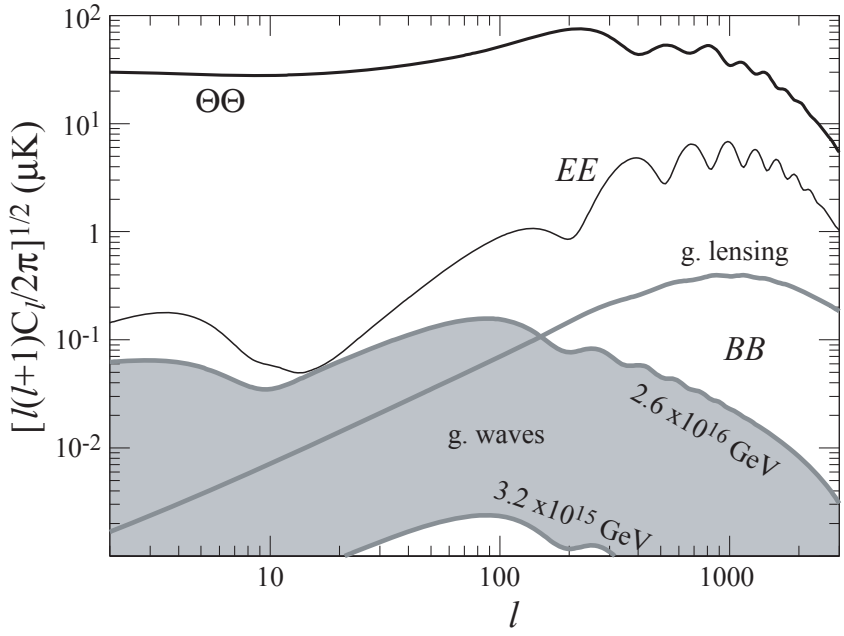


Figure 5.1: TT (labelled ΘΘ on the plot), EE, and BB power spectra. The EE spectrum is out of phase with respect to the TT spectrum. The grey region covers a number of possible BB power spectra, depending on the strength of the gravity waves in the LSS. The line labelled g. lensing is a predicted B mode spectrum generated purely by gravitational lensing (see Section 5.4.2). Credit: Hu et al. 2003.

the sky to which a pixel is instantaneously exposed), the transformation is

$$C_{pp'} = \sum_{\ell m \ell' m'} B^*_{\ell m} B_{\ell' m'} Y^*_{\ell m}(\mathbf{p}) Y_{\ell' m'}(\mathbf{p}') C_{\ell m \ell' m'}. \tag{5.23}$$

Note that the transformation of the pixel beam from pixel to harmonic space is

$$B_{\ell m} = \sum_p B_p(\mathbf{p}) Y_{\ell m}(\mathbf{p}). \tag{5.24}$$

It is usually easier to work in harmonic space, but it can be helpful to plot maps of the correlation matrix (which is done in pixel space) in order to visualise the correlations (e.g., see Niarchou 2006).

5.4 Taking Measurements

A measurement of the CMB consists of more than just the signal that we are interested in. We can parameterise the data (e.g., see Jaffe et al. 1999, Wandelt et al. 2004) as

$$d_i = \sum_p A_{ip}(s_p + f_p) + n_i \tag{5.25}$$

where i labels increments of time ($t = i\delta t$), s_p and f_p are the signal and foregrounds detected in pixel p respectively, n_i is the instrumental noise and A_{ip} is the pointing operator ($A_{ip} = 1$ at times when pixel p is observed and $A_{ip} = 0$ at times when pixel p is not observed). The data d_i will be measurements of either temperature, T, or polarisation, in the form of Stokes' parameters Q or U. The temperature data, d_i^T can be used directly in analyses of the temperature anisotropies. However, the orientation of the axes against which Q and U were measured must be taken into account before d_i^Q and d_i^U can be used in analyses of E and B modes (see Section 4.2.2).

We have used a time-ordering label, i, on the data, d_i, and noise, n_i. The signal, s_p, and foregrounds, f_p, are not time ordered, but we effectively make them so by multiplying by A_{ip} and summing over p. We can do the opposite and find the non-time-ordered data, d_p, and noise, n_p. For example, the non time ordered noise is

$$n_p = \sum_i A_{ip} n_i, \tag{5.26}$$

or, in matrix form,

$$\mathbf{n} = \mathbf{A}\mathbf{n}_{\rm tod}. \tag{5.27}$$

where $\mathbf{n}_{\rm tod}$ is the time-ordered noise correlation matrix. To convert between time ordered and non time ordered correlation matrices ($\mathbf{N}_{\rm tod}$ and $\mathbf{N}$ respectively in the case of noise), we use

$$N_{pp'} = \sum_{i,i'} A_{ip} N_{ii'} A_{i'p'}, \tag{5.28}$$

or, in matrix form,

$$\mathbf{N} = (\mathbf{A}^{\rm T}\mathbf{N}_{\rm tod}^{-1}\mathbf{A})^{-1}. \tag{5.29}$$

5.4.1 Uncertainties

We wish to infer C_ℓ or $C_{\ell m \ell' m'}$ from measurements described in Eq. (5.25). There are many sources of uncertainty that we should consider:

- **Noise n_i and foregrounds f_p** can be estimated but, as we cannot know them precisely, will always be responsible for some degree of uncertainty in our knowledge of the signal, s_p. Note that we usually assume the noise to have a Gaussian distribution.

- **The pixel beam $B_p(\mathbf{x})$** (which describes which directions $\mathbf{x}$ are seen by pixel p) not only takes into account the resolution of a pixel, but the shape of the area that it sees. The uncertainty lies in not knowing exactly where the signal has come from. The signal and foregrounds measured in pixel p both involve the pixel beam:

$$s_p = \int_{\mathbf{x}} d\mathbf{x} B_p(\mathbf{x}) s(\mathbf{x}) \tag{5.30}$$

$$f_p = \int_{\mathbf{x}} d\mathbf{x} B_p(\mathbf{x}) f(\mathbf{x}). \tag{5.31}$$

  However, on the scales considered in this thesis, the area covered by the the pixel beam for pixel p is negligible (i.e. $B_p(x) \simeq \delta(x - x_p)$).

- **Sample variance of C_ℓ or $C_{\ell m \ell' m'}$.** The variance of a population $\{x_n\}$ can be estimated from a sample $\{y_n\}$ of that population (see any standard

undergraduate text book on statistics):

$$\mathrm{Var}(\{x_n\}) \simeq \mathrm{Var}(\{y_n\}). \tag{5.32}$$

However, we are interested in the variance of the mean value of our sample (as a measure of the uncertainty on the mean):

$$\mathrm{Var}(\overline{y}) = \frac{\mathrm{Var}(\{y_n\})}{N}, \tag{5.33}$$

where N is the size of the sample, $\overline{y}$ is the mean of the sample, and the y_n must not be correlated in order for the equation to hold. Eq. (5.32) and Eq. (5.33) lead to the result

$$\mathrm{Var}(\overline{y}) \simeq \frac{\mathrm{Var}(\{x_n\})}{N}. \tag{5.34}$$

The sample variance improves (i.e. decreases) as the sample increases in size. Regardless of how many universes there may be, we can only observe one which, unfortunately, means that our sample size is only one for $C_{\ell m \ell' m'}$. But the situation can be better for C_ℓ, for which there are $2\ell + 1$ moments (m) for each multipole (ℓ); the sample size in this case is $2\ell + 1$.

- **Sky coverage** is also an example of sample variance. Sky coverage may not be complete if not using an all-sky survey or if a mask has been used to "remove" foregrounds. Either way, the sample variance scales as $1/f_{\mathrm{sky}}$, where f_{sky} is the fraction of the sky covered by the data being used.

5.4.2 Challenges

There are many challenges facing any experiment which trys to measure the CMB. Polarisation can be especially problematic as only about 10% of CMB photons are not randomly polarised resulting in a weaker signal than for temperature (Hu & White 1997). Foregrounds such as the Milky Way and other galaxies contaminate CMB measurements. They provide additional photons that can be mistaken for CMB photons. We also have to consider effects that change the properties of the CMB photons themselves as they travel from the LSS to Earth. The main such effects are reionisation, the ISW effect and gravitational lensing.

Foregrounds can be removed to an extent by estimating the microwave contribution from sources surveyed at other wavelengths (Tegmark 1998). This is very difficult for the Milky Way and this region of the sky tends to be masked (essentially ignored). The removal of foregrounds is considerably more problematic for polarisation than for temperature. However, the Planck satellite has been designed to address these issues (Planck Collaboration 2005): it is much more sensitive to polarisation than its predecessors; it has A range of frequency bands that, together, should aid the removal of foregrounds.

- **Cleaning**: This is where an attempt is made to remove a foreground, revealing the uncontaminated, 'clean', signal of interest. However, this requires estimation of the foregrounds and inevitably results in some degree of uncertainty.

- **Masking**: The implementation of a mask involves multiplying the data by a mask function, $M(\hat{\mathbf{n}})$, which is zero in masked regions and one elsewhere. e.g., for temperature, Eq. (5.7) becomes

$$a^T_{\ell m} = \frac{1}{T_0} \int d\hat{\mathbf{n}}\, \Delta T(\hat{\mathbf{n}})\, Y^*_{\ell m}(\hat{\mathbf{n}})\, M(\hat{\mathbf{n}} \tag{5.35}$$

  Alternatively, we can used a weight function, $W(\hat{\mathbf{n}})$, which is zero where masked but not necessarily one elsewhere, providing smoother transitions instead of sharp cuts around masked regions. Masking introduces anisotropies to the CMB that can result in coupling of modes. Gruetjen & Shellard 2012 discuss methods of optimising masking in order to account for these effects.

Reionisation: some part of the CMB may have been scattered at the epoch of reionisation, long after recombination/last scattering. This would cause damping in both the temperature and polarisation spectra at small angular scales and induce a bump in the polarisation at a large angular scale (see Fig. (5.2)). The ℓs at which these effects take place depend on the optical depth to reionisation, τ (Zaldarriaga 1997). τ is incorporated in the ΛCDM model and can be estimated as part of the ΛCDM fit to the CMB data. In this way, the effects of reionisation can to some extent be accounted for (as well as gaining valuable information about the epoch). The bump should appear below $\ell = 20$ (Kaplan et al. 2003).

ISW effect: as CMB photons propagate to Earth, changes in gravitational potential induce variations in the photon energy. Fortunately, this does not affect polarisation. Temperature is affected on scales of ℓ less than about 5 (Rassat &

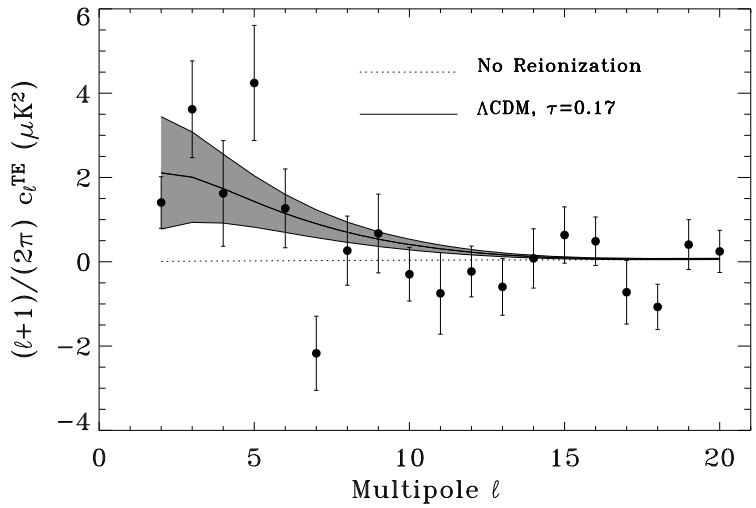


Figure 5.2: Reionisation bump in the low ℓ region of the TE power spectrum. Credit: Kogut et al. 2003.

Starck 2013); the contribution to the power spectrum can be modelled from theory or estimated using galaxy surveys (Fig. (5.3)).

Weak gravitational lensing smooths out acoustic oscillations in TT, EE and TE power spectra, generates power at arcmin scales in TT, EE and TE power spectra, and effectively creates a spectrum of B-mode polarisation from E-modes (Hirata et al. 2005). There is not much that can be done about the smoothing of the power spectra, but there are ways to estimate the lensing structure of the Universe (Lewis & Challinor 2006, Smith et al. 2012) and the contribution of lensing to the power spectra. The lensing contribution only becomes significant at large scales, when ℓ reaches the order 1000 (Fig. (5.4)); this threshold is likely much lower for the generation of B modes, since the CMB B mode signal is not thought be very strong (Fig. (5.1)).

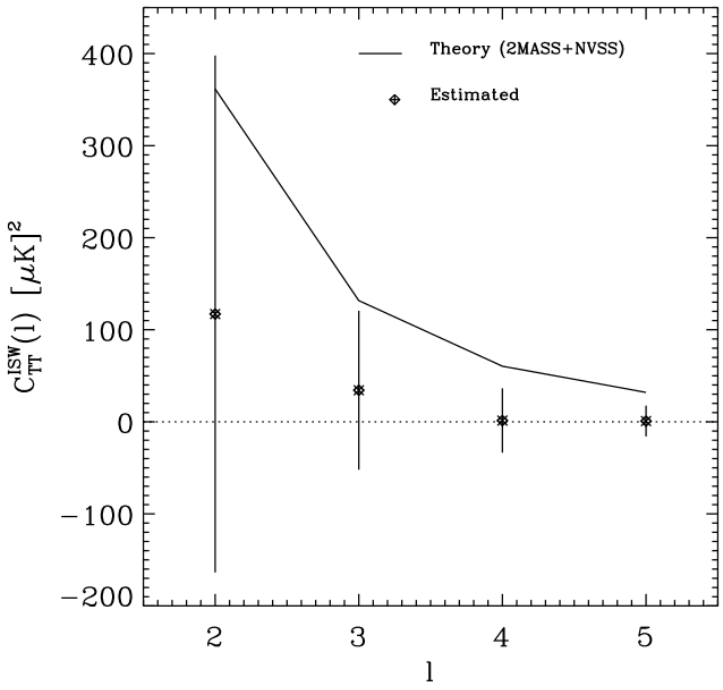


Figure 5.3: ISW contribution to the TT power spectrum at low ℓ. The solid line is the theoretical prediction and the data points are estimated from 2MASS and NVSS galaxies with WMAP9 data. Credit: Rassat & Starck 2013.

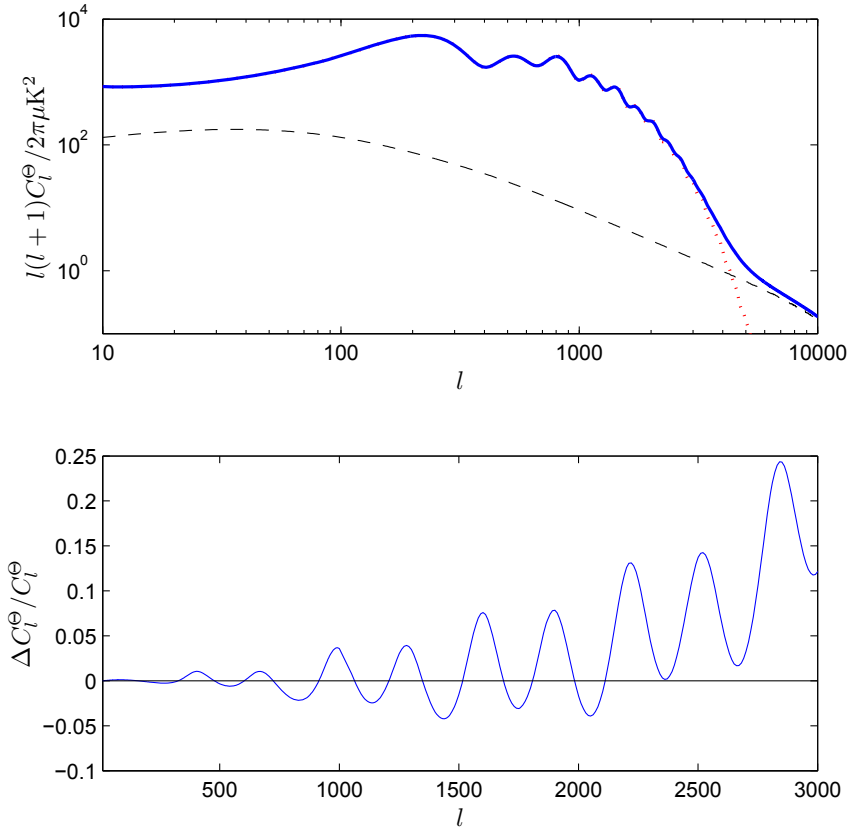


Figure 5.4: Contribution of gravitational lensing to the temperature power spectrum, based on a typical ΛCDM model. Top: the lensed temperature power spectrum (blue, solid line), unlensed spectrum (red, dotted line) and power from lensing only (black, dashed line). Bottom: the fractional change in the power spectrum due to lensing. Credit: Lewis & Challinor 2006.

Part III

Bayesian Analysis

Chapter 6

The General Bayesian Approach

Bayesian analyses determine a probability distribution for the parameters of a model. This is different to frequentist approaches, which propose a hypothesis, or model with parameters in some fixed range, to be true (or not) and then attempt to determine whether or not this is the case to some confidence level. The frequentist confidence level can be considered the probability of the model parameters lying in a fixed range, but is not a probability distribution for the parameters. An advantage of the Bayesian approach is that it has the potential to take account of as much available information as possible. Many believe that there is no need for this gain in complexity, but Bayesian analyses are increasingly employed to investigate astrophysical data for a number of reasons (Trotta 2008). As theories become more complicated and observations more precise, models will become more complex and require more sophisticated analysis. Due to direct incorporation of models in the analysis, Bayesian statistics often offer a more intuitive interpretation of the data they describe. When there are vast amounts of data, sensible Bayesian and frequentist approaches will generally lead to similar conclusions. But all too often, in cosmology and UHECR research for example, the data available is sparse or poor quality. In these cases, the extra information utilised by Bayesian methodologies is crucial.

6.1 Bayes' Theorem

Bayesian statistics is so-called as it makes use of Bayes' theorem, but it still incorporates other statistics. Bayes' theorem states the probability of a hypothesis given relevant data. One way to interpret this is: the probability that the parameters of a model have certain values, given that certain data (regarding the system that the model describes) have been obtained (Loredo 1990). In this case, Bayes' theorem

can be written:

$$\Pr(\text{parameters}|\text{data}) = \frac{\Pr(\text{data}|\text{parameters})\Pr(\text{parameters})}{\Pr(\text{data})} \tag{6.1}$$

where Pr(parameters|data) is known as the posterior, Pr(data|parameters) the likelihood, Pr(parameters) the prior, and Pr(data) the evidence. Note that $\Pr(x = X|y)$ is the probability (density) that, given y, x is essentially X but, more precisely, describes the probability (density) that x lies in some small range about X (Jaffe 1996).

Technically, all these probabilities take account of some background information, I. Using the notation θ for the parameters and D for the data, we write Bayes' theorem as:

$$\boxed{\Pr(\theta|D, I) = \frac{\Pr(D|\theta, I)\Pr(\theta|I)}{\Pr(D|I)}. \quad \textbf{Bayes' theorem}} \tag{6.2}$$

The evidence acts to normalise the posterior and can be found by *marginalising* (essentially removing) θ from the product of the likelihood and prior. This is done by integrating this product over θ:

$$\Pr(D|I) = \int_{\theta-space} \Pr(D|\theta, I)\Pr(\theta|I)d\theta. \tag{6.3}$$

Different parameters of the same model have the same evidence and so the evidence is often neglected when comparing the probabilities of different parameters. This is helpful as it is often difficult to find the evidence analytically.

6.1.1 Choosing Priors

The most problematic part of the Bayesian method, and arguably the one that provokes the most criticism, is the choice of prior (Loredo 1990). One must be careful to select a prior that truly describes previous knowledge of the parameter space, but it is not always clear what mathematical form best reflects this knowledge.

In addition, a prior should technically be normalised; if it cannot be normalised, i.e. if it integrates to infinity, it is known as *improper*. This is a problem if it yields an improper posterior, which must be normalisable. However, an improper prior can yield a proper posterior if it provides an evidence which is finite for all D. Consider the uniform prior, $\Pr(\theta|I) = \text{const.}$. This is an improper prior as it covers all real space (i.e. $-\infty < \theta < \infty$) and so integrates to infinity. But it can lead to a finite evidence for all D, i.e.

$$\Pr(D|I) = \int_{-\infty}^{\infty} \Pr(D|\theta, I)\Pr(\theta|I)d\theta = A(D), \tag{6.4}$$

where A is some finite value dependent on D. If the evidence can be found analytically, it is fairly straightforward to implement this prior, but this is often not the case. If we are to find the evidence numerically, we cannot integrate between infinite limits; we must take some other limits, $\pm a$ where $a > 0$, being careful to choose a to be large enough for the evidence to have converged to A (to a large number of significant figures) for all D.

So, if the posterior is independent (to very many significant figures) of the limits placed on the prior, we can just as well use an improper prior. In these cases, the information provided by the data via the likelihood "overwhelms" that provided by the prior. If, however, the prior information and the associated ranges of parameters is important, allowing the parameter range to become infinite leads to vanishing or unnormalisable posterior probability. Proper priors must be used in these situations.

Uniform and logarithmic priors are popular choices where there is a complete lack of prior knowledge. The natural logarithm is uniform over natural logarithmic scales, often making it more appropriate in astrophysical/cosmological situations. A drawback of the logarithmic prior is that, unlike the uniform prior, it does not allow for parameters with a value of zero.

6.2 Model Comparison

If we wish to compare how well two models j and k fit experimental data, we can take the ratio of their posterior probabilities: the posterior "odds" (Jaffe 1996, Drell et al. 2000) can be expressed as:

$$\frac{\Pr(j|D, I)}{\Pr(k|D, I)} = \frac{\Pr(D|j, I)}{\Pr(D|k, I)}\frac{\Pr(j|I)}{\Pr(k|I)} = \frac{\Pr(D|I_j)}{\Pr(D|I_k)}\frac{\Pr(j|I)}{\Pr(k|I)} = B_{jk}\frac{\Pr(j|I)}{\Pr(k|I)}. \tag{6.5}$$

where $B_{jk} = \Pr(D|I_j)/\Pr(D|I_k)$ is "Bayes' factor", a ratio which depends only on the experimental data. Eq. (6.5) shows the odds expressed in terms of Bayes' factor and a term which is dependent on the theoretical model only. Bayes' factor favours a model whose average likelihood (with respect to the prior distribution) is greater, i.e. a model with a strongly favoured maximum likelihood will still be disfavoured overall if its likelihood is very low over large areas of the allowed parameter space

(Jaffe 1996).

The favouring of simpler theories (Ockham's razor), unless a more complex model (e.g. one with more parameters) is significantly better at describing the data, is an advantage of the Bayesian approach (Jaffe 1996, Trotta 2008).

6.3 Bayesian Inference in Astrophysics and Cosmology

Over the last decade or so, Bayesian methods have been increasingly adopted in astrophysics and cosmology. Some examples are listed below.

- Cosmological parameter estimation from various observations such as CMB and supernova. Cosmo MC is a Monte Carlo code that performs this parameter estimation and is publicly available (Lewis & Bridle 2002).
- Constraining topology using CMB temperature observations (e.g. Niarchou 2006). Most work to date has focused on the use of temperature anisotropies. With the advent of Planck, it will soon be possible to perform a full investigation with polarisation measurements.
- Weak gravitational lensing (shear) from deep sky galaxy surveys (e.g. Miller & CFHTLenS Collaboration 2012).
- The origin of UHECRs (Ultra High Energy Cosmic Rays) from UHECR observations. The use of a Bayesian approach for this application is introduced in Watson et al. 2011, a paper describing the work in Ch. 7 of this thesis.
- Constraining neutrino mass using various experimental data, including astrophysical, (e.g. Archidiacono et al. 2012).
- MultiNest, a publicly available, general Bayesian code for cosmology and particle physics (Feroz et al. 2009).

Chapter 7

Example Application of the Bayesian Approach: The Origin of UHECRs

The work in this chapter has been published in a paper in collaboration with Daniel Mortlock and Andrew Jaffe (Watson et al. 2011). The results were generated from code written by myself to perform the statistical analysis (described in Section 7.4) on the data (Section 7.3), which was provided by Daniel Mortlock in a form consistent with our models (Section 7.4.1).

7.1 Brief Introduction to UHECRs

Cosmic rays (CRs) are highly accelerated protons and nuclei that reach Earth with arrival energies in the wide range $\sim 10^8$ eV $\leq E_{\rm arr} \leq\sim 10^{20}$ eV (see, e.g., Stoker 2009). They were discovered in 1912 by Victor Hess, and the less well known Domenico Pacini. Hess performed balloon experiments that detected a greater amount of "penetrative radiation" at higher altitudes than recorded at sea level (Hess 1912). Pacini submerged a copper box containing an electroscope in the ocean, measuring less penetrative radiation than at sea level (Pacini 1912). These results led both men, separately, to the conclusion that these rays must originate in space: hence the name "cosmic rays". It wasn't until 1932 that Jacob Clay demonstrated that CRs are not photons but must be extremely energetic particles. He found that the intensity of CRs was lower nearer the equator than at more northern latitudes. He realised that this could be explained by the difference in the geomagnetic field at these locations, but only if the CRs were charged particles (Clay & Berlage 1932).

The origin of ultra-high energy cosmic rays (UHECRs) with $E_{\rm arr} \geq\sim 10^{19}$ eV (about 10^6 times greater than energies that can be achieved by CERN's LHC), in particular, remains uncertain. Suggested sources include gamma ray bursts (Wick et al. 2004, Waxman 1995) and new physics (e.g. Sarkar 2002). The most promising theory is that UHECRs are generated by active galactic nuclei (AGNs). There are several physical models to motivate this idea (e.g., Diehl 2009; Protheroe & Szabo 1992; Fraschetti & Melia 2008), but the hypothesis requires empirical verification.

Various correlation analyses of UHECR arrival directions and locations of possible progenitors have been performed using data from different observatories and progenitor catalogues. An analysis by the Pierre Auger Collaboration (Abraham et al., 2007b) found the first 27 UHE PAO (Pierre Auger Observatory) events to be strongly correlated with a sample of local AGNs in the Veron-Cetty & Veron 2006, VCV, catalogue; this was the first strong empirical confirmation of the hypothesis that UHECRs are generated by AGNs.

The PAO has continued to operate in the time since these results were obtained; subsequent data (Abreu et al. 2010) show a much weaker correlation. Beatty & Westerhoff 2009 discuss the many attempts to find a correlation between AGNs and UHECRs, using a variety of techniques and data, such as those reported by Nemmen et al. 2010, Abraham et al. 2008, 2007b, Abbasi et al. 2008, Ghisellini et al. 2008 and George et al. 2008. In particular, Abbasi et al. 2008 claim no significant correlation.

7.1.1 The GZK Effect

Cosmic rays above energies of $E_{\rm GZK} \simeq 5{\times}10^{19}$ eV are highly relativistic and see CMB photons blueshifted. Protons at these energies can interact with the blueshifted photons to produce pions, losing energy in the process; this is know as the GZK effect (Greisen 1966; Zatsepin & Kuz'min 1966). The GZK mean free path between interactions for an $E \simeq 10^{20}$ eV proton is only about 4 Mpc, and each interaction typically reduces a CR's energy by approximately 20 per cent (Achterberg et al. 1999). So any observed UHECRs must have originated within an effective 'GZK horizon' of about 100 Mpc (see Fig. (7.1)). If UHECRs are primarily Fe nuclei, the GZK horizon is expected to be even smaller, since these nuclei are heavier and slower than protons of the same energy. However, this also means that the deflection due to magnetic fields, and thus uncertainty in arrival direction, is greatly increased.

So, while the GZK effect reduces the number of detectable UHECRs, a fortunate consequence is that it also reduces the number of plausible AGN sources to the few thousand with distances less than about 100 Mpc or, equivalently, redshifts less

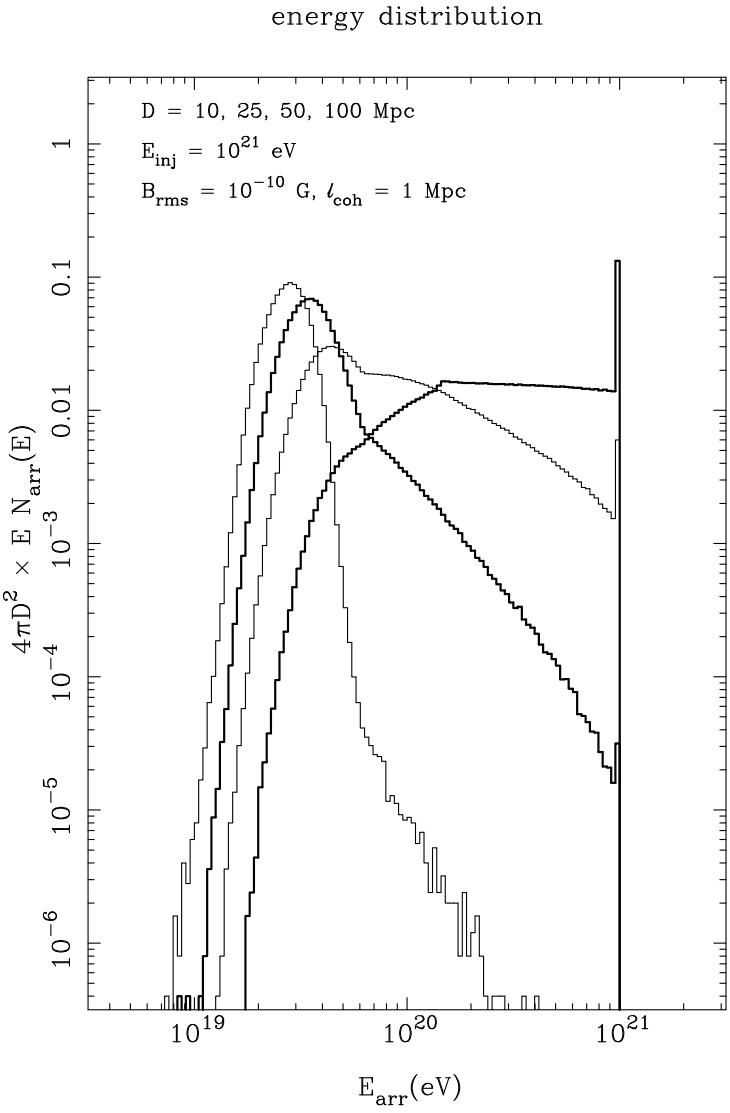


Figure 7.1: Plot of simulated received energy spectra from mono energetic sources at different distances from Earth. Each source emits cosmic rays at the energy of $E_{\rm emit} = 10^{21}$ eV. Curves from rightmost to leftmost represent sources at increasing distances. The received spectrum from a source at only 10 Mpc has a distinct spike containing cosmic rays that have reached us without losing energy to the GZK effect. By 100 Mpc, the fraction of cosmic rays whose energy remains at $E_{\rm emit} = 10^{21}$ eV is negligible. i.e., the probability of an $E_{\rm emit} = 10^{21}$ eV cosmic ray reaching us from a source more distant than 100 Mpc is negligible. Credit: Achterberg et al. 1999.

than about 0.03. This makes it plausible to search for a correlation between the arrival directions of UHECRs and locations of local AGNs, provided sufficiently many UHECRs can be observed.

7.2 Purpose

Abraham et al. 2007b have reported the strongest correlation between arrival directions of UHECRs and locations of AGNs to date (at the time of publication of Watson et al. 2011). Given the small numbers of UHECRs on which these results are based, some care must be taken with the statistical methods employed. This is both to ensure that all the available information is utilised and to avoid over interpretation. These aims can be achieved by adopting a Bayesian approach in which the relevant stochastic processes (e.g., the GZK interactions of the UHECRs with the CMB, deflection by the Galaxy's magnetic field, and measurement errors) are explicitly modelled. The details of some of these processes are not known (most relevantly, the strength of the magnetic fields and the energy calibration of the UHECRs), but such uncertainties can be accounted for by marginalisation.

Our aim is to provide an improved method for testing whether UHECRs originate from AGNs, addressing some issues with the analysis presented by Abraham et al. 2007b:

- They use circular angular matching regions around each UHECR: all AGNs within these regions are considered to be equally probable sources; any AGN outside these regions is completely disregarded as a progenitor. This approach is suboptimal as real matches would tend to be more centrally concentrated, with the source probability decreasing gradually as the angular distance from the CR increases.

- They ignore the radial distance to the AGNs. This is potentially misleading because their analysis is as sensitive to physically implausible correlations (i.e. those involving AGNs too distant to be progenitors) as it is to those that would be expected if the AGNs are in fact the UHECRs' progenitors.

- Their simple correlation analysis ignores the arrival energy E_{arr} of the individual UHECRs. A likelihood based approach can incorporate the fact that the higher the energy of the event, the nearer its progenitor is expected to be (due to the GZK effect).

These points can be addressed, and the constraining power of an UHECR data set increased, by using a physical model of UHECR generation, propagation and observation, thereby extracting all the valuable information in the data set (albeit at the small price of increased complexity).

In this work, the first steps are taken to developing a comprehensive Bayesian formalism for analysing UHECR data. The starting point is to reanalyse the UHECR and AGN samples used by Abraham et al. 2007b, changing only the statistical method. This is so that, aside from providing a direct answer to the question of whether the 27 PAO UHECRs come from the local VCV AGNs, it will show directly how the results depend on the statistical method used to analyse such data sets. After describing the UHECR and AGN samples in Section 7.3, the Bayesian method and CR propogation model are presented in Section 7.4. The results of applying this methodology are given in Section 7.5 and the overall conclusions are summarised in Section 7.6.

7.3 Experiments and Data

A number of difficulties hinder efforts to gain experimental evidence about UHECRs. The most fundamental problem is that CRs are deflected by the Galaxy's magnetic field. The arrival directions of lower energy extragalactic protons are essentially independent of their point of origin, although UHECRs are expected to be deflected by no more than a few degrees (Achterberg et al., 1999).

It is also problematic that UHECRs are very rare, with the observed number flux falling off with energy as

$$\frac{d\Gamma_{\rm obs}}{dE_{\rm arr}} \simeq \left(\frac{E_{\rm arr}}{10^{19}\ {\rm eV}}\right)^{-2.6}\ {\rm s}^{-1}\ {\rm m}^{-2}\ {\rm sr}^{-1} \tag{7.1}$$

(e.g., Abraham et al. 2010), where $\Gamma_{\rm obs}$ is the observed number flux and $E_{\rm arr}$ is the arrival energy. The fall off is expected to be even more extreme above energies of $E_{\rm GZK} \simeq 5 \times 10^{19}$ eV due to the GZK effect. Fig. (7.2) shows the cosmic ray spectrum, consisting of events detected by various observatories.

The problem of the low UHECR arrival rate can only be overcome by using a large collecting area, and by observing for long periods of time. Large ground based observatories such as HiRes Fly's Eye and AGASA (e.g. see Nagano & Watson 2000), have made significant progress in obtaining UHECR data. At present, the largest CR observatory is the Pierre Auger Observatory (Abraham et al. 2004), which is located near Malargüe in Argentina, at a longitude of 69°4 and a latitude

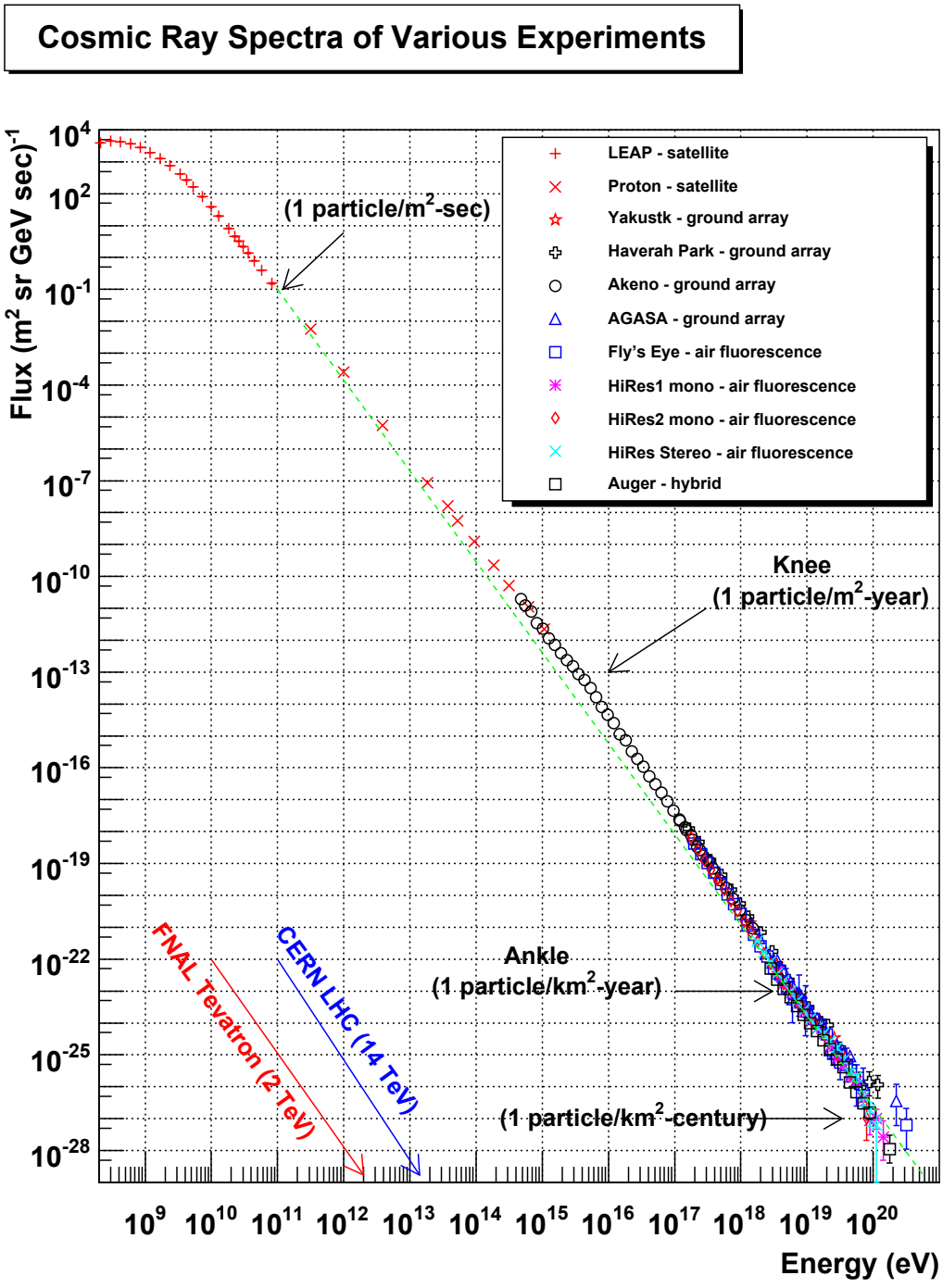


Figure 7.2: Cosmic ray energy spectrum containing events measured by various different experiments. Credit: William F. Hanlon, University of Utah.

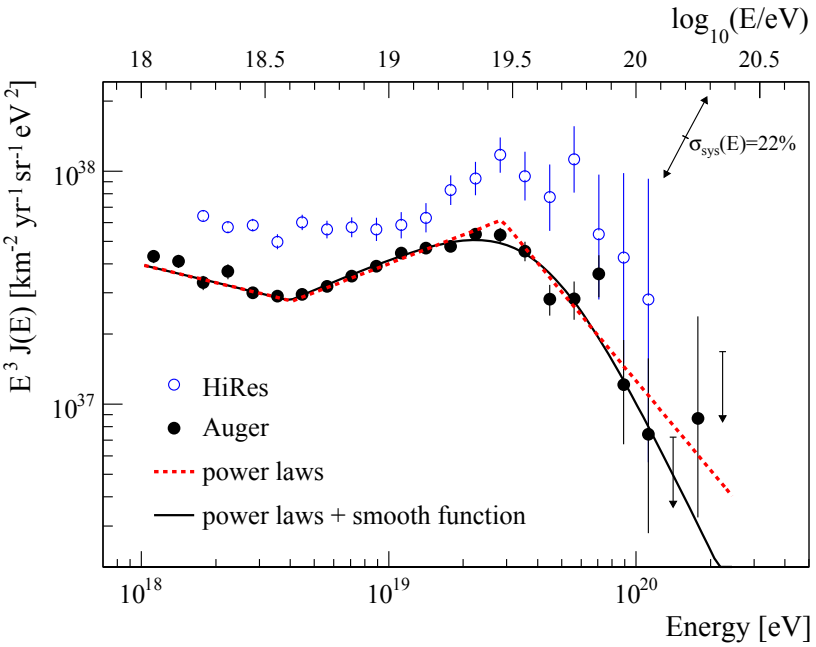


Figure 7.3: Spectrum of PAO UHECRs, with HiRes detections for comparison. Credit: Abraham et al. 2010.

of $-35°2$, and has been operational since January 2004. It has 1600 surface detectors (SDs) that cover an area of 3000 km^2, as well as four arrays of six atmospheric fluorescence telescopes.

The sample of UHECRs (Section 7.3.1) and the AGN catalogue (Section 7.3.2) analysed here are the same as those used by Abraham et al. 2007b.

7.3.1 PAO Observations of UHECRs

During its first approximately 3.6 years of observing, the PAO made reliable detections of the arrival directions and energies of 81 UHECRs, of which 27 had an (estimated) arrival energy of $E_{\rm arr} \geq 5.7 \times 10^{19}$ eV (see Fig. (7.3)).

The arrival directions are measured accurate to about 1°. There is an additional effective uncertainty in the progenitor direction due to deflection of the UHECR by Galactic and intergalactic magnetic fields. The magnitude of this effect is somewhat uncertain, with estimates of the typical deflection angles ranging from 2° (e.g. Dolag et al. 2005; Medina Tanco et al. 1998) to 10° (e.g. Sigl et al. 2004) for $E_{\rm arr} \simeq 10^{20}$ eV UHECRs. The combined effect is that the observed arrival direction, $\hat{\boldsymbol{r}}_{\rm arr}$, and the direction to the progenitor, $\hat{\boldsymbol{r}}_{\rm src}$, are typically separated by a smearing angle of a few degrees.

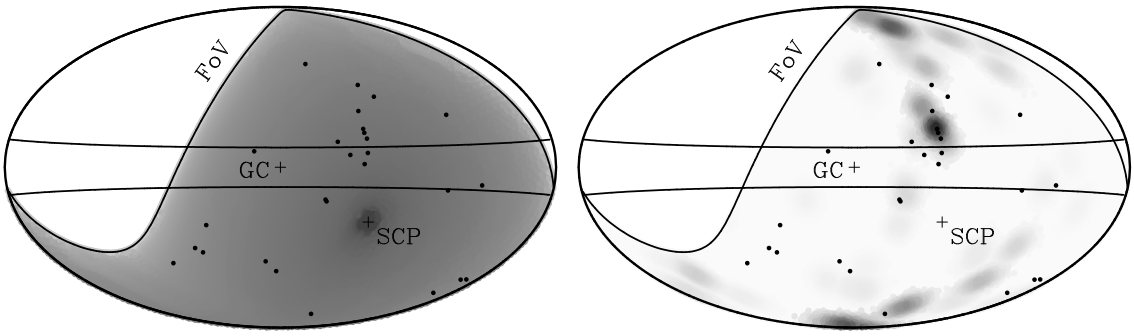


Figure 7.4: The arrival directions of the $N_c = 27$ PAO UHECRs (black points) and the source-weighted exposure (greyscale: darker indicates greater exposure) for the background-only model (left) and the AGN-only model (right), in Galactic coordinates. The Galactic Centre (GC), South Celestial Pole (SCP) and PAO's field of view (FoV) are all indicated. Lines of constant Galactic latitude $|b| = 10°$ are also shown. Credit: Watson et al. 2011.

As the construction of the PAO continued, its effective detector area increased steadily over the time during which the 27 UHECRs were detected. The evolution was sufficiently gradual that the exposure per unit solid angle, $d\epsilon/d\Omega$ (which has units of area $\times$ time), is a function of declination only. The angular dependence of the PAO exposure can be approximated by assuming that the instantaneous exposure is constant within 60° of the zenith and zero otherwise. (The detailed angular dependence is dominated by the cross sectional area of the SD array, and there are smaller corrections due to the various PAO data cuts, but these secondary effects are ignored here.) Integrating the instantaneous exposure over time to account for the Earth's rotation (cf. Fodor & Katz 2001) yields the declination dependent exposure $\epsilon(\hat{\boldsymbol{r}})$ shown in the left panel of Fig. (7.4), which contains plots of the UHECR and AGN data used in this analysis. The total exposure considered here is

$$\epsilon_{\rm tot} = \int \frac{d\epsilon}{d\Omega}\, d\Omega = 9000 \text{ yr km}^2 \text{ sr}^{-1} \tag{7.2}$$

(Abraham et al., 2007a).

7.3.2 Local AGNs

This work follows Abraham et al. 2007b in considering only AGNs in the 12th edition of the Veron-Cetty & Veron 2006 catalogue as possible sources for the PAO UHECRs. The distance to each source, D, is calculated from the quoted absolute and apparent magnitudes in the VCV catalogue, and AGNs without absolute magnitudes are omitted. The full catalogue contains 108,014 AGNs, but only $N_s = 921$ have $z_{\rm obs} \leq$

0.03 and are hence plausible UHECR progenitors inside the GZK horizon of about 100 Mpc.

The VCV catalogue is heterogeneous, having been compiled from a variety of AGN and quasar surveys, and as such it is not ideal for statistical studies. It is, however, expected to be close to complete for the local AGNs of interest here, except close to the Galactic plane. Moreover, as emphasized in Section 7.2, the VCV sample was chosen specifically to facilitate comparison with the results of Abraham et al. 2007b.

7.4 Method

Source and background rates, $\Gamma_{\rm src}$ and $R_{\rm bkg}$: we use a model (detailed in Section 7.4.1) characterised by the rate at which UHECRs are emitted by each AGN, $\Gamma_{\rm src}$, and the rate at which an isotropic background of UHECRs arrive at Earth, $R_{\rm bkg}$. If none of the UHECRs come from the candidate AGNs then the data should be consistent with $\Gamma_{\rm src} = 0$. Conversely, if all the UHECRs come from the AGNs in the catalogue, then the data should be consistent with $R_{\rm bkg} = 0$. By determining the most probable values for theses rates, given the PAO data, we can assess the candidacy of the AGNs as PAO UHECR progenitors. Note that the two rates have different units: $\Gamma_{\rm src}$ is the average number of UHECRs emitted per unit time by an AGN, and is given in units of $\rm s^{-1}$; $R_{\rm bkg}$ is the average number of UHECRs per unit time, per unit area, per unit solid angle, arriving at Earth, and is given in units of $\rm s^{-1}\ m^{-2}\ sr^{-1}$.

The joint posterior probability distribution of the rates $\Gamma_{\rm src}$ and $R_{\rm bkg}$, given the PAO data, summarises the full constraints on these rates and is given by

$$\Pr(\Gamma_{\rm src}, R_{\rm bkg}|\text{data}) = \frac{\Pr(\text{data}|\Gamma_{\rm src}, R_{\rm bkg})\,\Pr(\Gamma_{\rm src}, R_{\rm bkg})}{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\Pr(\text{data}|\Gamma_{\rm src}, R_{\rm bkg})\,\Pr(\Gamma_{\rm src}, R_{\rm bkg})\,d\Gamma_{\rm src}\,dR_{\rm bkg}}, \quad (7.3)$$

where:

- $\Pr(\Gamma_{\rm src}, R_{\rm bkg})$ is the prior distribution that encodes any external constraints on the rates. A uniform prior over $R_{\rm bkg} \geq 0$ and $\Gamma_{\rm src} \geq 0$ is adopted, which reflects the lack of knowledge of what the true rates might be (whilst allowing zero rates, unlike the logarithmic prior). This choice of prior also has the advantage that the posterior contour plots show the likelihood and, hence,

the constraining power of the PAO data directly. The one prior restriction assumed is that both $\Gamma_{\rm src}$ and $R_{\rm bkg}$ are non-negative.

- $\Pr(\text{data}|\Gamma_{\rm src}, R_{\rm bkg})$ is the likelihood of obtaining the measured data given particular values for $R_{\rm bkg}$ and $\Gamma_{\rm src}$.
- The integral in the denominator is the evidence. As we are not making any comparisons to other models, the only role that the evidence plays here is to ensure that the posterior is correctly normalised. Hence it can be ignored when investigating the shape of the posterior.

