Thermal SZ sky maps and cross-correlation with weak lensing surveys - Implications for cosmology and baryonic physics

Pranav S
Supervisor: Prof Subhabrata Majumdar

September 9, 2019

Abstract

The knowledge of the baryonic and non-baryonic distribution of the universe is fundamental in understanding evolution and structure formation in the Universe. However, a large fraction of them cannot be detected directly. One of the major ways of indirectly detecting the baryon distribution, is cross correlation between the thermal Sunyaev Zeldovich effect (tSZ) and Weak Gravitational Lensing. Doing an independent analysis of this cross correlation will also be useful in detecting any systematic errors, if any, in the existing skymaps. We compute the tSZ skymaps using a methodology independent from the Planck collaboration's piplines, and then cross correlate these skymaps with tangential shear to compare with the existing constrains on halo astrophysics and cosmology [1]. This work consists of three parts, a) Independent generation of tSZ skymaps using machine learning. b) Cross-correlating the tSZ skymaps with weak-lensing maps. c) Comparison with theory using Halo models. This work was done partially in collaboration with Prof. Rishi Khatri.

Acknowledgements

I would like to thank my supervisor, Prof Subhabrata Majumdar and Dr Shadab Alam for all the discussions and help regarding the work on Cross-correlations. I would also like to thank Prof Rishi Khatri for his help and support on the generation of tSZ skymaps using machine learning. **thank cluster**

Contents

Chapter 1

Introduction

Though the general processes driving cosmological evolution and large scale structure formation is reasonably well understood, and the model of the universe is agreed upon to be the Λ-CDM model. There exists a disagreement among various probes on the precise values of the parameters contributing to the model. Therefore, It is of importance to test for any systematic biases which might exist among the existing cosmological data.
SZ-Effect and Gravitational Lensing are both powerful probes to explore the baryonic content of the universe. With the existance of hubble tension, there exists a possibility of systematic biases in our data. Cross-correlations provide a powerful tool for us to avoid these systematic biases, since the systematic biases of these two independent probes will not be correlated to each other.
This work consists of 2 parts. a) We use an independent methodology to exact the tSZ maps from Planck's frequency data. This provides us with hints, incase there exists systematic deviations from the planck skymaps. b) We compute cross-correlations between tSZ maps and Weak Lensing maps by various sky surveys.
This thesis is divided as follows, We initially review the relevant physics of the data we would be using (2). After which, We explain our component separation technique which also includes a quick introduction to the various machine learning aspects used in this work (3). We then, compute the cross-correlation with various maps and datasets. (4). We finally present our results and concluding remarks (5, 6)

Chapter 2

Relevant Physics

In this section, We explain the relevant physics, which are related to the physics quantities we are computing.

2.1 Weak Lensing

Weak Lensing is the phenomenon when a gravitational potential distorts the shape of astrophysical sources. We know that according to general relativity, light bends when passing though a gravitational potential. This creates an lens-like effect which distorts the sources in the background. If the lensing is weak enough that, the images are simply distorted instead of forming multiple images, It is known as weak lensing.

We can represent this lensing effect by the distortion tensor (Ψ_{ij}). Starting with the lensing equation $\vec{\beta} = \vec{\theta} - \vec{\alpha}$, where, $\vec{\theta}$ is the true position of the source, $\vec{\beta}$ is the observed position of the source and $\vec{\alpha}$ is the deviation. The deviation $\vec{alpha}$ can be derived as a function of the gravitational potential, Using the geodesic equation.

$$\alpha^i(\vec{\theta}) = \frac{\partial \Phi(\vec{\theta})/c^2}{\partial \theta^i} \tag{2.1}$$

Where, Φ is the projected gravitational potential, Related to the gravitational potential as,

We get,

$$\frac{\partial \beta_i}{\partial \theta_j} = \begin{pmatrix} 1 - \dfrac{\partial \alpha_1}{\partial \theta_1} & \dfrac{\partial \alpha_1}{\partial \theta_2} \\ \dfrac{\partial \alpha_2}{\partial \theta_1} & 1 - \dfrac{\partial \alpha_2}{\partial \theta_2} \end{pmatrix} \tag{2.2}$$

We then define the distortion tensor (Ψ_{ij}) as,

$$\frac{\partial \beta_i}{\partial \theta_j} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \Psi_{ij} \tag{2.3}$$

and then the elements of the distortion tensor[1] are defined as,

$$\Psi_{ij} = \begin{pmatrix} \kappa + \gamma_1 & \gamma_2 \\ \gamma_2 & \kappa - \gamma_1 \end{pmatrix} \tag{2.4}$$