Applying the above simplifications, Eq. (7.3) reduces to

$$\Pr(\Gamma_{\rm src}, R_{\rm bkg}|\text{data}) \propto \Theta(\Gamma_{\rm src})\Theta(R_{\rm bkg})\Pr(\text{data}|\Gamma_{\rm src}, R_{\rm bkg}), \tag{7.4}$$

where $\Theta(x)$ is the Heavyside step function.

Data parameterisation: a 'counts in cells' approach is employed, dividing the sky into $N_p = 180 \times 360 = 64800$ pixels distributed uniformly in right ascension and declination. The raw data take the form of the measured arrival directions, $\{\hat{\boldsymbol{r}}_c\}$, and number, N_c, of UHECRs. (It would also be possible to use the measured arrival energies of the UHECRs.) So the data are recast as the set of UHECR counts in each pixel, $\{N_{{\rm c},p}\}$. In the limit of infinitely small pixels, this is mathematically equivalent to using the arrival directions, but is more straightforward to analyse and simulate.

$\overline{N}_{{\rm bkg},p}$ and $\overline{N}_{{\rm src},p}$ are the expected number of background and source UHECRs in pixel p, respectively. The expected number of background UHECRs in pixel p is

$$\overline{N}_{{\rm bkg},p} = R_{\rm bkg} \int_p \frac{d\epsilon}{d\Omega}\, d\Omega_{\rm obs}, \tag{7.5}$$

where the integral is over the p'th pixel and $d\epsilon/d\Omega$ is the PAO exposure per unit solid angle (see Section 7.3.1). The expected number of UHECRs from known sources in pixel p is

$$\overline{N}_{{\rm src},p} = \sum_{s=1}^{N_s} \left[\left(\int_{E{\rm min-arr}}^{\infty} \frac{dN_{\rm arr}(E_{\rm arr}, D_s)}{dt\, dA} dE_{\rm arr} \right) \left(\int_p \frac{d\epsilon}{d\Omega} \Pr(\hat{\boldsymbol{r}}_{\rm arr}|\hat{\boldsymbol{r}}_{\rm s})\, d\Omega_{\rm obs}, \right) \right] \tag{7.6}$$

where the sum is over the AGN sources, $\Pr(\hat{\boldsymbol{r}}_{\rm arr}|\hat{\boldsymbol{r}}_{\rm s})$ is the smearing probability (Eq. (7.20)), and the first integral is the rate (number per area and time) of UHECRs

from a source at distance D_s arriving at Earth above the cut-off energy, $E_{\mathrm{min-arr}}$. This rate is proportional to the source rate, Γ_{src}, but further depends on both the shape of the AGN UHECR injection spectrum and the distance-dependence of the GZK energy losses, and so requires an explicit UHECR model (see Eq. (7.19)).

The positional dependence of $\overline{N}_{\mathrm{bkg},p}$ and $\overline{N}_{\mathrm{src},p}$ are both shown in Fig. (7.4). The right panel is a combination of both the PAO exposure and the local distribution of AGNs, although comparing the left and the right panel it is clear that the latter dominates. In particular, the strongest source by far is Centaurus A (with $l = 309°5$ and $b = 19°4$), which has previously been suggested as the dominant source of UHECRs (e.g. Abraham et al. 2007b).

The likelihood is a product of the independent Poisson likelihoods in each pixel, and is hence given by

$$\Pr(\{N_{\mathrm{c},p}\}|\Gamma_{\mathrm{src}}, R_{\mathrm{bkg}}) = \prod_{p=1}^{N_P} \frac{(\overline{N}_{\mathrm{bkg},p} + \overline{N}_{\mathrm{src},p})^{N_{\mathrm{c},p}} \exp[-(\overline{N}_{\mathrm{bkg},p} + \overline{N}_{\mathrm{src},p})]}{N_{\mathrm{c},p}!}. \qquad (7.7)$$

Section 7.4.1 describes the parts of the likelihood calculation which depend on our model of AGN UHECR production and propagation. In Section 7.4.2, we simulate PAO UHECR data in order to investigate the properties of the likelihood function.

The fraction of UHECRs that have come from AGNs, F_{AGN} can be found from the expected number of source and background events in any sample, which can be calculated from the rates. The constraints on the expected UHECR numbers are simply proportional to those on the relevant rates; F_{AGN} is given by the ratio of the expected number of AGN UHECRs to the expected total number.

Note that it is crucial to begin by parameterising the problem with the fundamental physical quantities, the rates Γ_{src} and R_{bkg}, rather than the F_{AGN} (as done in Abreu et al. 2010). This is because in small samples, in which the total arrival rate of UHECRs has a significant Poisson uncertainty, the only way to consistently account for the (independent) fluctuations in the source and background UHECRs is to parameterise their rates explicitly.

Individual UHECR-source probabilities, P_{src}, (i.e. whether any single UHECR came from a particular source) can be found within the Bayesian approach. A useful estimate of the probability that a UHECR, with measured arrival direction in pixel

p, has come from one of the sources under consideration is

$$P_{\rm src} = \Pr(\text{from source}|p, \Gamma_{\rm src}, R_{\rm bkg}) = \frac{\overline{N}_{{\rm src},p}}{\overline{N}_{{\rm src},p} + \overline{N}_{{\rm bkg},p}}, \tag{7.8}$$

given values for the two rates. As the rates inferred from a sample of even just 27 UHECRs are not sensitive to any one event, it is reasonable to evaluate $P_{\rm src}$ using the best-fit values of $\Gamma_{\rm src}$ and $R_{\rm bkg}$ to assess the likely origin of each UHECR in turn. This is done for the PAO data in Section 7.5.

7.4.1 UHECR Model

AGN source rate: it assumed here for simplicity that all AGNs emit UHECRs at the same overall rate and with an energy flux proportional to $E^{-\gamma}$, i.e.

$$J(E) = CE^{-\gamma} \tag{7.9}$$

where C is some constant and $\gamma = 3.6$ (Abraham et al., 2010). Therefore the number flux is given by

$$\frac{dN_{\rm emit}(E_{\rm emit})}{dtdA} = \frac{J(E_{\rm emit})}{E_{\rm emit}} = CE_{\rm emit}^{-\gamma-1} \tag{7.10}$$

and the total emission rate of UHECRs with energy greater than E is

$$\int_E^\infty \frac{dN_{\rm emit}(E_{\rm emit})}{dt} dE_{\rm emit} = CA \int_E^\infty E_{\rm emit}^{-\gamma-1} dE_{\rm emit} = \frac{CA}{\gamma} E^{-\gamma}, \tag{7.11}$$

where A is the area over which UHECRs are emitted.

The source rate is defined as

$$\Gamma_{\rm src} = \int_{E{\rm min-emit}}^\infty \frac{dN_{\rm emit}(E_{\rm emit})}{dt} dE_{\rm emit} = \frac{CA}{\gamma} E_{\rm min-emit}^{-\gamma}. \tag{7.12}$$

Here, $\Gamma_{\rm src}$ is the same for all AGNs, although a more realistic model would allow $\Gamma_{\rm src}$ to vary with source; it is plausible that the UHECR emission rate scales with an AGN's hard X-ray luminosity (e.g., Protheroe & Szabo 1992). Now Eq. (7.11) becomes

$$\int_E^\infty \frac{dN_{\rm emit}(E_{\rm emit})}{dt} dE_{\rm emit} = \Gamma_{\rm src} \left(\frac{E}{E_{\rm min-emit}} \right)^{-\gamma}. \tag{7.13}$$

AGN luminosity: the definition of $\Gamma_{\rm src}$ also means that the UHECR luminosity can be cast as

$$L_{\rm src} = \int_{E{\rm min-emit}}^{\infty} L(E_{\rm emit}) dE_{\rm emit} = \int_{E{\rm min-emit}}^{\infty} AJ(E_{\rm emit}) dE_{\rm emit} \\ = \frac{\gamma}{\gamma - 1} \Gamma_{\rm src} E_{\rm min-emit}. \tag{7.14}$$

GZK effect: the dominant energy loss mechanism of UHECRs is the GZK interaction with the CMB photons. Whilst this is stochastic, its most important feature is the exponential reduction in the probability of an UHECR travelling more than about 100 Mpc without dropping below $E_{\rm GZK}$. This can be accounted for by adopting a continuous loss approximation (cf. Achterberg et al. 1999) in which a UHECR's arrival energy is given by

$$E_{\rm arr} = \max \left[E_{\rm GZK}, E_{\rm emit} (1 - f_{\rm GZK})^{D/L_{\rm GZK}} \right], \tag{7.15}$$

where D is the distance to the source, $f_{\rm GZK} = 0.2$ is the average fractional energy loss per GZK interaction, $L_{\rm GZK} = 4$ Mpc is the GZK mean free path. It is also assumed that there are no further energy losses once a CR reaches $E_{\rm GZK}$, although this is unimportant for UHECRs with $E_{\rm min} > E_{\rm GZK}$ (such as those in the PAO sample).

AGN arrival rate: the quantity that we want is the arrival rate (per unit area and time) of UHECRs with arrival energy above $E_{\rm min-arr}$ from a source at distance D. For $E_{\rm arr} > E_{\rm GZK} \simeq 5 \times 10^{19}$ eV, which is the case for the PAO data in question, Eq. (7.15) can be used to express this quantity as

$$\int_{E\mathrm{min-arr}}^{\infty} \frac{dN_{\mathrm{arr}}(E_{\mathrm{arr}}, D)}{dtdA} dE_{\mathrm{arr}}$$

$$= \int_{E\mathrm{min-arr}}^{\infty} \frac{1}{4\pi D^2} \frac{dN_{\mathrm{emit}}(E_{\mathrm{arr}}/(1-f_{\mathrm{GZK}})^{D/L_{GZK}})}{dt} dE_{\mathrm{arr}} \tag{7.16}$$

$$= \frac{(1-f_{\mathrm{GZK}})^{\gamma D/L_{GZK}}}{4\pi D^2} \int_{\frac{E\mathrm{min-arr}}{(1-f_{\mathrm{GZK}})^{D/L_{GZK}}}}^{\infty} \frac{dN_{\mathrm{emit}}(E_{\mathrm{emit}})}{dt} dE_{\mathrm{emit}}.$$

Using Eq. (7.13) in Eq. (7.16) yields

$$\int_{E\mathrm{min-arr}}^{\infty} \frac{dN_{\mathrm{arr}}(E_{\mathrm{arr}}, D)}{dtdA} dE_{\mathrm{arr}}$$

$$= \Gamma_{\mathrm{src}} \frac{(1-f_{\mathrm{GZK}})^{\gamma D/L_{GZK}}}{4\pi D^2} \left(\frac{E_{\mathrm{min-arr}}}{E_{\mathrm{min-emit}}(1-f_{\mathrm{GZK}})^{D/L_{GZK}}} \right)^{-\gamma}. \tag{7.17}$$

Since $E_{\mathrm{arr}} > E_{\mathrm{GZK}}$ in the PAO sample then, according to the simple GZK model, Eq. (7.15),

$$E_{\mathrm{min-arr}} = E_{\mathrm{min-emit}}(1-f_{\mathrm{GZK}})^{D/L_{\mathrm{GZK}}} \tag{7.18}$$

which, in Eq. (7.17), finally gives

$$\int_{E\mathrm{min-arr}}^{\infty} \frac{dN_{\mathrm{arr}}(E_{\mathrm{arr}}, D)}{dtdA} dE_{\mathrm{arr}} = \Gamma_{\mathrm{src}} \frac{(1-f_{\mathrm{GZK}})^{\gamma D/L_{GZK}}}{4\pi D^2}. \tag{7.19}$$

This can be used in Eq. (7.6) to calculate the expected number of CRs in each pixel and, therefore, the likelihood function, Eq. (7.7).

Deflection: the final ingredient to Eq. (7.6) is the deflection probability. The arrival directions are measured accurate to about 1°, although there is an additional effective uncertainty in the progenitor direction as UHECRs are deflected by Galactic and inter-galactic magnetic fields. The combined effect can be modelled by defining the probability distribution of observed arrival directions of UHECRs from a source at $\hat{\boldsymbol{r}}_{\mathrm{src}}$ as a two-dimensional Gaussian on the sphere,

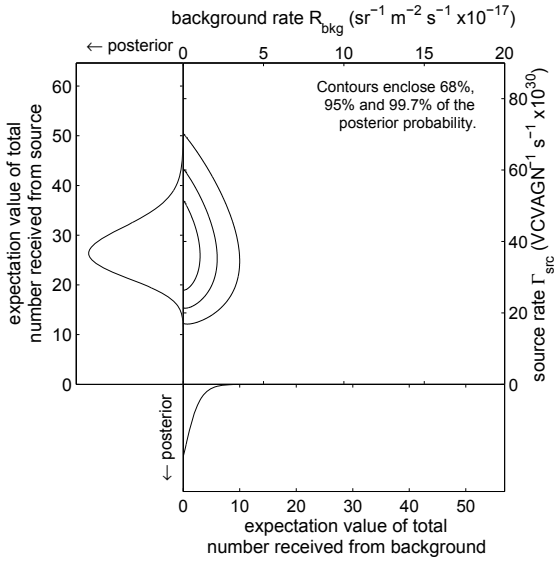


Figure 7.5: The posterior probability of the UHECR rate from VCV AGNs, $\Gamma_{\rm src}$, and the uniform background rate, $R_{\rm bkg}$, implied from a simulated sample of 27 UHECRs, all of which were emitted by VCV AGNs. The contours enclose 68%, 95% and 99.7% of the posterior probability, and the line plots show the marginalised probability for each rate.

$$\Pr(\hat{\boldsymbol{r}}_{\rm arr}|\hat{\boldsymbol{r}}_{\rm src}) = \frac{1}{2\pi\sigma^2(1-e^{-2/\sigma^2})} \exp\left(-\frac{1-\hat{\boldsymbol{r}}_{\rm arr}\cdot\hat{\boldsymbol{r}}_{\rm src}}{\sigma^2}\right). \tag{7.20}$$

A fiducial smearing angle of $\sigma = 3^\circ$ is assumed unless otherwise stated, but results using $\sigma = 6^\circ$ and $\sigma = 10^\circ$ are also calculated for comparison purposes.

7.4.2 Simulations

It is useful to test the constraining power of a small number of UHECRs by generating mock PAO samples with known progenitor properties. Simulations of the two extreme cases were created (credit: Daniel Mortlock). In the AGN-only sample, all the UHECRs were sourced from the nearby VCV AGNs and propagated using the simple GZK model described in Section 7.4.1. In the all-background sample, the arrival directions are random. In both cases, the incident UHECRs were subject to the PAO's measurement errors and declination-dependent exposure. Both samples were constrained to have exactly 27 events so as to provide parameter constraints that can be compared directly with those from the real PAO sample.

The results of the AGN-only simulation are shown in Fig. (7.5). As expected, the constraints on $\Gamma_{\rm src}$ match the naive Poisson expectation. Note the rejection of the possibility that more than a few of the PAO UHECRs are not from the

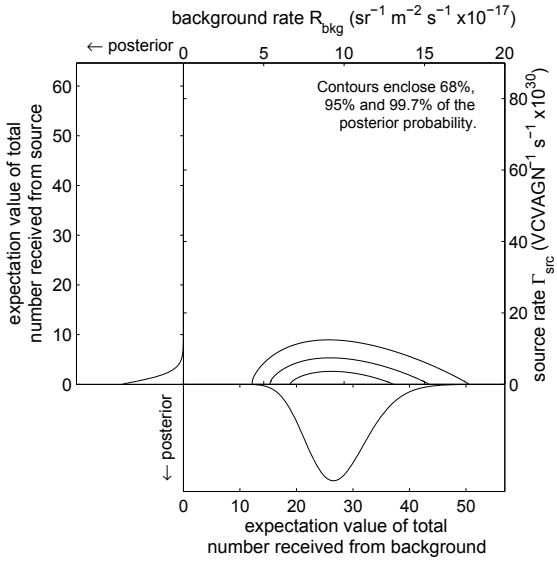


Figure 7.6: Same as Fig. (7.5), but for a simulated sample of 27 isotropically distributed UHECRs.

VCV AGNs. The constraints on the AGN fraction (see Fig. (7.9)) from such a data-set would be $F_{\rm AGN} = 1.00^{+0.00}_{-0.07}$, where the intervals enclose the most probable 68% of the posterior probability. This strong result implies that if AGNs source all UHECRs, even a sample of 27 events would be sufficient to confirm this hypothesis, if a complete sample of the progenitors was available.

The results of the background-only simulation are shown in Fig. (7.6). Again, the constraints on $R_{\rm bkg}$ match the Poisson expectation and this time the possibility of more than a few UHECRs arriving from AGNs is rejected. The resultant constraints on the AGN fraction (see Fig. (7.9)) are $F_{\rm AGN} = 0.00^{+0.07}_{-0.00}$. It is also important to note that some pixels (very far away from any AGN) have negligible contribution from the VCV AGNs and, because some of the UHECRs in this sample fell in those pixels, there is an absolute hard upper bound on $F_{\rm AGN}$ that is significantly lower than unity.

The fact that the posteriors from the AGN-only and the background-only simulations are almost completely disjoint implies that even a sample of just 27 UHECRs might be sufficient to provide a definitive answer as to their origin. The parameter constraints from the real data should lie between the two extremes shown in Fig. (7.5) and Fig. (7.6).

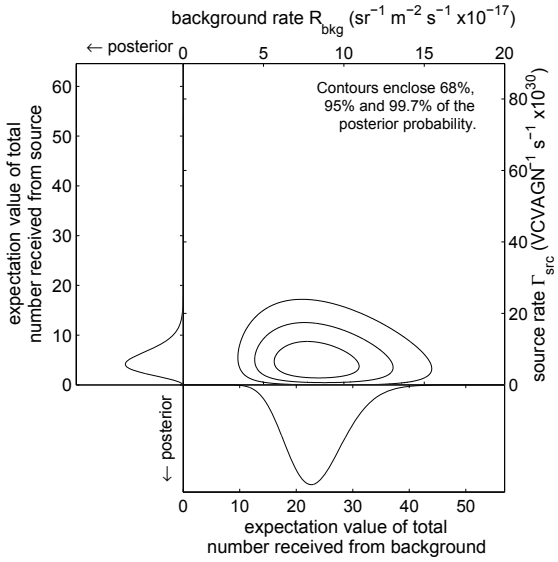


Figure 7.7: Same as Fig. (7.5), but for all 27 PAO UHECRs.

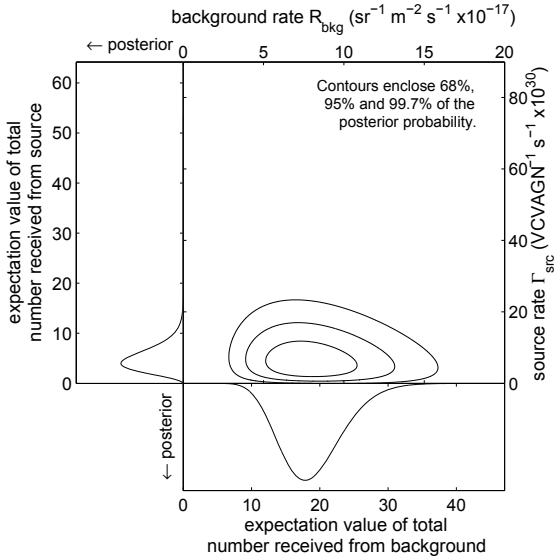


Figure 7.8: Same as Fig. (7.5), but for the 22 PAO UHECRs with arrival directions at least 10° from the Galactic plane.

7.5 Results

The posterior probability distribution in $\Gamma_{\rm src}$ and $R_{\rm bkg}$ given the PAO UHECR sample is shown in Fig. (7.7). As expected, the posterior is intermediate between the extreme cases shown in Fig. (7.5) and Fig. (7.6). The most likely rates are $\Gamma_{\rm src} = (5.8^{+4.0}_{-2.9}) \times 10^{30}$ s^{-1} (equivalent to UHECR source luminosity of $L_{\rm src} = 7.4^{+5.1}_{-3.7} \times 10^{31}$ W) and $R_{\rm bkg} = (8.0^{+1.9}_{-1.6}) \times 10^{-17}$ sr^{-1} m^{-2} s^{-1}. Also calculated is the posterior distribution of the fraction of the PAO UHECRs that come from VCV AGNs, shown in Fig. (7.9). The most probable value is $F_{\rm AGN} = 0.15$, and the constraints can be summarised by the interval $F_{\rm AGN} = 0.15^{+0.10}_{-0.07}$ (where, again, the limits enclose the most likely 68% of the posterior probability).

As most extragalactic catalogues are incomplete close to the Galactic plane, the above analysis was repeated on a reduced data-set from which the region with Galactic latitudes of $|b| \leq 10^\circ$ had been removed (see Fig. (7.8)). The PAO exposure in the retained regions is 7480 yr km^2 sr and the number of UHECRs included was reduced from 27 to 22. The lower numbers resulted in slightly broader constraints on $F_{\rm AGN}$, as can be seen from Fig. (7.9). From this cut data, we find $\Gamma_{\rm src} = (5.6^{+3.9}_{-2.8}) \times 10^{30}$ s^{-1}, $R_{\rm bkg} = (7.6^{+2.0}_{-1.7}) \times 10^{-17}$ sr^{-1} m^{-2} s^{-1} and $F_{\rm AGN} = 0.18^{+0.11}_{-0.09}$.

The analysis was also repeated using larger mean smearing angles of $\sigma = 6^\circ$ and $\sigma = 10^\circ$. The limits on the AGN fraction in these models are $F_{\rm AGN} = 0.22^{+0.12}_{-0.09}$ ($\sigma = 6^\circ$) and $F_{\rm AGN} = 0.31^{+0.12}_{-0.12}$ ($\sigma = 10^\circ$). In both cases, the most probable value of $F_{\rm AGN}$ is higher, although the range of values compatible with the data is broader, than in the fiducial model. It is natural that a higher AGN fraction be compatible with the data given larger values of σ, as a greater fraction of the sky is within σ of at least one source. This effect has been seen by, e.g., Kim & Kim 2011 and Abraham et al. 2010. In particular, Kim & Kim 2011 report the fraction of observed UHECRs that originate from AGNs to be 0.45 for a smearing angle of 6°. However, the best-fit value of $F_{\rm AGN}$ increases less strongly with σ in the Bayesian formalism presented in this chapter than found by using other methods. This is because the inherent self-consistency of the Bayesian approach ensures that the correct balance is struck between the compatibility of this more forgiving model and the lack of predictivity.

There is strong evidence of a UHECR signal from the known VCV AGNs, which manifests in the result that $F_{\rm AGN} = 0.15$ is 200 times more probable than $F_{\rm AGN} = 0.00$, but not all the PAO UHECRs can be explained this way. Note that the

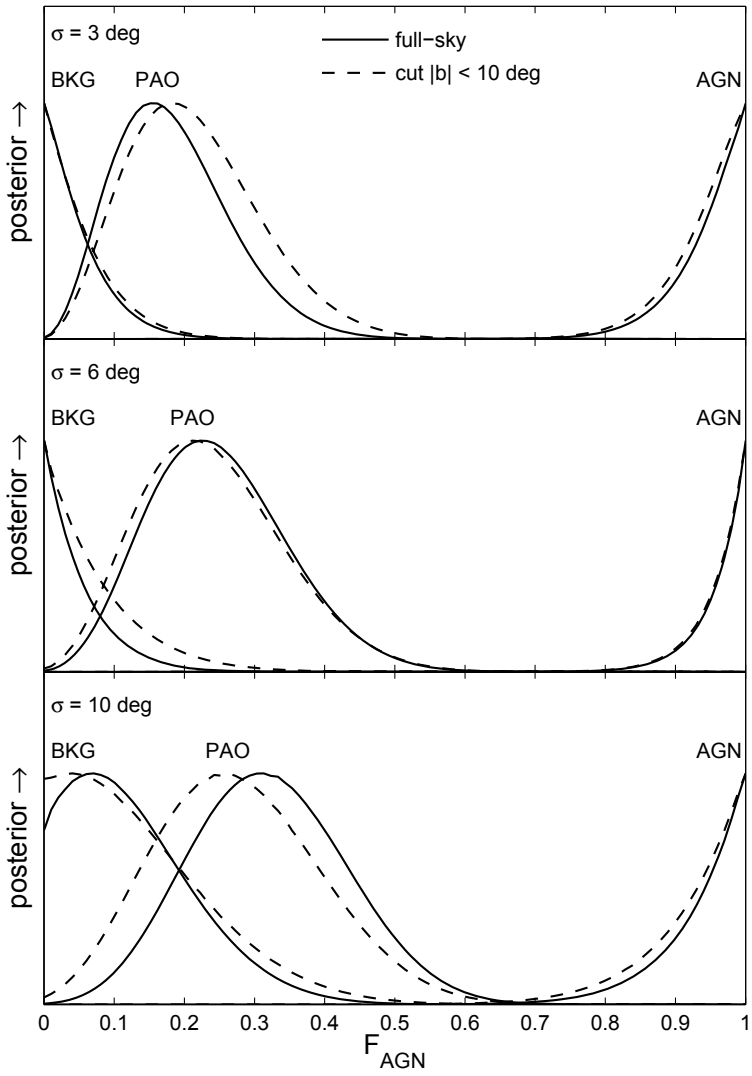


Figure 7.9: Posterior distributions of the fraction of observed UHECRs that are from the population of VCV AGNs, $F_{\rm AGN}$, shown for simulated samples (both isotropic and AGN-only) and for the real PAO data. Curves for both the full sample of 27 UHECRs and the cut sample of 22 UHECRs (with arrival directions at least 10° from the Galactic plane) are shown in the same panel. Each panel represents a different smearing angle.

background-only (i.e. $F_{\rm AGN} = 0$) case, the null hypothesis rejected by Abraham et al. 2007b, is actually reasonably consistent with the data. However, the hypothesis that all the PAO UHECRs come from VCV AGNs (i.e. $F_{\rm AGN} = 1$) is completely ruled out because there are several events with no plausible AGN progenitor in the VCV catalogue.

The probability that each of the 27 UHECRs came from one of the VCV AGNs was calculated explicitly according to Eq. (7.8) by adopting the best-fit values for $R_{\rm bkg}$ and $\Gamma_{\rm src}$ given above; these probabilities are given in Table 7.1. For $\sigma = 3^\circ$, only 9 events have $P_{\rm src} \geq\sim 0.1$, all of which were identified as being within 3.2° of an AGN with $z_{\rm obs} \leq 0.017$ by Abraham et al. 2007b. However the other 11 events which Abraham et al. 2007b identified as AGN correlated have very low values of $P_{\rm src}$. In most cases, this is because the angular correlation is with an AGN that is close to their maximum redshift and so has a significantly reduced UHECR flux at Earth. Moreover, 14 of the UHECRs have $P_{\rm src} < 0.001$, with no plausible AGN progenitor, at least within the VCV catalogue. As also shown in Table 7.1, the results are similar, but less conclusive, for larger smearing angles. The AGN hypothesis cannot be ruled out for the low $P_{\rm src}$ events, however: these UHECRs could have come from AGNs that are not in the VCV catalogue (and some could have come from VCV AGNs if deflected by more than a few degrees). Of course, $P_{\rm src}$ may not be so low for these events if our simple model is adjusted to allow the source rate $\Gamma_{\rm src}$ to vary with source.

7.6 Discussion

This chapter details a Bayesian analysis to test whether the first 27 UHECRs with $E_{\rm arr} \geq 5.7 \times 10^{19}$ eV detected by the PAO have come from the known local AGNs in the VCV catalogue. The first main conclusion from this analysis is that either at least some do come from VCV AGNs or at least some come from progenitors within a few degrees of the VCV AGNs. A more realistic model should help to distinguish between these two possibilities. Within the model presented here, the fraction of UHECRs that come from the VCV AGNs is constrained to be $0.15^{+0.10}_{-0.07}$ (where the limits enclose the most probable 68 percent of the posterior). Conversely, the second important result is that many of the PAO UHECRs have not come from AGNs in the VCV catalogue, either because of incompleteness (most obviously close to the Galactic plane) or because there is another source of UHECRs, possibly in our own Galaxy.

The results differ somewhat from those presented by Abraham et al. 2007b,

Table 7.1: The measured arrival directions of the 27 PAO UHECRs listed in Abraham et al. (2007b) along with their assessment of AGN correlation (PAO corr.) and values from the method presented in this chapter of the AGN progenitor probability (which is rounded to zero if less than 0.0005) for the three different smearing angles. The CRs marked with *1, *2 and *3 in the b column are those closest to Centaurus A, with angular separations of 0.9, 2.3 and 5.8 deg respectively.

l deg	b deg	PAO corr.	$P_{\rm src}$ $\sigma = 3$ deg	$P_{\rm src}$ $\sigma = 6$ deg	$P_{\rm src}$ $\sigma = 10$ deg
15.4	8.4	no	0.000	0.000	0.000
−50.8	27.6	yes	0.559	0.761	0.681
−49.6	1.7	yes	0.000	0.134	0.387
−27.7	−17.0	yes	0.099	0.067	0.033
−34.4	13.0	yes	0.078	0.171	0.424
−75.6	−78.6	yes	0.380	0.528	0.493
58.8	−42.4	yes	0.000	0.000	0.000
−52.8	14.1*3	yes	0.870	0.836	0.711
4.2	−54.9	yes	0.000	0.004	0.008
48.8	−28.7	yes	0.000	0.000	0.000
−103.7	−10.3	no	0.000	0.000	0.001
−165.9	−46.9	yes	0.000	0.003	0.010
−27.6	−16.5	yes	0.099	0.067	0.033
−52.3	7.3	no	0.167	0.533	0.577
88.8	−47.1	yes	0.000	0.000	0.002
−170.6	−45.7	yes	0.000	0.006	0.011
−51.2	17.2*2	yes	0.952	0.873	0.735
−57.2	41.8	no	0.005	0.123	0.294
63.5	−40.2	yes	0.000	0.000	0.000
−51.4	19.2*1	yes	0.964	0.881	0.742
−109.4	23.8	yes	0.000	0.000	0.002
−163.8	−54.4	yes	0.001	0.006	0.020
−41.7	5.9	no	0.002	0.208	0.454
12.1	−49.0	yes	0.000	0.001	0.003
−21.8	54.1	yes	0.000	0.005	0.088
−65.1	34.5	no	0.000	0.049	0.321
−125.2	−7.7	no	0.001	0.002	0.002

because more explicit models of background and source events are used here, as well as different statistical methods. The starting point of their analysis is the null hypothesis that the UHECRs have not come from local AGNs; they find that this is rejected 'at the 99% level' given the number of the UHECRs that are within 3° of a VCV AGN. The strength of the correlation makes it clear that there is at least some connection between the two populations. But it is impossible to go beyond this limited statement due to the use of arbitrary cuts in their correlation analysis (both in angular radius and AGN redshift); the equal weighting of the nearest known AGN, Centaurus A, with the hundreds of AGNs at distances of about 100 Mpc, and the equal value placed on any angular match out to 3°, dilutes whatever correlation signal is present. The simulations of AGN-only and background-only UHECR samples shown here demonstrate that even a sample of just 27 events is in fact sufficient to decisively distinguish between these two extreme possibilities, but that the apparent strong correlation inferred by Abraham et al. 2007b is in part due to the analysis method.

During the final preparation of the paper on this work (Watson et al. 2011), the Pierre Auger Collaboration presented an extended analysis of an enlarged set of 69 UHECRs (Abreu et al. 2010). Aside from the correlation-based methods they had used previously, they also included a likelihood-based formalism that has some similarities to the method presented in this chapter. The results of the two likelihood approaches are broadly similar (and differ from the earlier correlation-based analyses), primarily because they both include a physical model of UHECR propagation. They hence go closer to the ideal of including all the available information (i.e. not just the data, but knowledge of the CR physics) and so produce more robust results.

There are several extensions to the analysis of even the 27 PAO UHECRs that might allow stronger conclusions regarding the origins of these particles. Most importantly, the energy of individual CRs can be accounted for in the likelihood, rather than just demanding they are above the $E_{\mathrm{min-arr}} = 5.7 \times 10^{19}$ eV cut. This will make it more important to use a more realistic, stochastic calculation of the GZK effect, as well as the energy dependent deflection due to magnetic fields. It will similarly be more important to investigate the possibility that the AGN UHECR emission rate scales with AGN luminosity; a corollary is that it may be possible to discriminate between different AGN emission models.

The strength of any such inferences will be increased as the PAO continues to take data, increasing the number of detected UHECRs. However, it is also possible that even the current data-set could be used more efficiently by including lower

energy events. This would obviously increase the numbers, although there is the potentially severe penalty of diluting the angular signal by including UHECRs that have either been deflected by more than about 10° or have come from the many AGNs at distances greater than about 100 Mpc. These trade-offs can be evaluated objectively (to the degree that the CR propagation and deflection models are accurate), following the underlying principle of extracting as much information as possible from the UHECR measurements.

Another way to potentially make better use of the existing PAO UHECRs would be to use a more homogeneous AGN sample than the VCV catalogue. One such example is the catalogue of AGNs from the *Swift* Burst Alert Telescope (BAT) survey, which has nearly uniform selection criteria outside the Galactic plane. Both George et al. 2008 and the latest PAO analysis from Abreu et al. 2010 compare UHECR data to this catalogue. In particular, George et al. 2008 approach the analysis in a fashion similar to that of Abraham et al. 2007b and found correlation at the '98% level'. It would be valuable to apply the Bayesian method discussed here to this data-set.

There are still many problems facing the determination of the origin of UHECR events, but the Bayesian approach offers clear advantages in tackling these issues over other statistical methodologies. The fully Bayesian analysis demonstrated in this work provides encouraging results and presents a strong case for developing the method further.

Chapter 8

CMB Probability Density Functions

We want to find the probability of a specific cosmology, Ω, and topology, Ξ, given a set of CMB data, $\mathbf{d}$, i.e.

$$\Pr(\Omega, \Xi|\mathbf{d}) = \frac{\Pr(\mathbf{d}|\Omega, \Xi)\Pr(\Omega, \Xi)}{\int d(\Omega, \Xi)\Pr(\mathbf{d}|\Omega, \Xi)\Pr(\Omega, \Xi)}. \tag{8.1}$$

This can be recast in terms of the correlation matrix, which depends on cosmology and topology $C_{\ell m \ell' m'} = C_{\ell m \ell' m'}(\Omega, \Xi) = C^{\Omega,\Xi}_{\ell m \ell' m'}$, although care must be taken to check for degeneracy, where a correlation matrix corresponds to more than one set of cosmological and topological parameters. First, a note on notation: depending on the context, $C_{\ell m \ell' m'}$ (or, in pixel space, $C_{pp'}$) can be read as the entire matrix or as an individual element of the matrix with specific $\ell m \ell' m'$ (or pp'). To avoid ambiguity, $C_{\ell m \ell' m'}$ ($C_{pp'}$) will denote only a matrix element and $\mathbf{C}$ the entire matrix for the rest of this thesis. In terms of the correlation matrix, Eq. (8.1) becomes

$$\boxed{\Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{d}) = \frac{\Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi})\Pr(\Omega, \Xi)}{\int d\mathbf{C}^{\Omega,\Xi}\Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi})\Pr(\Omega, \Xi)} \quad \textbf{CMB posterior}.} \tag{8.2}$$

The main term of interest is the likelihood,

$$\boxed{\Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi}) \quad \textbf{CMB likelihood}.} \tag{8.3}$$

This is because the eventual goal is to perform Bayesian model comparison and so any terms that are the same for different (Ω, Ξ) are of little interest. This is the case for the prior, $\Pr(\Omega, \Xi)$, if we take it to be uniform in Ω and Ξ. A uniform

prior in Ξ is appropriate given our lack of knowledge of topology; see Section 6.1.1). Regardless of the choice of prior, the evidence (the denominator in Eq. (8.2)) is the the same for all (Ω, Ξ). Hence, in this case, the only term in Eq. (8.2) which varies with (Ω, Ξ) is the likelihood.

The specific form of the CMB likelihood is detailed in Section 8.1. Section 8.2 describes methods for sampling from the likelihood.

8.1 The CMB Likelihood

The usual choice of CMB likelihood is a multivariate Gaussian (see, e.g., Jaffe et al. 1999). In matrix form, the general expression of a multivariate Gaussian probability distribution is

$$\Pr(\mathbf{d}|\boldsymbol{\mu}, \mathbf{D}) = \mathrm{G}(\mathbf{d} - \boldsymbol{\mu}, \mathbf{D}) = \frac{1}{|2\pi\mathbf{D}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{d} - \boldsymbol{\mu})^{\dagger}\mathbf{D}^{-1}(\mathbf{d} - \boldsymbol{\mu})\right] \tag{8.4}$$

where $\mathbf{d}$ is a vector of variables d_i, $\boldsymbol{\mu}$ is a vector containing the mean values μ_i of the variables d_i, and $\mathbf{D}$ is a covariance matrix containing the correlations D_{ij} of variables d_i and d_j. Recast directly in terms of the variables d_i, and the corresponding μ_i and D_{ij}, Eq. (8.4) becomes

$$\begin{aligned}\Pr(\{d_i\}|\{\mu_i\}, \{D_{ij}\}) \\ = \frac{1}{\left(2\pi \sum_{i_1,\ldots,i_n=1}^{n} \left(\epsilon_{i_1\ldots i_n} \prod_{k=1}^{n} D_{k\, i_k}\right)\right)^{1/2}} \exp\left[-\frac{1}{2}\sum_{i,j}(d_i - \mu_i)D^{-1}_{ij}(d_j - \mu_j)\right]\end{aligned} \tag{8.5}$$

where $\epsilon_{i_1\ldots i_n}$ is the Levi-Civita, or permutation, symbol and i labels ℓm or p (j labels $\ell' m'$ or p') according to whether we are working in harmonic or pixel space.

8.2 Sampling Methods

The CMB likelihood and, hence, posterior cannot generally be found analytically and finding them numerically (i.e. sampling values of the likelihood at various points in the parameter space) is computationally expensive. So it is worth careful consideration of which sampling methods to use. If uniformly interested in the dis-

tribution over the entire parameter space, one can sample the likelihood or posterior over some uniform grid (Section 8.2.1). If, however, one is interested in only the maximum likelihood or posterior, a random walk algorithm such as Markov Chain Monte Carlo (MCMC; Section 8.2.2) sampling may be more efficient. Gridding is the simpler, more accurate route, but MCMC sampling is (if done correctly) less expensive.

8.2.1 Gridding

Gridding involves sampling directly from the likelihood at intervals in the parameter space. The explicit form of the likelihood is given here, building from a simplistic, pure CMB signal, case to a more realistic situation that accounts for noise and masking of foregrounds.

Pure signal: Let us first consider the (unrealistic) case where the measured data $\mathbf{d}$ are pure CMB signal $\mathbf{s}$. The covariance matrix for $\mathbf{d}$, $\mathbf{D}$, is simply equal to that for $\mathbf{s}$, $\mathbf{C}^{\Omega,\Xi}$. Also, since we are dealing with temperature anisotropies, not absolute temperature, the mean, $\boldsymbol{\mu}$, is zero. Hence the Gaussian distribution of Eq. (8.4) directly provides us with the CMB likelihood Eq. (8.6):

$$\Pr(\mathbf{d}|\mathbf{D}) = \Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi}) = \mathrm{G}(\mathbf{d}, \mathbf{C}^{\Omega,\Xi}). \tag{8.6}$$

Pure signal plus noise: The introduction of noise complicates matters. The noise $\mathbf{n}$ is assumed to be Gaussian with some mean $\overline{\mathbf{n}}$ and covariance matrix $\mathbf{N}$. We now have data $\mathbf{d} = \mathbf{s}+\mathbf{n}$, with covariance matrix $\mathbf{D} = \mathbf{C}^{\Omega,\Xi}+\mathbf{N}$. Now, Eq. (8.4) becomes

$$\Pr(\mathbf{d}|\mathbf{D}) = \Pr(\mathbf{d}|\overline{\mathbf{n}}, \mathbf{C}^{\Omega,\Xi}, \mathbf{N}) = \mathrm{G}(\mathbf{d} - \overline{\mathbf{n}}, \mathbf{C}^{\Omega,\Xi} + \mathbf{N}). \tag{8.7}$$

This is not the CMB likelihood. To get the likelihood, we must marginalise over the noise, i.e.

$$\Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi}) = \int d\overline{\mathbf{n}}\, d\mathbf{N}\, \Pr(\mathbf{d}|\overline{\mathbf{n}}, \mathbf{C}^{\Omega,\Xi}, \mathbf{N})\Pr(\overline{\mathbf{n}}, \mathbf{N}). \tag{8.8}$$

However, if the mean and covariance matrix of the noise are known, we can assign a delta function to the joint prior $\Pr(\overline{\mathbf{n}}, \mathbf{N})$. Then, upon integrating Eq. (8.8), we get

$$\Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi}) = \Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi}, \overline{\mathbf{n}}, \mathbf{N}) = \mathrm{G}(\mathbf{d} - \overline{\mathbf{n}}, \mathbf{C}^{\Omega,\Xi} + \mathbf{N}). \tag{8.9}$$

where $\overline{\mathbf{n}}$ and $\mathbf{N}$ are now fixed and part of the background information I.

Signal plus noise with masks (or cut sky): If we are not using a map "cleaned" of foregrounds, we must remove the parts that are contaminated using a mask (see Section 5.4.2). Pixels cut from the data using a mask should also be cut from the covariance and noise matrices. Once the cuts have been made, the likelihood can be obtained through Eq. (8.9).

8.2.2 MCMC Sampling

Markov Chain Monte Carlo (MCMC) sampling is often adopted in order to find the maximum likelihood of a likelihood distribution. The difference between MCMC and a general Monte Carlo method is:

- Monte Carlo method: very broad term used to describe algorithms which sample numerical results from a distribution at random, most commonly using a random walk approach.
- Markov Chain Monte Carlo (MCMC) method: a Monte Carlo method that is designed to converge on stationary points in the distribution of interest. See Section 8.2.2.1.

An MCMC approach often adopted for CMB analyses is Gibbs sampling (e.g. Wandelt et al. 2004, Eriksen et al. 2004 for temperature; Larson et al. 2007 for polarisation). However, there is a low signal-to-noise inefficiency problem with the Gibbs sampler at high ℓ. This is addressed by Jewell et al. 2009, who build upon previous Gibbs sampling methods by combining with a Metropolis-Hastings sampling approach. A further extension (Rudjord et al. 2009) uses a modified Blackwell-RAO (BR) estimator to improve the characterisation of the joint posteriors used in Gibbs sampling. In summary, the different sampling methods and estimators involved are:

- Gibbs method: an MCMC method for multi-variate probability distributions which samples from a joint probability distribution of two or more variables when sampling from the full distribution is difficult. See Section 8.2.2.2.
- Metropolis-Hastings method: an MCMC method for multi-variate probability distributions for which the form of neither the full nor joint probability distributions are well known.
- Blackwell-RAO estimator (BR): is an observable quantity that is used to estimate an unobservable quantity (e.g. the probability of a bus arriving in a given period of time is not observable, but the number arriving in that time is),

where the estimation of the unobservable can be improved by taking its expectation value given a sufficient statistic of the observable (e.g. we can measure the number of buses arriving during many time intervals (observable) to find a mean rate of arrival (statistic) that can be used to estimate the probability of a bus arriving (unobservable)).

8.2.2.1 Simple MCMC Sampling

A simple MCMC code follows the following steps:

(i) Choose seed(s), or initial value(s), for the parameter(s) for which you wish to find the maximum likelihood.

(ii) Take a random step away from the current parameter value(s) to find candidate value(s):

- This step is typically drawn from a gaussian distribution of appropriate standard deviation. (However, the choice of distribution to draw from can be influenced by knowledge of the distribution of the parameter(s) that the MCMC is exploring.)
- Add the random step(s) to the current parameter values(s) to yield candidate values(s).

(iii) Test for which value(s) you wish to keep, the current or the candidate. Accept the candidate if:

- The likelihood of the candidate is greater than that of the current value(s).
- The candidate likelihood is smaller than that of the current but greater than the current value multiplied by a random number drawn from a uniform distribution between 0 and 1. (This step can help reduce the risk of the code getting "stuck" in a local maximum.)

If the candidate is rejected, keep the current value(s). If it is accepted, the candidate becomes the current values(s)

(iv) Repeat steps (ii) and (iii) until the routine has converged on a value for the maximum likelihood.

The following can affect the values obtained for the maximum likelihood:

- The **seed(s)**: a seed which lies near a local maximum, which is not the global maximum, can result in convergence to the local instead of global maximum.

- The **standard deviation** of the gaussian distribution from which the parameter steps are drawn: this affects the step size between iterations - too big and it may miss the maximum, too small and the risk of getting stuck at a local maximum is increased.
- The **number of iterations**: if this is set too low, the routine may not converge on a maximum. However, setting it too high is needlessly computationally expensive.

In the case of the CMB likelihood, we may want to explore the likelihood as a function of the orientation of the topology, i.e. Euler angles α, β and γ. The MCMC process here would be:

(i) Choose seed values for α, β and γ, e.g.:

- $\alpha = \alpha_{\text{seed}}$
- $\beta = \beta_{\text{seed}}$
- $\gamma = \gamma_{\text{seed}}$

(ii) Draw a random step from a guassian distribution of given standard deviation for each parameter and add to the respective parameter to obtain a candidate value:

- $\alpha_{\text{candidate}} = \alpha + r_\alpha$
- $\beta_{\text{candidate}} = \beta + r_\beta$
- $\gamma_{\text{candidate}} = \gamma + r_\gamma$

Note that r_α, r_β and r_γ must be re-drawn every time this step is performed.

(iii) If $\Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi(\alpha_{\text{candidate}()},\beta_{\text{candidate}},\gamma_{\text{candidate}})}) \geq \Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi(\alpha,\beta,\gamma)})$:

- $\alpha \rightarrow \alpha_{\text{candidate}}$
- $\beta \rightarrow \beta_{\text{candidate}}$
- $\gamma \rightarrow \gamma_{\text{candidate}}$

Else, if $\Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi(\alpha_{\text{candidate}()},\beta_{\text{candidate}},\gamma_{\text{candidate}})}) \geq r \times \Pr(\mathbf{d}|\mathbf{C}^{\Omega,\Xi(\alpha,\beta,\gamma)})$:

- $\alpha \rightarrow \alpha_{\text{candidate}}$
- $\beta \rightarrow \beta_{\text{candidate}}$
- $\gamma \rightarrow \gamma_{\text{candidate}}$

Else α, β and γ remain unchanged.
Note that r is a random number drawn from a uniform distribution between 0 and 1 which must be re-drawn every time this step is performed.

(iv) Repeat steps (ii) and (iii) for n iterations, or until the routine has converged on a value for the maximum likelihood.

This process should be repeated for a selection of different seeds.

If the maximum likelihood is found, the values of α, β and γ that correspond to it yield the correlation matrix $\mathbf{C}^{\Omega,\Xi(\alpha,\beta,\gamma)}$ for which the data $\mathbf{d}$ is most likely. Note that, in this example, we have varied only the orientation of the topology: the cosmology, topology type and dimensions of the fundamental domain have been fixed.

8.2.2.2 Gibbs Sampling

Gibbs sampling works with the joint posterior $\Pr(\mathbf{C}^{\Omega,\Xi}, \mathbf{s}|\mathbf{d})$, instead of $\Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{d})$. This may not appear any easier to deal with. But the premise of Gibbs sampling is that one can sample from this joint distribution by alternately sampling from the conditional distributions $\Pr(\mathbf{s}|\mathbf{C}^{\Omega,\Xi}, \mathbf{d})$ and $\Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{s}, \mathbf{d})$, i.e. start with some estimate of the correlation matrix, $\mathbf{C}^{(\Omega,\Xi)\,0}$, then iterate as follows:

$$\mathbf{s}^{i+1} \leftarrow \Pr(\mathbf{s}|\mathbf{C}^{\Omega,\Xi\,i}, \mathbf{d}) \tag{8.10}$$

$$\mathbf{C}^{\Omega,\Xi\,i+1} \leftarrow \Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{s}^{i+1}, \mathbf{d}) \propto \Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{s}^{i+1}). \tag{8.11}$$

where, in Eq. (8.11), $\Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{s}, \mathbf{d}) = \Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{s})\Pr(\mathbf{s}|\mathbf{d})$, but we are not interested in $\Pr(\mathbf{s}|\mathbf{d})$ since it is independent of the model. The form of the conditional density in Eq. (8.10) is a special case of the Wiener filter posterior density where the signal covariance $\mathbf{S}$ is taken to be $\mathbf{C}^{\Omega,\Xi}$:

$$\Pr(\mathbf{s}|\mathbf{C}^{\Omega,\Xi}, \mathbf{d}) \propto \mathrm{G}(\mathbf{C}^{\Omega,\Xi}(\mathbf{C}^{\Omega,\Xi} + \mathbf{N})^{-1}\mathbf{d}, ((\mathbf{C}^{\Omega,\Xi})^{-1} + \mathbf{N}^{-1})^{-1}). \tag{8.12}$$

The conditional density of Eq. (8.11) is proportional to a multivariate Gaussian:

$$\Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{s}) \propto \Pr(\mathbf{s}|\mathbf{C}^{\Omega,\Xi})\Pr(\mathbf{C}^{\Omega,\Xi}) = \Pr(\mathbf{C}^{\Omega,\Xi})\mathrm{G}(\mathbf{s}, \mathbf{C}^{\Omega,\Xi}). \tag{8.13}$$

Note that $\Pr(\mathbf{C}^{\Omega,\Xi}|\mathbf{s})$ in Eq. (8.13) is usually taken to be an inverse gamma distribution (Wandelt et al. 2004). However, this is when working with the power spectrum

(C_ℓ), for which there is a $C_{\ell m \ell' m'}$ sample of size $2\ell + 1$. The gamma distribution involves averaging over the sample and so is not appropriate when the sample size is only one, as is the case when working with the correlation matrix.

8.2.3 Choice of Sampling Method

As seen in Section 8.2.2.2, Gibbs methods are not ideal for full correlation matrix. In this thesis, a combination of gridding and simple MCMC is adopted. The CMB likelihood tends to have a narrow peak, which means care must be taken with MCMC methods not to miss it.

Part IV

Constraining Topology with the Polarised CMB

Chapter 9

Prescription

Tools for constraining cosmic topology using a Bayesian approach were discussed in Ch. 6 and Ch. 8; the best constraints should be achieved by utilising the complete correlation matrix of the CMB. Surprisingly few attempts have been made to do this due to the process being computationally expensive. Many analyses have been performed using the CMB power spectrum (a reduced form of the correlation matrix), but this ignores valuable information. Analyses such as Niarchou (2006) have made use of the full correlation matrix, but very few have made use of the information contained in the polarisation data of the CMB (an example where this has been done is Aslanyan et al. (2013), but with the focus on constraining the size of the Universe for each topology, rather than constraining topology itself). This is because, up until recently, the best measurements of CMB polarisation have been obtained by WMAP, and these are rather noisy. The most recent CMB space observatory, Planck, promises much improved measurements. While these data are yet to be released, it makes the case for utilising CMB polarisation even stronger, as any methods that are developed for WMAP data could quickly be applied to Planck data upon release.

In this thesis, a method is adopted that harnesses the full CMB correlation matrix for both temperature and polarisation. This is described in Section 9.3 and applied in Ch. 10. Four flat spaces are investigated, the flat torus (E_1), half turn space (E_2), Klein space (E_7) and Klein space with vertical flip (E_9), the reasons for which are given in Section 2.3. The full range of cases explored here are listed in Section 9.2. Note that the noise in WMAP's polarisation measurements renders only E modes useable in our investigation, but the method described can also be applied to B modes. In fact, there are many extensions that could be made to this work; in the following sections, where methods and/or data used in this thesis are described, details of prudent extensions are also given.

9.1 Data Used in this Thesis

The data used here is the latest, nine year, WMAP data, which has been foreground reduced by the WMAP collaboration (Bennett et al. 2012) and is publicly available on NASA's CMB data website, LAMBDA (lambda.gsfc.nasa.gov). Since the maximum ℓ that we are using is 30, the maximum resolution will be an angular area of very roughly $(180/\ell)^2 \sim 36$ square degrees. The sky can be broken into 1,800 such regions of equal angular area (note that this translates into areas of differing physical size depending on how close to the pole/z-axis you are). The WMAP data used has a HEALPix resolution of 4 (Gorski et al. 1999), which is equivalent to dividing the sky into 3,072 equal physical-area pixels and is appropriate for the maximum ℓ used.

The data is provided in the form of Stokes' parameters I, Q and U, along with the effective number of observations N_{obs}, for each pixel. The signal data are easily converted into harmonic space T, E and B values (the respective $a_{\ell m}$s) using HEALPix. An inverse covariance matrix for noise (N^{-1}) is also provided. This is not convenient for our purposes, as we want N, and inverting matrices is very computationally expensive. We can (crudely) assume the noise to be negligible for T and E. However, the WMAP noise for B is too high to make it worth investigating here.

Finally, WMAP has five frequency bands. We will initially focus on the W band as it appears to have the strongest signal.

EXTENSIONS (FUTURE WORK)

Ext. **1.1** Use data from all WMAP frequency bands.

Ext. **1.2** Estimate the WMAP noise covariance matrix.

Ext. **1.3** Use Planck T and E data.

Ext. **1.4** Use Planck B data.

9.2 Cases Investigated in this Thesis

The cases investigated in this thesis are discussed below and summarised in Table 9.1.

Topologies: the topologies investigated in this thesis are E_1, E_2, E_7 and E_9 (see Section 2.3), the flat torus, half turn space, Klein space and Klein space with vertical flip respectively.

EXTENSIONS (FUTURE WORK)

Ext. **2.1** Explore all flat spaces.

Ext. **2.2** Explore spherical and hyperbolic spaces.

Cosmologies: for each topology, the Hubble parameter is varied between 0.685 and 0.701, in increments of 0.002. These values represent the constraints obtained from the nine year WMAP observations (Hinshaw et al. 2012). Ideally, a smaller increment, say 0.001 or less, would be used, but the value of 0.002 was chosen due to time constraints of the investigation.

Other cosmological parameters are fixed to the default values set by CAMB. The size of the Universe comes under geometry, and is technically another cosmological parameter. However, it is usually assumed to be infinite. So size is discussed separately below.

EXTENSIONS (FUTURE WORK)

Ext. **2.3** Test values of h at smaller intervals.

Ext. **2.4** Vary other cosmological parameters.

Sizes: for each topology and cosmology, the size of the fundamental domain is varied between 21 Gpc and 35 Gpc, in increments of 2 Gpc. This is because it has been shown that a lower limit on the size of the Universe is around 25 Gpc; Key et al. (2007) reported a value of 24 Gpc, and Planck Collaboration et al. (2013d) a value of 26.4 Gpc, but the value can vary with topology. So we pitch our lower limit a little under these estimates at 21 Gpc. The upper limit on the size detectable depends on the method used; circles in the sky cannot detect a topology bigger than the observable universe, but it may be possible for a Bayesian analysis of the CMB correlation matrix to detect a little beyond the size of the observable universe (Kunz et al. 2006). So we explore a little beyond the distance to the LSS. The distance to the last scattering surface is thought to be about 14 Gpc, giving a diameter of $L_{LSS} \sim 28$ Gpc. So the range of sizes investigated approximately corresponds to the range 0.75 to 1.25 L_{LSS}. Again, it would be preferable to use a smaller increment but the value used was chosen due to time constraints. Finally, any signatures of topology in the CMB will be stronger for smaller L; to aid understanding of the effects of the different topologies on the CMB, for $h = 0.6953$, we also investigate L from 10 to 18 Gpc (in increments of 2 Gpc).

Note that topologies with equal dimensions, $L_x = L_y = L_z = L$ are explored

Topologies	E_1, E_2, E_7, E_9
h	0.685, 0.687, 0.689, 0.691, 0.693, 0.695, 0.697, 0.699, 0.701
L (Gpc)	21, 23, 25, 27, 29, 31, 32, 35 (10, 12, 14, 16, 18) for $h = 0.693$ only
$\alpha_{\max}$	For E_1 and E_2: $\pi/2$ For E_7 and E_9: π
$\beta_{\max}$	For E_1: $\pi/2$ For E_2, E_7 and E_9: π
$\gamma_{\max}$	2π

Table 9.1: Values of parameters used in this thesis. Unless stated otherwise, all combinations of parameters are investigated.

here, but a natural extension would be to vary the proportions of the dimensions of the fundamental domain.

EXTENSIONS (FUTURE WORK)

Ext. **2.5** Test a broader range of values of L, and take values at smaller intervals.

Ext. **2.6** Try values of L_x, L_y, L_z which are not all equal to one another.

Orientations: for each topology, there is a different, infinite, set of unique orientations. The MCMC likelihood code is designed to determine the optimum orientation. The range of unique orientations depends on the symmetries of the topology. The orientations can be characterised by an Euler rotation α-β-γ, where α is an angle of rotation about the z-axis, β is about the original (unrotated) y-axis, and γ is about the original (unrotated) z-axis; the rotation order is α-β-γ. The ranges of these angles are given by the values of $\alpha_{\max}$, $\beta_{\max}$ and $\gamma_{\max}$ in Table 9.1 (we take the minimum values to be zero).

9.3 Method for Constraining Topology using Polarisation

In this section, we bring together the tools and parameters found in previous chapters to form a complete method for constraining topology using both temperature and polarisation measurements of the CMB. This is depicted as a flow diagram in Fig. (9.1).

The prescription for constraining topology used in this thesis consists of the

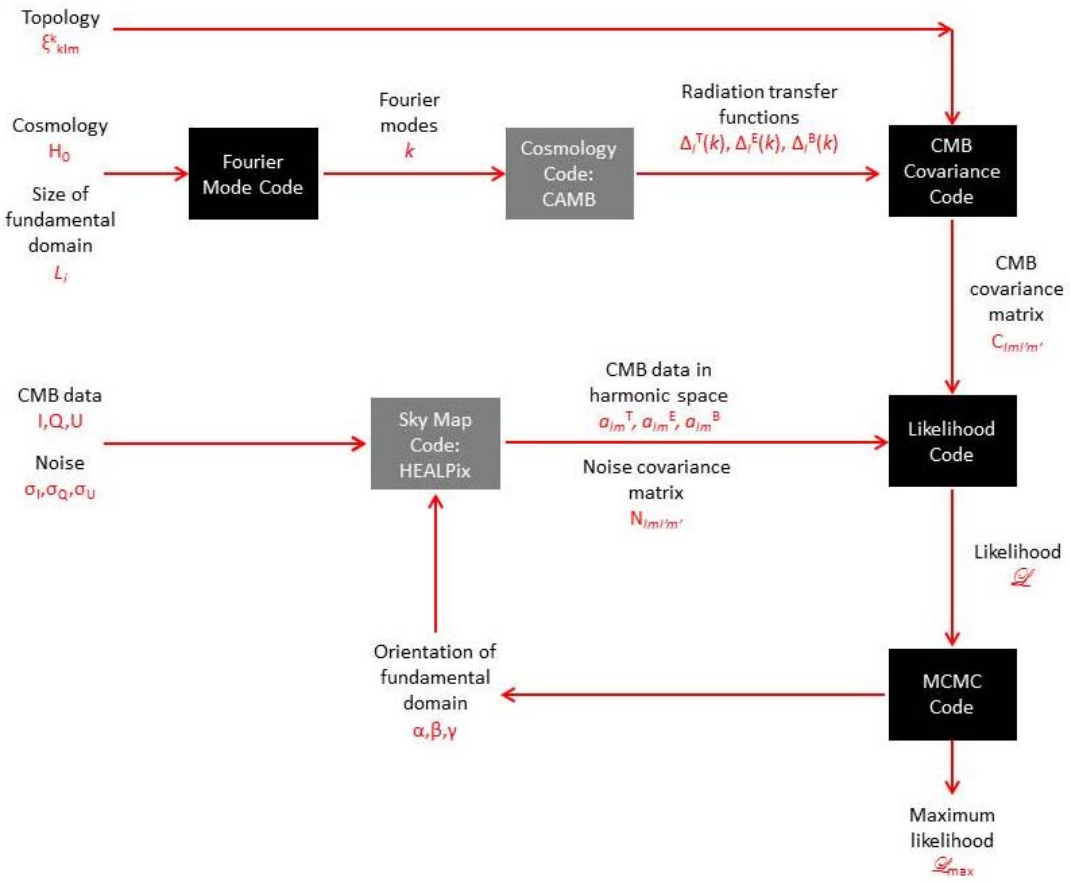


Figure 9.1: Flow diagram of method for constraining topology using the polarised CMB. The black boxes show code written as part of the work in this thesis. The Fourier mode and CMB covariance codes have been debugged but the likelihood and MCMC codes have not. Grey boxes depict freely available code from other sources. CAMB was modified to enable the Fourier modes to be read from file and the corresponding transfer functions to be written to file. A short routine was written in HEALPix to process the CMB data. Parameters, variables and data are shown in red, with the red arrows showing their progression through the code.

following steps:

1. Generate allowed Fourier modes: the first task is to calculate the Fourier modes, $\mathbf{k}$, which satisfy:

$$\mathbf{k} = 2\pi\mathbf{n} = 2\pi\left(\frac{n_x}{L_x}, \frac{n_y}{L_y}, \frac{n_z}{L_z}\right) \tag{9.1}$$

where L_i are the lengths of the sides of the fundamental domain.

Setting all L_i equal to L is the simplest place to start exploring these dimensions, and allows us to set all $n_{i,\max}$ equal to $n_{\max}$. The allowed values of n_i determine how straight-forward this step is. For spaces E_1, E_2, E_7 and E_9, n_i are always integers, and the effect of group generators, if anything, is only to change the sign of n_i. Note that, for the purposes of the next stage, which uses the publicly available code CAMB, only the magnitude k is required, which should be expressed in units of the Hubble parameter, h.

EXTENSIONS (FUTURE WORK)

Ext. **3.1** In order to investigate other topologies, non-integer n_i must be considered and/or n_i that experience more complicated transformations under the group generators.

Ext. **3.2** In order to try values of L_x, L_y, L_z which are not all equal to one another, $n_{i,\max}$ should be varied accordingly.

2. Generate radiative transfer functions: this is done through CAMB, which had to be modified to print out the transfer functions for a user-defined set of k values. The cosmological parameters are fixed to default values, with the exception of h, for which a range of values is explored (see Section 9.2). Transfer functions are calculated for all values of ℓ between 2 and 50.

3. Calculate correlation matrix: the major code written as part of the work of this thesis is that which calculates the full CMB temperature and polarisation correlation matrix for a given topology and cosmology. The algorithm is similar to that used by Phillips & Kogut (2006), which follows the following equation:

$$\mathbf{C}^{XY,L}_{\ell m,\ell' m'} = (4\pi)^2 \sum_n \Delta^{X*}_\ell(k_n)\Delta^Y_{\ell'}(k_n)\,\frac{\mathcal{P}(k)}{k^3}\,A^L_{\ell m,\ell' m'}(n) \tag{9.2}$$

where

$$A^L_{\ell m,\ell' m'}(n) = \sum_{\{|\mathbf{n}|=n\}} \xi^{\mathbf{n}*}_{k\ell m} \xi^{\mathbf{n}}_{k\ell' m'}. \tag{9.3}$$

These equations are described in more detail in Section 5.2.

EXTENSIONS (FUTURE WORK)

Ext. **3.3** Accommodate the different types of n_i required by Ext. **3.1** and Ext. **3.2** .

Ext. **3.4** Calculate TB, EB and BB matrices at the same time as the TT, TE and EE matrices that are currently calculated.

4. Convert CMB data into appropriate form: this is done using publicly available software HEALPix, Gorski et al. (1999). The form that the data must take for use in the likelihood code is spherical harmonic multipole moments $a_{\ell m}$ for T, E and B. See Section 9.1 for more detail about the data used in this thesis. The orientation of the data can also easily be changed using HEALPix.

5. Calculate likelihood: this is another code written for the work in this thesis. The most challenging part of the writing this code was solving y for $yC = d$, where C is the covariance matrix and d is the data. The code was tested using simple values of C and d to confirm that it did in fact solve for y correctly. Large determinants can take a long time to calculate, so the denominator of the likelihood is set to one. This should not affect the shape of the likelihood for the Euler angles, but will make it harder to compare other parameters, such as h. This likelihood function is implemented in a simple MCMC code that follows the prescription given in Ch. 8 in order to try to identify the maximum likelihood.

EXTENSIONS (FUTURE WORK)

Ext. **3.5** Perform more extensive MCMC investigations, trying more seeds and iterations. An appropriate convergence test would be helpful.

Ext. **3.6** Try using high resolution gridding (computationally expensive).

Ext. **3.7** Once a more detailed likelihood distribution is found, normalise the likelihood so that constraints can be attempted on L and h.

Ext. **3.8** Try using publicly available code MultiNest (Feroz et al. 2009) to determine likelihood - this code can calculate the evidence, which is needed for a proper comparison of different topologies.

6. Analysis: in order to aid the interpretation of the results, simulated data

is produced. This data is simple to generate using the lower triangle correlation matrix, $A_{\rm CMB} = A_{\ell m \ell' m'}$:

- Generate a vector of random numbers, $\mathbf{r}$, drawn from a standard normal distribution.
- Convert $\mathbf{r}$ into a vector of appropriately correlated numbers, or simulated CMB signal data, $\mathbf{s}_{\rm sim}$ via the Cholesky decomposition; $\mathbf{s}_{\rm sim} = A_{\rm CMB}\mathbf{r}$.

EXTENSIONS (FUTURE WORK)

Ext. **3.9** Run simulated data through the likelihood code to provide a guide as to what the likelihood distribution may look like.

Ext. **3.10** Produce detailed likelihoods plots, including marginalisations of parameters and confidence levels for values of parameters.

9.4 Covariance Matrix Code

This section mainly focuses on the performance of the code for calculating the CMB covariance matrices; for the code itself, see appendix Ch. A.

9.4.1 Hardware and Software Specifications

The machine used for calculating the covariance matrix is an entry level server with Intel Xeon series E3 quad core processors, a high capacity HDD drive and a 256 GB SSD, along with 24 GB of RAM. We chose not to access high powered computing facilities as a local machine grants the user more flexibility and autonomy. The operating system is Linux Debian and the compiler is the GNU C++ compiler. These selections were made because they are freely available and may encourage others to use and expand the code.

9.4.2 Calculations, Storage and Time

Phillips & Kogut (2006) find and store the topoterms from Eq. (9.3) before calculating the correlation matrix, Eq. (9.2). They argue that this provides a computational advantage as the topoterms are independent of cosmology and L and therefore only need finding once for a particular topology (and ratio L_x:L_y:L_z).

In our work, preliminary tests revealed that an element of the correlation matrix converges to at least four significant figures for $n_{\rm max} = 100$; for integer-valued

n_x, n_y and n_z, this corresponds to 4.2×10^6 allowed combinations of n_x, n_y and n_z. It has also been suggested that $\ell_{\max} = 30$ is sufficient for detecting the signature of a topology in the CMB via a statistical analysis of the CMB correlation matrix, although an $\ell_{\max}$ of 40 or 50 may be preferable (see. e.g., Kunz et al. 2006); starting from $\ell = 2$, this yields 957 ℓ-m combinations. Therefore, if each topoterm is a complex double, 16 bytes, then the storage required for the topoterms of one topology (and ratio L_x:L_y:L_z), is 114 GB. (Note that, if a higher $\ell_{\max}$ of, say, 50 is required for some topologies, this becomes 791 GB).

For the simple case of the flat torus, calculating the correlation matrix involves summing over all combinations of n_x-n_y-n_z, ℓ-m and ℓ'-m', each time taking the product of two spherical harmonics. That means calculating $4.2 \times 10^6 \times 2 \times 957 \times 957/2 = 1.9 \times 10^{14}$ spherical harmonics (division by two accounts for the fact that only combinations where $m+m'$ is even is allowed). On our machine, each spherical harmonic calculation takes 440 ns (actually 1.76 μs, but four calculations can be done at the same time). Hence, the total time to calculate the spherical harmonics would be 1.7×10^6 s, or 20 days. If we were to store all these harmonics as complex doubles, each taking up 16 bytes, the total required storage would be 57,000 GB.

This demonstrates the point that care must be taken not to unnecessarily repeat calculations. If we store the spherical harmonics, the number of calculations required can be dramatically reduced. There are three reasons for this: harmonics calculated for ℓ-m can be re-used for ℓ'-m'; due to symmetry between m and $-m$, only harmonics with $m \geq 0$ need be stored; only the ratios n_x:n_y:n_z are needed to calculate the harmonics, and many combinations n_x-n_y-n_z share the same ratio. This reduces the calculation time to just 12 minutes per correlation matrix, and requires 25 GB for storage of the spherical harmonics. (For $\ell_{\max} = 50$, the reduction would be 5 months and 420,000 GB to 33 minutes and 67 GB).

So far, we have worked out the time to calculate the spherical harmonics, but have not taken into account the time it takes to read the stored values. On a HDD with a typical data access rate of 100 MB.s^{-1}, the access time would be 50 hours. However, if we could store all the harmonics in RAM, with an access rate of 12800 MB.s^{-1}, the access time would be reduced to 35 minutes. (In fact, this could be reduced further to 18 minutes if we use a motherboard with dual channel capability, which are now commonly available). Our machine has 24 GB RAM, which is not enough to store 25 GB of spherical harmonics. Instead, the harmonics are permanently stored on an SSD and read into RAM in batches (see appendix Ch. A for more detail on how this is done), which has an insignificant effect on the total access time. The code is

designed to check the RAM available before deciding what size batches to use; this means that, if the RAM was increased to 32 GB (usually the maximum motherboard capacity), all the spherical harmonic data could be in RAM simultaneously.

The total time for the calculation of a single correlation matrix is now dominated by looping over the necessary parameters to find the allowed values of n_x-n_y-n_z and perform the sums in Eq. (9.2) and Eq. (9.3). The time taken varies slightly according to the complexity of the ξs associated with each topology (Ch. 2), but for all topologies explored here (E_1, E_2, E_7 and E_9) is well within a day (or two days, for).

Finally, the values of h and L explored in this thesis are fairly restricted, but the number of combinations is still 77. With three different types of covariance matrix to calculate (TT, TE and EE), the total number of covariance matrices calculated for each topology is 231. The most efficient way to find all these matrices is to do so simultaneously, so that the topoterms need only be calculated once. A small chunk of each matrix is kept in RAM at any one time, and written to the HDD once completed, freeing up the RAM for the next chunk. This is quite efficient, and all 231 correlation matrices for a particular topology are calculated within a day (or two days, for $\ell_{\max} = 50$). Each correlation matrix file is quite small: 14 MB for $\ell_{\max} = 30$; 103 MB for $\ell_{\max} = 50$.

The advantage of this approach over the one used by Phillips & Kogut (2006) is that little permanent storage is needed. The spherical harmonics can be stored (at 25 GB for $\ell_{\max} = 30$, or 67 GB $\ell_{\max} = 50$) but only take about 30 minutes to pre-calculate. In contrast, Phillips & Kogut (2006) require 114 GB, or 791 GB, permanent storage. On a standard HDD, these would take 20 minutes, or 2.5 hours, to read. The lack of storage required by the method presented in this thesis makes it more portable - it can be quickly and easily shared.

9.4.3 Testing

Values generated by the code were tested at several stages. This involved comparing the output with values calculated in Mathematica (this is generally very accurate, but is not a fast as C++, and is not freely available). The same tests were also checked against outputs from the freely available SciLab. The terms tested were:

- spherical harmonics
- topoterms

- elements of the covariance matrix

All the tests showed agreement to at least six significant figures. Mathematica and SciLab were, however, noticeably slower than the C++ code.

Chapter 10

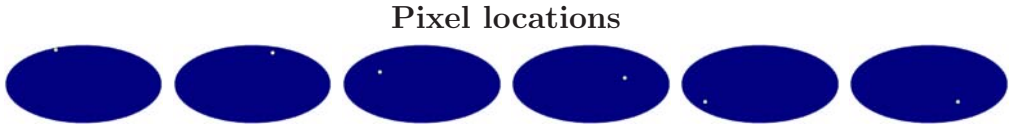
Results

10.1 Covariance Matrices

This section contains maps of the covariance matrices generated by the code described in Ch. 9. A map consists of one row of the covariance matrix in pixel space, for a particular pixel p. Fig. (10.1) shows the location on the sky of pixels for which the covariance matrix is plotted. The pixels are numbered according to the HEALPix nest system, with a HEALPix resolution of 4.

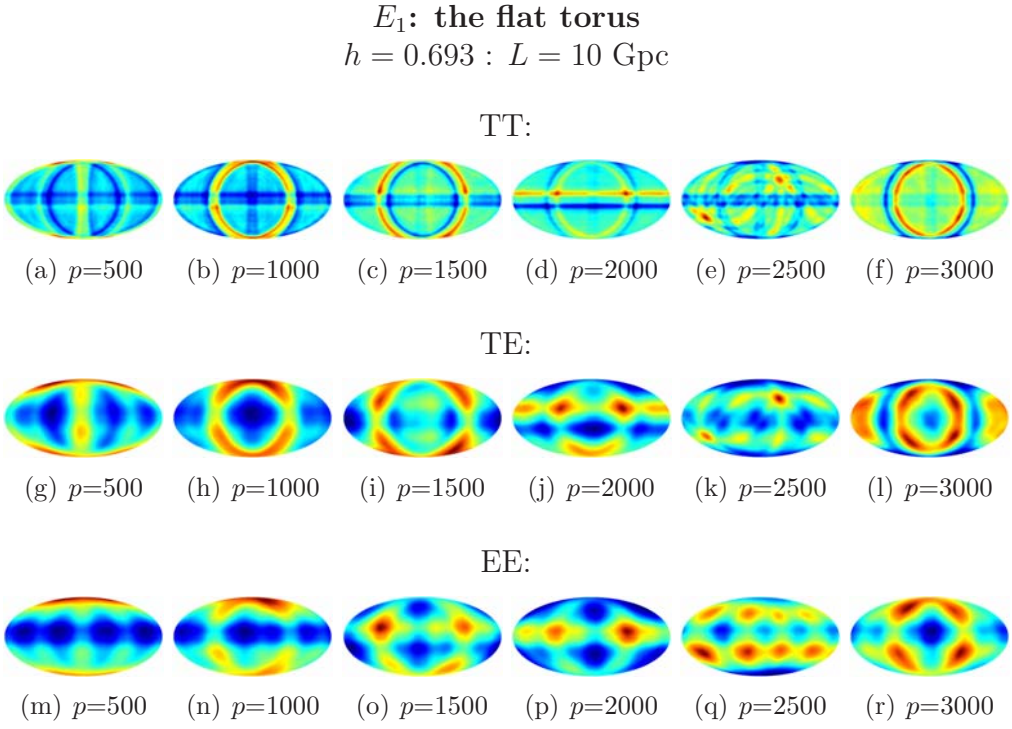
Fig. (10.2) to Fig. (10.13) each displays a set of these maps. Each figure corresponds to a particular topology-h-L combination and displays maps for TT, TE and EE covariance matrices. The values of (h,L), where L is in Gpc, for which the maps are given are (0.693,10), (0.693,25) and (0.701,25). The is little difference between maps which differ in h only. L, however, has a large influence over the appearance of the maps. Smaller values of L should contain a stronger signature of topology, and this appears to be the case here.

The flat torus maps are consistent with what one would expect; a region of high probability where the pixel that the map corresponds to resides, along with circular regions of higher correlation. The effects of other topologies are harder to visualise, but half turn space should contain similar features to the flat torus, which is the case with these maps.



(a) p=500 (b) p=1000 (c) p=1500 (d) p=2000 (e) p=2500 (f) p=3000

Figure 10.1: Maps showing the locations of pixels 500, 1000, 1500, 2000, 2500 and 3000.



(a) p=500 (b) p=1000 (c) p=1500 (d) p=2000 (e) p=2500 (f) p=3000

(g) p=500 (h) p=1000 (i) p=1500 (j) p=2000 (k) p=2500 (l) p=3000

(m) p=500 (n) p=1000 (o) p=1500 (p) p=2000 (q) p=2500 (r) p=3000

Figure 10.2: Maps of the correlation matrices for the flat torus, E_1, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

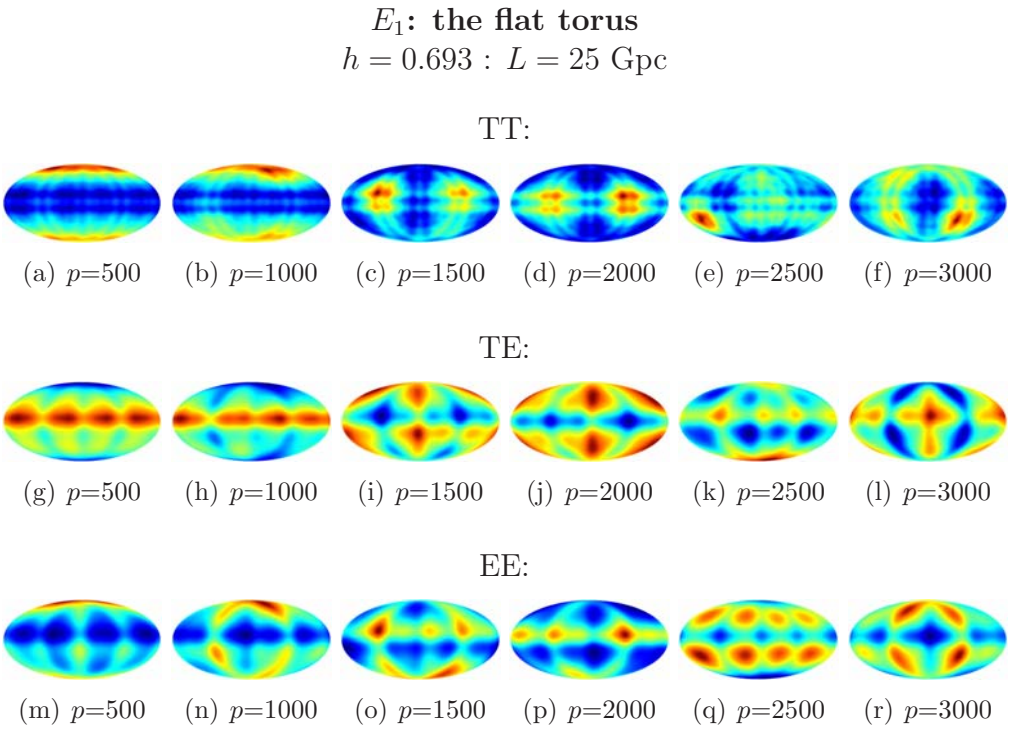


Figure 10.3: Maps of the correlation matrices for the flat torus, E_1, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

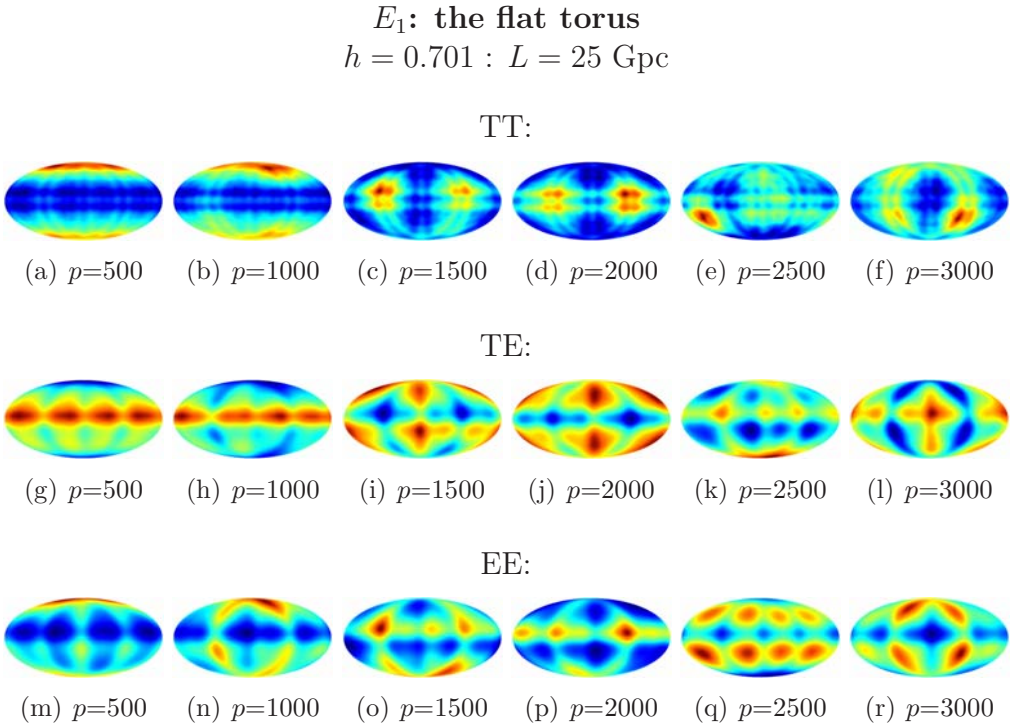


Figure 10.4: Maps of the correlation matrices for half turn space, E_2, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

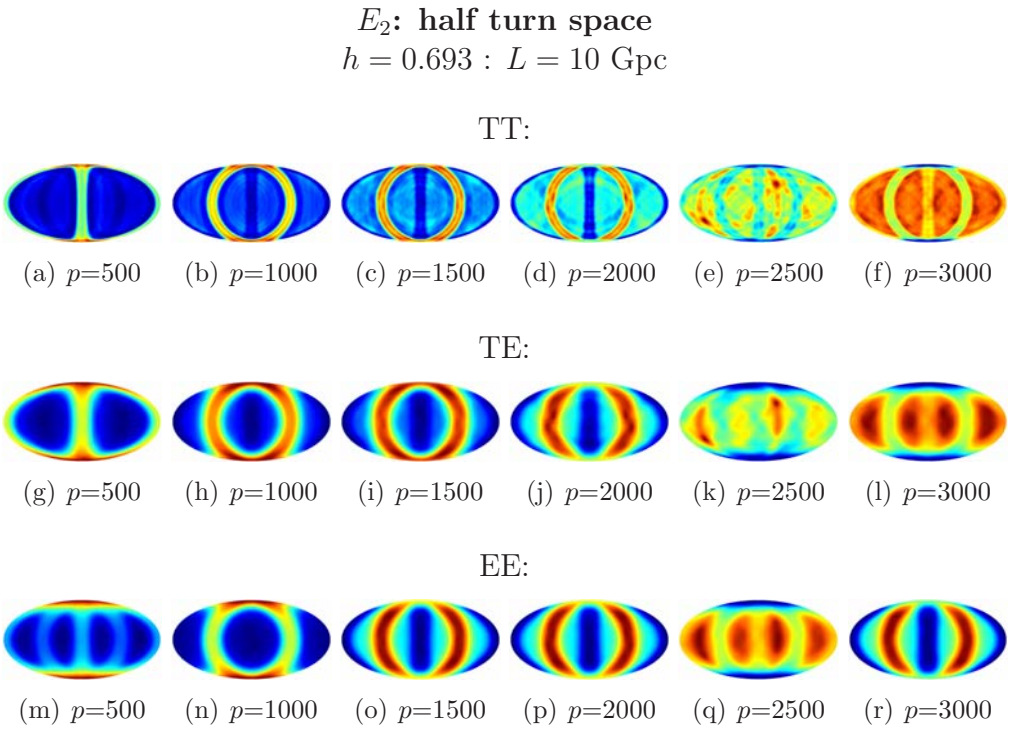


Figure 10.5: Maps of the correlation matrices for the flat torus, E_1, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

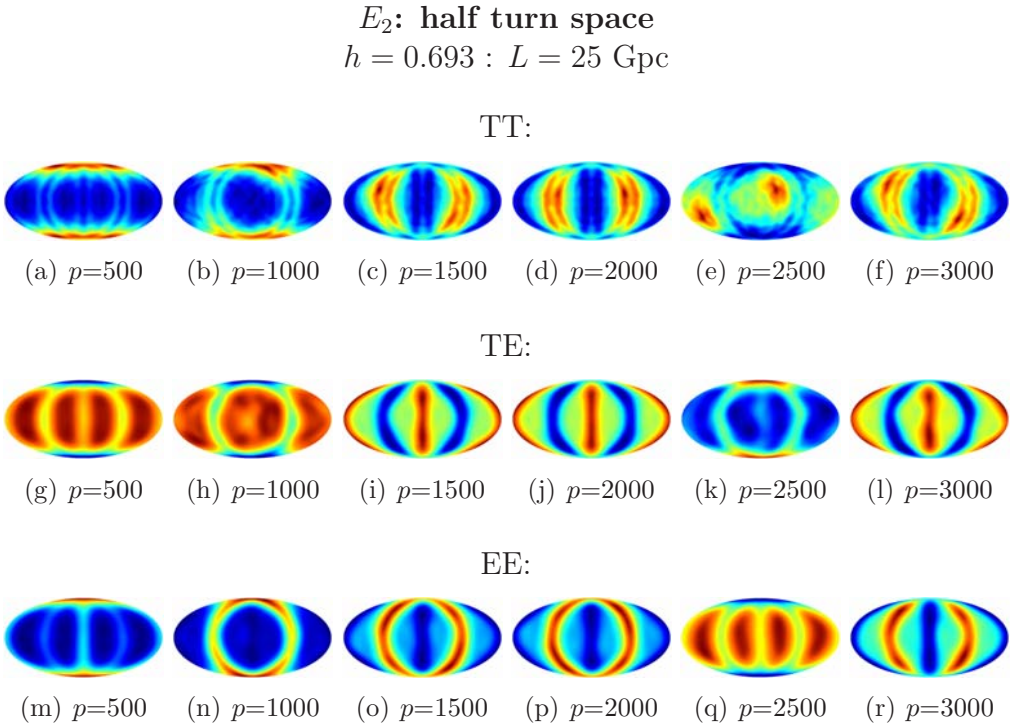


Figure 10.6: Maps of the correlation matrices for half turn space, E_2, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

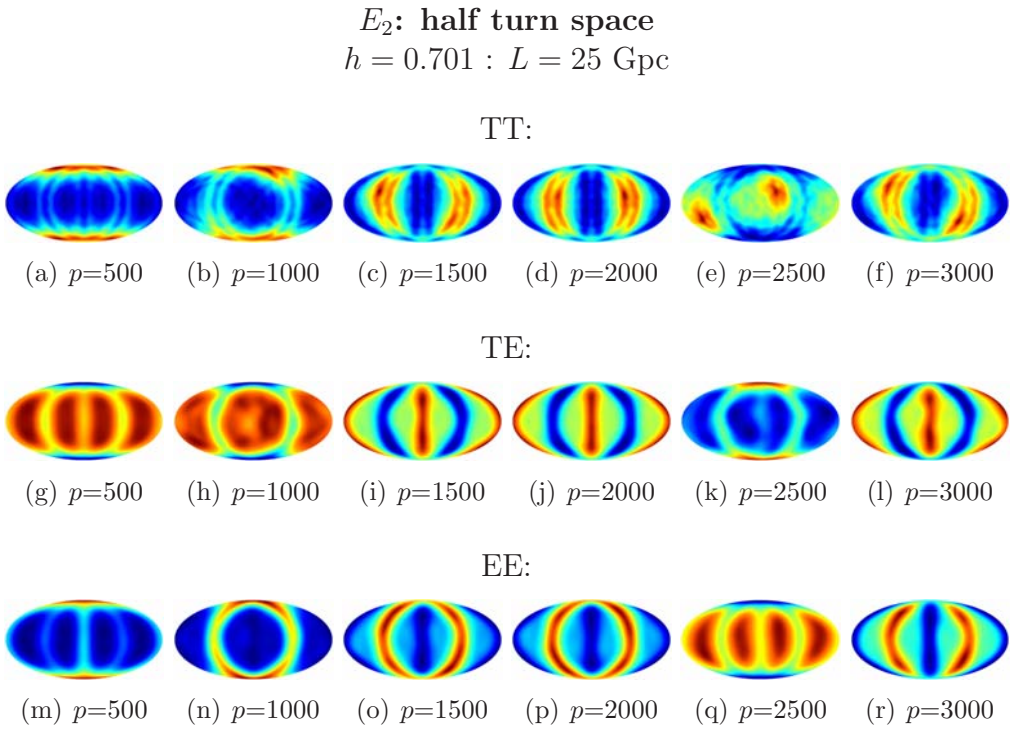


Figure 10.7: Maps of the correlation matrices for half turn space, E_2, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

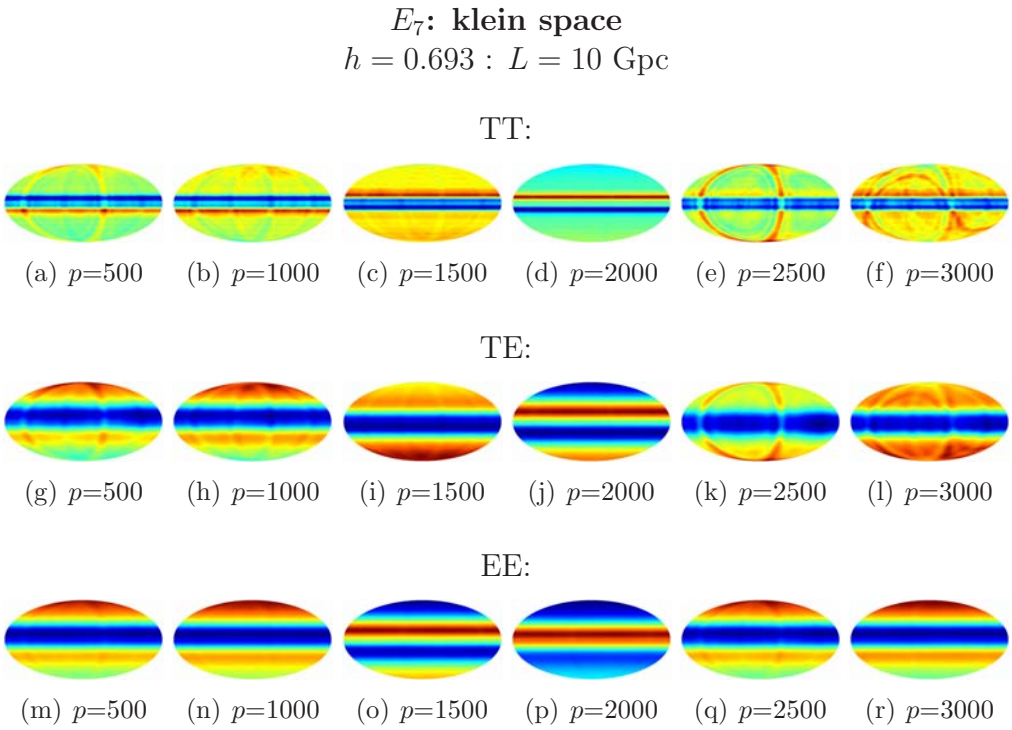


Figure 10.8: Maps of the correlation matrices for klein space, E_7, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

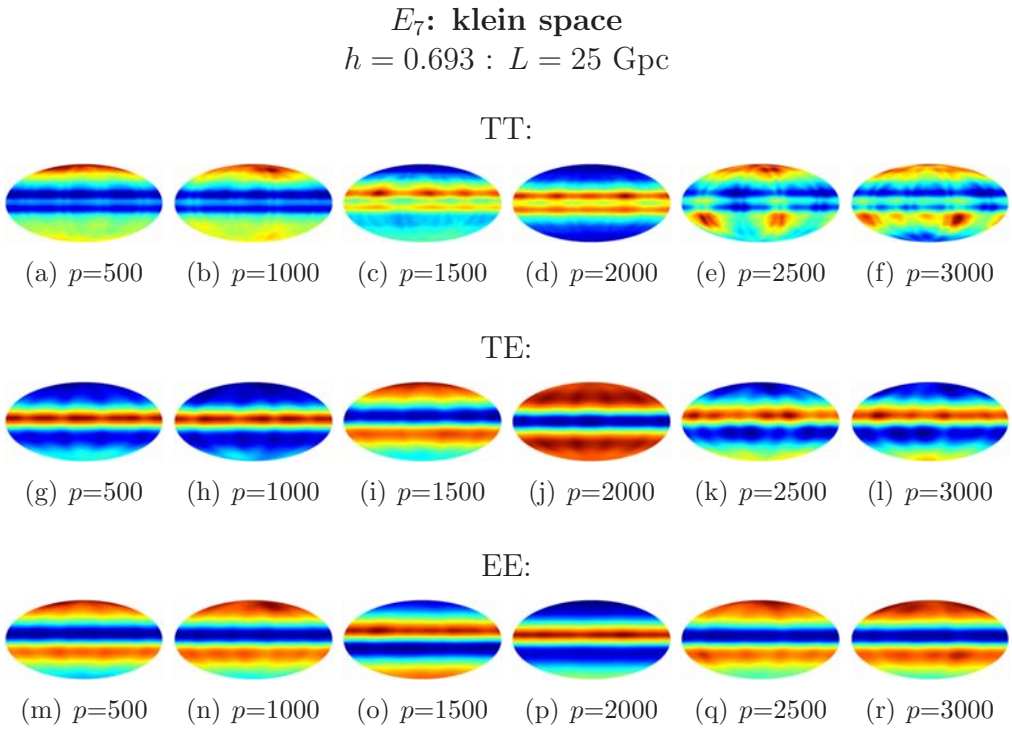


Figure 10.9: Maps of the correlation matrices for klein space, E_7, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

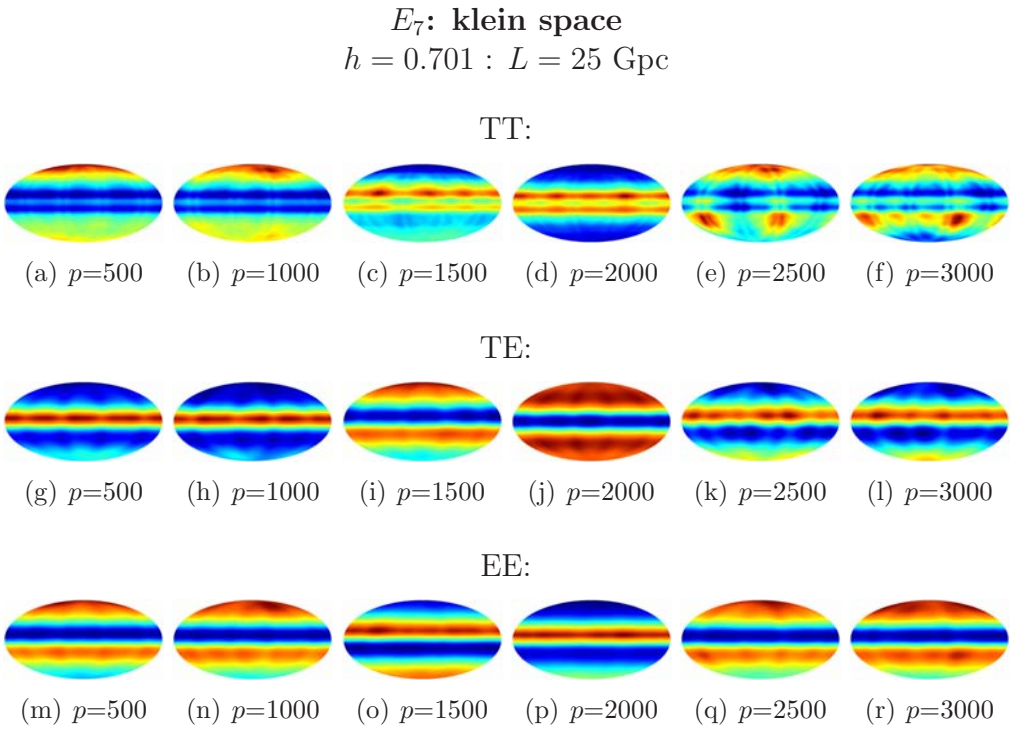


Figure 10.10: Maps of the correlation matrices for klein space, E_7, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

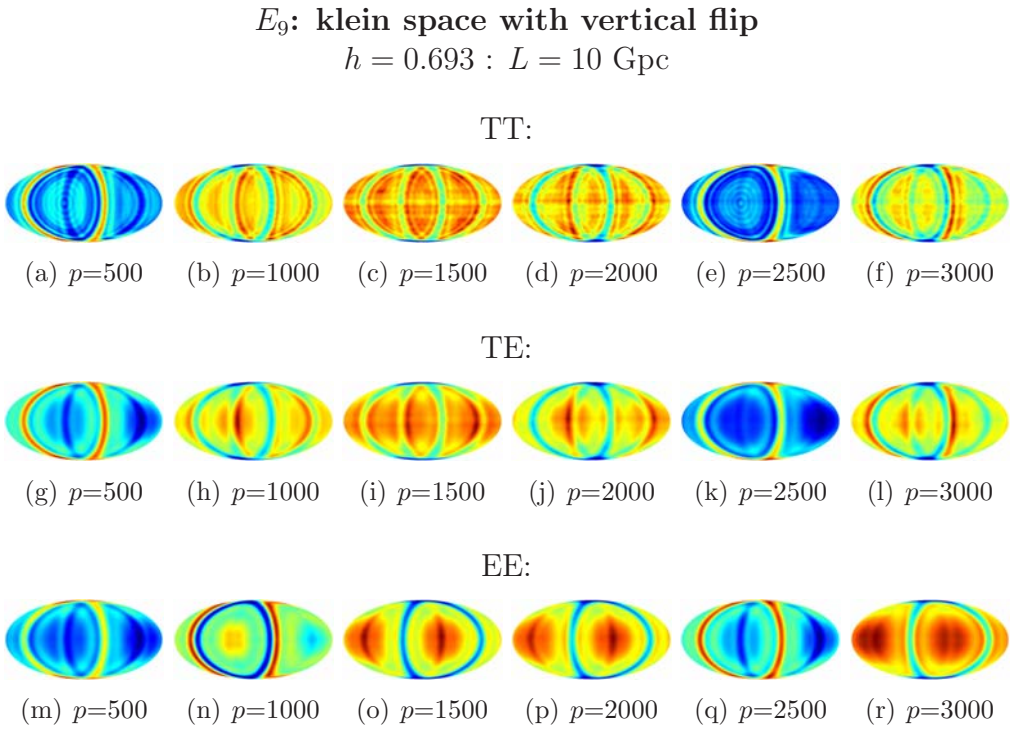


Figure 10.11: Maps of the correlation matrices for klein space with vertical flip, E_9, with $h = 0.693$ and $L = 10$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

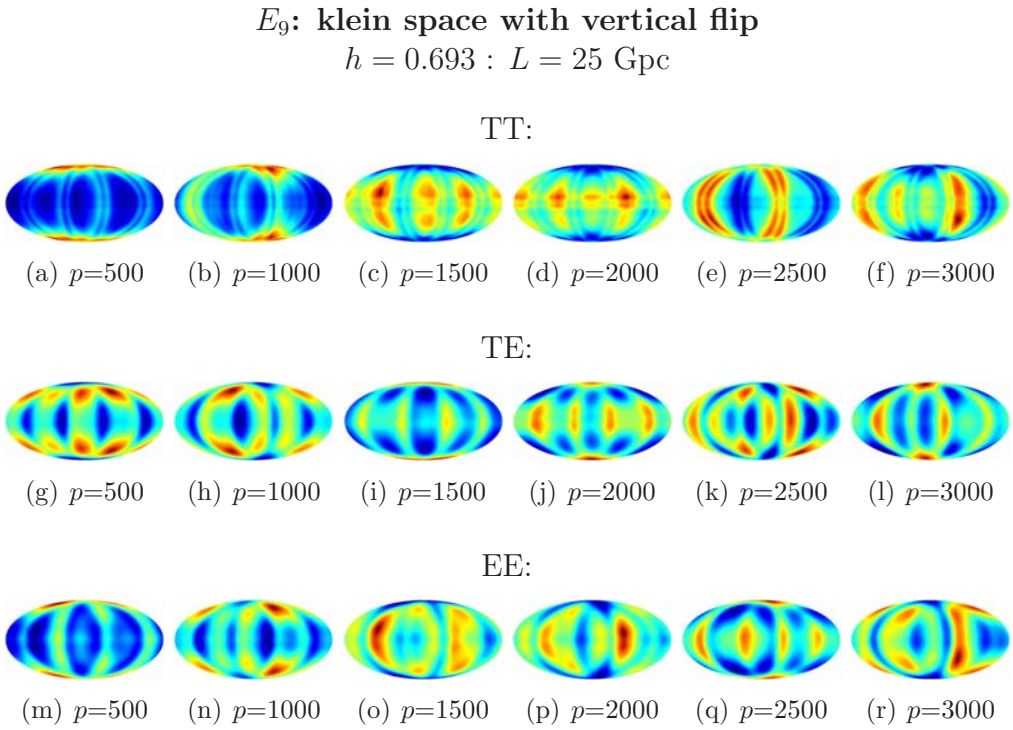


Figure 10.12: Maps of the correlation matrices for klein space with vertical flip, E_9, with $h = 0.693$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

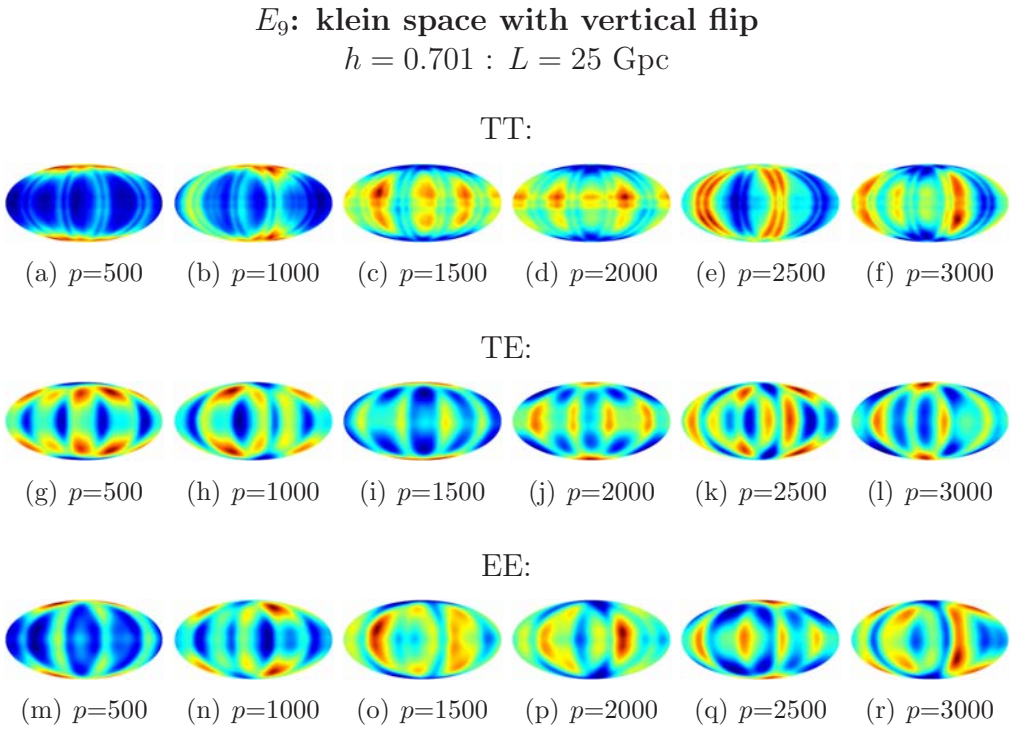


Figure 10.13: Maps of the correlation matrices for klein space with vertical flip, E_9, with $h = 0.701$ and $L = 25$ Gpc. Each map represents one row of the correlation matrix in pixel space. p is the pixel that this row corresponds to. The first row contains maps for the TT covariance matrix, the second for TE and the third for EE.