In the weak lensing regime, $\gamma_1 \approx e_1/2$ and $\gamma_2 \approx e_2/2$ where, e_1 and e_2 are the ellipticities of the galaxies [2]

[1]The distortion tensor is symmetric because we can write the deviation $\vec{\alpha}$ as a gradient of the projected gravitational potential

Now, Writing the shear using the terms in equation (2.2), We get,

$$\begin{aligned}\kappa &= \frac{1}{2}\left(\frac{\partial\alpha_1}{\partial\theta_1} + \frac{\partial\alpha_2}{\partial\theta_2}\right) \\ \gamma_1 &= \frac{1}{2}\left(\frac{\partial\alpha_1}{\partial\theta_1} - \frac{\partial\alpha_2}{\partial\theta_2}\right) \\ \gamma_2 &= \frac{\partial\alpha_1}{\partial\theta_2} = \frac{\partial\alpha_2}{\partial\theta_1}\end{aligned} \tag{2.5}$$

Now, Writing these in terms of the projected gravitational potential,

$$\begin{aligned}\kappa &= \frac{1}{2c^2}\left(\frac{\partial^2\Phi}{\partial\theta_1^2} + \frac{\partial^2\Phi}{\partial\theta_1^2}\right) \\ \gamma_1 &= \frac{1}{2c^2}\left(\frac{\partial^2\Phi}{\partial\theta_1^2} - \frac{\partial^2\Phi}{\partial\theta_1^2}\right) \\ \gamma_2 &= \frac{1}{2c^2}\left(\frac{\partial^2\Phi}{\partial\theta_1\partial\theta_2}\right)\end{aligned} \tag{2.6}$$

In order to remove the polar dependence of these terms, We make different combinations of γ_1 and γ_2.

$$\begin{aligned}\gamma_T &= -\gamma_1\cos(2\phi) - \gamma_2\sin(2\phi) \\ \gamma_X &= -\gamma_1\cos(2\phi) + \gamma_2\sin(2\phi)\end{aligned} \tag{2.7}$$

Since, For the case of spherically symmetric mass distributions, γ_x is zero, It serves as a very useful *null test.* Any deviation from zero for γ_x can be considered as systematic error in our calculations.

Since we have no way of knowing in advance, The intrinsic ellipticities of the galaxies to study the effect of shear, We use correlation functions to capture the effect of shear on these galaxies.
The observed ellipticities of the galaxies can be considered as having 2 parts, The Shear part and the Intrinsic Part, in order to remove the contribution from the Intrinsic part we compute correlation functions, so that the intrinsic part which is randomly oriented cancel out.

For example, For the case of Cross Correlation between γ_T and y[2],

$$\begin{aligned}\langle\gamma_T y\rangle &= \langle\gamma_T^{shear} y\rangle + \langle\gamma_T^{intrinsic} y\rangle \\ &= \langle\gamma_T^{shear} y\rangle + \cancelto{0}{\langle\gamma_T^{intrinsic} y\rangle} \\ &= \langle\gamma_T^{shear} y\rangle\end{aligned} \tag{2.8}$$

this way, We can directly compute the effect of weak lensing directly from the ellipticities without worrying about accounting for the intrinsic shapes of galaxies.

2.2 SZ Effect

Compton Scattering is one of the major astrophysical processes which couple matter and radiation. Sunyeav - Zel'dovich effect (SZ Effect) is one such example of Compton Scattering at low energies, which electrons in clusters of galaxies get scattered by the Cosmic Microwave background radiation.

We would specifically be in interested in the *comptonization parameter* y, which is a dimensionless measure of the time spend by the radiation in an electron distribution along a particular line of sight.

$$y(\vec{r}) = \int n_e(\vec{r})\sigma_T dl \frac{k_B T_e(\vec{r})}{m_e c^2} \tag{2.9}$$

Where,

- n_e is the electron concentration

[2]which for our purposes now is some scalar field

- σ_T is the thompson scattering cross section[3].
- T_e is the temperature of the electron cloud.

When we construct skymaps for the SZ effect, We are using the change in intensity to measure this comptonization parameter to give us insights into the distribution of electron clouds in the universe.

Now, to use these maps to compute astrophysical or cosmological parameters, We compute the cross-correlation between the two quantities. Cross-correlation is an incredibly powerful statistical tool since it helps us avoid systematic biases, which might be present in these two quantities individually. Cross-correlating the quantities mean that the systematic biases won't affect our final outcome as the systematic biases from two different experiments could be assumed to be uncorrelated with each other.