10.2 Simulated Data

This section contains simulated maps generated using the covariance matrices via a Cholesky decomposition (see Ch. 9) for $2 \geq \ell \geq 10$. No noise has been added to any of these maps. Fig. (10.14) to Fig. (10.17) each show a collection of simulations for one topology; within the figure are maps for TT, TE and EE, as well as (h,L) values of (0.693,10), (0.693,25) and (0.701,25). Each column of maps was generated using the same set of random numbers, which is clear from the similarities in the patterns. Again, there is little difference between maps which differ in h only. L, however, has a large influence over the appearance of the maps. A 45 degree alignment is apparent between the Q and U maps, which is to be expected from the definitions of Stokes' parameters Q and U.

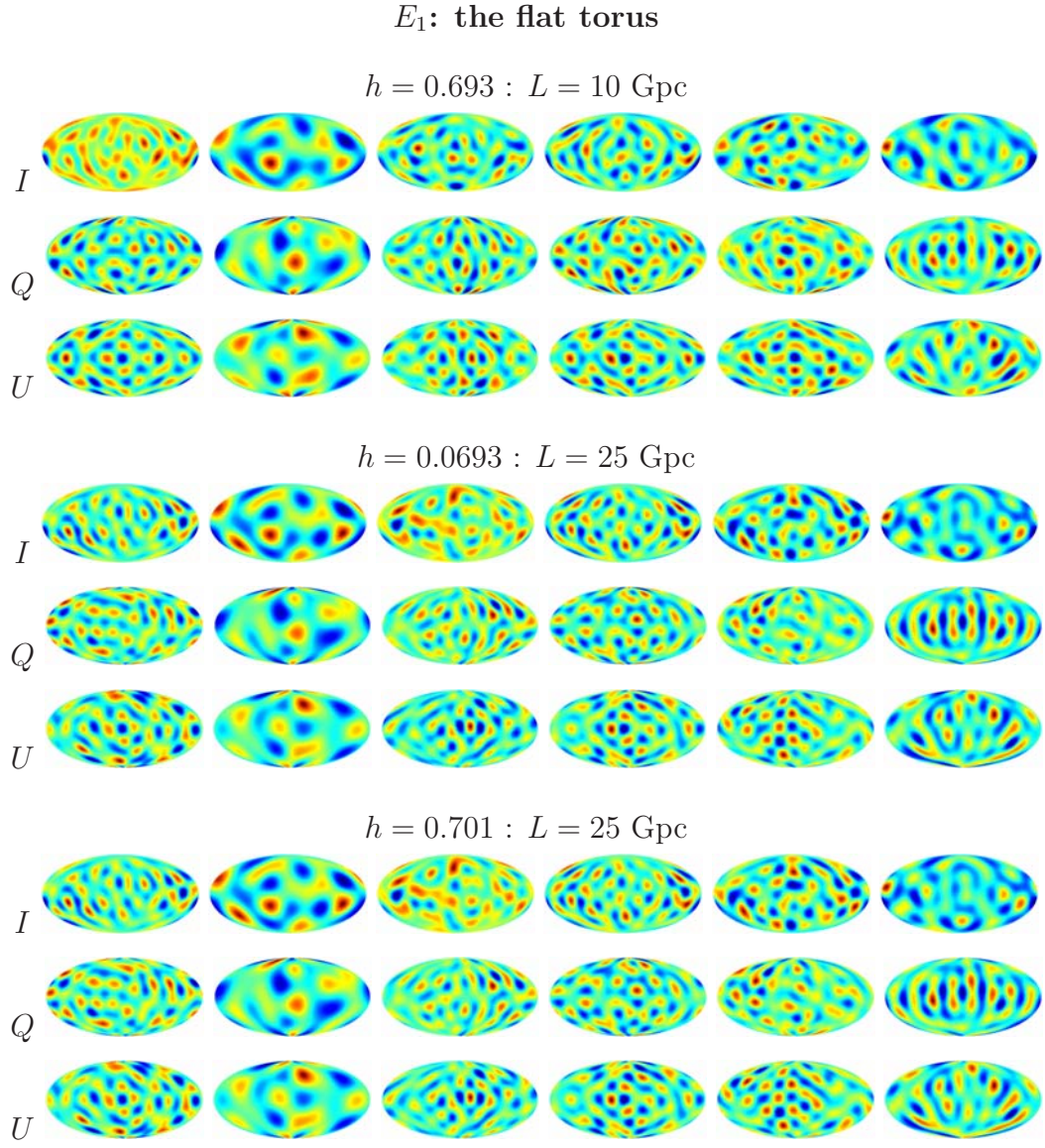


Figure 10.14: Simulated maps for the flat torus, E_1, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters.

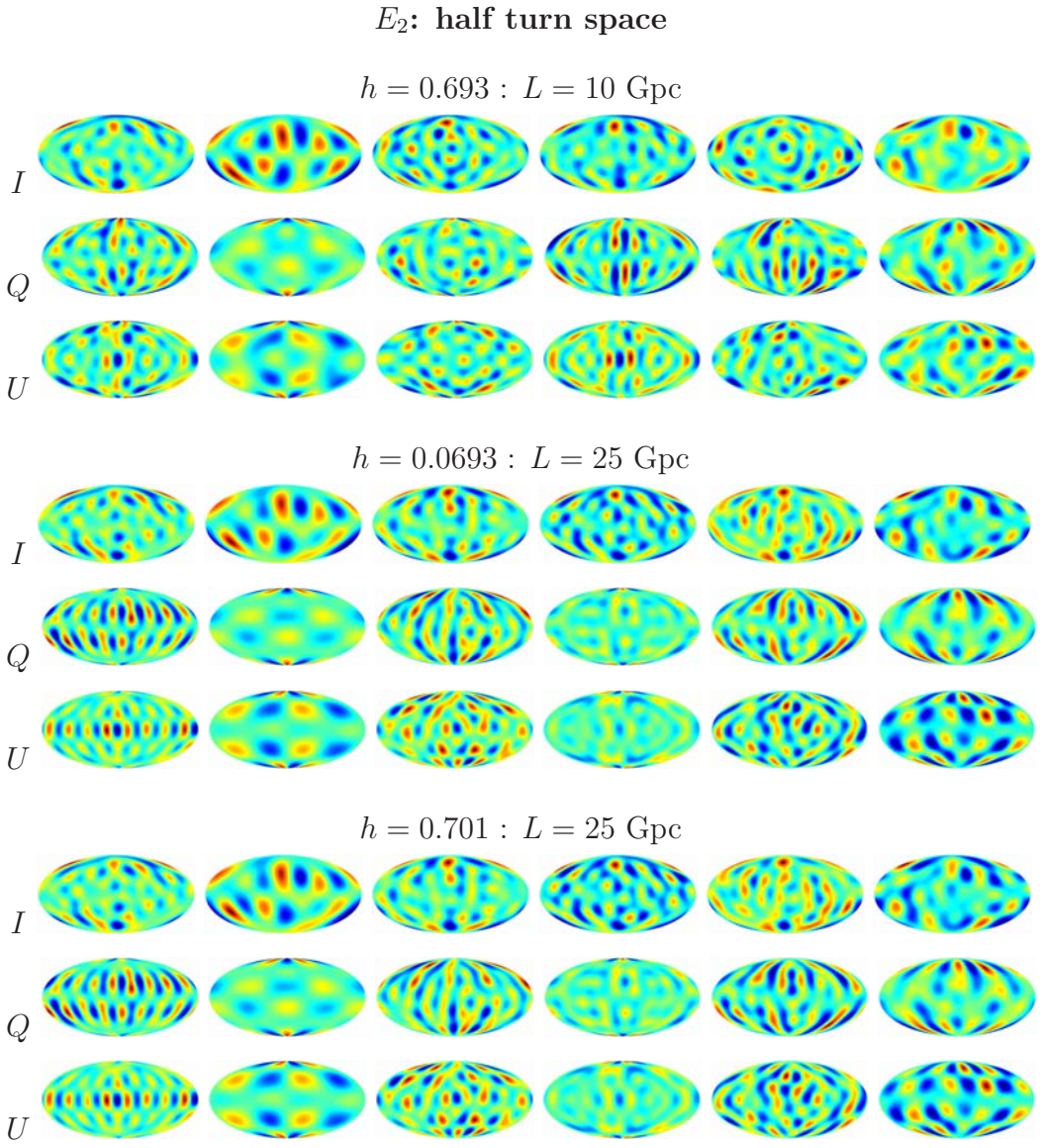


Figure 10.15: Simulated maps for half turn space, E_2, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters.

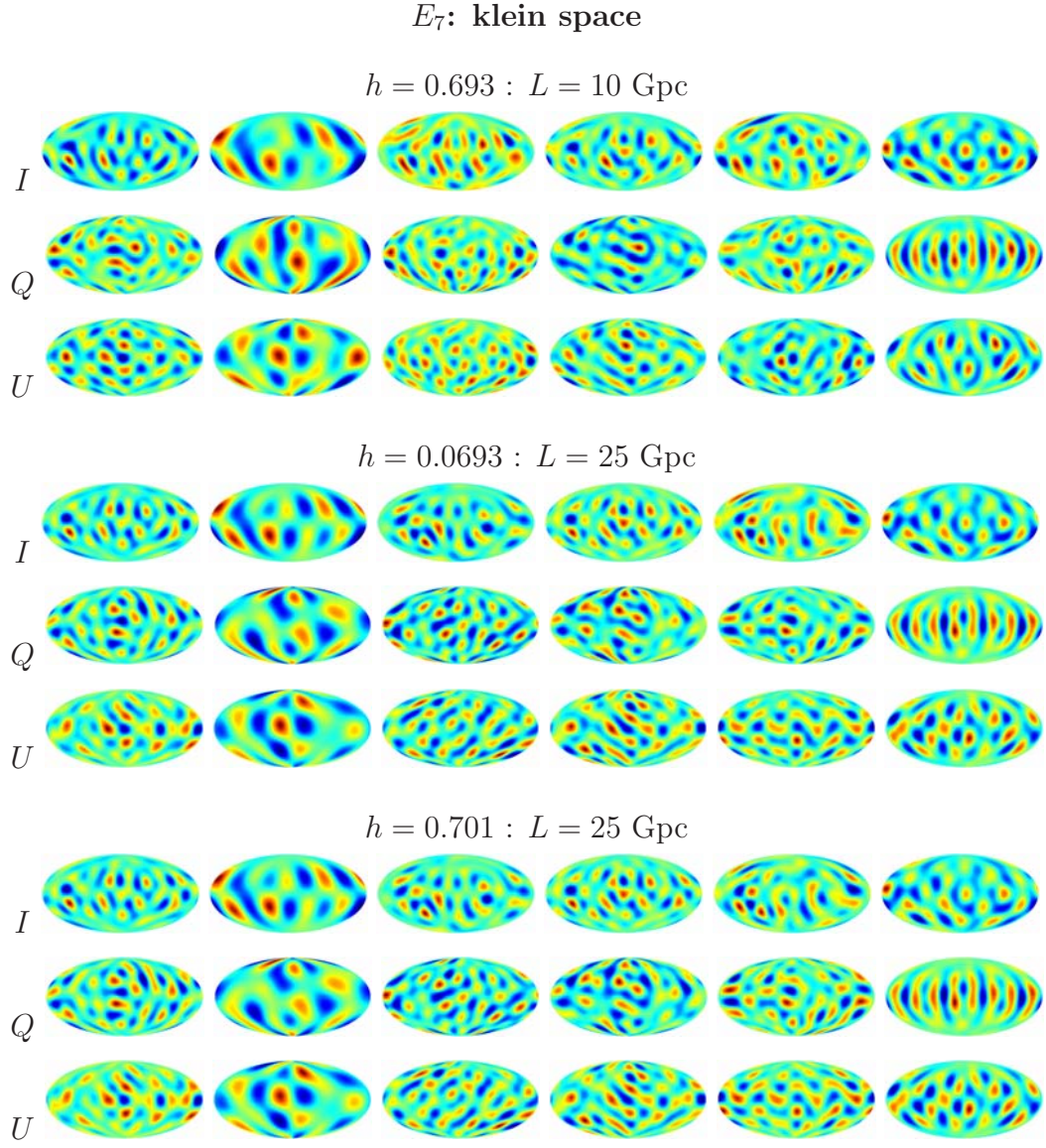


Figure 10.16: Simulated maps for klein space, E_7, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters.

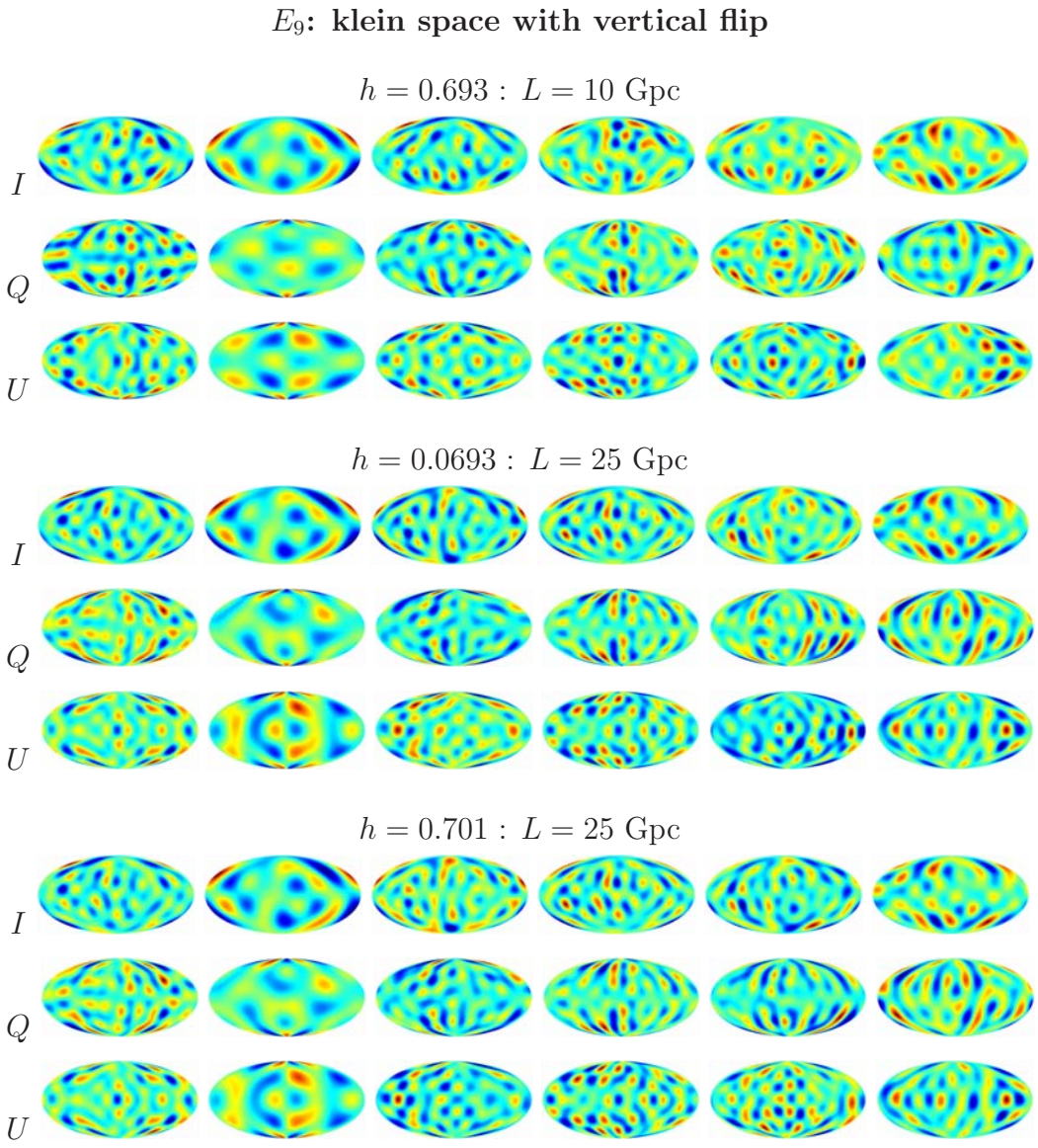


Figure 10.17: Simulated maps for klein space with vertical flip, E_9, for $2 \leq \ell \leq 10$. Six different simulations are shown for each combination of parameters.

10.2.1 Temperature Maps

This section breaks some of the simulated temperature maps from the previous section into individual multipole maps (left columns) and sequetially recombines them (right columns). This is in order to aid understanding of the contributions of individual mulipoles to the composite maps.

Similar maps can be found in Section 10.3 for the nine-year WMAP data. It is interesting to compare the simulated maps with these real data. By eye, it's hard to say which simulated maps most resemble the real data. However, this is not a sensible exercise as we only have one realisation for each topology, and other realisations could look rather different.

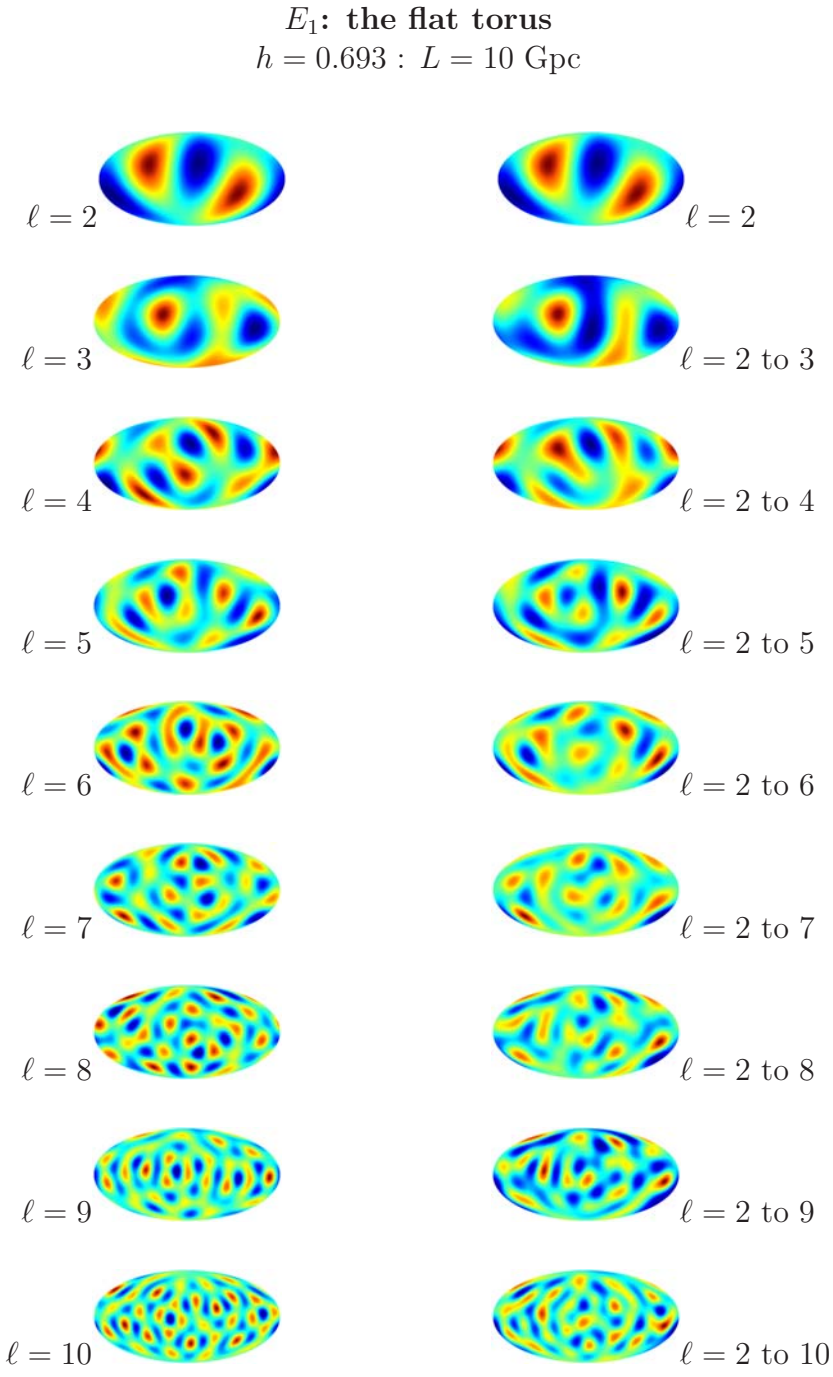


Figure 10.18: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.2). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

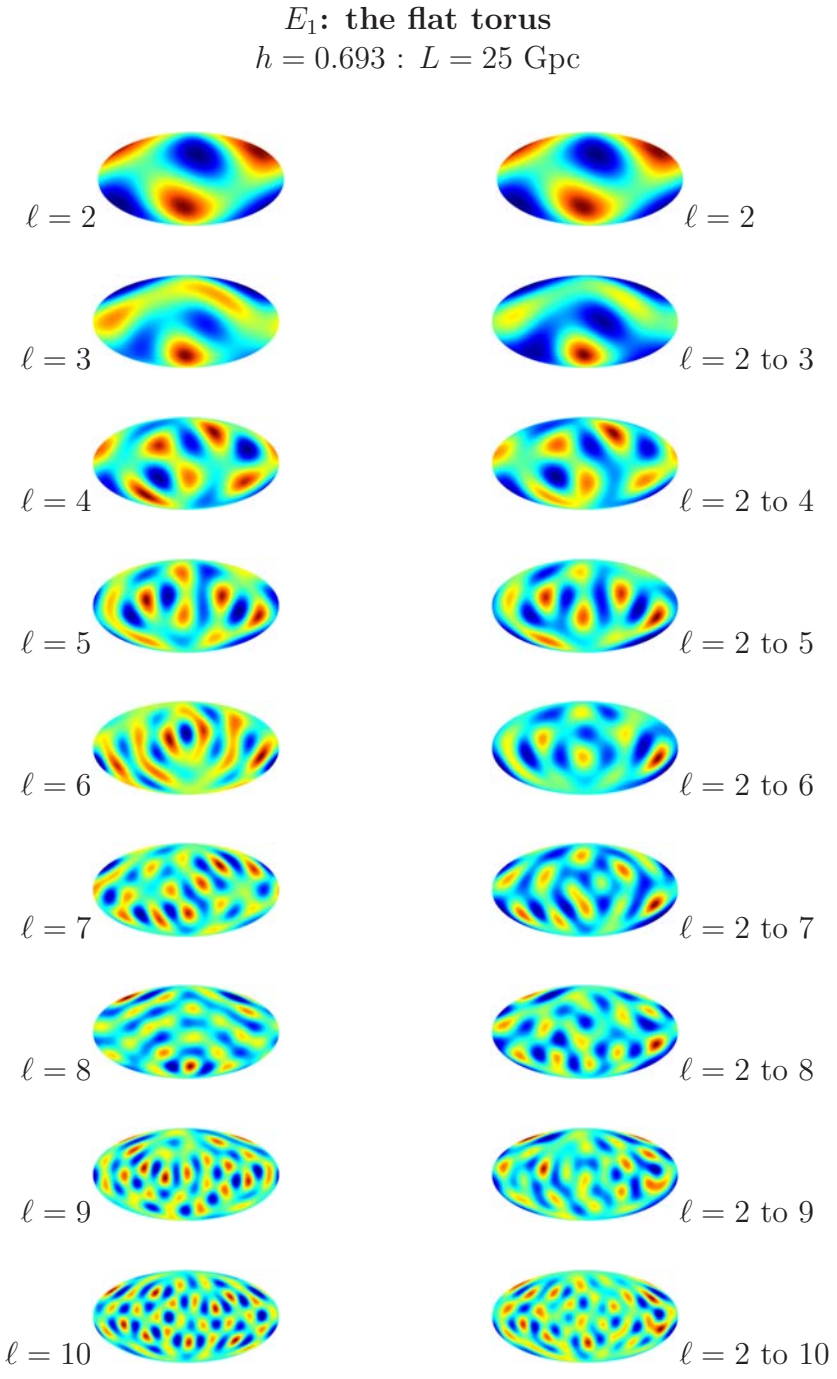


Figure 10.19: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.3). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

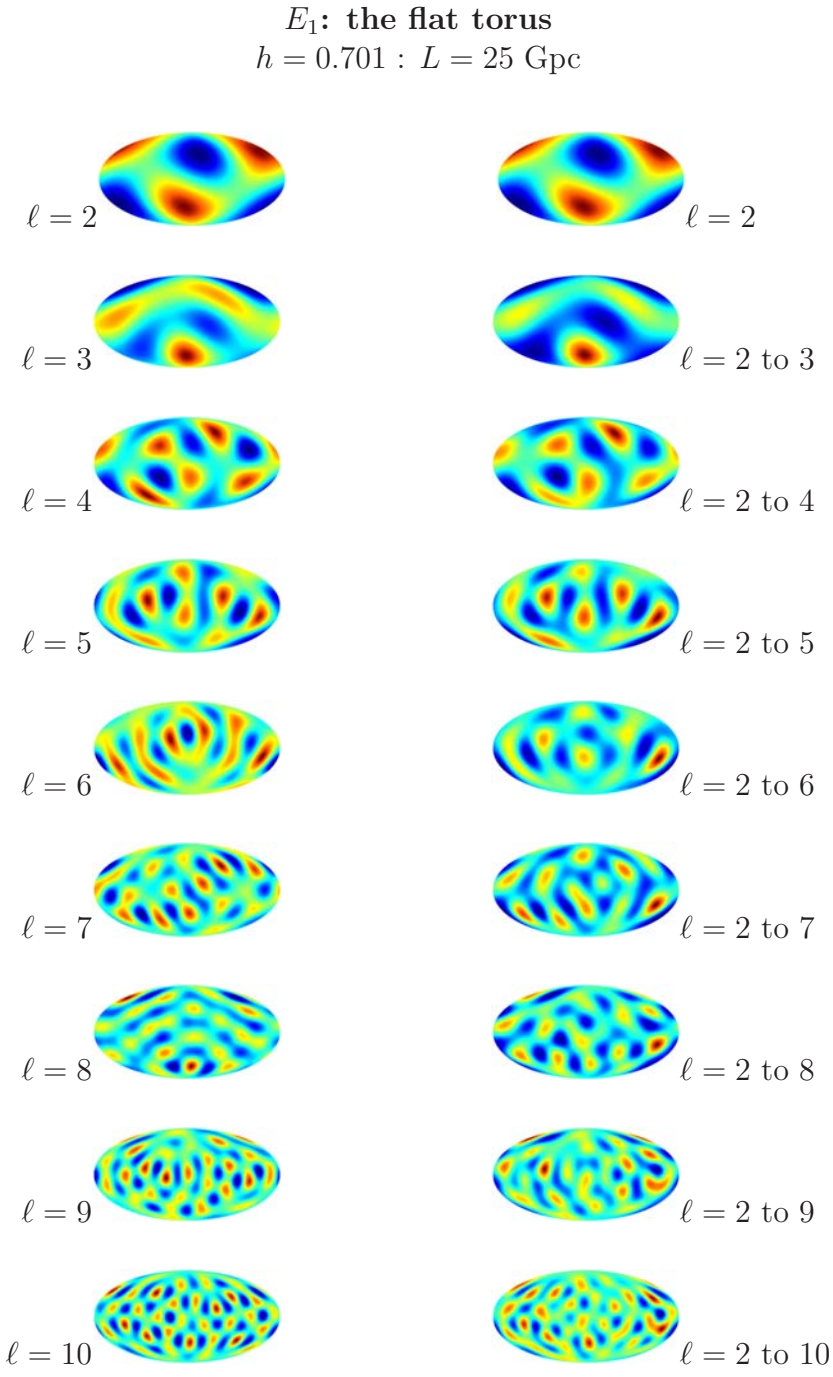


Figure 10.20: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.4). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

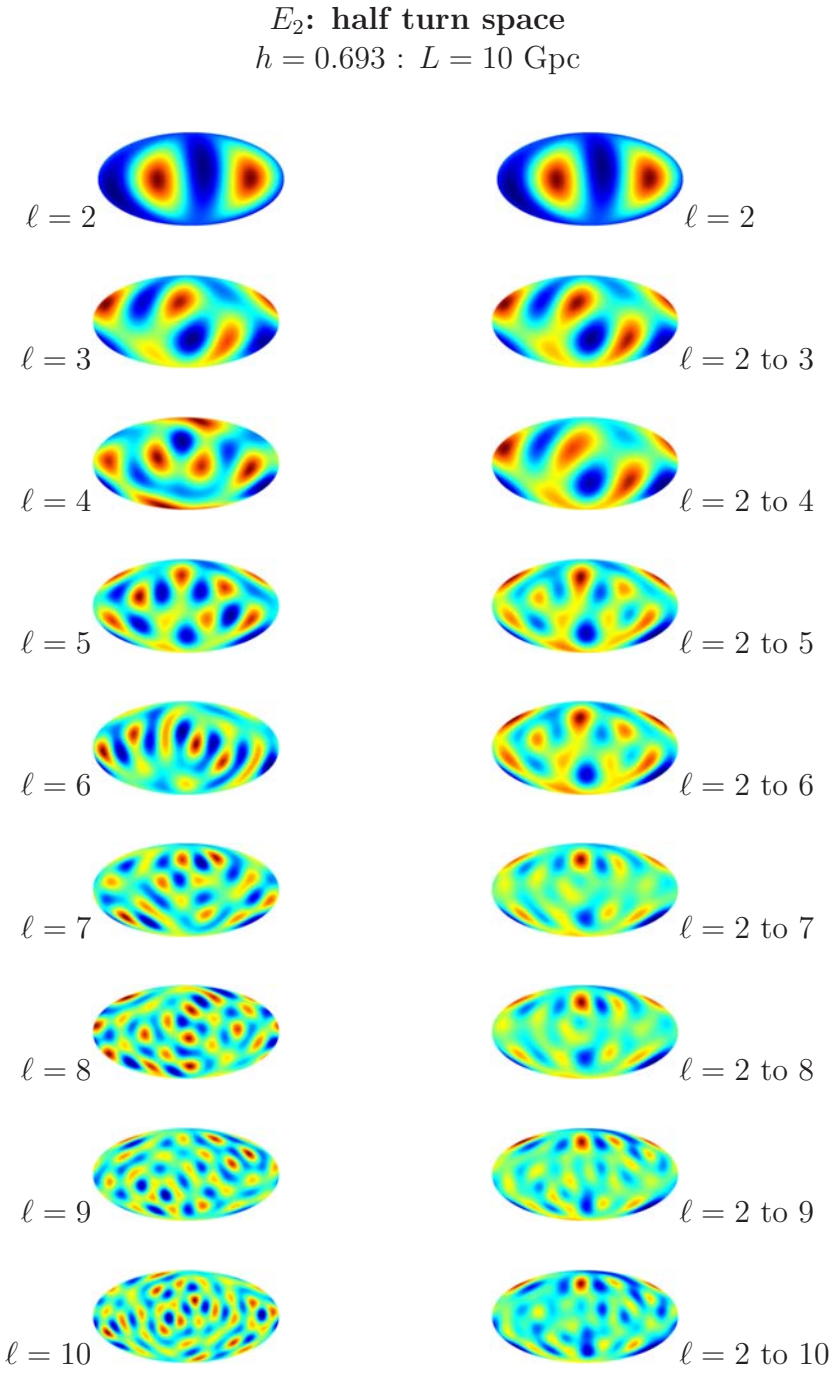


Figure 10.21: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.5). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

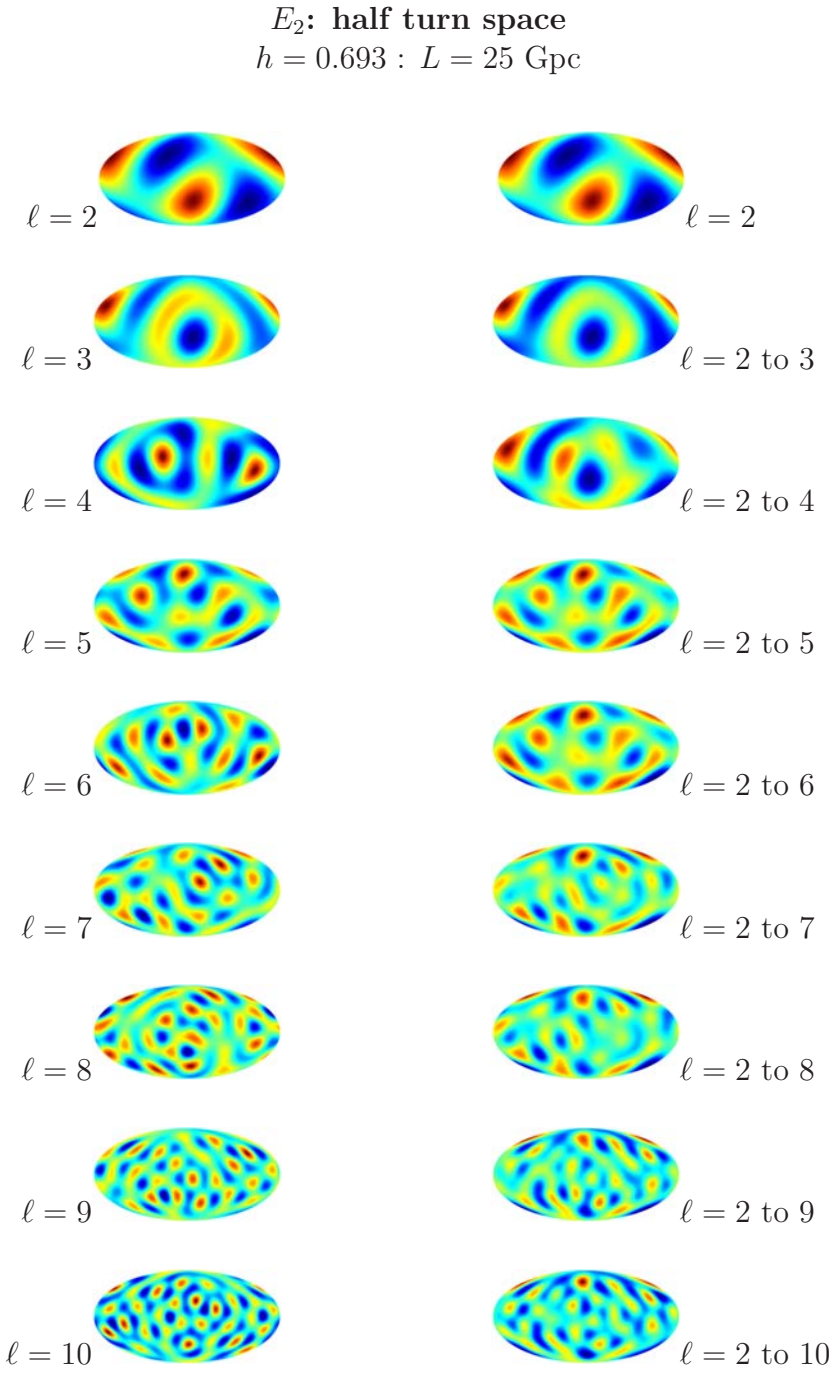


Figure 10.22: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.6). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

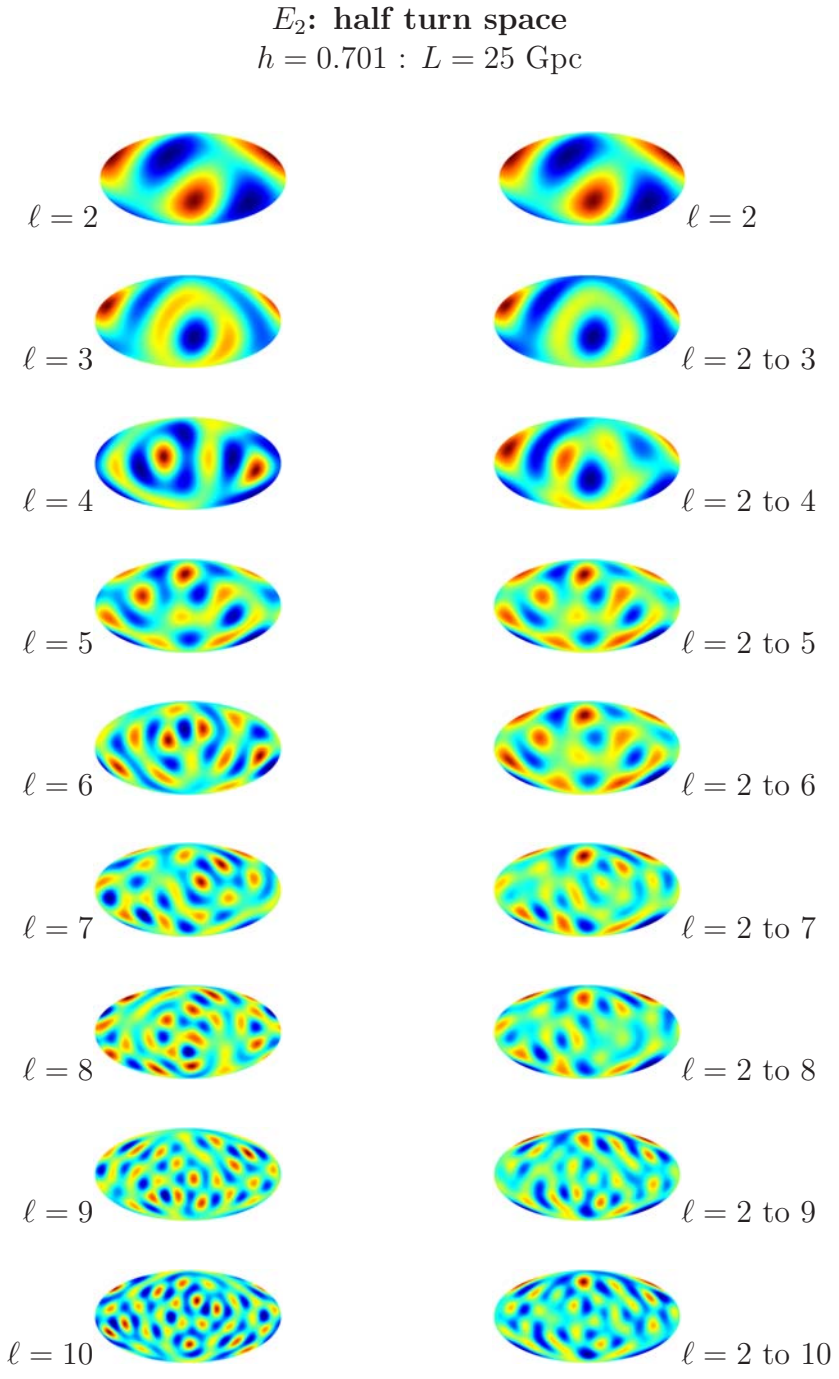


Figure 10.23: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.7). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

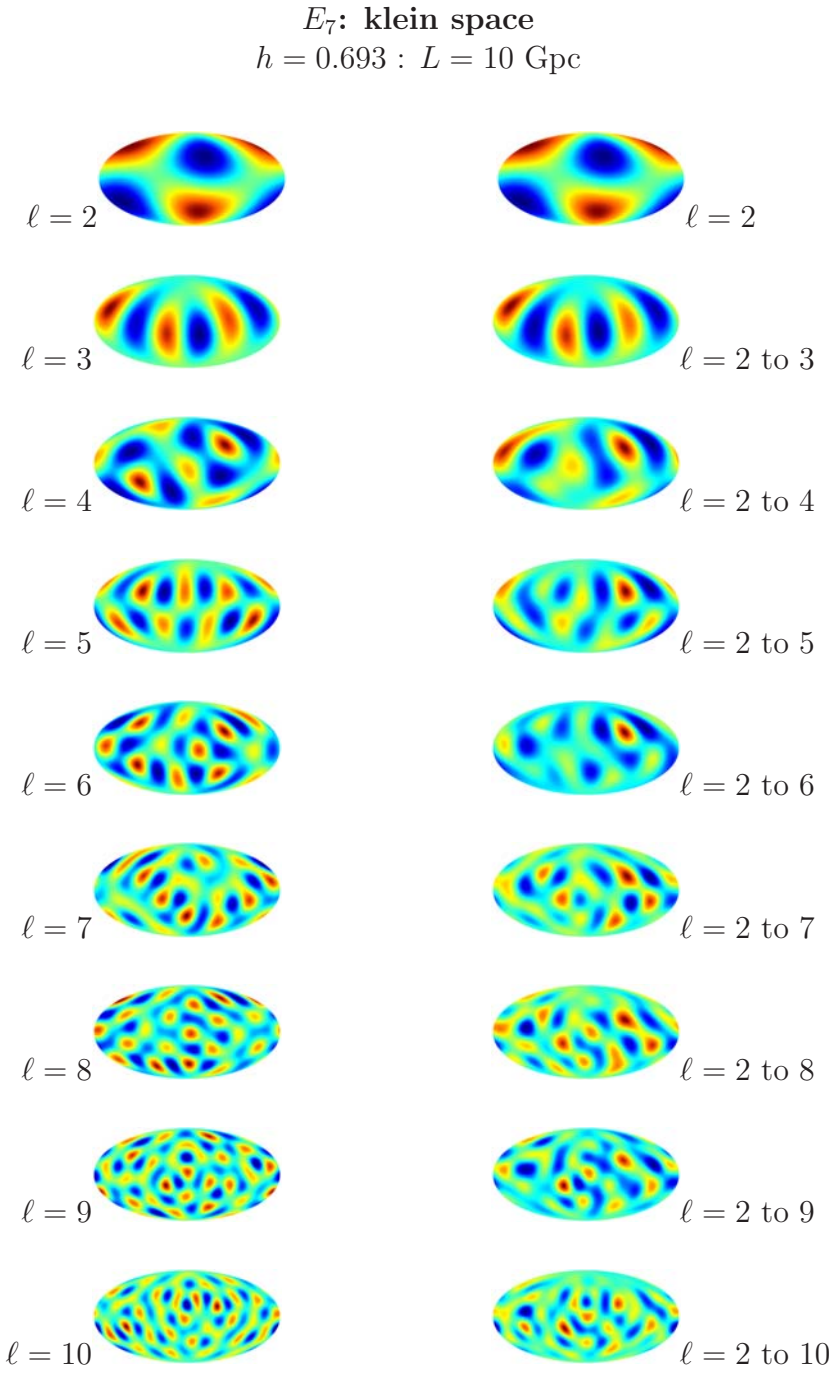


Figure 10.24: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.8). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

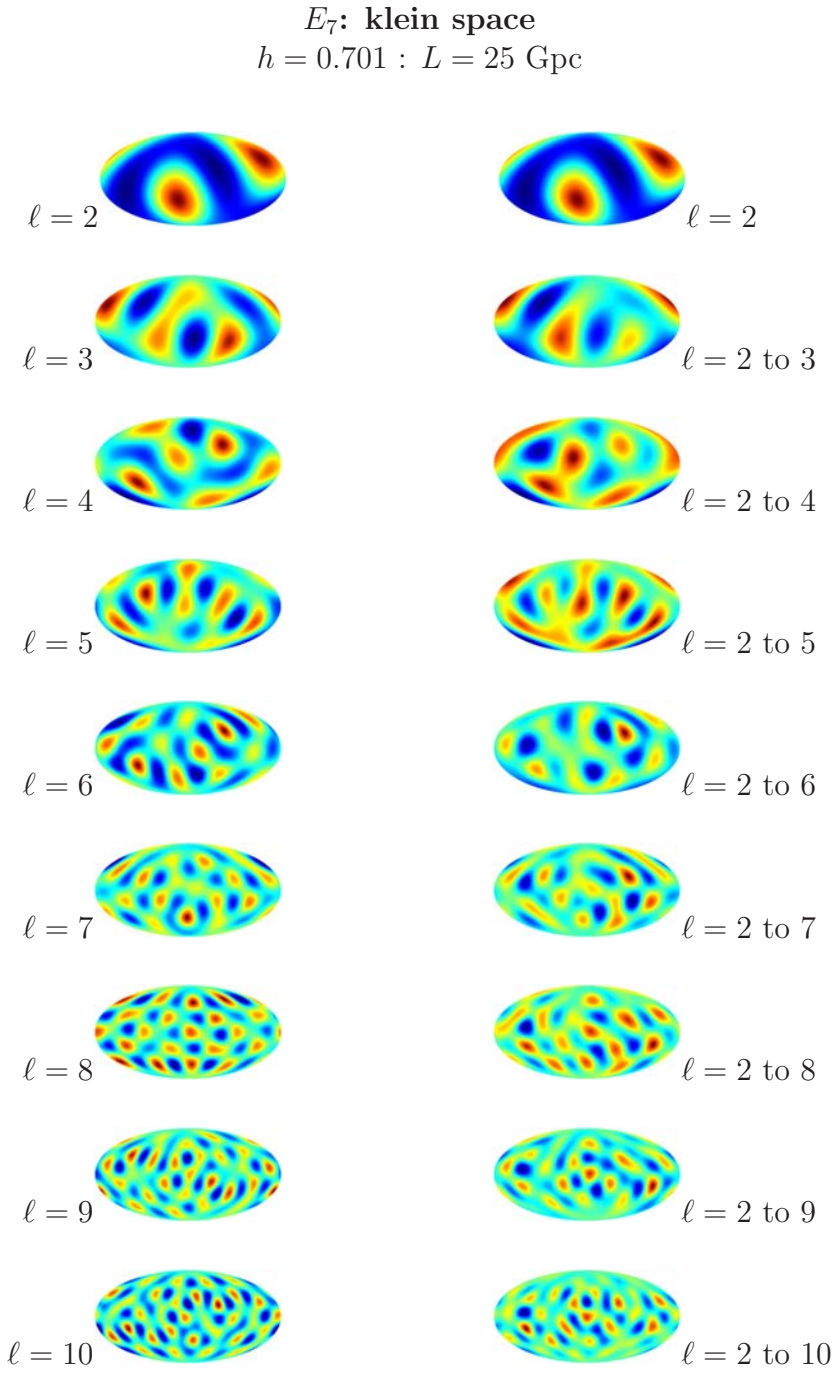


Figure 10.25: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.9). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

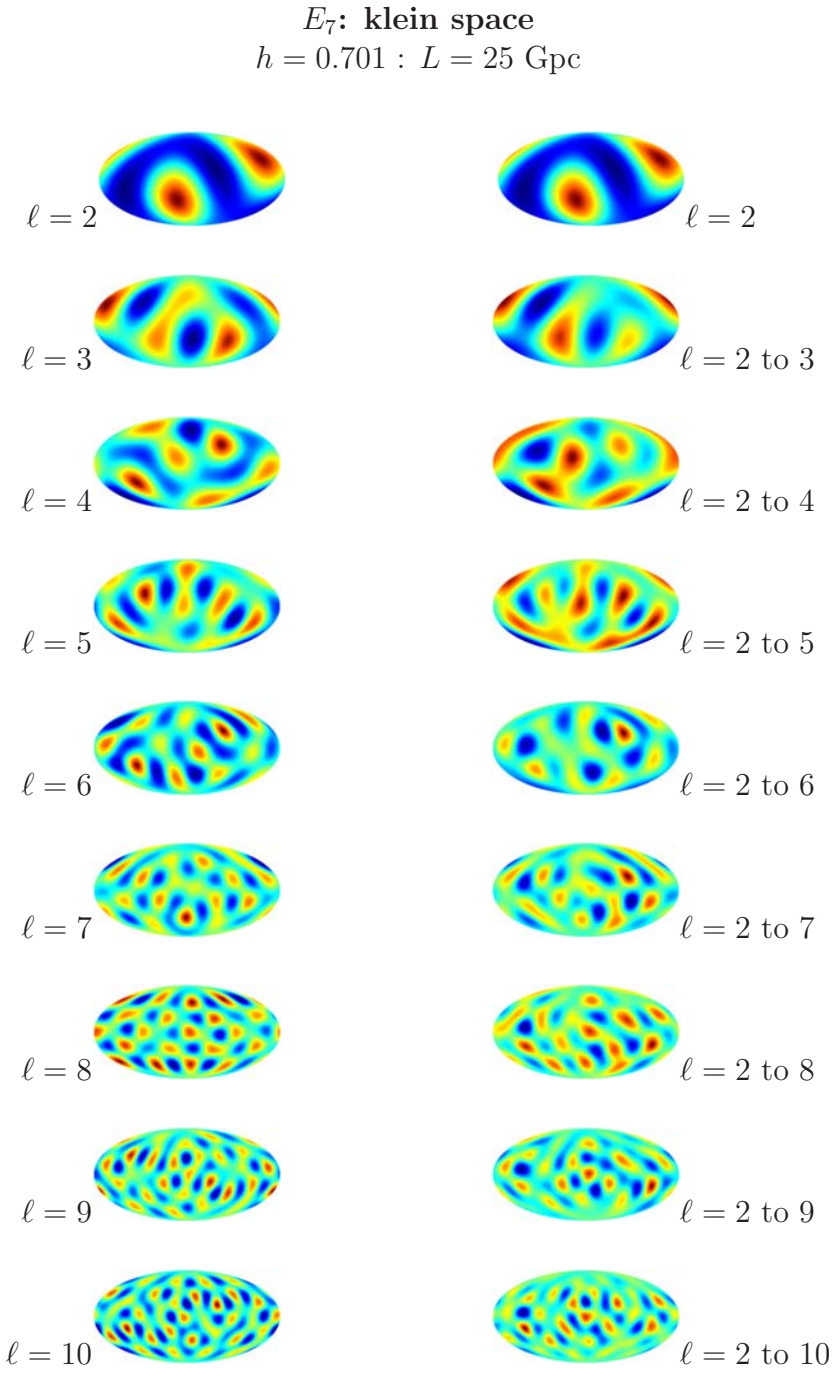


Figure 10.26: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.10). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

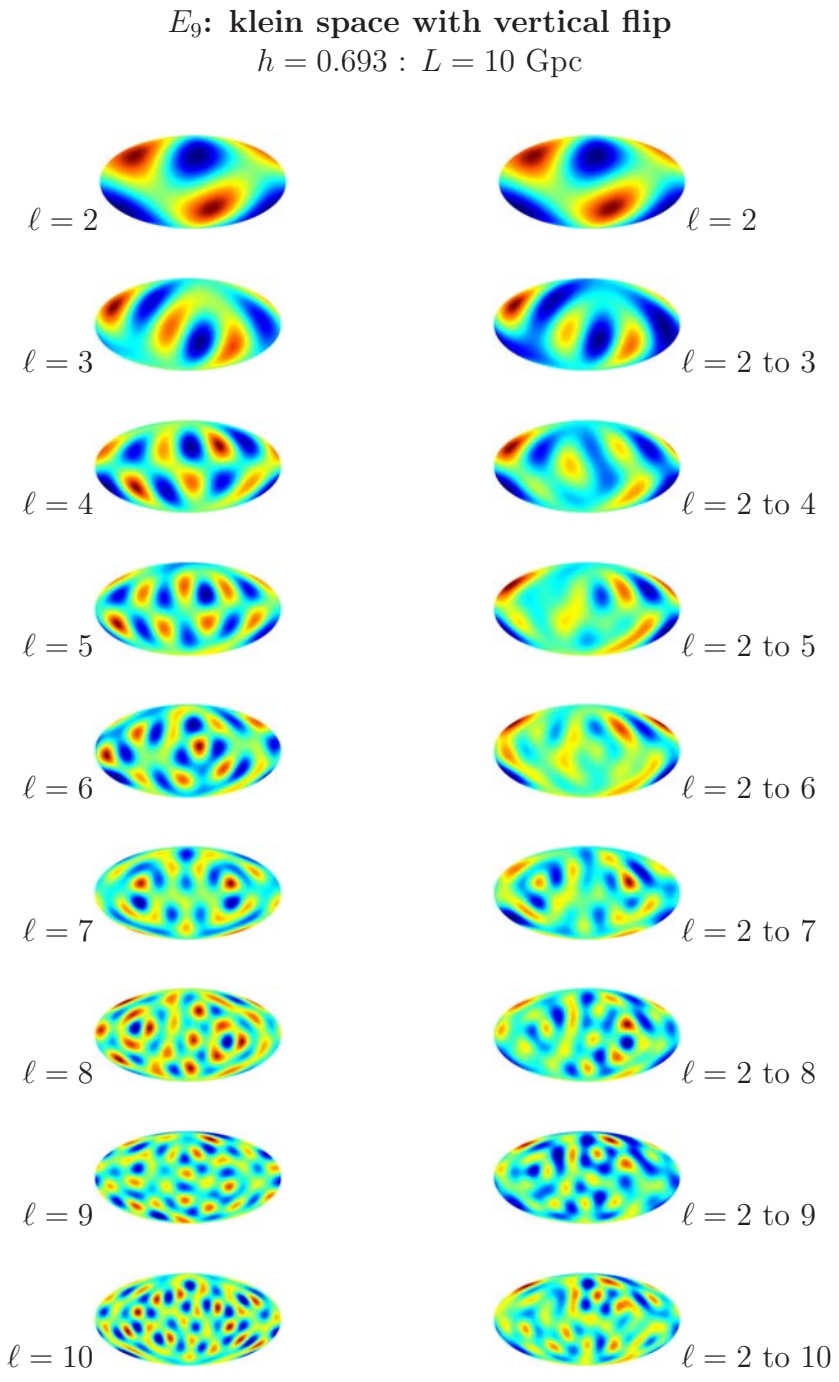


Figure 10.27: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.11). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

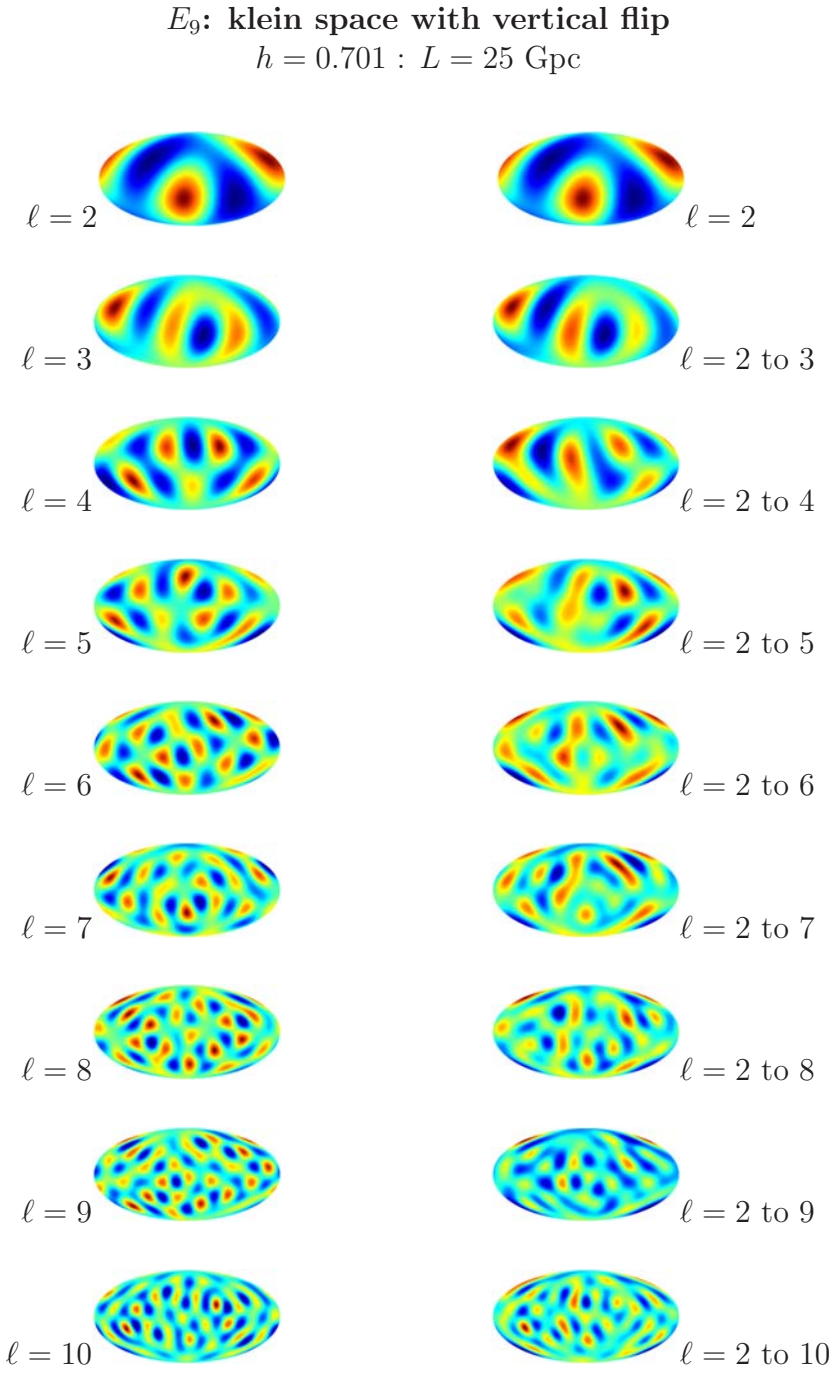


Figure 10.28: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.12). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

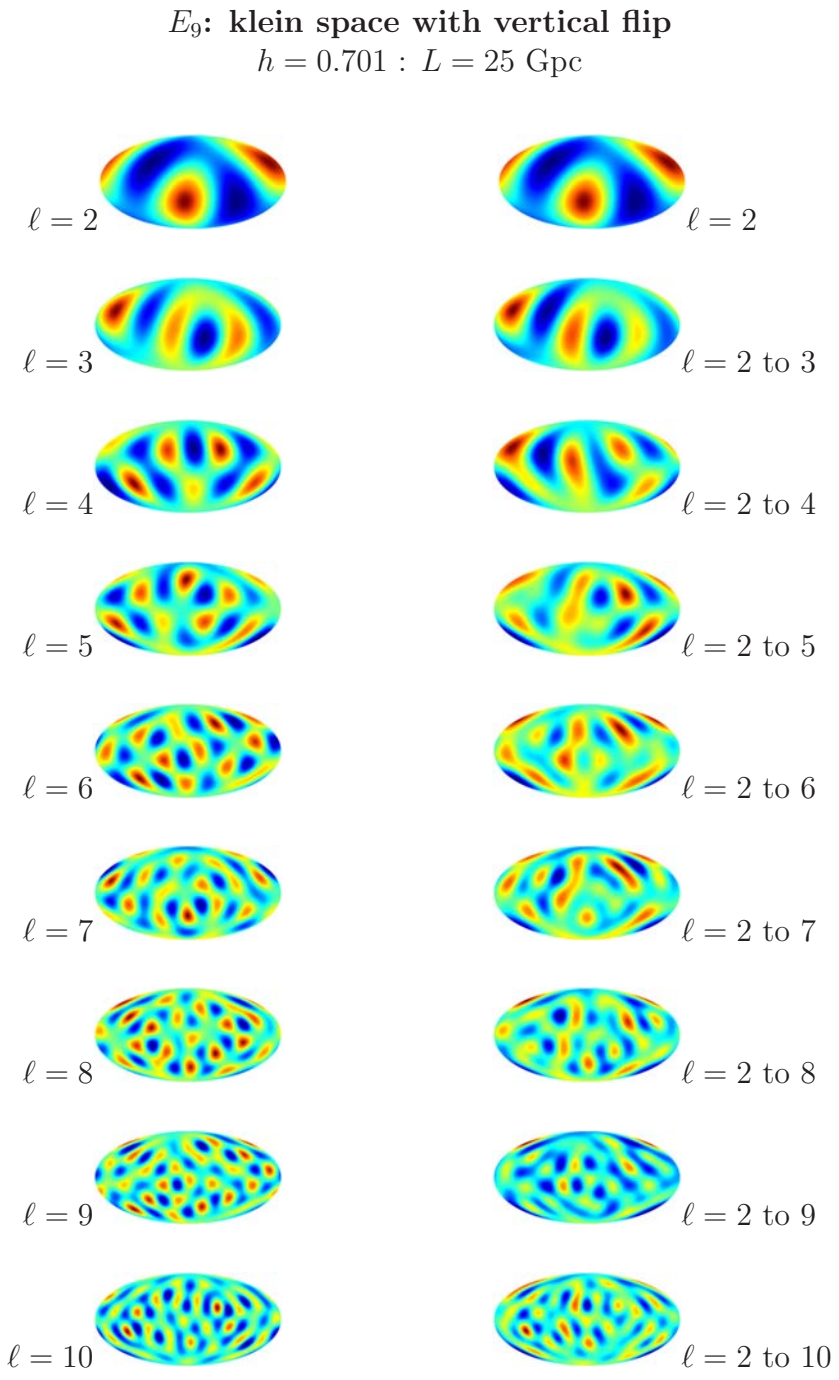


Figure 10.29: Simulated CMB maps generated using the TT correlation matrices represented in Fig. (10.13). The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

10.2.2 Polarisation Maps

This section breaks some of the simulated polarisation maps from into individual multipole maps (left columns) and sequentially recombines them (right columns). Again, this is in order to aid understanding of the contributions of individual mulipoles to the composite maps.

Similar maps can be found in Section 10.3 for the nine-year WMAP data for comparison.

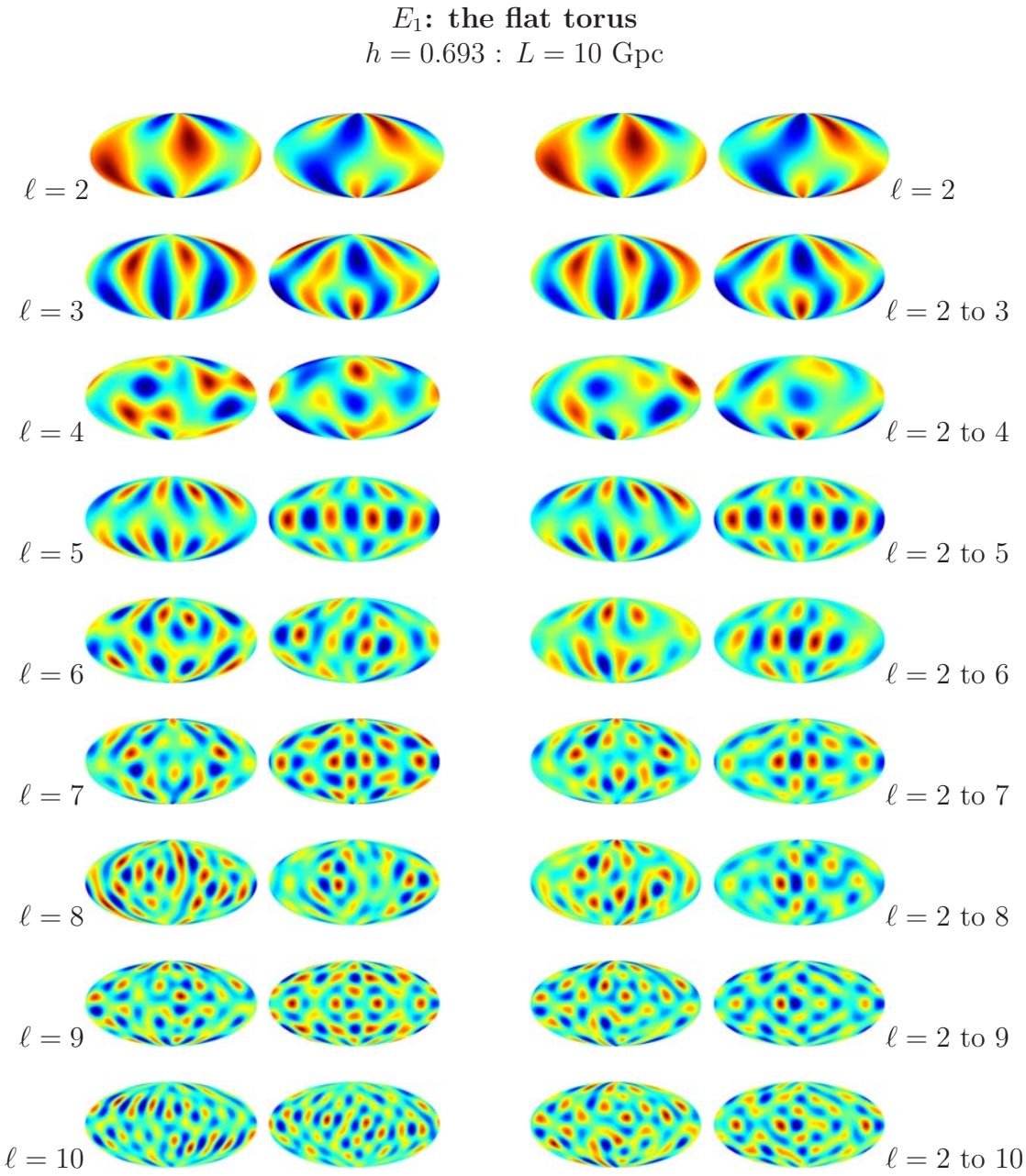


Figure 10.30: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.2)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

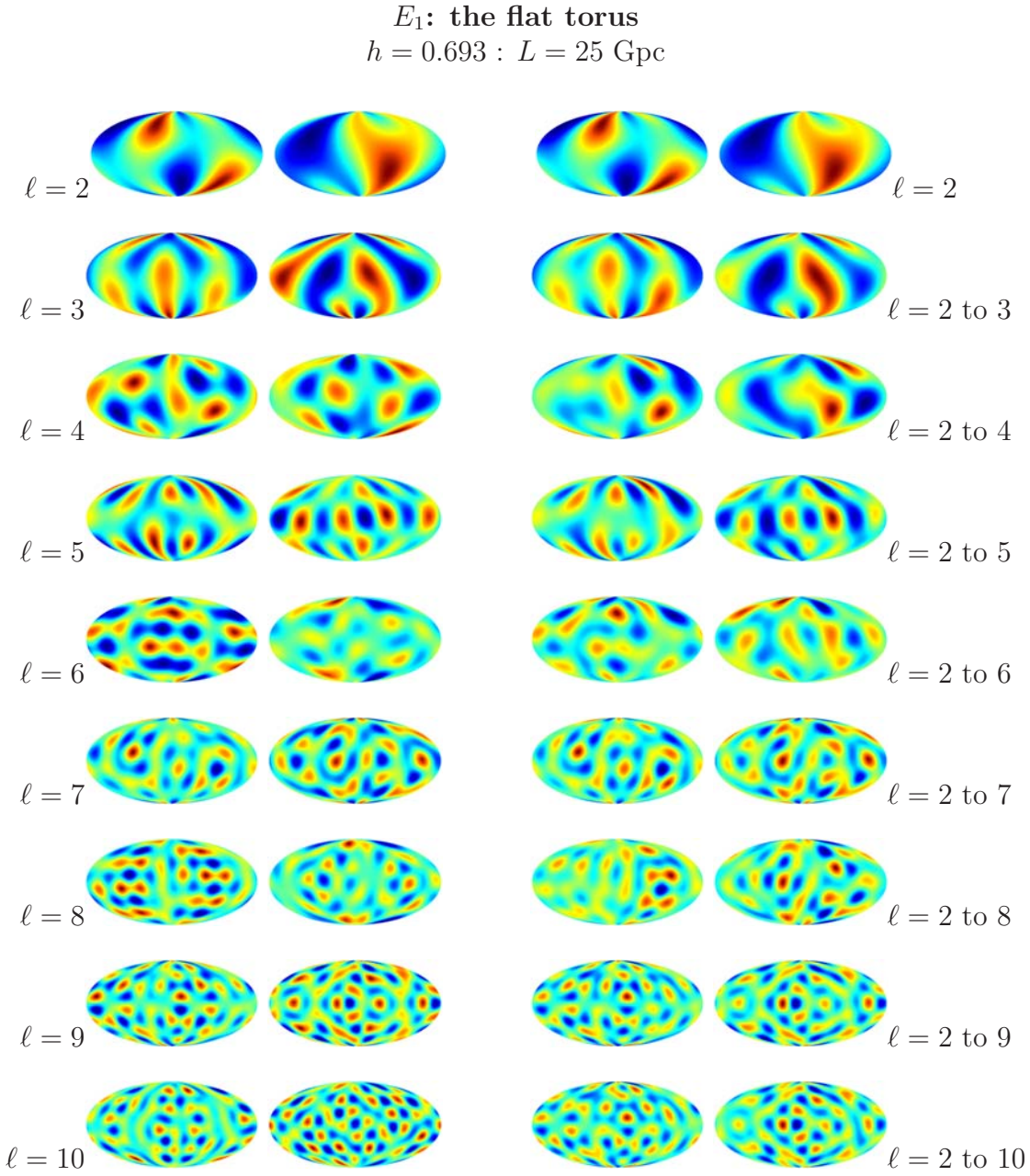


Figure 10.31: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.3)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

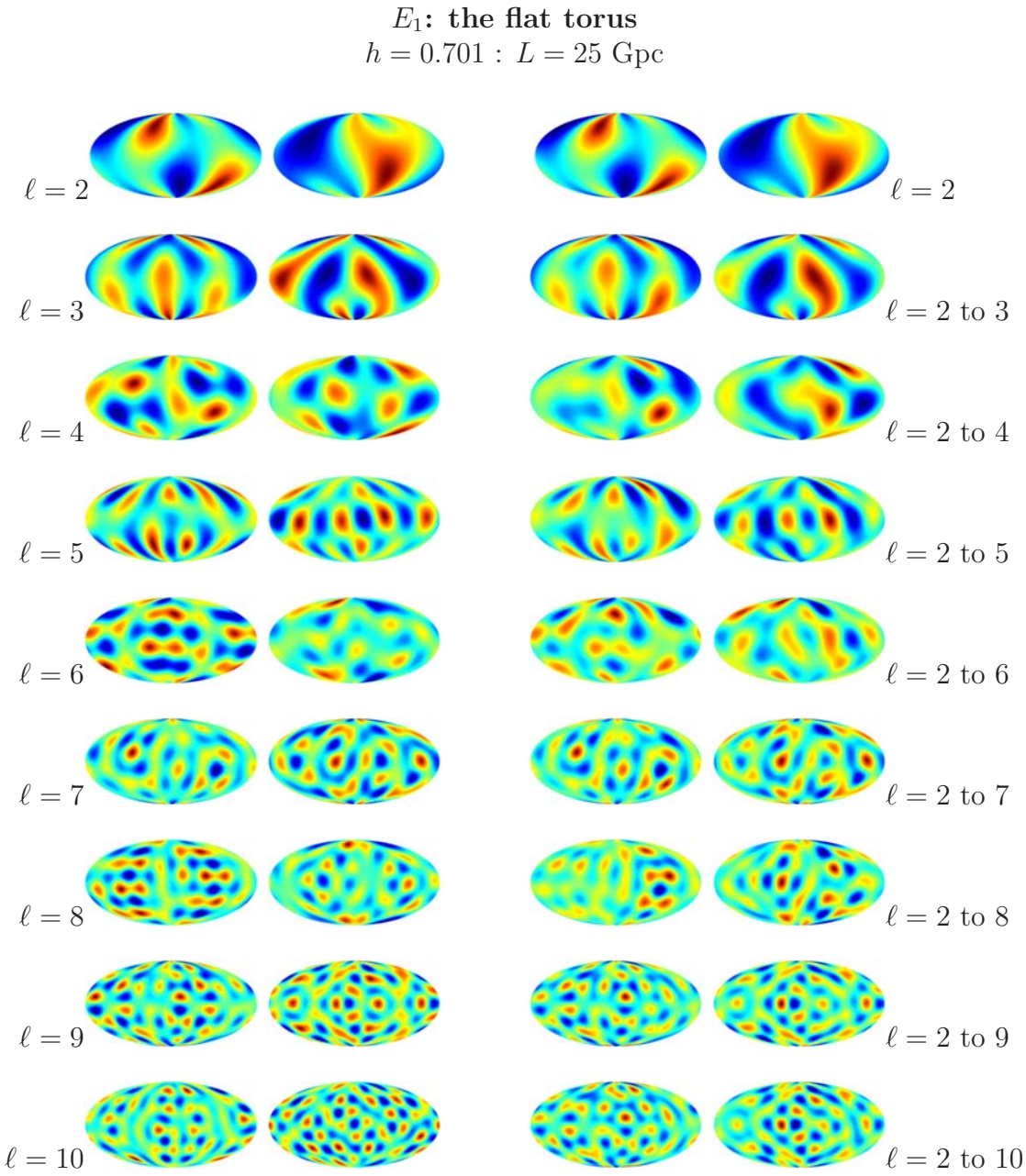


Figure 10.32: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.4)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

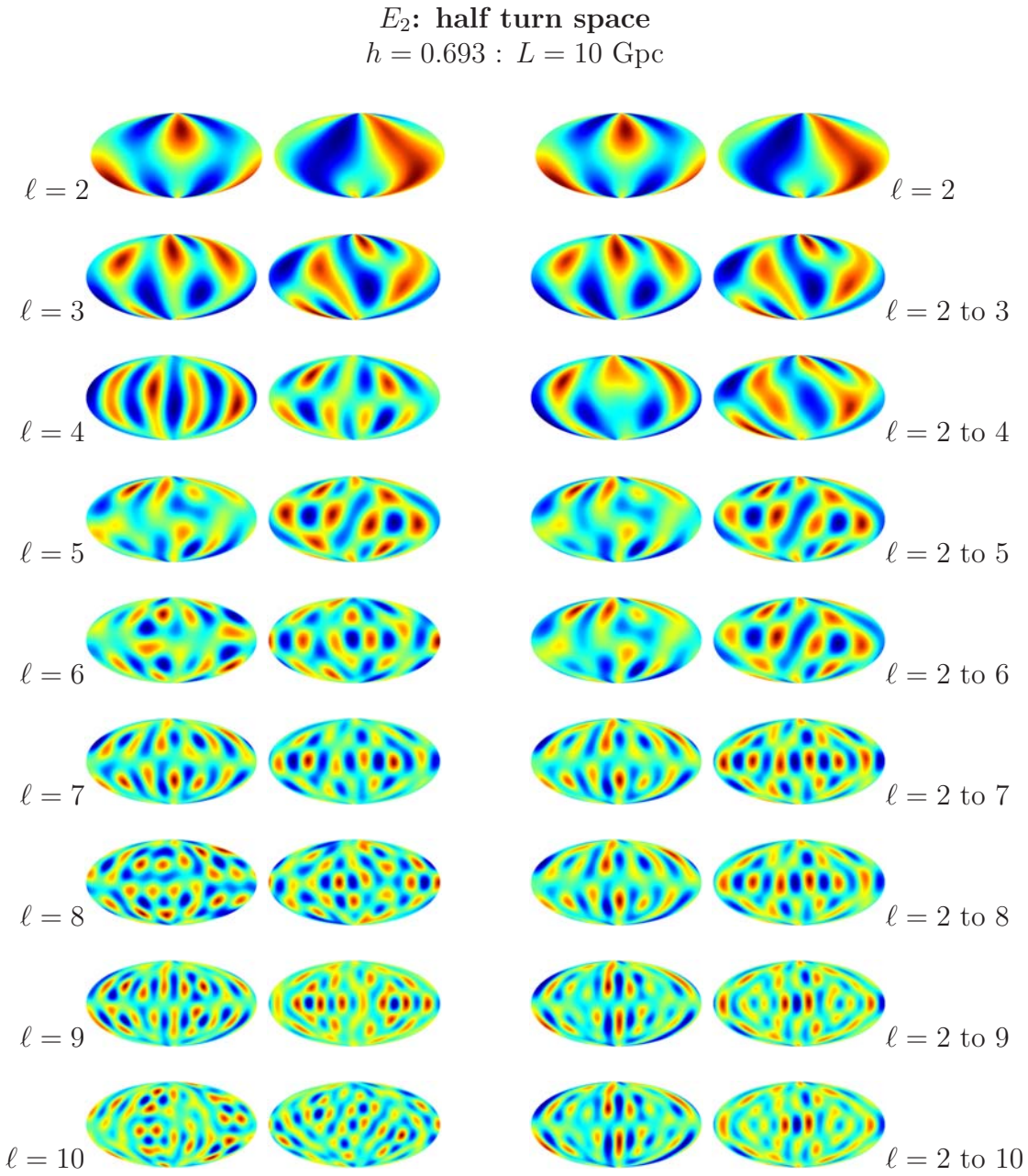


Figure 10.33: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.5)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

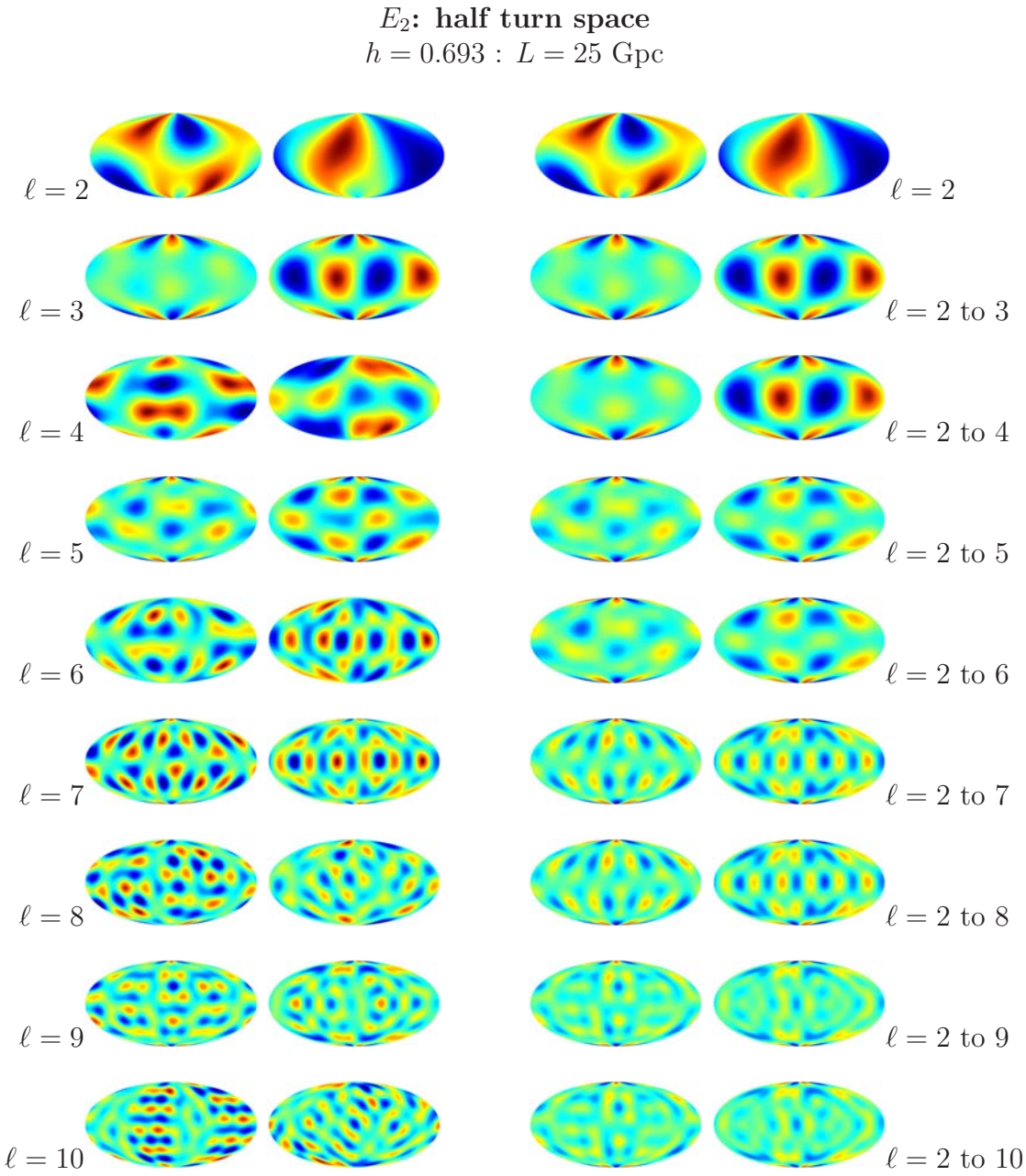


Figure 10.34: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.6)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

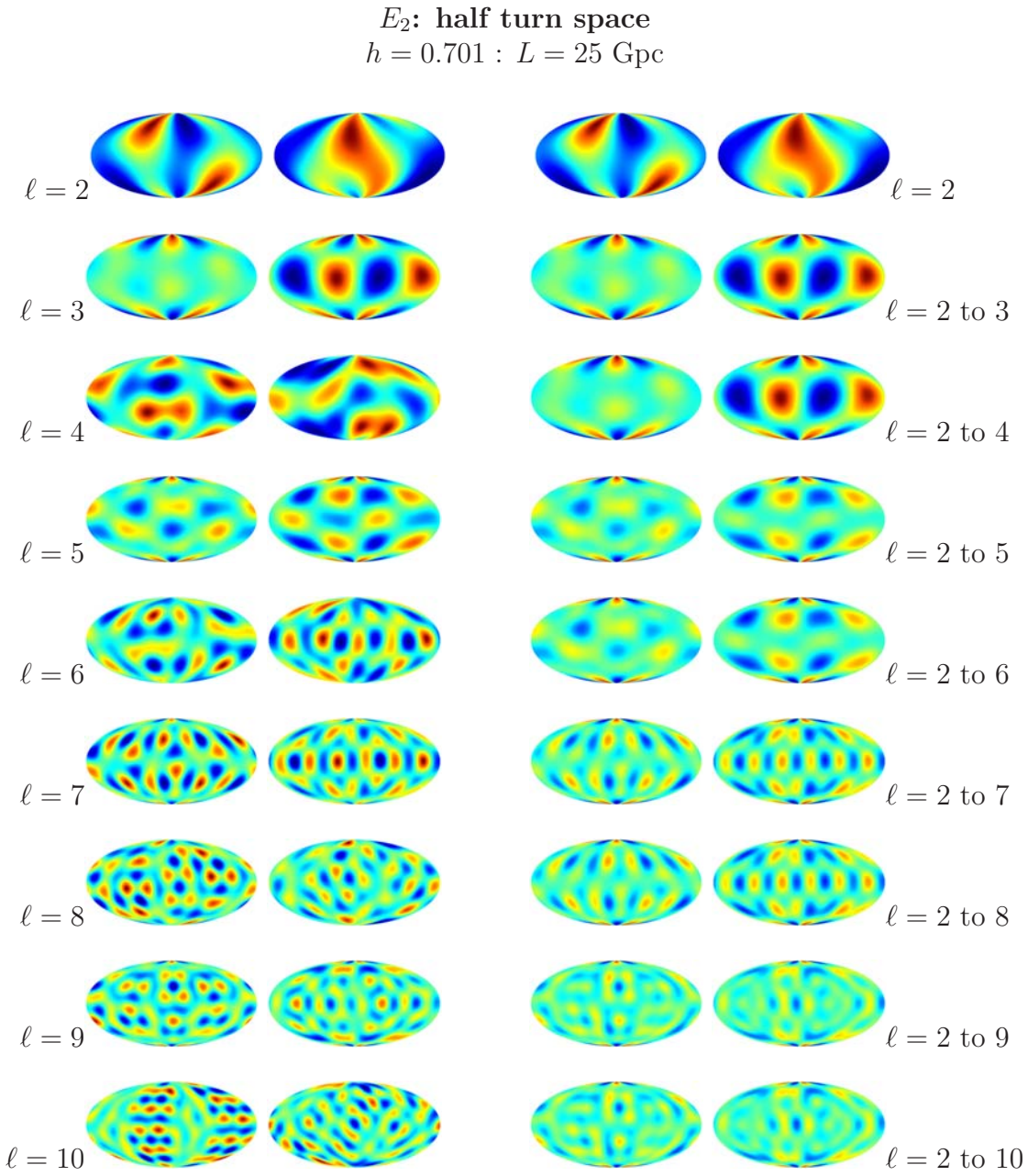


Figure 10.35: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.7)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

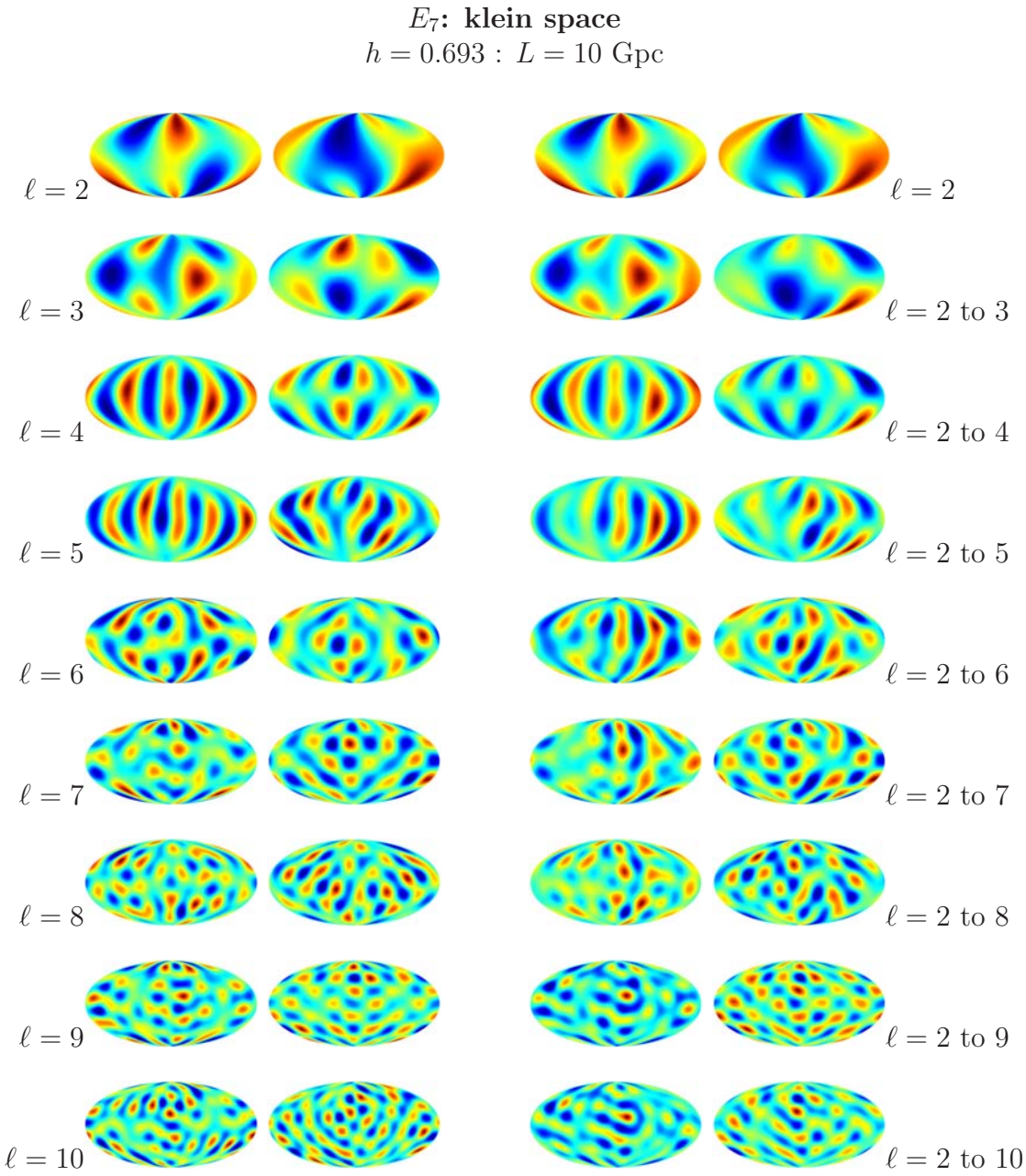


Figure 10.36: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.8)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

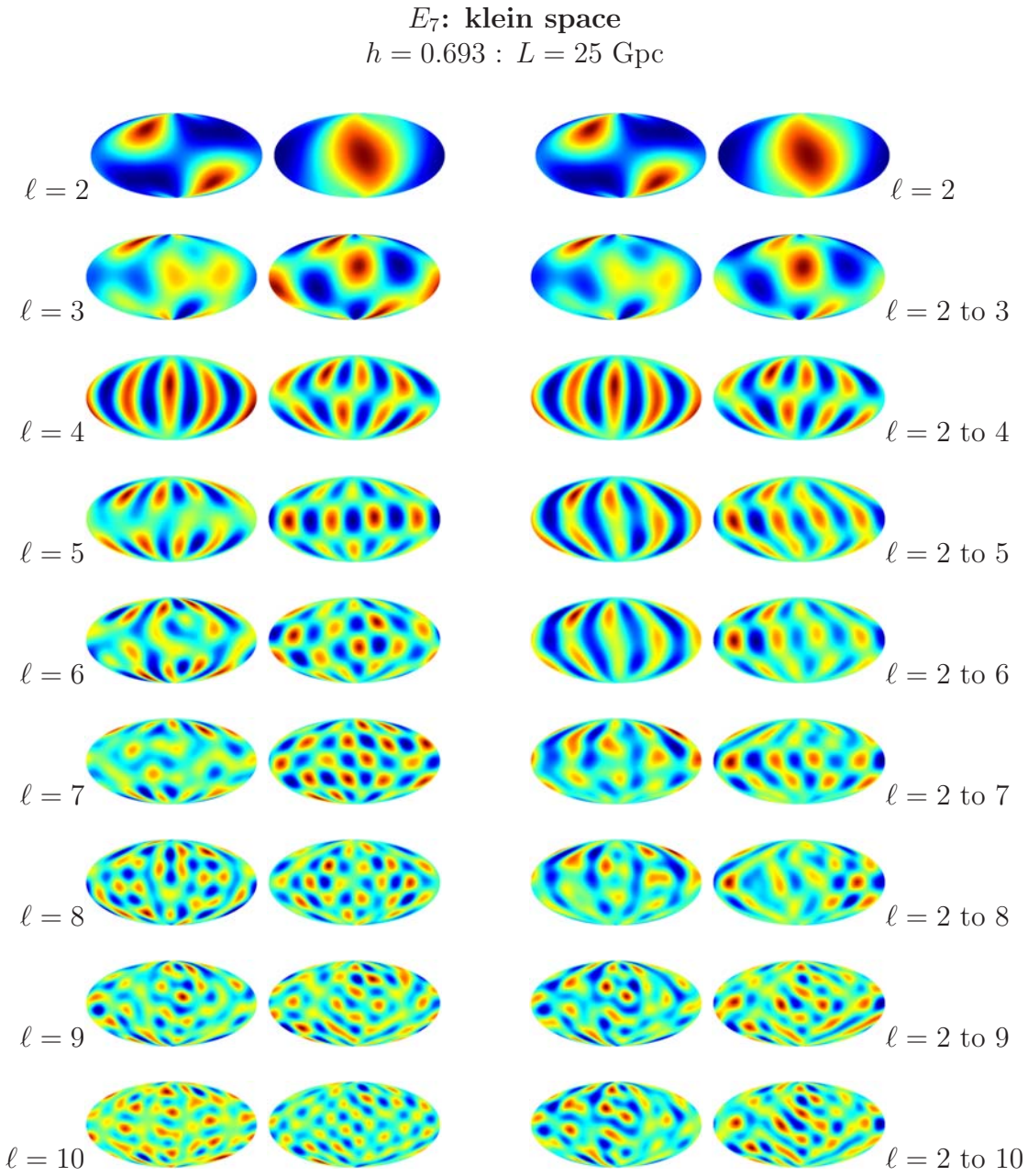


Figure 10.37: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.9)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

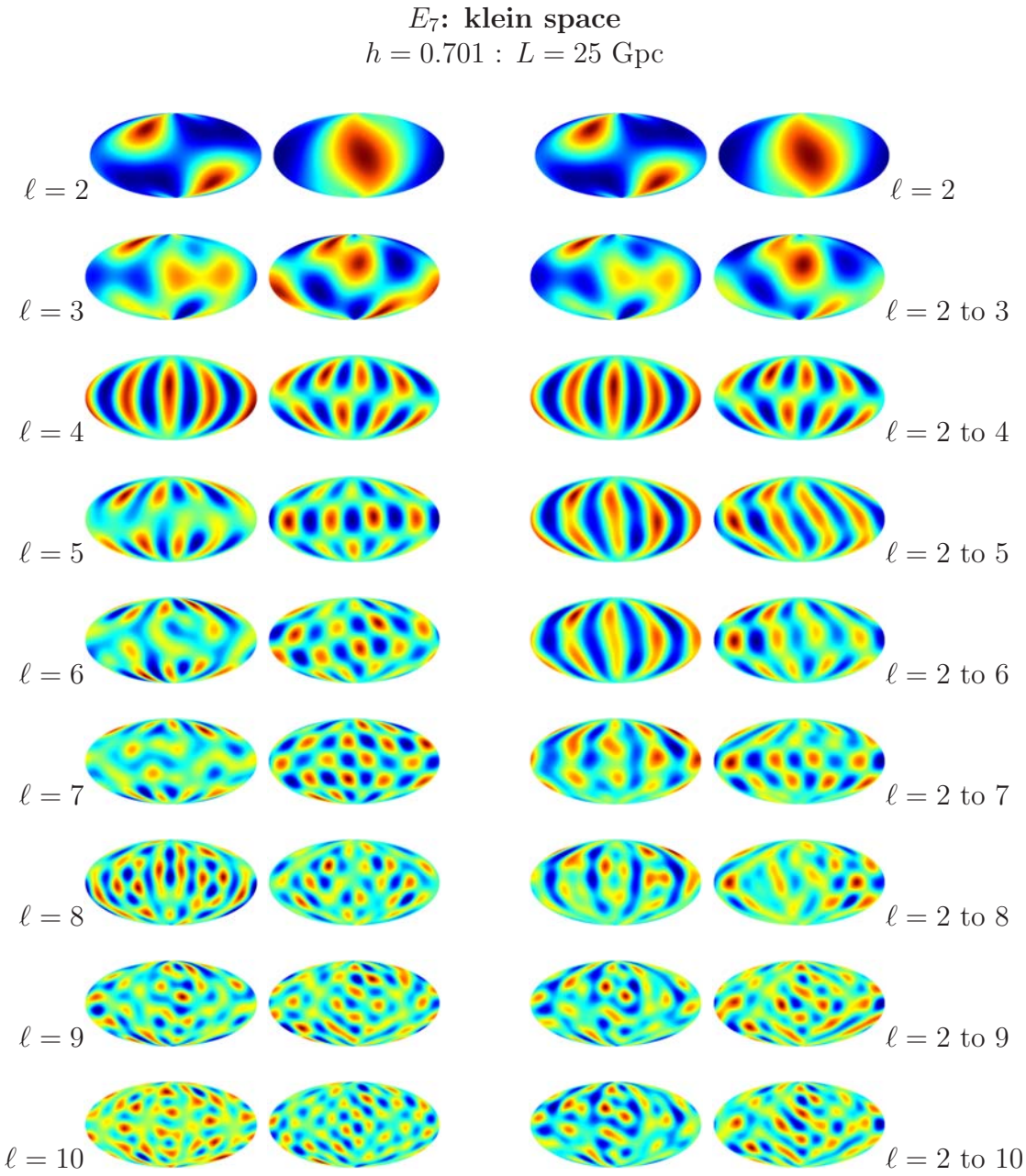


Figure 10.38: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.10)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

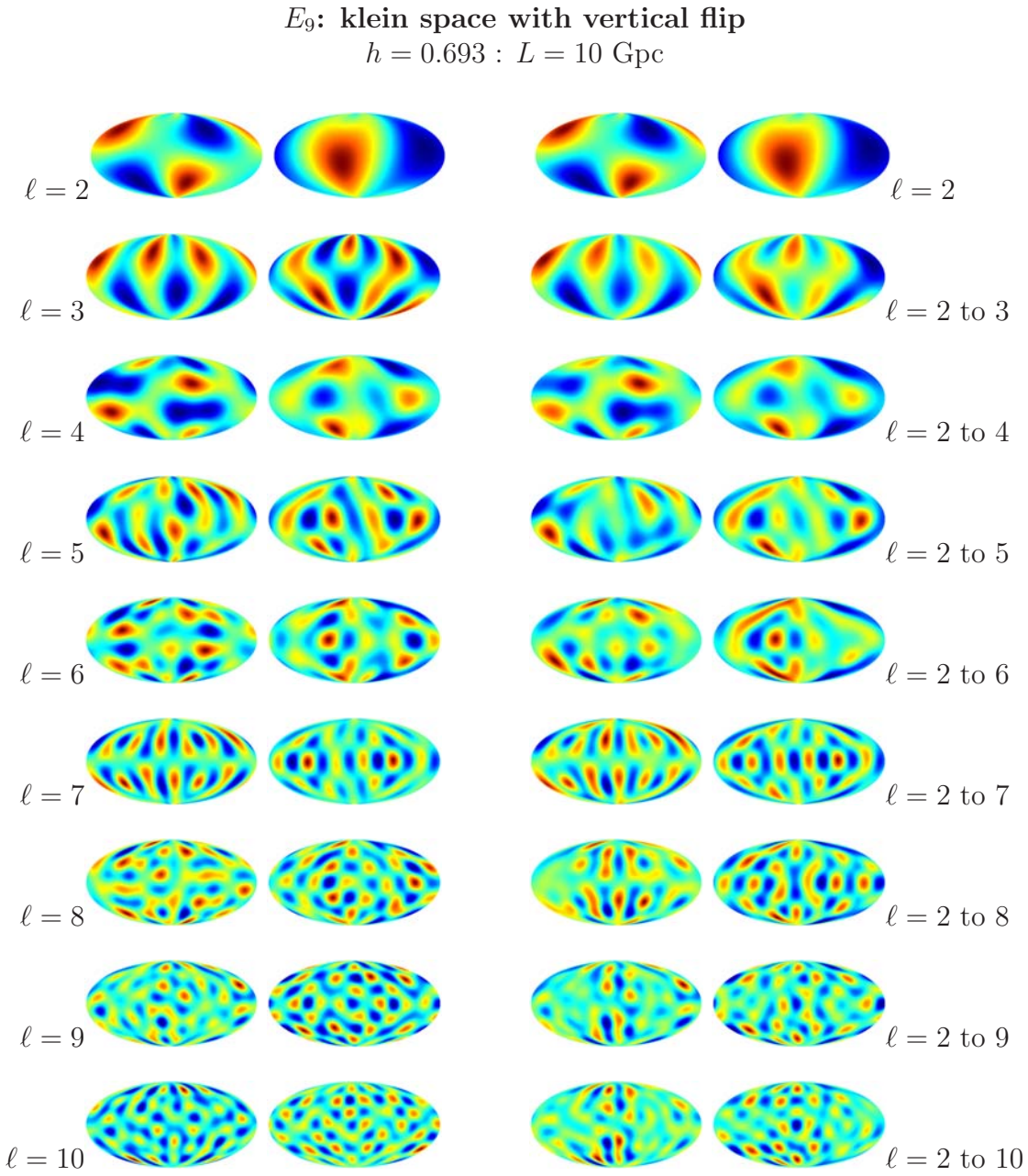


Figure 10.39: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.11)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

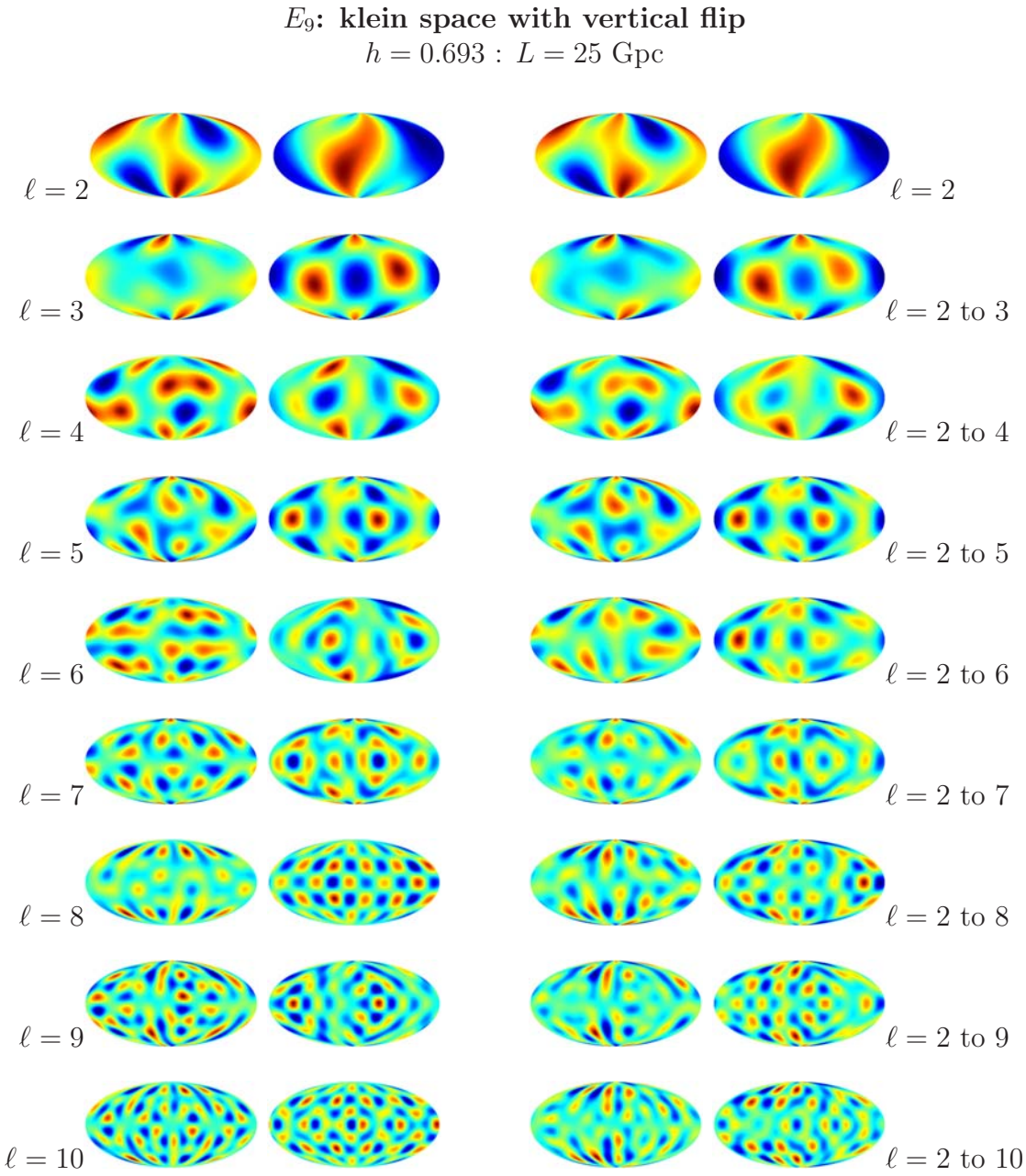


Figure 10.40: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.12)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

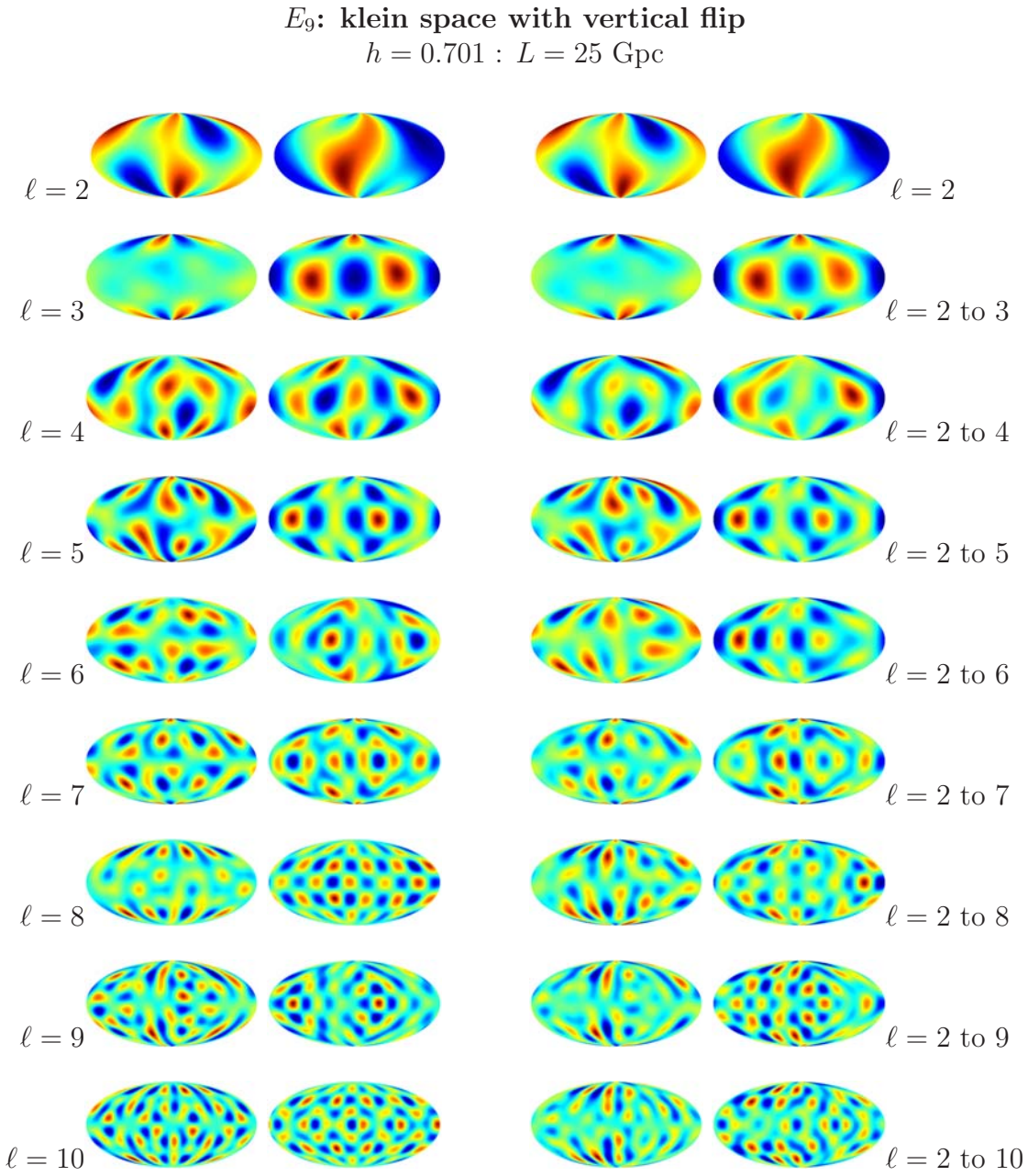


Figure 10.41: Simulated Q and U polarisation maps (generated using the EE correlation matrices represented in Fig. (10.13)). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

10.3 WMAP Data

This section breaks some of the real-data nine-year WMAP maps into individual multipole maps (left columns) and sequentially recombines them (right columns). They provide an interesting comparison to the simulated maps from the previous sections.

On visual inspection, it's hard to tell if the simulated data has the same underlying characteristics as the WMAP data. Other realisations of the simulated data may be a better visual match. In order to make a proper comparison, the statistics of the maps need to be compared. This is where a thorough Bayesian analysis would come in; unfortunately, the simple MCMC analysis attemped could not detect a maximum likelihood.

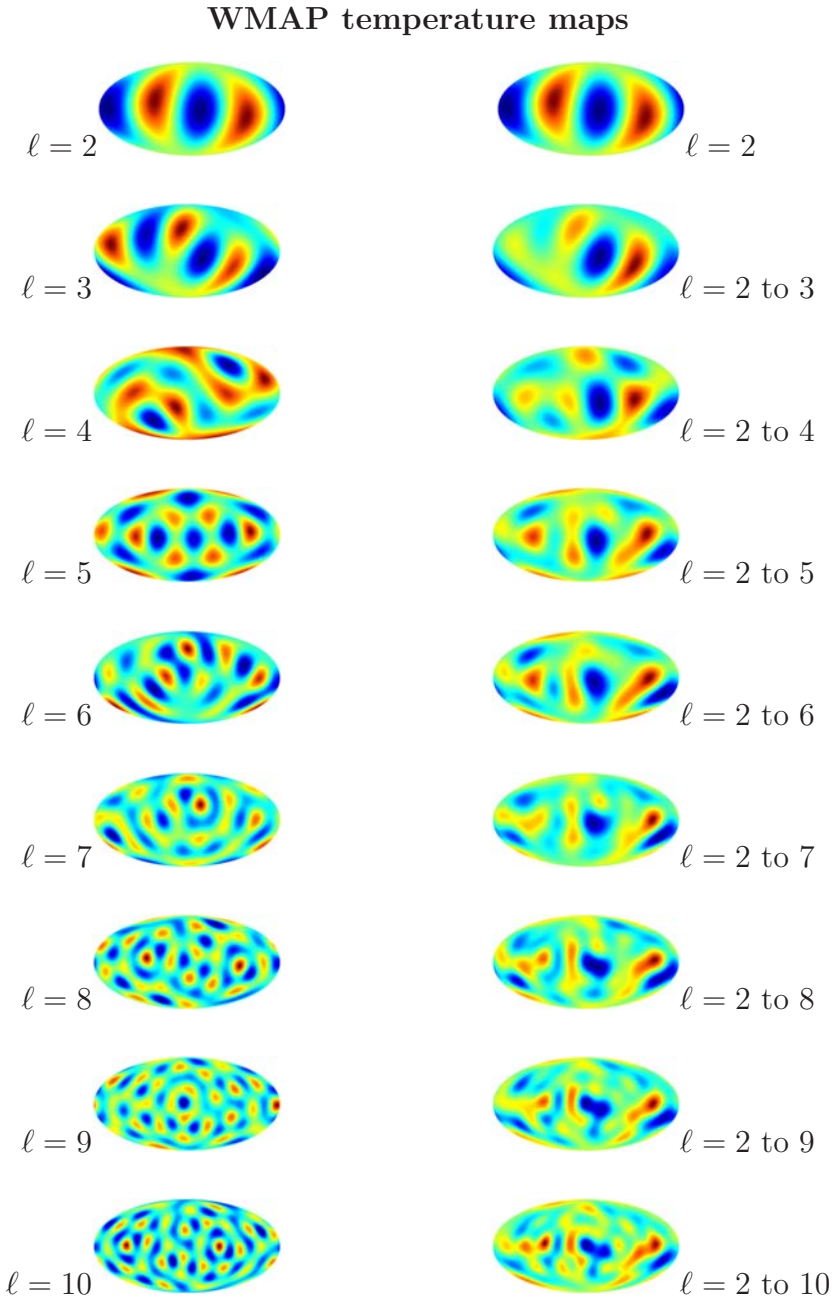


Figure 10.42: WMAP 9-year temperature maps. The left column contains maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added.

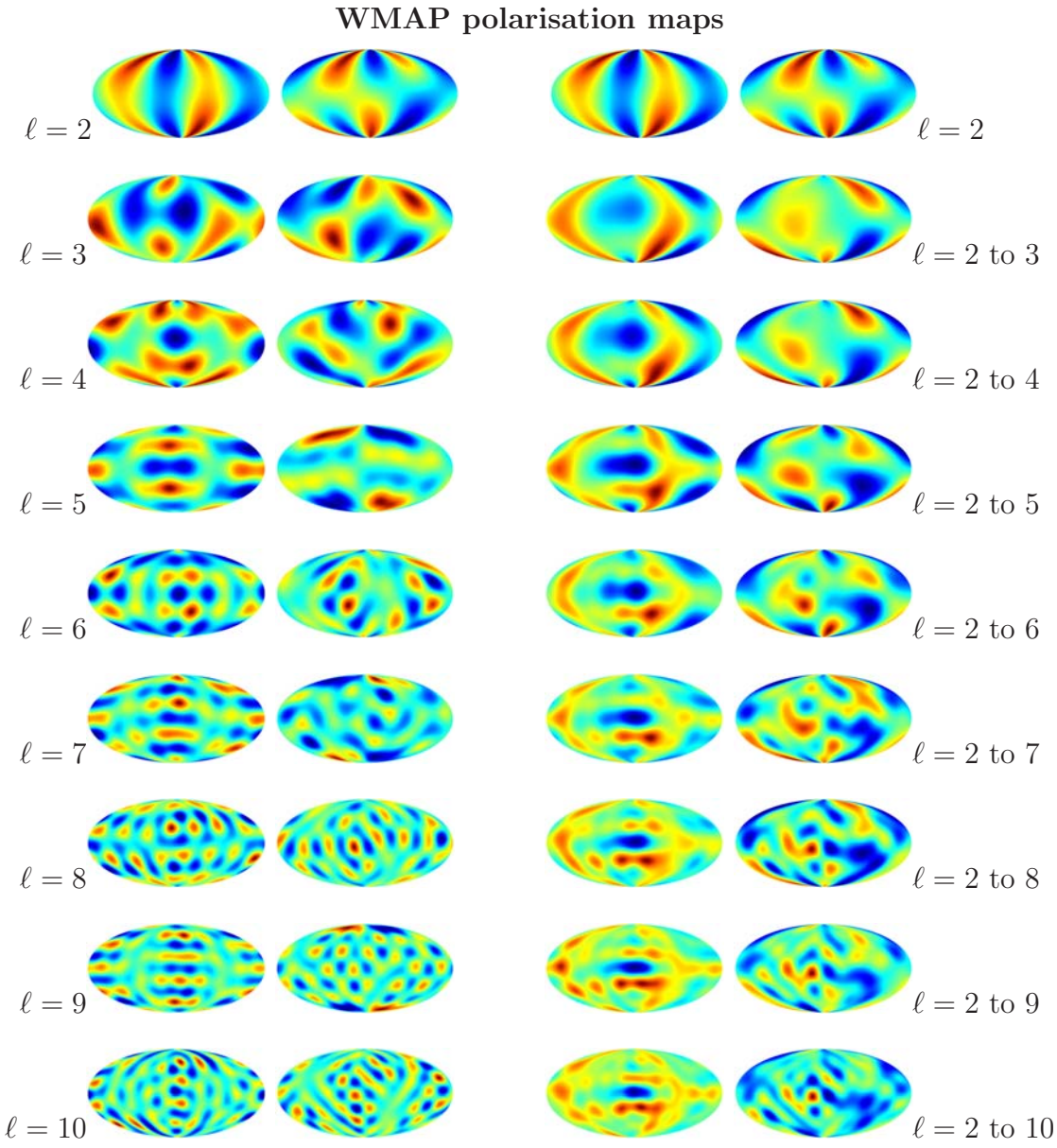


Figure 10.43: WMAP 9-year Q and U polarisation maps (for E-mode polarisation only). The two left columns contain polarisation maps for individual multipoles; the right column shows how the combined map changes as the individual multipole maps are added. Q is represented by columns 1 and 3, U by columns 2 and 4.

Chapter 11

Discussion

In this thesis, we set out to develop a method for constraining cosmic topology by utilising as much information as possible from the CMB; this involves making use of polarisation data, in addition to temperature, and taking the full CMB covariance matrix, instead of the power spectrum.

In Ch. 2, we produced a consistent catalogue of the properties and mathematical descriptors of all possible flat topologies. Such explicit lists are difficult to find in the literature, but it has proven very useful in deciding which topologies to begin the investigation with, and what steps need to be taken in order to extend the investigation to other flat spaces. A similar catalogue of spherical and hyperbolic spaces would surely be invaluable.

In Ch. 7, we were diverted by an investigation into the origins of UHECRs. We concluded that there is a strong chance that at least some UHECRs are produced by AGNs. More relevant to the thesis as a whole, we demonstrated how Bayesian methods can successfully make use of as much valuable information as possible where data is relatively scarce.

Ch. 9 is where we described our method for constraining cosmic topology using the polarised CMB. The majority of efforts were spent developing a code for calculating the full CMB correlation matrix that is efficient, yet non-reliant on high-powered computing and potentially freely available to anyone who may wish to develop it further. On an entry-level server with 24 GB RAM, it is capable of generating in excess of 231 covariance matrices in under a day, and probably significantly more, for a given topology (for $\ell_{\max} = 30$). In addition, for the types of calculations involved, it requires relatively little disk space (25 GB) to run.

By the time work on the correlation matrix code was complete, the Bayesian analysis had been somewhat neglected. The simple likelihood codes adopted still require debugging and further development. Lower limits on the size of the Universe are generally accepted to be greater than the size of the LSS, meaning that any signatures of topology in the CMB could be very weak. In addition, the CMB presents a number of challenges when it comes to extracting the pure signal. Hence, devising a successful analysis that truly makes the most of all available information is a large undertaking; an intuitive, powerful, community-maintained code for finding the covariance matrices would enable a greater proportion of efforts to go into these statistical analyses. The covariance matrix code presented in this thesis has many features that are ideal for a community-maintained code. It is fast, requires little permanent storage, requires moderate computing power (which could be in the form of a high-end desktop machine) and uses a freely available compiler (GNU c++).

The next step in developing this program for computing the topology-dependent CMB covariance matrix is to make it as accessible as possible. This is to be done through improved commenting and variable names within the code itself, the creation of instructive documentation on its use, and making it publicly available online (through GNU licensing or similar). Also to be built in, are the flexibility to set more of the parameters from outside the code and enhancements to the base class to enable easier addition of more complex topologies. With these improvements in flexiblity and accessibility, this code has the potential to be a valuable aid in achieving the the ultimate goal of a thorough and exhaustive exploration of what the CMB can tell us about cosmic topology.

Epilogue

There have been many attempts to investigate cosmic topology, but each assesses a limited selection of topologies over different ranges of cosmological parameters and sizes of the fundamental domain, as well as each employing their own method of analysis. More consistency is needed in order to meaningfully compare results and gain more insight into what the cosmic topology might be. It may be possible to eliminate many candidate topologies, but of the many that remain, we have to deal with relative probabilities. The probabilities are not absolute, but depend on the method of analysis. This makes comparing results from different methods difficult.

In addition, in order to utilise as much information as possible from the CMB, the full correlation matrix must be calculated. This must to be done for each topology, cosmology and size of fundamental domain to be investigated; calculating just one of these can be computationally expensive. Hence, a large amount of time is invested in creating correlation matrices. Since cosmic topology will be challenging to constrain no matter how precise observations are, more time needs to be spent on developing and fine-tuning methods of analysis.

The number of potential cosmic topolgies is vast and the effort that goes into investigating just one of them is far from trivial. I believe that, in order to make significant progress in the near future, a more collaborative approach needs to be taken. This could involve:

- building a communal catalogue of the coefficients $\xi^{\hat{k}}_{k\ell m}$ (the coefficients of the eigenfunctions of the universal covering space of a topology) which encapsulate the properties of a topology. There are many more candidate topologies for which these have not been determined than those for which they have.

- making codes for calculating the CMB correlation matrix publicly available. Not only would this save time, enabling individuals to use this code rather than writing their own, it would enable testing of the codes alongside one another. Ultimately, hopefully, this would lead to the development of a single code that can be fine-tuned for efficiency and accuracy.

- building a communal database of correlation matrices; if the correlation matrices are readily available rather than calculated by every individual or group who wish to try their method of analysis, the same analysis may be applied to more topologies, providing a more consistent comparison.

Creating a communal database for the correlation matrix should be feasible as there is not much flexibility in the way in which it is calculated. Efforts should be focused on the method of analysis. Perhaps a series of community challenges could be set to devise the most successful method. This has been done effectively for improving measurements of weak gravitational lensing in the GREAT08 and GREAT10 challenges (see Bridle et al. (2010) and Kitching et al. (2012) respectively). Simulated CMB data could be generated for a range of topologies (without stating what those topologies are). The covariance matrices could also be provided. The challenge would be to devise the method most successful at identifying the simulated data with the covariance matrix that generated them. Initial challenges could give more information to participants, e.g. the cosmological parameters used, including the curvature, when generating the data. Subsequent challenges could reveal less information as participants refine their codes.

Whatever path the investigation into cosmic topology takes in the future, the idea of determining the shape of the Universe remains a fascinating one.

Bibliography

Abbasi R. U., et al., 2008, Astroparticle Physics, 30, 175

Abraham J., et al., 2004, Nuclear Instruments and Methods in Physics Research A, 523, 50

—, 2007a, Astroparticle Physics, 27, 244

—, 2007b, Science, 318, 938

—, 2008, Astroparticle Physics, 29, 188

—, 2010, Physics Letters B, 685, 239

Abreu P., et al., 2010, Astroparticle Physics, in press (arXiv:1009.1855)

Achterberg A., Gallant Y. A., Norman C. A., Melrose D. B., 1999, ArXiv Astrophysics e-prints

Alpher R. A., Herman R., 1948, Nature, 162, 774

Archidiacono M., Fornengo N., Giunti C., Melchiorri A., 2012, PhysRevD, 86, 065028

Aslanyan G., Manohar A. V., Yadav A. P. S., 2013, JCAP, 8, 9

Aurich R., Lustig S., Steiner F., 2006, MNRAS, 369, 240

Bartlett J. G., 2006, Brazilian Journal of Physics, 36, 1124

Beatty J. J., Westerhoff S., 2009, Annual Review of Nuclear and Particle Science, 59, 319

Bennett C. L., Larson D., Weiland J. L., Jarosik N., Hinshaw G., Odegard N., Smith K. M., Hill R. S., Gold B., Halpern M., Komatsu E., Nolta M. R., Page L., Spergel D. N., Wollack E., Dunkley J., Kogut A., Limon M., Meyer S. S., Tucker G. S., Wright E. L., 2012, ArXiv e-prints

Bielewicz P., Banday A. J., 2011, MNRAS, 412, 2104

Bierberbach L., 1911, Mathematische Annalen, 70

Bond J. R., Efstathiou G., 1984, ApJL, 285, L45

Bond J. R., Pogosyan D., Souradeep T., 2000a, PhysRevD, 62, 043005

—, 2000b, PhysRevD, 62, 043006

Bridle S., Balan S. T., Bethge M., Gentile M., Harmeling S., Heymans C., Hirsch M., Hosseini R., Jarvis M., Kirk D., Kitching T., Kuijken K., Lewis A., Paulin-Henriksson S., Schölkopf B., Velander M., Voigt L., Witherick D., Amara A., Bernstein G., Courbin F., Gill M., Heavens A., Mandelbaum R., Massey R., Moghaddam B., Rassat A., Réfrégier A., Rhodes J., Schrabback T., Shawe-Taylor J., Shmakova M., van Waerbeke L., Wittman D., 2010, MNRAS, 405, 2044

Bunn E. F., Bourdon A., 2008, PhysRevD, 78, 123509

Callin P., 2006, ArXiv Astrophysics e-prints

Chandrasekhar S., 1950, Radiative transfer.

Clay J., Berlage H. P., 1932, Naturwissenschaften, 20, 687

Cornish N. J., Spergel D. N., Starkman G. D., 1998, Classical and Quantum Gravity, 15, 2657

Crittenden R. G., 2004, Pramana, 63, 829

de Sitter W., 1917, Koninklijke Nederlandse Akademie van Wetenschappen Proceedings Series B Physical Sciences, 19, 1217

Demianski M., Lapucha M., 1987, MNRAS, 224, 527

Dicke R. H., Peebles P. J. E., Roll P. G., Wilkinson D. T., 1965, ApJ, 142, 414

Diehl R. L., 2009, Eur. Phys. J. D, 55, 509

Dodelson S., 2003, Modern cosmology, Dodelson, S., ed.

Dolag K., Grasso D., Springel V., Tkachev I., 2005, JCAP, 1, 9

Drell P. S., Loredo T. J., Wasserman I., 2000, ApJ, 530, 593

Dyer C. C., 1987, in Theory and Observational Limits in Cosmology, Stoeger W. R., ed., p. 467

Efstathiou G., 2004, MNRAS, 348, 885

Einstein A., 1917, Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften (Berlin), Seite 142-152., 142

Eisenstein D. J., Hu W., Tegmark M., 1998, ApJL, 504, L57

Ellis G. F. R., 1971, General Relativity and Gravitation, 2, 7

Eriksen H. K., O'Dwyer I. J., Jewell J. B., Wandelt B. D., Larson D. L., Górski K. M., Levin S., Banday A. J., Lilje P. B., 2004, ApJS, 155, 227

Fagundes H. V., 1983, Physical Review Letters, 51, 517

—, 1989, ApJ, 338, 618

Fagundes H. V., Wichoski U. F., 1987, ApJL, 322, L5

Feodoroff E., 1885, Russian journalfor crystallography and mineralogy, 21

Feroz F., Hobson M. P., Bridges M., 2009, MNRAS, 398, 1601

Fixsen D. J., Cheng E. S., Gales J. M., Mather J. C., Shafer R. A., Wright E. L., 1996, ApJ, 473, 576

Fodor Z., Katz S. D., 2001, PhysRevD, 63, 023002

Frankland W. B., 1913, Math. Gaz.

Fraschetti F., Melia F., 2008, MNRAS, 391, 1100

Friedmann A., 1922, Zeitschrift fur Physik, 10, 377

—, 1924, Zeitschrift fur Physik, 21, 326

Gamow G., 1948, Nature, 162, 680

George M., et al., 2008, MNRAS, 388, L59

Ghisellini G., Ghirlanda G., Tavecchio F., Fraternali F., Pareschi G., 2008, MNRAS, 390, L88

Gomero G. I., 2003, Classical and Quantum Gravity, 20, 4775

Gorski K. M., Wandelt B. D., Hansen F. K., Hivon E., Banday A. J., 1999, ArXiv Astrophysics e-prints

Gott III J. R., 1980, MNRAS, 193, 153

Greisen K., 1966, Physical Review Letters, 16, 748

Gruetjen H. F., Shellard E. P. S., 2012, ArXiv e-prints

Guth A. H., 1981, PhysRevD, 23, 347

Hearin A. P., Gibelyou C., Zentner A. R., 2011, JCAP, 10, 12

Hess V. F., 1912, Z. Phys., 13

Hinshaw G., Larson D., Komatsu E., Spergel D. N., Bennett C. L., Dunkley J., Nolta M. R., Halpern M., Hill R. S., Odegard N., Page L., Smith K. M., Weiland J. L., Gold B., Jarosik N., Kogut A., Limon M., Meyer S. S., Tucker G. S., Wollack E., Wright E. L., 2012, ArXiv e-prints

Hirata C. M., Loeb A., Afshordi N., 2005, PhysRevD, 71, 063531

Hu W., Hedman M. M., Zaldarriaga M., 2003, PhysRevD, 67, 043004

Hu W., White M., 1997, New Astronomy, 2, 323

Huffman C., 2005, Archytas of Tarentum, Pythagorean, Philosopher and Mathematician King

Jaffe A., 1996, ApJ, 471, 24

Jaffe A. H., Bond J. R., Ferreira P. G., Knox L. E., 1999, in American Institute of Physics Conference Series, Vol. 476, 3K cosmology, Maiani L., Melchiorri F., Vittorio N., eds., pp. 249–365

Jewell J. B., Eriksen H. K., Wandelt B. D., O'Dwyer I. J., Huey G., Górski K. M., 2009, ApJ, 697, 258

Jungman G., Kamionkowski M., Kosowsky A., Spergel D. N., 1996a, PhysRevD, 54, 1332

—, 1996b, Physical Review Letters, 76, 1007

Kamionkowski M., Kosowsky A., Stebbins A., 1997, PhysRevD, 55, 7368

Kaplan J., Delabrouille J., Fosalba P., Rosset C., 2003, Comptes Rendus Physique, 4, 917

Key J. S., Cornish N. J., Spergel D. N., Starkman G. D., 2007, PhysRevD, 75, 084034

Kim H. B., Kim J., 2011, JCAP, 3, 6

Kitching T. D., Balan S. T., Bridle S., Cantale N., Courbin F., Eifler T., Gentile M., Gill M. S. S., Harmeling S., Heymans C., Hirsch M., Honscheid K., Kacprzak T., Kirkby D., Margala D., Massey R. J., Melchior P., Nurbaeva G., Patton K., Rhodes J., Rowe B. T. P., Taylor A. N., Tewes M., Viola M., Witherick D., Voigt L., Young J., Zuntz J., 2012, MNRAS, 423, 3163

Knox L., Song Y.-S., 2002, Physical Review Letters, 89, 011303

Kogut A., Spergel D. N., Barnes C., Bennett C. L., Halpern M., Hinshaw G., Jarosik N., Limon M., Meyer S. S., Page L., Tucker G. S., Wollack E., Wright E. L., 2003, ApJS, 148, 161

Komatsu E., Dunkley J., Nolta M. R., Bennett C. L., Gold B., Hinshaw G., Jarosik N., Larson D., Limon M., Page L., Spergel D. N., Halpern M., Hill R. S., Kogut A., Meyer S. S., Tucker G. S., Weiland J. L., Wollack E., Wright E. L., 2009, ApJS, 180, 330

Kosowsky A., 1996, Annals of Physics, 246, 49

—, 1999, New A. Rev., 43, 157

Kovac J. M., Leitch E. M., Pryke C., Carlstrom J. E., Halverson N. W., Holzapfel W. L., 2002, Nature, 420, 772

Kunz M., Aghanim N., Cayon L., Forni O., Riazuelo A., Uzan J. P., 2006, PhysRevD, 73, 023511

Lachieze-Rey M., Luminet J.-P., 1995, Phys. Rept., 254, 135

Larson D. L., Eriksen H. K., Wandelt B. D., Górski K. M., Huey G., Jewell J. B., O'Dwyer I. J., 2007, ApJ, 656, 653

Lemaître G., 1927, Annales de la Société Scientifique de Bruxelles, 47, 49

—, 1958, La Structure et l'Evolution de l'Univers

Levin J., 2002, Physics Reports, 365, 251

Lewis A., Bridle S., 2002, Phys. Rev., D66, 103511

Lewis A., Challinor A., 2006, Physics Reports, 429, 1

Linde A. D., 1983, in Very Early Universe, Gibbons G. W., Hawking S. W., Siklos S. T. C., eds., pp. 205–249

—, 1984, Nuovo Cimento Lettere, 39, 401

—, 1986, Physics Letters B, 175, 395

Loredo T. J., 1990, Fundamental Theories of Physics, 39

Lubin P., Melese P., Smoot G., 1983, ApJL, 273, L51

Luminet J.-P., 1998, Acta Cosmologica, 24, 105

MacTavish C. J., Ade P. A. R., Bock J. J., Bond J. R., Borrill J., Boscaleri A., Cabella P., Contaldi C. R., Crill B. P., de Bernardis P., De Gasperis G., de Oliveira-Costa A., De Troia G., di Stefano G., Hivon E., Jaffe A. H., Jones W. C., Kisner T. S., Lange A. E., Lewis A. M., Masi S., Mauskopf P. D., Melchiorri A., Montroy T. E., Natoli P., Netterfield C. B., Pascale E., Piacentini F., Pogosyan D., Polenta G., Prunet S., Ricciardi S., Romeo G., Ruhl J. E., Santini P., Tegmark M., Veneziani M., Vittorio N., 2006, ApJ, 647, 799

Mainini R., Minelli D., Gervasi M., Boella G., Sironi G., Baú A., Banfi S., Passerini A., De Lucia A., Cavaliere F., 2013, JCAP, 8, 33

Medina Tanco G. A., de Gouveia dal Pino E. M., Horvath J. E., 1998, ApJ, 492, 200

Miller L., CFHTLenS Collaboration, 2012, in American Astronomical Society Meeting Abstracts, Vol. 219, American Astronomical Society Meeting Abstracts #219, p. 130.03

Montroy T. E., Ade P. A. R., Bock J. J., Bond J. R., Borrill J., Boscaleri A., Cabella P., Contaldi C. R., Crill B. P., de Bernardis P., De Gasperis G., de Oliveira-Costa A., De Troia G., di Stefano G., Hivon E., Jaffe A. H., Kisner T. S., Jones W. C., Lange A. E., Masi S., Mauskopf P. D., MacTavish C. J., Melchiorri A., Natoli P., Netterfield C. B., Pascale E., Piacentini F., Pogosyan D., Polenta G., Prunet S., Ricciardi S., Romeo G., Ruhl J. E., Santini P., Tegmark M., Veneziani M., Vittorio N., 2006, ApJ, 647, 813

Nagano M., Watson A. A., 2000, Reviews of Modern Physics, 72, 689

Nemmen R. S., Bonatto C., Storchi-Bergmann T., 2010, ApJ, 722, 281

Niarchou A., 2006, PhD thesis, Imperial College, London

Niarchou A., Jaffe A. H., 2006, in American Institute of Physics Conference Series, Vol. 848, Recent Advances in Astronomy and Astrophysics, N. Solomos, ed., pp. 774–778

Novacki W., 1934, Commentarii Mathematici Helvetici, 7

Pacini D., 1912, Nuovo Cimento

Peebles P. J. E., 1981, ApJL, 243, L119

Peebles P. J. E., Yu J. T., 1970, ApJ, 162, 815

Penzias A. A., Wilson R. W., 1965, ApJ, 142, 419

Phillips N. G., Kogut A., 2006, ApJ, 645, 820

Piacentini F., Ade P. A. R., Bock J. J., Bond J. R., Borrill J., Boscaleri A., Cabella P., Contaldi C. R., Crill B. P., de Bernardis P., De Gasperis G., de Oliveira-Costa A., De Troia G., di Stefano G., Hivon E., Jaffe A. H., Kisner T. S., Jones W. C., Lange A. E., Masi S., Mauskopf P. D., MacTavish C. J., Melchiorri A., Montroy T. E., Natoli P., Netterfield C. B., Pascale E., Pogosyan D., Polenta G., Prunet S., Ricciardi S., Romeo G., Ruhl J. E., Santini P., Tegmark M., Veneziani M., Vittorio N., 2006, ApJ, 647, 833

Planck Collaboration, 2005, The Scientific Programme of Planck. ESA

Planck Collaboration, Ade P. A. R., Aghanim N., Armitage-Caplan C., Arnaud M., Ashdown M., Atrio-Barandela F., Aumont J., Baccigalupi C., Banday A. J., et al., 2013a, ArXiv e-prints

—, 2013b, ArXiv e-prints

Planck collaboration, Ade P. A. R., Aghanim N., Armitage-Caplan C., Arnaud M., Ashdown M., Atrio-Barandela F., Aumont J., Baccigalupi C., Banday A. J., et al., 2013c, ArXiv e-prints

Planck Collaboration, Ade P. A. R., Aghanim N., Armitage-Caplan C., Arnaud M., Ashdown M., Atrio-Barandela F., Aumont J., Baccigalupi C., Banday A. J., et al., 2013d, ArXiv e-prints

Protheroe R. J., Szabo A. P., 1992, Physical Review Letters, 69, 2885

Rassat A., Starck J.-L., 2013, ArXiv e-prints

Riazuelo A., Caillerie S., Lachièze-Rey M., Lehoucq R., Luminet J., 2006, ArXiv Astrophysics e-prints

Rudjord Ø., Groeneboom N. E., Eriksen H. K., Huey G., Górski K. M., Jewell J. B., 2009, ApJ, 692, 1669

Sarkar S., 2002, ArXiv High Energy Physics - Phenomenology e-prints

Schwarzschild K., 1900, ViertelJAhrsschrift der Astronischen Gesellschaft, 35, 337

Scott D., Smoot G. F., 2008, Physics Letters B, 667, 1

Shankaranarayanan S., Sriramkumar L., 2005, in PASCOS 2004 Part I: Particles, Strings and Cosmology, Alverson G., Barberis E., Nath P., Vaughn M. T., eds., p. 38

Sigl G., Miniati F., Enßlin T. A., 2004, PhysRevD, 70, 043007

Smith K. M., Hanson D., LoVerde M., Hirata C. M., Zahn O., 2012, JCAP, 6, 14

Smoot G. F., 1999, COBE observations and results

Sokolov D. D., Shvartsman V. F., 1974, Soviet Journal of Experimental and Theoretical Physics, 39, 196

Sokolov D. D., Starobinski A. A., 1975, Astronomicheskii Zhurnal, 52, 1041

Sommerville D. M. Y., 1913, The Elements of Non-Euclidean Geometry

Stark R. F., 1981, MNRAS, 195, 115

Starkman G. D., 1998, Classical and Quantum Gravity, 15, 2529

Starobinski A. A., 1982, Physics Letters B, 117, 175

Stoker P. H., 2009, Advances in Space Research, 44, 1081

Tegmark M., 1998, ApJ, 502, 1

Thurston W. P., 1997, Three-dimensional Geometry and Topology, Volume 1

Thurston W. P., Weeks J. R., 1984, Scientific American, 251, 108

Tolman B. W., Matzner R. A., 1984, Royal Society of London Proceedings Series A, 392, 391

Trotta R., 2008, Contemporary Physics, 49, 71

Uzan J.-P., Riazuelo A., Lehoucq R., Weeks J., 2004, PhysRevD, 69, 043003

Varshalovich D. A., Moskalev A., Khersonski V., 1988, Quantum Theory of Angular Momentum. World Scientific, Singapore

Veron-Cetty M. P., Veron P., 2006, Astronomy and Astrophysics, 455, 773

Wandelt B. D., Larson D. L., Lakshminarayanan A., 2004, PhysRevD, 70, 083511

Wang X., Tegmark M., Zaldarriaga M., 2002, PhysRevD, 65, 123001

Watson L. J., Mortlock D. J., Jaffe A. H., 2011, MNRAS, 418, 206

Waxman E., 1995, Physical Review Letters, 75, 386

Weeks J., 2002, The Shape of Space. Marcel Dekker, New York

Wick S. D., Dermer C. D., Atoyan A., 2004, Astroparticle Physics, 21, 125

Zaldarriaga M., 1997, PhysRevD, 55, 1822

Zaldarriaga M., Harari D. D., 1995, PhysRevD, 52, 3276

Zaldarriaga M., Seljak U., 1997, PhysRevD, 55, 1830

—, 2000, ApJS, 129, 431

Zaldarriaga M., Spergel D. N., Seljak U., 1997, ApJ, 488, 1

Zarei M., Bavarsad E., Haghighat M., Mohammadi R., Motie I., Rezaei Z., 2010, PhysRevD, 81, 084035

Zatsepin G. T., Kuz'min V. A., 1966, Soviet Journal of Experimental and Theoretical Physics Letters, 4, 78

Zeldovich Y. B., Starobinskii A. A., 1984, Soviet Astronomy Letters, 10, 135

Appendices

Appendix A

Code for Computing the CMB Correlation Matrix for E_1, E_3, E_7 and E_9

Included here, is more detail on the key part of the code written as part of the work presented in this thesis; the code for calculating the CMB covariance matrices of different multi-connected flat spaces, E_1, E_3, E_7 and E_9. The code is currently in its pre-alpha stage; it needs further development to make it accessible to other users. This appendix is divided into the following sections:

- Section A.1 - Usage
- Section A.2 - Hierarchy
- Section A.3 - The Code

A.1 Usage

Usage: the code is executed at the command line in the following way:

topoterms ⟨topology_number⟩ ⟨ell_max⟩ ⟨nsq_min⟩ ⟨nsq_max⟩ [nsq_inc]

where the arguments are:

- topology_number: 1, 3, 7, 9 (representing E_1, E_3, E_7, E_9,)
- ell_min: $2 \leq$ ell_max ≤ 50

- nsq_min: $0 \leq$ nsq_min ≤ 10000 where nsq $= n$ from $k = 2\pi n/L$. Note, covariance matrices are calculated for $0 <$ nsq $\leq$ nsq_max; see nsq_inc for the role of the parameter nsq_min.

- nsq_max: nsq_min $<$ ell_max ≤ 10000

- nsq_inc: nsq_max $-$ nsq_min. Specifies how often (the interval in nsq) a convergence test is performed on the covariance matrix. nsq_min is the value of nsq at which convergence tests start; a convergence test is then performed every time nsq has increased by nsq_inc, until nsq_max is reached. Results of the tests are output to screen as they are done. If nsq_inc is zero or not defined by the user, no test is performed.

Input files: files containing the radiation transfer functions calculated using CAMB.

Output files: files containing the CMB covariance matrices.

A.2 Hierachy

The code files are listed below, with a brief description of what they are. Note that the original intention was to write a code to calculate the topoterms and save them to file. The topoterms would then be read by a code designed to calculate the covariance matrices. However, it proved more efficient to calculate the covariance matrix immediately, removing the need to write/read the topoterms to/from file. The code files, and objects and functions within, were named according to the original intention; the names have not been updated to make it clear that the covariance matrix is being calculated as well as the topoterms.

- topoterms.cpp - wrapper for classes listed below, sets values of parameters, tests for convergence of correlation matrices

- c_topoterms.cpp - base class, finds covariance matrices for E_1 (also has a header file, topoterms.h)

- c_topoterms3.cpp - class, inherits from topoterms.cpp, finds covariance matrices for E_3 (also has a header file, topoterms3.h)

- c_topoterms7.cpp - class, inherits from topoterms.cpp, finds covariance matrices for E_7 (also has a header file, topoterms7.h)
- c_topoterms9.cpp - class, inherits from topoterms.cpp, finds covariance matrices for E_9 (also has a header file, topoterms9.h)

An outline of the alogorithm employed in the topoterms classes is depicted in Fig. (A.1).

A.3 The Code

The following pages display the code itself with the files appearing in the following order:

- topoterms.cpp - 15 pages
- c_topoterms.h - 7 pages
- c_topoterms.cpp - 32 pages
- c_topoterms3.h - 1 page
- c_topoterms3.cpp - 17 pages
- c_topoterms7.h - 1 page
- c_topoterms7.cpp - 7 pages
- c_topoterms9.h - 1 page
- c_topoterms9.cpp - 7 pages

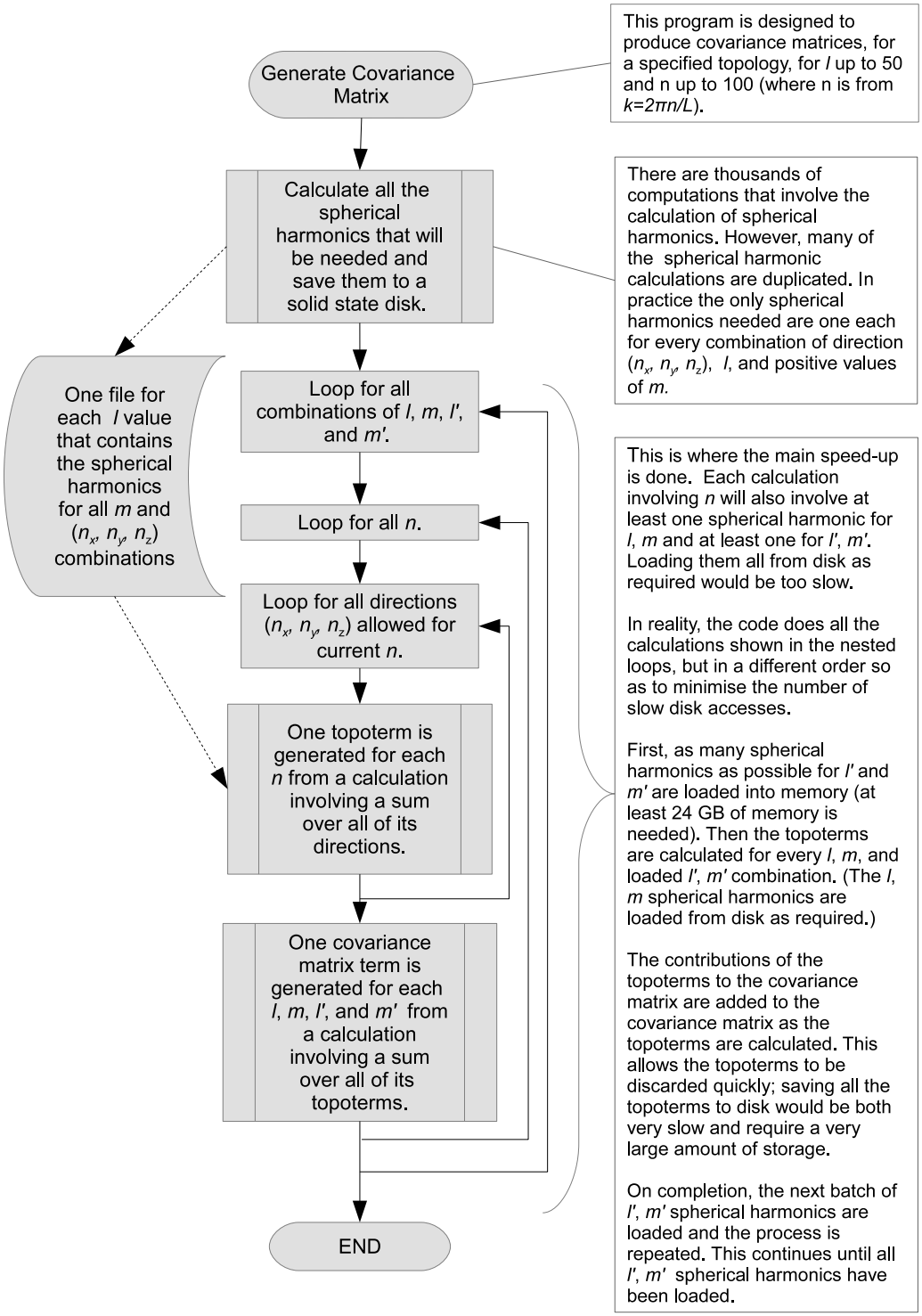


Figure A.1: Flow chart showing the alogorithm employed in c_topoterms.cpp, c_topoterms3.cpp, c_topoterms7.cpp and c_topoterms9.cpp.

topoterms.cpp

```
#include "c_topoterms.h"
#include "c_topoterms3.h"
#include "c_topoterms7.h"
#include "c_topoterms9.h"
void gen_covarmatrix(int TT_Number, int ellmin, int ellmax, int nsq_min, int nsq_max);
void add_to_covarmatrix_array(c_topoterms::covamatrix_t* covamatrix_dest, c_topoterms::covamatrix_t*
covamatrix_source);
bool compare_two_covarmatrix_array(c_topoterms::covamatrix_t* covamatrix_1, c_topoterms::covamatrix_t* covamatrix_2);
bool read_covarmatrix_file(std::string filename, c_topoterms::covamatrix_t *covamatrix_results);
bool write_covarmatrix_file(std::string filename, c_topoterms::covamatrix_t *covamatrix_data);


// Set values of L and h to calculate covariance matrix for:

#define INCLUDE_SPECIAL_L
const double L[] { 21000, 23000, 25000, 27000, 29000, 31000, 33000, 35000 }; // size of f.d. in Mpc
const double h[] = { 0.685, 0.687, 0.689, 0.691, 0.693, 0.695, 0.697, 0.699, 0.701 }; // Hubble parameter

#ifdef INCLUDE_SPECIAL_L
const double special_L[5] = { 10000, 12000, 14000, 16000, 18000 };
#endif

const int number_of_L_terms = sizeof(L) / sizeof(*L);
const int number_of_h_terms = sizeof(h) / sizeof(*h);

#ifdef INCLUDE_SPECIAL_L
const int number_of_special_L_terms = sizeof(special_L) / sizeof(*special_L);
const int number_of_h_L_terms = (number_of_L_terms * number_of_h_terms) + number_of_special_L_terms;
#else
const int number_of_h_L_terms = (number_of_L_terms * number_of_h_terms);
#endif
c_topoterms::h_L_terms_t h_L_terms[number_of_h_L_terms];
```

topoterms.cpp

```
int main(int argc, char* argv[])
{

    // Set values of ell and nsq (nsq=n*n, k=2*pi*n/L) to calculate covariance matrix for:
    // (User is prompted to input these on screen.)

    const int ellmin = 2;
    int ellmax = 50;
    int TT_Number, nsq_min, nsq_max, nsq_inc;   // nsq ranges from 0 to nsq_max. nsq_min is first value of nsq
                                                // for which convergence of covariance  is tested. nsq_inc defines
                                                // how often (in terms of increase in nsq) convergence is tested.
                                                // If nsq=0 or is undefined by user, no convergence test is done/

    try
    {
        if (argc != 5 && argc != 6)
            throw(1);

        TT_Number = atoi(argv[1]);
        ellmax = atoi(argv[2]);
        nsq_min = atoi(argv[3]);
        nsq_max = atoi(argv[4]);
        nsq_inc = (argc == 6) ? atoi(argv[5]) : (0);

        if ((TT_Number < 1 || TT_Number > 3) && TT_Number != 7 &&  TT_Number != 9)
            throw(1);
        if (ellmax < 2 || ellmax  > 50)
            throw(1);
        if (nsq_min < 0 || nsq_min > 10000)
```

topoterms.cpp

```
        throw(1);
    if (nsq_max < nsq_min || nsq_max > 10000)
        throw(1);
    if (nsq_inc > (nsq_max - nsq_min))
        throw(1);
}
catch (int errorcode)
{
    std::cout << "usage: topoterms topology_number ell_max nsq_min nsq_max [nsq_inc]\n";
    std::cout << "   :where\n";
    std::cout << "         topology number = 1, 3, 7 or 9 \n";
    std::cout << "         2 <= ell <= 50 \n";
    std::cout << "         0 <= nsq_min <= 10000\n";
    std::cout << "         nsq_min < nsq_max <= 10000 \n";
    std::cout << "         0 <= nsq_inc <= (nsq_max - nsq_min)\n";
    return (errorcode);
}
time_t t = time(0);   // get time now
struct tm * now = localtime(&t);
std::cout << "\n\nStarted at " << (now->tm_year + 1900) << '-' << (now->tm_mon + 1) << '-' << now->tm_mday << ' '
<< now->tm_hour << ':' << now->tm_min
        << std::endl;
//std::cout << sizeof(_Complex double) *((MAX_ELL * 2) + 1) << std::endl;

std::ostringstream filename;
std::string empty_str;
empty_str = "";

std::cout << "set " << number_of_h_L_terms << " h and L terms" << std::endl;
```

```
    for (int h_index = 0; h_index < number_of_h_terms; h_index++)
    {
        for (int L_index = 0; L_index < number_of_L_terms; L_index++)
        {
            h_L_terms[L_index + (h_index * number_of_L_terms)].h = h[h_index];
            h_L_terms[L_index + (h_index * number_of_L_terms)].L = L[L_index];
        }
    }

#ifdef INCLUDE_SPECIAL_L
    for (int special_L_index = 0; special_L_index < number_of_special_L_terms; special_L_index++)
    {
        h_L_terms[(number_of_L_terms * number_of_h_terms) + special_L_index].h = 0.693;
        h_L_terms[(number_of_L_terms * number_of_h_terms) + special_L_index].L = special_L[special_L_index];
    }
#endif

    c_topoterms::covamatrix_t *last_covarmatrix_results = (c_topoterms::covamatrix_t*) new c_topoterms::covamatrix_t;
    c_topoterms::covamatrix_t *current_covarmatrix_results = (c_topoterms::covamatrix_t*) new
c_topoterms::covamatrix_t;


    // Generate covariance matrices.
    // If nsq_inc is defined and non-zero; if so, do convergence test for covariance matrix.
    // Covergence test data is output to screen as matrices are calculated and tests are performed.

    if (nsq_inc == 0)
    {
        gen_covarmatrix(TT_Number, ellmin, ellmax, nsq_min, nsq_max);
    }
    else
```

topoterms.cpp

```
    {
        std::ostringstream prefix[3];
            prefix[0] << "topology" << TT_Number << "_TT";
            prefix[1] << "topology" << TT_Number << "_TE";
            prefix[2] << "topology" << TT_Number << "_EE";

        gen_covarmatrix(TT_Number, ellmin, ellmax, 0, nsq_min - 1);   //, last_covarmatrix_results);

    #if(1)
        for (int TF_index = 0; TF_index < number_of_h_L_terms; TF_index++)
            {
                for (int prefix_index = 0; prefix_index < 3; prefix_index++)
                {
                    filename.str(empty_str);
                    filename << "/PHD_Share/SSD/Temp/covarmatrix-p_" << prefix[prefix_index].str() << "_L:" << h_L_terms
    [TF_index].L << "_h:"
                         << h_L_terms[TF_index].h << "_nsq_min:0_nsq_max:" << nsq_min - 1 << ".dat";
                    read_covarmatrix_file(filename.str().c_str(), last_covarmatrix_results);
                    if (remove(filename.str().c_str()) != 0)
                    {
                        perror("Error deleting file");
                    }

                    filename.str(empty_str);
                    filename << "/PHD_Share/Covarmatrix_Files/covarmatrix-p_" << prefix[prefix_index].str() << "_L:" <<
    h_L_terms[TF_index].L << "_h:"
                         << h_L_terms[TF_index].h << "_nsq_min:0_nsq_max:" << nsq_min - 1 << ".dat";
                    write_covarmatrix_file(filename.str(), last_covarmatrix_results);
                }
            }
    #endif
        for (int local_nsq_max = nsq_min + nsq_inc, local_nsq_min = nsq_min; local_nsq_max <= nsq_max; local_nsq_min +=
```

Page 5

topoterms.cpp

```
nsq_max + 1, local_nsq_max += nsq_inc)
        {
            std::cout <<
        "**********************************************************************************************************" << std::endl;
            std::cout << "*  Doing covarmatrix for topology" << TT_Number << ", ellmax = " << ellmax << ", nsq_max = "
        << local_nsq_max << std::endl;
            std::cout <<
        "**********************************************************************************************************" << std::endl;
            gen_covarmatrix(TT_Number, ellmin, ellmax, local_nsq_min, local_nsq_max); //, current_covarmatrix_results);

            for (int TF_index = 0; TF_index < number_of_h_L_terms; TF_index++)
            {
                for (int prefix_index = 0; prefix_index < 3; prefix_index++)
                {
                    //load the last full file
                    filename.str(empty_str);
                    filename << "/PHD_Share/Covarmatrix_Files/covarmatrix-p_" << prefix[prefix_index].str() << "_L:" <<
        h_L_terms[TF_index].L << "_h:"
                            << h_L_terms[TF_index].h << "_nsq_min:0_nsq_max:" << local_nsq_min - 1 << ".dat";
                    read_covarmatrix_file(filename.str().c_str(), last_covarmatrix_results);
                    if (remove(filename.str().c_str()) != 0)
                    {
                        perror("Error deleting file");
                    }
                    //load the just generated partial file;
                    filename.str(empty_str);
                    filename << "/PHD_Share/SSD/Temp/covarmatrix-p_" << prefix[prefix_index].str() << "_L:" <<
        h_L_terms[TF_index].L << "_h:"
                            << h_L_terms[TF_index].h << "_nsq_min:" << local_nsq_min << "_nsq_max:" << local_nsq_max <<
        ".dat";
                    read_covarmatrix_file(filename.str().c_str(), current_covarmatrix_results);
                    if (remove(filename.str().c_str()) != 0)
```

topoterms.cpp

```
            {
                perror("Error deleting file");
            }
            //add the full file to the partial file and save to next in full file series.
            add_to_covarmatrix_array(current_covarmatrix_results, last_covarmatrix_results);

            filename.str(empty_str);
            filename << "/PHD_Share/Covarmatrix_Files/covarmatrix-p_" << prefix[prefix_index].str() << "_L:" <<
h_L_terms[TF_index].L << "_h:"
                    << h_L_terms[TF_index].h << "_nsq_min:0_nsq_max:" << local_nsq_max << ".dat";
            write_covarmatrix_file(filename.str(), current_covarmatrix_results);

            compare_two_covarmatrix_array(current_covarmatrix_results, last_covarmatrix_results);
        }
    }
    delete current_covarmatrix_results;
    delete last_covarmatrix_results;
}

t = time(0);   // get time now
now = localtime(&t);
std::cout << "\nEnded at " << (now->tm_year + 1900) << '-' << (now->tm_mon + 1) << '-' << now->tm_mday << ' ' <<
now->tm_hour << ':' << now->tm_min
     << "\n\n" << std::endl;

return 0;
}

// Function to generate covariance matrix:
void gen_covarmatrix(int TT_Number, int ellmin, int ellmax, int nsq_min, int nsq_max)
```

topoterms.cpp

```
{
    c_topoterms *topoterms;
    switch (TT_Number)
    {
    case 1:
        topoterms = new c_topoterms(ellmin, ellmax, nsq_min, nsq_max);
        break;
    case 3:
        topoterms = new c_topoterms3(ellmin, ellmax, nsq_min, nsq_max);
        break;
    case 7:
        topoterms = new c_topoterms7(ellmin, ellmax, nsq_min, nsq_max);
        break;
    case 9:
        topoterms = new c_topoterms9(ellmin, ellmax, nsq_min, nsq_max);
        break;
    default:
        topoterms = new c_topoterms(ellmin, ellmax, nsq_min, nsq_max);
        break;
    }

    topoterms->set_h_and_L_list(h_L_terms, number_of_h_L_terms);   // set L and h values to
                                                                   // find covarmatrix for

    topoterms->generate_sh();   // find spherical harmonics needed to
                                // calculate covarmatrix for current topology

#if(0)
    topoterms.generate_sh(FORCE_SAVE_TO_FILE);
#endif

#if(0)
```

Page 8

```
    _Complex double return_value = topoterms.generate_sh(2,0,1,-1,0);
    std::cout << "\nfor ell = 2, m = 0, n1 = 1, n2 = -1, n3 = 0  sh = " << __real__ return_value << " + " << __imag__
return_value << 'i' << std::endl;
#endif

    topoterms->generate_tt();    // *** the topoterms class fn generate_tt originally just computed
                                 //     topoterms but now goes on to find the covariance matrix ***

    delete topoterms;

}

// Function for adding to covariance matrix elements
void add_to_covarmatrix_array(c_topoterms::covamatrix_t* covamatrix_dest, c_topoterms::covamatrix_t* covamatrix_source)
{
    for (int ell = 0; ell <= MAX_ELL; ell++)
    {
        for (int m = 0; m <= (MAX_ELL * 2); m++)
        {
            for (int elldash = 0; elldash <= MAX_ELL; elldash++)
            {
                for (int mdash = 0; mdash <= (MAX_ELL * 2); mdash++)
                {
                    (*covamatrix_dest)[ell][m][elldash][mdash] += (*covamatrix_source)[ell][m][elldash][mdash];
                }
            }
        }
    }
```

topoterms.cpp

```
}

// Function for comparing two arrays (used to check for convergence of covaraince matrix):
bool compare_two_covarmatrix_array(c_topoterms::covamatrix_t* covamatrix_1, c_topoterms::covamatrix_t* covamatrix_2)
{
    double diff;
    double total_diff = 0;
    ;
    int actual_count = 0;
    int possible_count = 0;
    int match_count = 0;
    int no_match_count = 0;
    int both_zero_count = 0;
    int one_zero_count = 0;
    int sign_missmatch_count = 0;
    bool return_val = true;
    double max_diff = 0;
    double min_diff = std::numeric_limits<double>::max();
    const double compareLevel = 0.00001;
    double min_matched_value = std::numeric_limits<double>::max();
    double max_matched_value = std::numeric_limits<double>::min();
    double min_non_matched_value = std::numeric_limits<double>::max();
    double max_non_matched_value = std::numeric_limits<double>::min();

    for (int ell = 2; ell <= MAX_ELL; ell++)
    {
        for (int m = MAX_ELL - ell; m <= (MAX_ELL * 2); m++)
        {
            for (int elldash = 2; elldash <= MAX_ELL; elldash++)
            {
                for (int mdash = MAX_ELL - elldash; mdash <= (MAX_ELL * 2); mdash++)
                {
```

```
                topoterms.cpp

for (int part_id = 0; part_id < 2; part_id++)
{
   possible_count++;

      double covamatrix_1_val;
      double covamatrix_2_val;

      if (part_id == 0)
      {
         covamatrix_1_val = __real__ (*covamatrix_1)[ell][m][elldash][mdash];
         covamatrix_2_val = __real__ (*covamatrix_2)[ell][m][elldash][mdash];
      }
      else
      {
         covamatrix_1_val = __imag__ (*covamatrix_1)[ell][m][elldash][mdash];
         covamatrix_2_val = __imag__ (*covamatrix_2)[ell][m][elldash][mdash];
      }

      if (covamatrix_1_val != 0 && covamatrix_2_val != 0)
      {
         if (fabs(covamatrix_1_val) > fabs(covamatrix_2_val))
         {
            diff = (covamatrix_1_val / covamatrix_2_val) - 1;
         }
         else if (fabs(covamatrix_1_val) < fabs(covamatrix_2_val))
         {
            diff = (covamatrix_2_val / covamatrix_1_val) - 1;
         }
         else
         {
            diff = 0;
         }

                                Page 11
```

topoterms.cpp

```
if (diff >= 0)
{
    if (max_diff < diff)
        max_diff = diff;
    if (min_diff > diff)
        min_diff = diff;

    total_diff += diff;
    actual_count++;

    if (diff < compareLevel)
    {
        //treat as exact match
        match_count++;
        if (min_matched_value > fabs(covamatrix_1_val))
            min_matched_value = fabs(covamatrix_1_val);
        if (max_matched_value < fabs(covamatrix_1_val))
            max_matched_value = fabs(covamatrix_1_val);
        if (min_matched_value > fabs(covamatrix_2_val))
            min_matched_value = fabs(covamatrix_2_val);
        if (max_matched_value < fabs(covamatrix_2_val))
            max_matched_value = fabs(covamatrix_2_val);
    }
    else
    {
        return_val = false;
        no_match_count++;
        if (min_non_matched_value > fabs(covamatrix_1_val))
            min_non_matched_value = fabs(covamatrix_1_val);
        if (max_non_matched_value < fabs(covamatrix_1_val))
            max_non_matched_value = fabs(covamatrix_1_val);
```

Page 12

topoterms.cpp

```
            if (min_non_matched_value > fabs(covamatrix_2_val))
                min_non_matched_value = fabs(covamatrix_2_val);
            if (max_non_matched_value < fabs(covamatrix_2_val))
                max_non_matched_value = fabs(covamatrix_2_val);
        }
    }
    else
    {
        sign_missmatch_count++;
    }
}
else
{
    if (covamatrix_1_val == 0 && covamatrix_2_val == 0)
    {
        both_zero_count++;
    }
    else
    {
        one_zero_count++;
    }
}
}
}
}
}
}
}
std::cout << std::fixed;
std::cout << "-------- compare results --------" << std::endl;
std::cout << "comparerd terms = " << actual_count << ", " << match_count << " < " << compareLevel << " < " <<
no_match_count << ", min fractional error = "
```

topoterms.cpp

```
        << min_diff << ", max fractional error = " << max_diff << ", average fractional error = " << (total_diff /
    actual_count) << std::endl;
        std::cout << "magnitude range for values with fractional error less than " << compareLevel << std::scientific << "
    min:" << min_matched_value << " max:"
             << max_matched_value << std::endl;
        std::cout << std::fixed;
        std::cout << "magnitude range for values with fractional error greater than " << compareLevel << std::scientific <<
    " min:" << min_non_matched_value
             << " max:" << max_non_matched_value << std::endl;
        std::cout << "both terms zero = " << both_zero_count << ", one term zero = " << one_zero_count << std::endl;
        std::cout << "different signs = " << sign_missmatch_count << std::endl;

        return return_val;
    }

    // Function for reading covariance matrix from file:
    bool read_covarmatrix_file(std::string filename, c_topoterms::covamatrix_t *covamatrix_results)
    {
        std::ostringstream Covarmatrixfilename;
        std::ifstream Covarmatrixfile;
        Covarmatrixfilename << filename.c_str();
        if (access(Covarmatrixfilename.str().c_str(), R_OK) != -1)
        {

            Covarmatrixfile.open(Covarmatrixfilename.str().c_str(), std::ios::binary);
            Covarmatrixfile.seekg(0, std::ios::end);
            long long Covarmatrixfile_fsize = Covarmatrixfile.tellg();
            Covarmatrixfile.seekg(0, std::ios::beg);
            //before reading the file check to see it is the size we expect it to be;
            if (Covarmatrixfile_fsize != sizeof(c_topoterms::covamatrix_t))
            {
                std::cout << "Bad file size for " << Covarmatrixfilename.str() << " " << __FILE__ << " line " << __LINE__
```

```
<< "\n";
        return (1);
    }
    //read in the covarmatrix file
    std::cout << "reading covarmatrix file " << Covarmatrixfilename.str() << "\n";
    Covarmatrixfile.read((char*) covamatrix_results, sizeof(c_topoterms::covamatrix_t));
    Covarmatrixfile.close();
  }
  else
  {
     std::cout << "Could not open " << Covarmatrixfilename.str() << " " << __FILE__ << " line " << __LINE__ << "\n";
     return (1);
  }

  return (0);
}
// Function for writing covariance matrix to file:
bool write_covarmatrix_file(std::string filename, c_topoterms::covamatrix_t *covamatrix_data)
{
  if (access(filename.c_str(), R_OK) != -1)
  {
     std::cout << "file " << filename << " already exists " << __FILE__ << " line " << __LINE__ << "\n";
     return (1);
  }
  std::ofstream Covarmatrixfile(filename.c_str());
  std::cout << "writing covarmatrix file " << filename << "\n";
  Covarmatrixfile.write((char*) covamatrix_data, sizeof(c_topoterms::covamatrix_t));
  Covarmatrixfile.close();
  return (0);
}
```

c_topoterms.h

```
 * c_topoterms.h

#ifndef C_TOPOTERMS_H_
#define C_TOPOTERMS_H_

#include <iostream>
#include <fstream>
#include <iomanip>
#include <sstream>
#include <string>
#include <math.h>
#include <vector>
#include <cstdlib>
#include <complex>
#include <gsl/gsl_sf_legendre.h>
#include <time.h>
#include <pthread.h>
#include <string.h>
#include <unistd.h>
#include <sys/sysinfo.h>
#include <vector>
#include <complex>
#include <limits>

// shouldn't need this but the code analyser doen't seem to know about __builtin_cpow
_Complex double __builtin_cpow( _Complex double, _Complex double);

//#define NDEBUG
#include <assert.h>

#define MAX_NUMBER_OF_CORES 4
```

Page 1

c_topoterms.h

```
#define MAX_NMAX 100
#define MAX_ELL 50

#define NO_TRUNCATE

#define FORCE_SAVE_TO_FILE true

class c_topoterms
{

public:
    typedef _Complex double covamatrix_t [MAX_ELL +1 ][(MAX_ELL * 2) +1][MAX_ELL +1 ][(MAX_ELL * 2) +1];

    struct h_L_terms_t
    {
        double h;
        double L;
    };

    void set_h_and_L_list( h_L_terms_t *h_L_terms, int number_of_terms);

    c_topoterms(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value);

    virtual ~c_topoterms();
    int generate_sh(unsigned int ell_min = 2, unsigned int ell_max = MAX_ELL);

    _Complex double  generate_tt()
    {
        return generate_tt(m_ellmin, m_ellmax, m_ellmin, m_ellmax);
    }
    _Complex double  generate_tt(unsigned int ellmin, unsigned int ellmax, unsigned int elldash_min, unsigned int
```

c_topoterms.h

```
elldash_max, int m_test_value = -1, int m_dash_test_value = -1);
    int get_direction_index(int N1, int N2, int N3)
    {
        return m_first_found_direction_n1_n2_n3[N1 + MAX_NMAX][N2 + MAX_NMAX][N3 + MAX_NMAX];
    }

protected:

#ifdef DEBUG
    int number_of_cores = 1;
#else
    int number_of_cores = 4;
#endif

    std::string m_topology_name;

    const double m_Pi;


    const unsigned int m_ellmin;
    const unsigned int m_ellmax;
    const int m_nsq_min;
    const int m_nsq_max;
    int m_nnsq;

    struct covamatrix_data_t
    {

        double h;
        double L;
```

Page 3

c_topoterms.h

```
    double transfer_functions_T[MAX_ELL + 1][(MAX_NMAX * MAX_NMAX) + 1];
    double transfer_functions_E[MAX_ELL + 1][(MAX_NMAX * MAX_NMAX) + 1];
    double transfer_functions_Kpower[(MAX_NMAX * MAX_NMAX) + 1];
    _Complex double TT_covamatrix_for_one_ell_and_its_ms[(MAX_ELL * 2) + 1];
    _Complex double TE_covamatrix_for_one_ell_and_its_ms[(MAX_ELL * 2) + 1];
    _Complex double EE_covamatrix_for_one_ell_and_its_ms[(MAX_ELL * 2) + 1];
    std::ofstream *TT_outfile = NULL;
    std::ofstream *TE_outfile = NULL;
    std::ofstream *EE_outfile = NULL;
};

int m_number_of_h_l_terms = 0;
covamatrix_data_t * mp_covamatrix_data = NULL;

struct direction_data_t
{
    double theta;
    double phi;
    double costheta;
    _Complex double expl;
    int nsq;
    int N1;
    int N2;
    int N3;
};

struct single_direction_t
{
    int n1;
    int n2;
    int n3;
};
```

Page 4

c_topoterms.h

```
struct nsq_and_directions_t
{
    int nsq;
    int ndirecs;
    struct single_direction_t *direction_index_list;
};
enum thread_type_t
{
  TH_GTT, TH_GSH, TH_WTT
};

struct start_thread_data_t
{
    enum thread_type_t threadtype;
    c_topoterms * ctopoterms;
};

struct spherical_task_data_t: start_thread_data_t
{
    int *ell;
    int ell_max;
};

struct tt_file_write_task_data_t: start_thread_data_t
{
    std::ofstream *tt_outfile;
    long long int number_of_topoterms;
    const _Complex double *topoterms_for_one_ell_one_m_one_elldash_all_mdash;
};

struct topoterms_task_data_t: start_thread_data_t
```

Page 5

c_topoterms.h

```
    {
        int *mdash_count;
        int ell;
        int elldash;
        int m;
        int m_dash; //testing only
        int task_number;
        long long int number_of_topotetms;
        _Complex double const *one_ell_one_m_sh_for_unique_directions;
        _Complex double const *one_elldash_all_mdash_sh_for_unique_directions;
        covamatrix_data_t *transfer_functions;
        covamatrix_t *covamatrix;
        int unique_direction_list_count;
    };
public:
    typedef int direction_N1_N2_N3_t[(MAX_NMAX * 2) + 1][(MAX_NMAX * 2) + 1][(MAX_NMAX * 2) + 1];
    typedef direction_data_t unique_directions_list_t[((MAX_NMAX * 2) + 1) * ((MAX_NMAX * 2) + 1) * ((MAX_NMAX * 2) +
1)];
    typedef nsq_and_directions_t nsq_and_directions_list_t[MAX_NMAX * MAX_NMAX * MAX_NMAX];
protected:

    int unique_direction_list_count;
    unique_directions_list_t &unique_directions_list;
    nsq_and_directions_list_t &nsq_and_directions_list;

    direction_N1_N2_N3_t &m_first_found_direction_n1_n2_n3;

    pthread_mutex_t lock_ell;
    pthread_mutex_t lock_mdash_count;
```

Page 6

c_topoterms.h

```
public:
    int get_unique_directions_list(const unique_directions_list_t** unique_directions)
    {   *unique_directions = &unique_directions_list;
        return unique_direction_list_count;
    }
protected:

    virtual void calculate_topoterms(topoterms_task_data_t &tt_data);
    void write_tt_file_thread(tt_file_write_task_data_t &tfw_task_data);
    void calculate_spherical_harmonics(spherical_task_data_t &sph_data);
    void find_ratio(int &n1, int &n2, int &n3);
    int find_nsq_directions(int nsq, struct single_direction_t  ** direction_index_list);
    int find_all_nsq();
    static void * start_thread(void *arg);

};

#endif /* C_TOPOTERMS_H_ */
```

c_topoterms.cpp

```
#include "c_topoterms.h"
using namespace std;

c_topoterms::c_topoterms(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value) :
        m_Pi(acos(-1.0)), m_ellmin(ell_min_value), m_ellmax(ell_max_value), m_nsq_min(nsq_min_value), m_nsq_max
(nsq_max_value),
        //this list will hold all the unique directions and their associated data.
        unique_directions_list(*(unique_directions_list_t *) new unique_directions_list_t),
        //this list will hold all the nsq values and a list of direction indexes for each nsq.  The indexes index into
the
    //unique_directions list to provide the actual direction information.
    nsq_and_directions_list(*(nsq_and_directions_list_t*) new nsq_and_directions_list_t),
m_first_found_direction_n1_n2_n3(
            *(direction_N1_N2_N3_t *) new direction_N1_N2_N3_t)
    {

    m_topology_name = "topology1";
    unique_direction_list_count = 0;
    m_nnsq = find_all_nsq();

}
c_topoterms::~c_topoterms()
{
    delete unique_directions_list;
    delete m_first_found_direction_n1_n2_n3;

    for (int i = 0; i < m_nnsq; i++)
    {
        delete nsq_and_directions_list[i].direction_index_list;
    }
```

```
    if (mp_covamatrix_data != NULL)
    {
        for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
        {
            {
                if (mp_covamatrix_data[TF_index].TT_outfile != NULL)
                    delete mp_covamatrix_data[TF_index].TT_outfile;
                if (mp_covamatrix_data[TF_index].TE_outfile != NULL)
                    delete mp_covamatrix_data[TF_index].TE_outfile;
                if (mp_covamatrix_data[TF_index].EE_outfile != NULL)
                    delete mp_covamatrix_data[TF_index].EE_outfile;
            }
        }

        delete mp_covamatrix_data;
    }
    delete nsq_and_directions_list;
}


//---------------------------------------------------------------------
// Find nsq_list - all possible values of n^2 <= nmax^2
//---------------------------------------------------------------------

int c_topoterms::find_all_nsq()
{
    const int m_nmax=  sqrt(m_nsq_max) ;
    int TotalDirections = 0;
    int nnsq = 0;
    unique_direction_list_count = 0;
```

c_topoterms.cpp

```
//this three dimensional array is a quick way to mark all directions that have already been found with the
//index in the list of unique directions (defined above), where they were first found;.
//set all entries to -1 to indicate they are unused.
memset(m_first_found_direction_n1_n2_n3, 0xff, sizeof(direction_N1_N2_N3_t));

for (int n1 = 0; n1 <= m_nmax; n1++)
{
    for (int n2 = n1; n2 <= m_nmax; n2++)
    {
        for (int n3 = n2; n3 <= m_nmax; n3++)
        {
            int nsq = n1 * n1 + n2 * n2 + n3 * n3;
            if (nsq >= m_nsq_min && nsq <= m_nsq_max && nsq != 0)
            {
                // Check that nsq does not already exist in nsq_list before adding to list
                int jj;
                for (jj = 0; jj < nnsq; jj++)
                {
                    if (nsq == nsq_and_directions_list[jj].nsq)
                    {
                        break;
                    }
                }
                if (jj == nnsq)
                {
                    nsq_and_directions_list[nnsq].nsq = nsq;
                    nsq_and_directions_list[nnsq].ndirecs = find_nsq_directions(nsq, &(nsq_and_directions_list
[nnsq].direction_index_list));
                    TotalDirections += nsq_and_directions_list[nnsq].ndirecs;
                    nnsq++;
                }
            }
```

```
            }
        }
    }
    cout << "number of nsq = " << nnsq << " total Directions = " << TotalDirections << " Unique Directions = " <<
unique_direction_list_count << endl;

    return nnsq;

}

//---------------------------------------------------------------------------
// Function to find directions (n1,n2,n3) for given n where 0<=n1<=n2<=n3
//---------------------------------------------------------------------------

int c_topoterms::find_nsq_directions(int nsq, struct single_direction_t ** direction_index_list)
{
    //for nsq we will add any unique direction found to the 'directions_list' and update the 'direction_list_count'.
    //we will also return a list of indexes into the 'directions_list' to give a complete list of direction for nsq.

    int number_of_directions_indexes = 0;
    //As we don't know the number of directions that will be found we will make this buffer far too big.
    // We will will create a buffer of the correct size before assigning it to the direction_index_list pointer
    struct single_direction_t *temp_direction_index_list = new struct single_direction_t[((((unsigned int)(sqrt
(m_nsq_max) +1)* 2) + 1) * ((unsigned int)(sqrt((m_nsq_max) +1) * 2) + 1)) * 2];

    for (int n1 = -sqrt(nsq); n1 <= sqrt(nsq); n1++)
    {
        for (int n2 = -sqrt(nsq - n1 * n1); n2 <= sqrt(nsq - n1 * n1); n2++)
        {
            int n3 = sqrt(nsq - n1 * n1 - n2 * n2);

            if (n1 * n1 + n2 * n2 + n3 * n3 == nsq)
```

c_topoterms.cpp

```
{
    for (int loop_index = 0; loop_index < 2; loop_index++)
        {
            //add the new direction to the list if direction for the current nsq
            temp_direction_index_list[number_of_directions_indexes].n1 = n1;
            temp_direction_index_list[number_of_directions_indexes].n2 = n2;
            temp_direction_index_list[number_of_directions_indexes++].n3 = n3;

            int N1 = n1;
            int N2 = n2;
            int N3 = n3;

            find_ratio(N1, N2, N3);

            //check to see if the direction is already in the list and if so just register an index to it
            //otherwise add the direction to the list and add register its index.
            if (m_first_found_direction_n1_n2_n3[N1 + MAX_NMAX][N2 + MAX_NMAX][N3 + MAX_NMAX] != -1)
            {
                m_first_found_direction_n1_n2_n3[n1 + MAX_NMAX][n2 + MAX_NMAX][n3 + MAX_NMAX] =
                    m_first_found_direction_n1_n2_n3[N1 + MAX_NMAX][N2 + MAX_NMAX][N3 + MAX_NMAX];
            }
            else
            {
                unique_directions_list[unique_direction_list_count].nsq = nsq;
                unique_directions_list[unique_direction_list_count].N1 = N1;
                unique_directions_list[unique_direction_list_count].N2 = N2;
                unique_directions_list[unique_direction_list_count].N3 = N3;
                unique_directions_list[unique_direction_list_count].theta = atan2(sqrt(n1 * n1 + n2 * n2), n3);
                if (unique_directions_list[unique_direction_list_count].theta == 0 || unique_directions_list
[unique_direction_list_count].theta == m_Pi)
                    {
```

```
                                c_topoterms.cpp

                    unique_directions_list[unique_direction_list_count].phi = 0.0;
                }
                else
                {
                    unique_directions_list[unique_direction_list_count].phi = atan2((double)n2, (double)n1) +
m_Pi;
                }
                unique_directions_list[unique_direction_list_count].costheta = cos(unique_directions_list
[unique_direction_list_count].theta);
                __real__ unique_directions_list[unique_direction_list_count].exp1 = cos(unique_directions_list
[unique_direction_list_count].phi);
                __imag__ unique_directions_list[unique_direction_list_count].exp1 = sin(unique_directions_list
[unique_direction_list_count].phi);

                m_first_found_direction_n1_n2_n3[N1 + MAX_NMAX][N2 + MAX_NMAX][N3 + MAX_NMAX] =
unique_direction_list_count;
                m_first_found_direction_n1_n2_n3[n1 + MAX_NMAX][n2 + MAX_NMAX][n3 + MAX_NMAX] =
unique_direction_list_count++;
            }
            if (n3 == 0)
            {
                break;
            }
            n3 = -n3;
        }
    }
}
```

```
    struct single_direction_t *return_buffer = new struct single_direction_t[number_of_directions_indexes];

    memcpy(return_buffer, temp_direction_index_list, sizeof(struct single_direction_t) * number_of_directions_indexes);

    delete temp_direction_index_list;
    *direction_index_list = return_buffer;
    return number_of_directions_indexes;
}

//-----------------------------------------------------------------------------

// Function to find smallest values of (N1,N2,N3) with same ratio as (n1,n2,n3)
// where 0<=n1<=n2<=n3

void c_topoterms::find_ratio(int &n1, int &n2, int &n3)
{
    // Assign n1, n2, n3 to na, nb, nc where na<=nb<=nc
    int na, nb, nc;
    int abs_n1 = abs(n1);
    int abs_n2 = abs(n2);
    int abs_n3 = abs(n3);

    if (abs_n3 >= abs_n2 && abs_n3 >= abs_n1)
    {
        na = n3;
        if (abs_n2 >= abs_n1)
        {
            nb = n2;
            nc = n1;
        }
        else
        {
```

c_topoterms.cpp

```
            nb = n1;
            nc = n2;
        }
    }
    else if (abs_n3 >= abs_n2)
    {
        na = n1;
        nb = n3;
        nc = n2;
    }
    else if (abs_n3 >= abs_n1)
    {
        na = n2;
        nb = n3;
        nc = n1;
    }
    else
    {
        nc = n3;
        if (abs_n2 >= abs_n1)
        {
            na = n2;
            nb = n1;
        }
        else
        {
            na = n1;
            nb = n2;
        }
    }
// Find highest common factor d of na, nb, nc
```

Page 8

c_topoterms.cpp

```
int d = 99;
int testd;
int abs_na = abs(na);
int abs_nb = abs(nb);
int abs_nc = abs(nc);
ostringstream sph_file_name;
string empty_str;
ofstream outfile;
if (na == 0)
{
    if (nb == 0)
    {
        if (nc == 0)
        {
            d = 1;
        }
        else
        {
            d = abs_nc;
        }
    }
    else
    {
        for (testd = abs_nb; testd >= 1; testd--)
        {
            if (((abs_nb % testd) == 0) && ((abs_nc % testd) == 0))
            {
                d = testd;
                break;
            }
        }
    }
```

Page 9

```
    }
    else
    {
        for (testd = abs_na; testd >= 1; testd--)
        {
            if (((abs_na % testd) == 0) && ((abs_nb % testd) == 0) && ((abs_nc % testd) == 0))
            {
                d = testd;
                break;
            }
        }
    }

    // Find N1, N2, N3
    n1 = n1 / d;
    n2 = n2 / d;
    n3 = n3 / d;

}

//------------------------------------------------------------------------------------------------------------------
------------
//  test to see if all topoterms can be generated if(TT_Number = 1)
//------------------------------------------------------------------------------------------------------------------
------------
_Complex double c_topoterms::generate_tt(unsigned int ellmin, unsigned int ellmax, unsigned int elldash_min, unsigned
int elldash_max, int m_test_value, int m_dash_test_value)
{
    pthread_t thr[number_of_cores];
    ostringstream one_ell_all_m_sh_file_name;
    string empty_str;
    ifstream one_ell_all_m_sh_infile;
```

c_topoterms.cpp

```
    int mdash_start_count = 0;
    long long total_number_of_topoterms = 0;
    ostringstream tt_file_name;
    ofstream tt_outfile;
    topoterms_task_data_t tt_task_data[4];

#if defined( TESTING_TT) || defined(TESTING_CM)
    //when testing topoterm or covarmatrix data we only do it for one value of ell, ell_dash, m and m_dash
    ellmax = ellmin;
    elldash_max = elldash_min;
    if (m_test_value == -1)
    {
        m_test_value = 0;
    }
    if (m_dash_test_value == -1)
    {
        m_dash_test_value = 0;
    }
#else
    //normally when not in test mode we generate all the associated topotems for ell = 0 to MAX_ELL and elldash = 0 to
MAX_ELL.
    //it is possible to specify the ranges for ell and elldash, the ranges for m and m_dash will always be set to
maximum
    //However the size of the covarmatrix output file will always be [MAX_ELL +1 ][(MAX_ELL * 2) +1][MAX_ELL +1 ]
[(MAX_ELL * 2) +1]
    //where the indexes are ell, m, ell_dash and m dash.  Locations that do not lie within the calculated range will be
unspecified
    m_test_value = -1;
    if (mp_covamatrix_data == NULL)
    {
        cout << "no h and L terms set " << __FILE__ << " line " << __LINE__ << "\n";
        return (0 + 0i);
```

c_topoterms.cpp

```
    }
#endif

    if ((ellmin < m_ellmin) || (elldash_min < m_ellmin) || (ellmax > m_ellmax) || (elldash_max > m_ellmax))
    {
        cout << "bad arguments for '_Complex double c_topoterms::generate_tt(int ellmin, int ellmax, int elldash_min,
int elldash_max, int m_test_value)' "
             << endl;
        return (0 + 0i);
    }

    empty_str = "";

    {
        string empty_str;
        ostringstream tffilename;
        ifstream infile;
        double dump;

        for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
        {
#if not defined( TESTING_TT) && not defined(TESTING_CM)
            //check to see if the covarmatrix file already exists

            ostringstream Covarmatrixfilename;

            Covarmatrixfilename.str(empty_str);
            Covarmatrixfilename  << "/PHD_Share/SSD/Temp/covarmatrix-p_" << m_topology_name << "_TT_L:" <<
mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h
                 << "_nsq_min:" << m_nsq_min << "_nsq_max:" << m_nsq_max << ".dat";
            mp_covamatrix_data[TF_index].TT_outfile = new ofstream(Covarmatrixfilename.str().c_str(), ios::binary |
ios::trunc);
```

Page 12

```
        Covarmatrixfilename.str(empty_str);
        Covarmatrixfilename << "/PHD_Share/SSD/Temp/covarmatrix-p_" << m_topology_name << "_TE_L:" <<
mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h
                << "_nsq_min:" << m_nsq_min << "_nsq_max:" << m_nsq_max << ".dat";
        mp_covamatrix_data[TF_index].TE_outfile = new ofstream(Covarmatrixfilename.str().c_str(), ios::binary |
ios::trunc);

        Covarmatrixfilename.str(empty_str);
        Covarmatrixfilename << "/PHD_Share/SSD/Temp/covarmatrix-p_" << m_topology_name << "_EE_L:" <<
mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h
                << "_nsq_min:" << m_nsq_min << "_nsq_max:" << m_nsq_max << ".dat";
        mp_covamatrix_data[TF_index].EE_outfile = new ofstream(Covarmatrixfilename.str().c_str(), ios::binary |
ios::trunc);

#endif

        for (unsigned int transfer_file_ell_number = elldash_min; transfer_file_ell_number <= elldash_max;
transfer_file_ell_number++)
                {
                tffilename.str(empty_str);
                tffilename << "/PHD_Share/SSD/camb_radiation_transferfns/radiation-transferfns_ell:" <<
transfer_file_ell_number << "_nmax:100_L:"
                        << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << ".dat";
                if (access(tffilename.str().c_str(), R_OK) != 0)
                {
                        cout << "can't open " << tffilename.str().c_str() << " " << __FILE__ << " line " << __LINE__ <<
"\n";
                        abort();
                }
                // the transfer function file will contain k and a transfer functions for nsq =1 to nsq = (m_nmax *
m_nmax) in int steps.
```

c_topoterms.cpp

```
        // dump transfer_functions[transfer_file_ell_number] = new double[(MAX_NMAX * MAX_NMAX) + 1];
        cout << "loading transfer functions from file : " << tffilename.str().c_str() << "                \r" <<
flush;
        infile.open(tffilename.str().c_str());
        for (int transfer_function_index = 1; transfer_function_index <= (MAX_NMAX * MAX_NMAX);
transfer_function_index++)
            {
                infile >> dump; //the k value
                if (infile.eof())
                {
                    cout << "end of file reading " << tffilename.str().c_str() << " " << __FILE__ << " line " <<
__LINE__ << "\n";
                    abort();
                }
                infile >> mp_covamatrix_data[TF_index].transfer_functions_T[transfer_file_ell_number]
[transfer_function_index];
                infile >> mp_covamatrix_data[TF_index].transfer_functions_E[transfer_file_ell_number]
[transfer_function_index];
                infile >> dump; //the B value
                double k = 2 * m_Pi * sqrt(transfer_function_index) / (mp_covamatrix_data[TF_index].L *
mp_covamatrix_data[TF_index].h);
                mp_covamatrix_data[TF_index].transfer_functions_Kpower[transfer_function_index] = (pow(k, -3) / pow
(k, 3));
            }
            infile >> dump;
            if (!infile.eof())
            {
                cout << "expected end of file for " << tffilename.str().c_str() << " " << __FILE__ << " line " <<
__LINE__ << "\n";
                abort();
            }
            infile.close();
```

Page 14

```
                                c_topoterms.cpp

        }
    }
    cout << endl;
}

const unsigned int sh_unique_direction_list_buffer_size = unique_direction_list_count * sizeof(_Complex double);
_Complex double *one_elldash_all_mdash_sh_for_unique_directions[MAX_ELL + 2];  // 0 never used
_Complex double *one_ell_one_m_sh_for_unique_directions = (_Complex double *) new char
[sh_unique_direction_list_buffer_size];

for (int i = 0; i < number_of_cores; ++i)
{
    tt_task_data[i].mdash_count = &mdash_start_count;
    tt_task_data[i].number_of_topotetms = 0;
    tt_task_data[i].task_number = i;
    tt_task_data[i].one_ell_one_m_sh_for_unique_directions = one_ell_one_m_sh_for_unique_directions;
    tt_task_data[i].unique_direction_list_count = unique_direction_list_count;
    tt_task_data[i].transfer_functions = mp_covamatrix_data;
    tt_task_data[i].m_dash =m_dash_test_value;
}
unsigned int ell_dash_loop_start_value = elldash_min;
unsigned int ell_dash_loop_end_value = elldash_max;

do
{
    struct sysinfo info;
    sysinfo(&info);
    unsigned long int useable_memory = (info.totalram / 4) * 3;
    unsigned long int allocated_memeory = 0;
    ell_dash_loop_end_value = elldash_max;

                                Page 15
```

c_topoterms.cpp

```
//read as many of the spherical harmonic files into buffers as memory allows;
for (unsigned int ell_file_index = ell_dash_loop_start_value; ell_file_index <= elldash_max; ell_file_index++)
{
    unsigned long int elldash_calculated_file_size = (unsigned long int) (ell_file_index + 1)
                * sh_unique_direction_list_buffer_size;

    if (allocated_memeory + elldash_calculated_file_size > useable_memory)
    {
        ell_dash_loop_end_value = ell_file_index - 1;
        break;
    }
    allocated_memeory += elldash_calculated_file_size;

    one_elldash_all_mdash_sh_for_unique_directions[ell_file_index] = (_Complex double*) new char
[elldash_calculated_file_size];
        unsigned long int elldash_fsize;
        ostringstream one_elldash_all_mdash_sh_file_name;
        ifstream one_elldash_all_mdash_sh_infile;

        one_elldash_all_mdash_sh_file_name.str(empty_str);
        one_elldash_all_mdash_sh_file_name << "/PHD_Share/SSD/sph/sph_" << ell_file_index << "_" << m_nsq_min <<
    "_" << m_nsq_max;
        cout << "loading spherical harmonic from file : " << one_elldash_all_mdash_sh_file_name.str() <<
        "
        \r" << flush;
        one_elldash_all_mdash_sh_file_name.str(one_elldash_all_mdash_sh_file_name.str().c_str());
        one_elldash_all_mdash_sh_infile.open(one_elldash_all_mdash_sh_file_name.str().c_str(), ios::binary);
        one_elldash_all_mdash_sh_infile.seekg(0, ios::end);
        elldash_fsize = one_elldash_all_mdash_sh_infile.tellg();
        one_elldash_all_mdash_sh_infile.seekg(0, ios::beg);
        //before reading the file check to see it is the size we expect it to be;
        if (elldash_fsize != elldash_calculated_file_size)
        {
```

Page 16

c_topoterms.cpp

```
            cout << "\nBad file size for " << one_elldash_all_mdash_sh_file_name.str() << " " << __FILE__ << " line
" << __LINE__ << " expected " << elldash_calculated_file_size << endl;
                abort();
            }
            //read in spherical_harmonic for all of mdash's unique directions
            Complex double * current_file_ptr = one_elldash_all_mdash_sh_for_unique_directions[ell_file_index];
            one_elldash_all_mdash_sh_infile.read((char*) current_file_ptr, elldash_calculated_file_size);
            one_elldash_all_mdash_sh_infile.close();
        }
    cout << "\nStarting topoterms for eldash = " << ell_dash_loop_start_value << " to " << ell_dash_loop_end_value
<< "\n";
    for (unsigned int ell = ellmin; ell <= ellmax; ell++)
        {
        unsigned long int ell_calculated_file_size = (unsigned long int)(ell + 1) *
sh_unique_direction_list_buffer_size;
        unsigned long int ell_fsize;

        //read in from file the list of direction derived spherical harmonics for all values of m associated with
    the current ell
        one_ell_all_m_sh_file_name.str(empty_str);
        one_ell_all_m_sh_file_name << "/PHD_Share/SSD/sph/sph_" << ell << "_" << m_nsq_min << "_" << m_nsq_max;
        one_ell_all_m_sh_file_name.str(one_ell_all_m_sh_file_name.str().c_str());
        one_ell_all_m_sh_infile.open(one_ell_all_m_sh_file_name.str().c_str(), ios::binary);
        one_ell_all_m_sh_infile.seekg(0, ios::end);
        ell_fsize = one_ell_all_m_sh_infile.tellg();
        one_ell_all_m_sh_infile.seekg(0, ios::beg);
        //before reading the file check to see it is the size we expect it to be;
        if (ell_fsize != ell_calculated_file_size)
        {
            cout << "Bad file size for " << one_ell_all_m_sh_file_name.str() << " " << __FILE__ << " line " <<
__LINE__ << " expected " << ell_calculated_file_size << "\n";
```

Page 17

c_topoterms.cpp

```
            abort();
        }
        one_ell_all_m_sh_infile.close();

        //if the value of m_test_value is not -1 the we are testing topoterm for a single ell, single m (i.e.
m_test_value), single elldash
            //and all mdashes.  The values of ellmax and elldash will have been set to ellmin and elldashmin
respectively by the conditional
            //build time switch 'TESTING_TT'

#if defined( TESTING_TT) || defined(TESTING_CM)
            for (int m = m_test_value; m <= m_test_value; m++)
#else
            for (int m = -ell; m <= (signed int)ell; m++)
#endif
            {
                cout << "doing topoterms for ell = " << ell << " and m = " << m << "                \r" << flush;
                one_ell_all_m_sh_infile.open(one_ell_all_m_sh_file_name.str().c_str(), ios::binary);
                //seek to the direction list for m
                one_ell_all_m_sh_infile.seekg(abs(m) * sh_unique_direction_list_buffer_size, ios::beg);
                //read in spherical harmonic for all of m's unique directions
                one_ell_all_m_sh_infile.read((char*) one_ell_one_m_sh_for_unique_directions,
sh_unique_direction_list_buffer_size);
                //Start one or more threads to work through the ell_dash and m_dash combinations to produce the
topoterms.
                one_ell_all_m_sh_infile.close();

                for (unsigned int elldash = ell_dash_loop_start_value; elldash <= ell_dash_loop_end_value; elldash++)
                {
                    {
                        pthread_mutex_init(&lock_mdash_count, NULL);
                        mdash_start_count = (((m - elldash) & 1) == 1) ? (1 - elldash) : (-elldash);
```

c_topoterms.cpp

```
                //added 5/2/2014 so that unused values are zero.
                for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
                    {
                        memset(mp_covamatrix_data[TF_index].TT_covamatrix_for_one_ell_and_its_ms, 0, sizeof
    (mp_covamatrix_data[TF_index].TT_covamatrix_for_one_ell_and_its_ms));
                        memset(mp_covamatrix_data[TF_index].TE_covamatrix_for_one_ell_and_its_ms, 0, sizeof
    (mp_covamatrix_data[TF_index].TE_covamatrix_for_one_ell_and_its_ms));
                        memset(mp_covamatrix_data[TF_index].EE_covamatrix_for_one_ell_and_its_ms, 0, sizeof
    (mp_covamatrix_data[TF_index].EE_covamatrix_for_one_ell_and_its_ms));
                    }

                for (int i = 0; i < number_of_cores; ++i)
                    {
                        tt_task_data[i].one_elldash_all_mdash_sh_for_unique_directions =
    one_elldash_all_mdash_sh_for_unique_directions[elldash];
                                tt_task_data[i].ell = ell;
                                tt_task_data[i].elldash = elldash;
                                tt_task_data[i].m = m;
                                tt_task_data[i].ctopoterms = this;
                                tt_task_data[i].threadtype = TH_GTT;

                                if (pthread_create(&thr[i], NULL, start_thread, &(tt_task_data[i])))
                                {
                                    return EXIT_FAILURE;
                                }
                            }
                        }
                    }

                // block until all threads complete
                for (int i = 0; i < number_of_cores; ++i)
                {
```

c_topoterms.cpp

```
                    pthread_join(thr[i], NULL);
                }

#if not defined( TESTING_TT) && not defined(TESTING_CM)
                int element_d_size = sizeof(_Complex double) * ((MAX_ELL * 2) + 1);
                int element_c_size = (MAX_ELL + 1) * element_d_size;
                int element_b_size = ((MAX_ELL * 2) + 1) * element_c_size;

                int seekpos = (elldash * element_d_size) + ((MAX_ELL + m) * element_c_size) + (ell *
element_b_size);

                for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
                {
                    if (mp_covamatrix_data[TF_index].TT_outfile != NULL)
                    {
                        mp_covamatrix_data[TF_index].TT_outfile->seekp(seekpos, ios::beg);
                        mp_covamatrix_data[TF_index].TT_outfile->write((char*) mp_covamatrix_data
[TF_index].TT_covamatrix_for_one_ell_and_its_ms,
                                element_d_size);
                    }
                    if (mp_covamatrix_data[TF_index].TE_outfile != NULL)
                    {
                        mp_covamatrix_data[TF_index].TE_outfile->seekp(seekpos, ios::beg);
                        mp_covamatrix_data[TF_index].TE_outfile->write((char*) mp_covamatrix_data
[TF_index].TE_covamatrix_for_one_ell_and_its_ms,
                                element_d_size);
                    }
                    if (mp_covamatrix_data[TF_index].EE_outfile != NULL)
                    {
                        mp_covamatrix_data[TF_index].EE_outfile->seekp(seekpos, ios::beg);
                        mp_covamatrix_data[TF_index].EE_outfile->write((char*) mp_covamatrix_data
[TF_index].EE_covamatrix_for_one_ell_and_its_ms,
```


249

```
                c_topoterms.cpp

                    element_d_size);
                }
            }
#endif      }
        }
    }
    for (unsigned int ell_file_index = ell_dash_loop_start_value; ell_file_index <= ell_dash_loop_end_value;
ell_file_index++)
    {
        delete one_elldash_all_mdash_sh_for_unique_directions[ell_file_index];
    }
    ell_dash_loop_start_value = ell_dash_loop_end_value + 1;

} while (ell_dash_loop_start_value <= elldash_max);

    for (int i = 0; i < number_of_cores; ++i)
    {
        total_number_of_topoterms += tt_task_data[i].number_of_topotetms;
    }

#if not defined( TESTING_TT) && not defined(TESTING_CM)
    cout << "\ntotal number of topoterms = " << total_number_of_topoterms << "\n";
#endif

    //give back the memory we borrowed
    delete one_ell_one_m_sh_for_unique_directions;

#if not defined( TESTING_TT) && not defined(TESTING_CM)
    for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)

                    Page 21
```

c_topoterms.cpp

```
    {
      if (mp_covamatrix_data[TF_index].TT_outfile != NULL)
        {
          mp_covamatrix_data[TF_index].TT_outfile->close();
        }
      if (mp_covamatrix_data[TF_index].TE_outfile != NULL)
        {
          mp_covamatrix_data[TF_index].TE_outfile->close();
        }
      if (mp_covamatrix_data[TF_index].EE_outfile != NULL)
        {
          mp_covamatrix_data[TF_index].EE_outfile->close();
        }
    }
#endif
  return (0);
}

//-----------------------------------------------------------------------
void
c_topoterms::calculate_topoterms(topoterms_task_data_t &tt_data)
{
  /* There is a topoterm for each combination of ell, m, elldash, mdash and n which is calculated using the
   * following equation:
   *                                                  ----
   *  i^ell  >    (sph_harm(ell,m,n1,n2,n3) * complex_conjugate(sph_harm(elldash,mdash,n1,n3)))
   *                ----
   *                direcs(n1,n2,n3) for given n    where n = sqrt(n1^2 * n2^2 * n3^2)
   */
  int local_mdash;
```

c_topoterms.cpp

```
const _Complex double I = 0.0 + 1.0i;
_Complex double tt;
_Complex double TTcovar[m_number_of_h_l_terms];
_Complex double TEcovar[m_number_of_h_l_terms];
_Complex double EEcovar[m_number_of_h_l_terms];
double ct;
const _Complex double const_a = (16 * __builtin_cpow(m_Pi, 2));

do
{
  pthread_mutex_lock(&lock_mdash_count);

    local_mdash = *(tt_data.mdash_count);
    (*(tt_data.mdash_count)) += 2;
    //The topoterm for ell=a, m=b, elldash=x, mdash=y  is the the same as the topoterm for ell=x, m=y, elldash=a,
mdash=b.
    //So there's no point in calculating both of them.

  pthread_mutex_unlock(&lock_mdash_count);

  if (local_mdash > tt_data.elldash)
  {
    break;
  }

  memset(TTcovar, 0, sizeof(TTcovar));
  memset(TEcovar, 0, sizeof(TEcovar));
  memset(EEcovar, 0, sizeof(EEcovar));

  for (int nsq_count = 0; nsq_count < m_nnsq; nsq_count++)
  {
    tt = 0 + 0i;
```

c_topoterms.cpp

```
    for (int direcindx = 0; direcindx < nsq_and_directions_list[nsq_count].ndirecs; direcindx++)
      {
        struct single_direction_t single_direction = nsq_and_directions_list[nsq_count].direction_index_list
[direcindx];
        int current_dir = m_first_found_direction_n1_n2_n3[single_direction.n1 + MAX_NMAX][single_direction.n2 +
MAX_NMAX][single_direction.n3 + MAX_NMAX];
        _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
        _Complex double shf2 =
            (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
tt_data.unique_direction_list_count)))[current_dir];

        if (tt_data.m < 0)
        {
          if ((local_mdash & 1) == 0) //i.e even
            {
              __imag__ shf1 = -__imag__ shf1;
            }
          else
            {
              __real__ shf1 = -__real__ shf1;
            }
          }

        if (local_mdash < 0)
        {
          if ((local_mdash & 1) == 0) //i.e even
            {
              __imag__ shf2 = -__imag__ shf2;
            }
          else
            {
              __real__ shf2 = -__real__ shf2;
```

Page 24

```
                                c_topoterms.cpp

        }
    }

    tt += (__builtin_conj(shf1) * shf2);
  }
  tt *= (__builtin_cpow(-I, tt_data.ell) *  __builtin_cpow(I, tt_data.elldash));
#ifdef TESTING_TT
    if (local_mdash == tt_data.m_dash)
    {
      pthread_mutex_lock(&lock_mdash_count);
      cout <<"**************************************************" << endl;
      cout << "ell:" << tt_data.ell << " m:" << tt_data.m << " elldash:" << tt_data.elldash << " mdash:" <<
local_mdash;
      cout << "   nsq = " << nsq_and_directions_list[nsq_count].nsq << "   TT1 = " << __real__ tt << " "<< showpos <<
__imag__ tt << "i"<<endl;
      cout << "**************************************************" << endl;
      pthread_mutex_unlock(&lock_mdash_count);
    }
#endif

#ifndef NO_TRUNCATE
    //make very small numbers zero.
    if (fabs(__imag__ tt) < 0.0000000001)
      __imag__ tt = 0;
    if (fabs(__real__ tt) < 0.0000000001)
      __real__ tt = 0;
#endif
    tt_data.number_of_topotetms++;

    // Find cosmoterm associated with nsq
    int nsq = nsq_and_directions_list[nsq_count].nsq;

                                Page 25
```

c_topoterms.cpp

```
    for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
    {
      ct = tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.ell][nsq]
          * tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.elldash][nsq]
          * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

      TTcovar[TF_index] += ct * tt;

      ct = tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.ell][nsq]
          * tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.elldash][nsq]
          * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

      TEcovar[TF_index] += ct * tt;

      ct = tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.ell][nsq]
          * tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.elldash][nsq]
          * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

      EEcovar[TF_index] += ct * tt;
    }
  }
    for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
    {
#ifndef TESTING_CM
      tt_data.transfer_functions[TF_index].TT_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = TTcovar
[TF_index] * const_a;
      tt_data.transfer_functions[TF_index].TE_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = TEcovar
[TF_index] * const_a;
      tt_data.transfer_functions[TF_index].EE_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = EEcovar
```

c_topoterms.cpp

```
[TF_index] * const_a;

#else
    if (local_mdash == tt_data.m_dash)
        {
          TTcovar[TF_index] *= const_a;
          TEcovar[TF_index] *= const_a;
          EEcovar[TF_index] *= const_a;
          //when testing print out the covarmatrix data
          cout << "TT_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "  " <<
              __real__ TTcovar[TF_index] << " " << showpos << __imag__ TTcovar[TF_index] << noshowpos << endl;
          cout << "TE_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "  " <<
              __real__ TEcovar[TF_index] << " "  << showpos << __imag__ TEcovar[TF_index] << noshowpos << endl;
          cout << "EE_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "  " <<
              __real__ EEcovar[TF_index] << " "  << showpos << __imag__ EEcovar[TF_index] << noshowpos << endl;
        }
#endif
    }
  }
  while (true);
}

int c_topoterms::generate_sh(unsigned int ell_min, unsigned int ell_max)
{
   pthread_t thr[number_of_cores];
   if(ell_max > m_ellmax) ell_max = m_ellmax;
   if(ell_min < m_ellmin) ell_min = m_ellmin;
   int ell_start = ell_min;

   spherical_task_data_t sph_task_data;
   sph_task_data.ell = &ell_start;
```

```
    sph_task_data.ell_max = ell_max;
    sph_task_data.ctopoterms = this;
    sph_task_data.threadtype = TH_GSH;

    pthread_mutex_init(&lock_ell, NULL);

    for (int i = 0; i < number_of_cores; ++i)
    {
        if (pthread_create(&thr[i], NULL, start_thread, &sph_task_data))
        {
            return EXIT_FAILURE;
        }
    }
    // block until all threads complete
    for (int i = 0; i < number_of_cores; ++i)
    {
        pthread_join(thr[i], NULL);
    }

    return 0;
}

//------------------------------------------------------------------------------

void c_topoterms::calculate_spherical_harmonics(spherical_task_data_t &sph_data)
{

    int local_ell;
    _Complex double *buffer;

    ostringstream sph_file_name;
    string empty_str;
```

```
ofstream outfile;
int generate_sh(bool force_save_tt_to_file = false);
do
{
    pthread_mutex_lock(&lock_ell);
    if (*(sph_data.ell) > sph_data.ell_max)
    {
        pthread_mutex_unlock(&lock_ell);
        break;
    }
    local_ell = (*(sph_data.ell))++;

    //write all the spherical harmonics for the current ell , m combination to a file.
    //the name of the file is sph_ell_m_nmax.dat where ell, m and namx are numbers.
    empty_str = "";
    sph_file_name.str(empty_str);
    sph_file_name << "/PHD_Share/SSD/sph/sph_" << local_ell << "_" << m_nsq_min << "_" << m_nsq_max;
    if (access(sph_file_name.str().c_str(), R_OK) != -1)
    {
        //do nothing if the file already exists.
        cout << "found spherical harmonics for  ell = " << local_ell << "                \r" << flush;
        pthread_mutex_unlock(&lock_ell);
        continue;
    }
    cout << "generating spherical harmonics for  ell = " << local_ell << "                \r" << flush;
    pthread_mutex_unlock(&lock_ell);
    buffer = new _Complex double[(local_ell + 1) * unique_direction_list_count];

    int m = 0;
    for (m = 0; m <= local_ell; m++)
    {
        // Loop over all the different directions possible for nsq
```

c_topoterms.cpp

```
            for (int direcindx = 0; direcindx < unique_direction_list_count; direcindx++)
            {
                _Complex double shf = (__builtin_cpow(unique_directions_list[direcindx].exp1, m))
                        * gsl_sf_legendre_sphPlm(local_ell, m, unique_directions_list[direcindx].costheta);
#ifndef NO_TRUNCATE
                //make very small numbers zero.
                if (fabs(__imag__ shf) < 0.0000000001)
                {
                    __imag__ shf = 0;

                }
                if (fabs(__real__ shf) < 0.0000000001)
                {
                    __real__ shf = 0;

                }
#endif
                buffer[(m * unique_direction_list_count) + direcindx] = shf;
            }

        }


        sph_file_name.str(sph_file_name.str().c_str());
        outfile.open(sph_file_name.str().c_str(), ios::binary);
        outfile.write((char *) buffer, (local_ell + 1) * unique_direction_list_count * sizeof(_Complex double));
        outfile.close();

        delete buffer;
    } while (true);

}
```

```
void * c_topoterms::start_thread(void *arg)
{
    start_thread_data_t & start_thread_data = *(start_thread_data_t*) arg;

    switch (start_thread_data.threadtype)
    {
    case TH_GTT:
        start_thread_data.ctopoterms->calculate_topoterms(*(topoterms_task_data_t*) arg);
        break;
    case TH_GSH:
        start_thread_data.ctopoterms->calculate_spherical_harmonics(*(spherical_task_data_t*) arg);
        break;
    default:
        cout << "\n unknown task\n";
        abort();
        break;
    }

    pthread_exit(NULL);
}
//
//-----------------------------------------------------------------------
void c_topoterms::set_h_and_L_list(h_L_terms_t *h_L_terms, int number_of_terms)
{
    if (mp_covamatrix_data != NULL)
    {
        for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
        {
            if (mp_covamatrix_data[TF_index].TT_outfile != NULL)
```

c_topoterms.cpp

```
            delete mp_covamatrix_data[TF_index].TT_outfile;
          if (mp_covamatrix_data[TF_index].TE_outfile != NULL)
            delete mp_covamatrix_data[TF_index].TE_outfile;
          if (mp_covamatrix_data[TF_index].EE_outfile != NULL)
            delete mp_covamatrix_data[TF_index].EE_outfile;
        }
      }
    delete mp_covamatrix_data;
  }
m_number_of_h_l_terms = number_of_terms;
mp_covamatrix_data = new covamatrix_data_t[m_number_of_h_l_terms];

for (int copy_index = 0; copy_index < number_of_terms; copy_index++)
{
    mp_covamatrix_data[copy_index].h = h_L_terms[copy_index].h;
    mp_covamatrix_data[copy_index].L = h_L_terms[copy_index].L;
}

}
```

Page 32

c_topoterms3.h

```
 * ctopoterms3.h

#ifndef CTOPOTERMS3_H_
#define CTOPOTERMS3_H_

#include "c_topoterms.h"

class c_topoterms3: public c_topoterms
{
public:
    c_topoterms3(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value);
    virtual ~c_topoterms3();

protected:
    virtual void calculate_topoterms(topoterms_task_data_t &tt_data);
};

#endif /* CTOPOTERMS3_H_ */
```

Page 1

c_topoterms3.cpp

```
 * ctopoterms3.cpp

#include "c_topoterms3.h"
using namespace std;

c_topoterms3::c_topoterms3(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value) :
      c_topoterms(ell_min_value, ell_max_value, nsq_min_value, nsq_max_value)
{
   m_topology_name = "topology3";

}

c_topoterms3::~c_topoterms3()
{
}

void c_topoterms3::calculate_topoterms(topoterms_task_data_t &tt_data)
{
   /* There is a topoterm for each combination of ell, m, elldash, mdash and n equal to:
    *                                    ----
    *  i^ell  >   (if n1 = 0, n2 = 0, and n3 is even then add (1) to sum else add (2) to sum
    *              -----
    *       direcs(n1,n2,n3) for given n    where distance n = sqrt(n1^2 * n2^2 * n3^2)
    *
    * (1) (sph_harm(ell,m,n1,n2,n3) * cpmplex_conjugate(sph_harm(elldash,mdash,n1.n3))
    * (2) 1/sqrt(2) * (sph_harm(ell,m,n1,n2,n3) + (-1)^n3 * sph_harm(ell,m,n1,-n2,-n3)) *
    *                  (cpmplex_conjugate(sph_harm(ell,m,n1,n2,n3)) + (-1)^n3 * cpmplex_conjugate(sph_harm(ell,m,n1,-
n2,-n3)))
    */
   int local_mdash;
   const _Complex double I = 0.0 + 1.0i;
```

```
                    c_topoterms3.cpp
const _Complex double powers_of_i[4] = { { 1 + 0i }, { 0 + 1i }, { -1 + 0i }, { 0 + -1i } };
const _Complex double powers_of_Neg_i[4] = { { 1 + 0i }, { 0 + -1i }, { 1 + 0i }, { 0 + 1i } };
_Complex double tt;
_Complex double TTcovar[m_number_of_h_l_terms];
_Complex double TEcovar[m_number_of_h_l_terms];
_Complex double EEcovar[m_number_of_h_l_terms];
double ct;
const _Complex double const_a = (16 * __builtin_cpow(m_Pi, 2));

do
{
    pthread_mutex_lock(&lock_mdash_count);

    local_mdash = *(tt_data.mdash_count);
    (*(tt_data.mdash_count)) += 2;

    pthread_mutex_unlock(&lock_mdash_count);

    if (local_mdash > tt_data.elldash)
    {
        break;
    }

    memset(TTcovar, 0, sizeof(TTcovar));
    memset(TEcovar, 0, sizeof(TEcovar));
    memset(EEcovar, 0, sizeof(EEcovar));

    for (int nsq_count = 0; nsq_count < m_nnsq; nsq_count++)
    {
        tt = 0 + 0i;
        for (int direcindx = 0; direcindx < nsq_and_directions_list[nsq_count].ndirecs; direcindx++)
        {
                                    Page 2
```

c_topoterms3.cpp

```
    struct single_direction_t single_direction = nsq_and_directions_list[nsq_count].direction_index_list
[direcindx];
            int n1 = single_direction.n1;
            int n2 = single_direction.n2;
            int n3 = single_direction.n3;

            int current_dir = m_first_found_direction_n1_n2_n3[n1 + MAX_NMAX][n2 + MAX_NMAX][n3 + MAX_NMAX];
            assert(current_dir < tt_data.unique_direction_list_count);

            if (n1 != 0 || n2 != 0 || (n3 % 4) != 0)
            {
                if (n1 > 0 && n2 >= 0)
                {
                    _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
                    _Complex double shf2 = (tt_data.one_elldash_all_mdash_sh_for_unique_directions
                        + ((abs(local_mdash) * tt_data.unique_direction_list_count)))[current_dir];
                    int dir_related_to_current_dir = m_first_found_direction_n1_n2_n3[n2 + MAX_NMAX][-n1 +
MAX_NMAX][-n3 + MAX_NMAX];
                    _Complex double shf3 = tt_data.one_ell_one_m_sh_for_unique_directions
[dir_related_to_current_dir];
                    _Complex double shf4 = (tt_data.one_elldash_all_mdash_sh_for_unique_directions
                        + ((abs(local_mdash) * tt_data.unique_direction_list_count)))
[dir_related_to_current_dir];
                    dir_related_to_current_dir = m_first_found_direction_n1_n2_n3[-n1 + MAX_NMAX][-n2 + MAX_NMAX]
[n3 + MAX_NMAX];
                    _Complex double shf5 = tt_data.one_ell_one_m_sh_for_unique_directions
[dir_related_to_current_dir];
                    _Complex double shf6 = (tt_data.one_elldash_all_mdash_sh_for_unique_directions
                        + ((abs(local_mdash) * tt_data.unique_direction_list_count)))
[dir_related_to_current_dir];
                    dir_related_to_current_dir = m_first_found_direction_n1_n2_n3[-n2 + MAX_NMAX][n1 + MAX_NMAX][-
n3 + MAX_NMAX];
```

Page 3

```
                                                c_topoterms3.cpp
                    __Complex double shf7 = tt_data.one_ell_one_m_sh_for_unique_directions
[dir_related_to_current_dir];
                          __Complex double shf8 = (tt_data.one_elldash_all_mdash_sh_for_unique_directions
                                  + ((abs(local_mdash) * tt_data.unique_direction_list_count)))
[dir_related_to_current_dir];
                        if (tt_data.m < 0)
                        {
                            if ((local_mdash & 1) == 0) //i.e even
                            {
                                __imag__ shf1 = -__imag__ shf1;
                                __imag__ shf3 = -__imag__ shf3;
                                __imag__ shf5 = -__imag__ shf5;
                                __imag__ shf7 = -__imag__ shf7;
                            }
                            else
                            {
                                __real__ shf1 = -__real__ shf1;
                                __real__ shf3 = -__real__ shf3;
                                __real__ shf5 = -__real__ shf5;
                                __real__ shf7 = -__real__ shf7;
                            }
                        }
                        if (local_mdash < 0)
                        {
                            if ((local_mdash & 1) == 0) //i.e even
                            {
                                __imag__ shf2 = -__imag__ shf2;
                                __imag__ shf4 = -__imag__ shf4;
                                __imag__ shf6 = -__imag__ shf6;
                                __imag__ shf8 = -__imag__ shf8;
                                                Page 4
```

c_topoterms3.cpp

```
        }
        else
        {
            __real__ shf2 = -__real__ shf2;
            __real__ shf4 = -__real__ shf4;
            __real__ shf6 = -__real__ shf6;
            __real__ shf8 = -__real__ shf8;
        }
    }

    int n3_Power = n3 & 3;
    int n3X2_Power = (2 * n3) & 3;
    int n3X3_Power = (3 *  n3) & 3;

    tt += (0.25 + 0i)
                * ((__builtin_conj(shf1) + (powers_of_Neg_i[n3_Power] *  __builtin_conj(shf3)) +
(powers_of_Neg_i[n3X2_Power] *  __builtin_conj(shf5)) + (powers_of_Neg_i[n3X3_Power] *  __builtin_conj(shf7)))
                    * (shf2 + (powers_of_i[n3_Power] * shf4) + (powers_of_i[n3X2_Power] * shf6) +
(powers_of_i[n3X3_Power] * shf8)));

            }
        }
        else if (n1 == 0 && n2 == 0 && (n3 & 1) == 0)  // if true do the same as for topoterm 1
        {
            //same as for topoterm 1
            _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
            _Complex double shf2 =
                    (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
tt_data.unique_direction_list_count)))[current_dir];

            if (tt_data.m < 0)
```

c_topoterms3.cpp

```
{
    if ((local_mdash & 1) == 0) //i.e even
    {
        __imag__ shf1 = -__imag__ shf1;
    }
    else
    {
        __real__ shf1 = -__real__ shf1;
    }
}
if (local_mdash < 0)
{
    if ((local_mdash & 1) == 0) //i.e even
    {
        __imag__ shf2 = -__imag__ shf2;
    }
    else
    {
        __real__ shf2 = -__real__ shf2;
    }
}
tt += (__builtin_conj(shf1) *  shf2 );
}
else if (n1 > 0 || (n1 == 0 && n2 > 0))  //if true to the same as for topoterm 2
{
    int Neg1_pow_n3 = ((n3 & 1) == 0) ? (1) : (-1);
    _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
    _Complex double shf2 =
            (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
```

Page 6

c_topoterms3.cpp

```
tt_data.unique_direction_list_count)))[current_dir];

        int dir_related_to_current_dir = m_first_found_direction_n1_n2_n3[n1 + MAX_NMAX][-n2 + MAX_NMAX][-
n3 + MAX_NMAX];
        __Complex double shf3 = tt_data.one_ell_one_m_sh_for_unique_directions[dir_related_to_current_dir];
        __Complex double shf4 =
                (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
tt_data.unique_direction_list_count)))[dir_related_to_current_dir];

        if (tt_data.m < 0)
        {
            //if(m<0){shf1=pow(-1,m)*conj(shf1);}
            if ((local_mdash & 1) == 0) //i.e even
            {
                __imag__ shf1 = -__imag__ shf1;
                __imag__ shf3 = -__imag__ shf3;
            }
            else
            {
                __real__ shf1 = -__real__ shf1;
                __real__ shf3 = -__real__ shf3;
            }
        }
        if (local_mdash < 0)
        {
            //if(mdash<0){shf2=pow(-1,mdash)*conj(shf2);}
            if ((local_mdash & 1) == 0) //i.e even
            {
                __imag__ shf2 = -__imag__ shf2;
                __imag__ shf4 = -__imag__ shf4;
            }
```

Page 7

c_topoterms7.h

```
 * ctopoterms7.h

#ifndef CTOPOTERMS7_H_
#define CTOPOTERMS7_H_

#include "c_topoterms.h"

class c_topoterms7: public c_topoterms
{
public:
    c_topoterms7(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value);
    virtual ~c_topoterms7();

protected:
    virtual void calculate_topoterms(topoterms_task_data_t &tt_data);
};

#endif /* CTOPOTERMS3_H_ */
```

Page 1

c_topoterms7.cpp

```
 * ctopoterms3.cpp

#include "c_topoterms7.h"
using namespace std;

c_topoterms7::c_topoterms7(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value) :
        c_topoterms(ell_min_value, ell_max_value, nsq_min_value, nsq_max_value)
{
    m_topology_name = "topology7";

}

c_topoterms7::~c_topoterms7()
{
}

void c_topoterms7::calculate_topoterms(topoterms_task_data_t &tt_data)
{

    int local_mdash;
    const _Complex double I = 0.0 + 1.0i;
    _Complex double tt;
    _Complex double TTcovar[m_number_of_h_l_terms];
    _Complex double TEcovar[m_number_of_h_l_terms];
    _Complex double EEcovar[m_number_of_h_l_terms];
    double ct;
    const _Complex double const_a = (16 * __builtin_cpow(m_Pi, 2));

    do
    {
        pthread_mutex_lock(&lock_mdash_count);
```

c_topoterms7.cpp

```
local_mdash = *(tt_data.mdash_count);
(*(tt_data.mdash_count)) += 2;

pthread_mutex_unlock(&lock_mdash_count);

if (local_mdash > tt_data.elldash)
{
   break;
}

memset(TTcovar, 0, sizeof(TTcovar));
memset(TEcovar, 0, sizeof(TEcovar));
memset(EEcovar, 0, sizeof(EEcovar));

for (int nsq_count = 0; nsq_count < m_nnsq; nsq_count++)
{
   tt = 0 + 0i;
   for (int direcindx = 0; direcindx < nsq_and_directions_list[nsq_count].ndirecs; direcindx++)
      {
         struct single_direction_t single_direction = nsq_and_directions_list[nsq_count].direction_index_list
[direcindx];

         int n1 = single_direction.n1;
         int n2 = single_direction.n2;
         int n3 = single_direction.n3;

         int current_dir = m_first_found_direction_n1_n2_n3[n1 + MAX_NMAX][n2 + MAX_NMAX][n3 + MAX_NMAX];
         assert(current_dir < tt_data.unique_direction_list_count);

         if (n2 > 0)
         {
            _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
```

c_topoterms7.cpp

```
        _Complex double shf2 =
                (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
tt_data.unique_direction_list_count)))[current_dir];
            if (tt_data.m < 0)
            {
                if ((local_mdash & 1) == 0) //i.e even
                {
                    __imag__ shf1 = -__imag__ shf1;
                }
                else
                {
                    __real__ shf1 = -__real__ shf1;
                }
            }
            if (local_mdash < 0)
            {
                if ((local_mdash & 1) == 0) //i.e even
                {
                    __imag__ shf2 = -__imag__ shf2;
                }
                else
                {
                    __real__ shf2 = -__real__ shf2;
                }
            }
            int Neg1PowerN1PlusN2 = (((n1 + n2) & 1) == 0) ? (1) : (-1);
            tt += (0.5 + 0i) * (__builtin_conj(shf1) + (Neg1PowerN1PlusN2 * __builtin_conj(shf1))) * (shf2 +
(Neg1PowerN1PlusN2 * shf2));
```

Page 3

```
                                                c_topoterms7.cpp
        }
        else if ((n1 & 1) == 0 && n2 == 0)  // if true do the same as for topoterm 1
        {
            //same as for topoterm1
            _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
            _Complex double shf2 =
                    (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
tt_data.unique_direction_list_count)))[current_dir];
                if (tt_data.m < 0)
                {
                    if ((local_mdash & 1) == 0) //i.e even
                    {
                        __imag__ shf1 = -__imag__ shf1;
                    }
                    else
                    {
                        __real__ shf1 = -__real__ shf1;
                    }
                }
                if (local_mdash < 0)
                {
                    if ((local_mdash & 1) == 0) //i.e even
                    {
                        __imag__ shf2 = -__imag__ shf2;
                    }
                    else
                    {
                        __real__ shf2 = -__real__ shf2;
                    }
                }
                                                Page 4
```

c_topoterms7.cpp

```
            tt += (__builtin_conj(shf1) * shf2);
        }
    }

    tt *= (__builtin_cpow(-I, tt_data.ell) * __builtin_cpow(I, tt_data.elldash));

#ifdef TESTING_TT
    if (local_mdash == tt_data.m_dash)
        {
        pthread_mutex_lock(&lock_mdash_count);
        cout << "**************************************************" << endl;
        cout << "ell:" << tt_data.ell << " m:" << tt_data.m << " elldash:" << tt_data.elldash << " mdash:" << local_mdash;
        cout << "   nsq = " << nsq_and_directions_list[nsq_count].nsq << "   TT7 = " << __real__ tt << " " << showpos << __imag__ tt << "i"
             << endl;
        cout << "**************************************************" << endl;
        pthread_mutex_unlock(&lock_mdash_count);
        }
#endif

#ifndef NO_TRUNCATE
    //make very small numbers zero.
    if (fabs(__imag__ tt) < 0.00000000001)
        __imag__ tt = 0;
    if (fabs(__real__ tt) < 0.00000000001)
        __real__ tt = 0;
#endif

    tt_data.number_of_topotetms++;

    // Find cosmoterm associated with nsq
```

```
                c_topoterms7.cpp

        int nsq = nsq_and_directions_list[nsq_count].nsq;

        for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
        {
            ct = tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.ell][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.elldash][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

            TTcovar[TF_index] += ct * tt;

            ct = tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.ell][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.elldash][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

            TEcovar[TF_index] += ct * tt;

            ct = tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.ell][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.elldash][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

            EEcovar[TF_index] += ct * tt;
        }
    }
    for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
    {
#ifndef TESTING_CM
        tt_data.transfer_functions[TF_index].TT_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = TTcovar
[TF_index] * const_a;
        tt_data.transfer_functions[TF_index].TE_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = TEcovar
[TF_index] * const_a;
        tt_data.transfer_functions[TF_index].EE_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = EEcovar

                                    Page 6
```

c_topoterms7.cpp

```
[TF_index] * const_a;
#else
        if (local_mdash == tt_data.m_dash)
            {
                TTcovar[TF_index] *= const_a;
                TEcovar[TF_index] *= const_a;
                EEcovar[TF_index] *= const_a;
                //when testing print out the covarmatrix data
                cout << "TT_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "
                  real__ TTcovar[TF_index] << " " << showpos << __imag__ TTcovar[TF_index] << noshowpos << endl;
                __real__"TE_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "
                  real__ TEcovar[TF_index] << " " << showpos << __imag__ TEcovar[TF_index] << noshowpos << endl;
                cout << "EE_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "
                  __real__ EEcovar[TF_index] << " " << showpos << __imag__ EEcovar[TF_index] << noshowpos << endl;
            }
#endif
    }
    while (true);
}
```

c_topoterms9.h

```
 * ctopoterms9.h

#ifndef CTOPOTERMS9_H_
#define CTOPOTERMS9_H_

#include "c_topoterms.h"

class c_topoterms9: public c_topoterms
{
public:
    c_topoterms9(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value);
    virtual ~c_topoterms9();

protected:
    virtual void calculate_topoterms(topoterms_task_data_t &tt_data);
};

#endif /* CTOPOTERMS8_H_ */
```

Page 1

c_topoterms9.cpp

```
 * ctopoterms3.cpp

#include "c_topoterms9.h"
using namespace std;

c_topoterms9::c_topoterms9(int ell_min_value, int ell_max_value, int nsq_min_value, int nsq_max_value) :
        c_topoterms(ell_min_value, ell_max_value, nsq_min_value, nsq_max_value)
{
    m_topology_name = "topology9";

}

c_topoterms9::~c_topoterms9()
{
}

void c_topoterms9::calculate_topoterms(topoterms_task_data_t &tt_data)
{

    int local_mdash;
    const _Complex double I = 0.0 + 1.0i;
    _Complex double tt;
    _Complex double TTcovar[m_number_of_h_l_terms];
    _Complex double TEcovar[m_number_of_h_l_terms];
    _Complex double EEcovar[m_number_of_h_l_terms];
    double ct;
    const _Complex double const_a = (16 * __builtin_cpow(m_Pi, 2));

    do
    {
        pthread_mutex_lock(&lock_mdash_count);
```

c_topoterms9.cpp

```
        local_mdash = *(tt_data.mdash_count);
        (*(tt_data.mdash_count)) += 2;

        pthread_mutex_unlock(&lock_mdash_count);

        if (local_mdash > tt_data.elldash)
        {
            break;
        }

        memset(TTcovar, 0, sizeof(TTcovar));
        memset(TEcovar, 0, sizeof(TEcovar));
        memset(EEcovar, 0, sizeof(EEcovar));

        for (int nsq_count = 0; nsq_count < m_nnsq; nsq_count++)
        {
            tt = 0 + 0i;
            for (int direcindx = 0; direcindx < nsq_and_directions_list[nsq_count].ndirecs; direcindx++)
            {
                struct single_direction_t single_direction = nsq_and_directions_list[nsq_count].direction_index_list
    [direcindx];

                int n1 = single_direction.n1;
                int n2 = single_direction.n2;
                int n3 = single_direction.n3;

                int current_dir = m_first_found_direction_n1_n2_n3[n1 + MAX_NMAX][n2 + MAX_NMAX][n3 + MAX_NMAX];
                assert(current_dir < tt_data.unique_direction_list_count);

                if (n2 > 0 && ((n1 + n2) & 1) == (n3 & 1))
                {
                    _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
```

c_topoterms9.cpp

```
            _Complex double shf2 =
                    (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
tt_data.unique_direction_list_count)))[current_dir];
                if (tt_data.m < 0)
                {
                    if ((local_mdash & 1) == 0) //i.e even
                    {
                        __imag__ shf1 = -__imag__ shf1;
                    }
                    else
                    {
                        __real__ shf1 = -__real__ shf1;
                    }
                }
                if (local_mdash < 0)
                {
                    if ((local_mdash & 1) == 0) //i.e even
                    {
                        __imag__ shf2 = -__imag__ shf2;
                    }
                    else
                    {
                        __real__ shf2 = -__real__ shf2;
                    }
                }
                int Neg1PowerN1PlusN2 = (((n1 + n2) & 1) == 0) ? (1) : (-1);
                tt += (0.5 + 0i) * (__builtin_conj(shf1) + (Neg1PowerN1PlusN2 * __builtin_conj(shf1))) * (shf2 +
(Neg1PowerN1PlusN2 * shf2));
```

Page 3

```
                                c_topoterms9.cpp
        }
        else if (((n1 | n3) & 1) == 0 && n2 == 0)  // if true do the same as for topoterm 1
        {
            //same as for topoterm1
            _Complex double shf1 = tt_data.one_ell_one_m_sh_for_unique_directions[current_dir];
            _Complex double shf2 =
                    (tt_data.one_elldash_all_mdash_sh_for_unique_directions + ((abs(local_mdash) *
tt_data.unique_direction_list_count)))[current_dir];
                if (tt_data.m < 0)
                {
                    if ((local_mdash & 1) == 0) //i.e even
                    {
                        __imag__ shf1 = -__imag__ shf1;
                    }
                    else
                    {
                        __real__ shf1 = -__real__ shf1;
                    }
                }
                if (local_mdash < 0)
                {
                    if ((local_mdash & 1) == 0) //i.e even
                    {
                        __imag__ shf2 = -__imag__ shf2;
                    }
                    else
                    {
                        __real__ shf2 = -__real__ shf2;
                    }
                }
                                                                Page 4
```

c_topoterms9.cpp

```
            tt += (__builtin_conj(shf1) * shf2);
        }
    }

    tt *= (__builtin_cpow(-I, tt_data.ell) * __builtin_cpow(I, tt_data.elldash));

#ifdef TESTING_TT
    if (local_mdash == tt_data.m_dash)
        {
        pthread_mutex_lock(&lock_mdash_count);
        cout << "**************************************************" << endl;
        cout << "ell:" << tt_data.ell << " m:" << tt_data.m << " elldash:" << tt_data.elldash << " mdash:" << local_mdash;
        cout << "  nsq = " << nsq_and_directions_list[nsq_count].nsq << "   TT7 = " << __real__ tt << " " << showpos << __imag__ tt << "i"
             << endl;
        cout << "**************************************************" << endl;
        pthread_mutex_unlock(&lock_mdash_count);
        }
#endif

#ifndef NO_TRUNCATE
    //make very small numbers zero.
    if (fabs(__imag__ tt) < 0.00000000001)
        __imag__ tt = 0;
    if (fabs(__real__ tt) < 0.00000000001)
        __real__ tt = 0;
#endif

    tt_data.number_of_topotetms++;

    // Find cosmoterm associated with nsq
```

```
                c_topoterms9.cpp

        int nsq = nsq_and_directions_list[nsq_count].nsq;

        for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
        {
            ct = tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.ell][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.elldash][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

            TTcovar[TF_index] += ct * tt;

            ct = tt_data.transfer_functions[TF_index].transfer_functions_T[tt_data.ell][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.elldash][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

            TEcovar[TF_index] += ct * tt;

            ct = tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.ell][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_E[tt_data.elldash][nsq]
                    * tt_data.transfer_functions[TF_index].transfer_functions_Kpower[nsq]; // cosmoterm

            EEcovar[TF_index] += ct * tt;
        }
    }
    for (int TF_index = 0; TF_index < m_number_of_h_l_terms; TF_index++)
    {
#ifndef TESTING_CM
        tt_data.transfer_functions[TF_index].TT_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = TTcovar
[TF_index] * const_a;
        tt_data.transfer_functions[TF_index].TE_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = TEcovar
[TF_index] * const_a;
        tt_data.transfer_functions[TF_index].EE_covamatrix_for_one_ell_and_its_ms[MAX_ELL + local_mdash] = EEcovar

                                    Page 6
```

c_topoterms9.cpp

```
[TF_index] * const_a;
#else
        if (local_mdash == tt_data.m_dash)
            {
                TTcovar[TF_index] *= const_a;
                TEcovar[TF_index] *= const_a;
                EEcovar[TF_index] *= const_a;
                //when testing print out the covarmatrix data
                cout << "TT_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "
                  __real__ TTcovar[TF_index] << " " << showpos << __imag__ TTcovar[TF_index] << noshowpos << endl;
                cout << "TE_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "
                  __real__ TEcovar[TF_index] << " " << showpos << __imag__ TEcovar[TF_index] << noshowpos << endl;
                cout << "EE_L:" << mp_covamatrix_data[TF_index].L << "_h:" << mp_covamatrix_data[TF_index].h << "
                  __real__ EEcovar[TF_index] << " " << showpos << __imag__ EEcovar[TF_index] << noshowpos << endl;
            }
#endif
        }
    }
    while (true);
}
```