[3]since for low energies, We can take the non relativistic approximation

Chapter 3

Foreground Component Separation

One of the ways of removing foregrounds from our data is to have a parametric model for the foregrounds based on known physics and fit the data to these foreground models to get an estimate of the parameters and use them to eliminate the foregrounds in the data. This is the methodology followed by Commander pipeline of the Planck collaboration [3] and LIL Method [4]. These methods suffer from being model dependent and the parameters are insufficient to model the foregrounds because various physical processes contribute to the foreground in a single pixel, making it hard to model these with a single parametric model. Therefore, it is useful to look at algorithms which are model independent and just use the data to estimate the foreground characteristics.
Various such *blind* component separation algorithms have been proposed where only the spectrum of the signal is needed [5, 6] and have been applied in various variations such as Harmonic space, Needlet frame etc [7, 8]. Since these blind algorithms require the number of foreground components to be lesser than the number of spectral channels available, it is necessary to divide the sky with similar foreground properties into different regions and apply the algorithm separately on those regions to reduce the number of foreground components. These existing variations try to divide the data into different clusters with similar foregrounds based on heuristic arguments and our current understanding of the properties of the foregrounds. We provide a spectral data based approach extending upon this data driven foreground clustering approach [9], which uses the signature of the foregrounds available in the data. Since the data is clustered only based on the spectra, different regions of the sky can still be in the same cluster based on their foreground properties. To the best of our knowledge, this is the first attempt at using *unsupervised* machine learning for foreground component separation.

3.1 Generalized ILC

We want to separate out the cosmological signal of interest from the different frequency maps provided by the Planck collaboration, by subtracting out the different foreground signals. The Generalized ILC is used to separate out any signal with a known spectrum.
The observed temperature data, in different pixels(positions) and frequency channels can be written as,

$$T(\nu, p) = S(\nu)A(p) + F(\nu, p) + N(\nu, p) \tag{3.1}$$

Where,

- $S(\nu)$ is the spectrum of the signal we are looking for. This is essentially the unit conversion which let's us perform ILC to extract any signal of a known spectrum. For eg, In the case of CMB, in K_{cmb} units, it is equal to one. ie, The CMB Temperature is independent of the frequency and is only a function of the pixel p.
- $A(p)$ is the position dependent signal (sky-map) we are looking for.
- $F(\nu, p)$ is the sum of all the foreground components in a particular pixel. Since, different foregrounds with different spectra contribute in each pixel, this is dependent on both the frequency and the pixel position in the sky.

- $N(\nu, p)$ is the noise which is in general dependent on both the frequency channel and the pixel position.

Now, Let us say that the signal is a linear combination of the frequency maps.

$$s_{sig}(p) = \sum_{\nu} w(\nu)T(\nu, p) \tag{3.2}$$

knowing that $s_{sig}(p) = A(p)$, in the absence of foregrounds and noise, We see that,

$$\sum_{\nu} w(\nu)S(\nu) = 1 \tag{3.3}$$

If we knew the true signal $s_{true}(p)$, We could find the weights $(w(\nu))$ assigned to the different frequencies by minimising a cost function,

$$C = \sum_{p} (s_{true}(p) - s_{sig}(p))^2 \tag{3.4}$$

$$C = \sum_{p}\sum_{\nu} (F(\nu, p) + N(\nu, p)) \tag{3.5}$$

Since, We don't know the true value of the signal, Let us consider an alternate cost function

$$C' = \sum_{p} (s_{sig}(p))^2 \tag{3.6}$$

$$C' = \sum_{p}(s_{true}(p))^2 + C - \sum_{p} s_{true}(p) \sum_{\nu} (F(\nu, p) + N(\nu, p)) \tag{3.7}$$

Since the first term is a additive constant, which is irrelevant to the minimization problem. We see that $C = C'$ if, the third term is zero (ie, The foregrounds and Noise are not correlated with the signal). Though the Noise is in general not correlated with the signal, We can't say the same about the foregrounds. This leads to a bias, where the ILC also will remove the part of the signal correlated with the foregrounds. This results in a solution which has lesser power than the actual signal. [9]

It has also been previously studied that, this ILC Bias is significant only for $l = 2$, and is negligible for all the higher multipoles. It can however, start affecting the higher multipoles if we increase the number of partitions to a high number. [9]

By minimising the cost, with respect to the constraint (3.3), We get the following equations for the weight.

$$w(i) = \frac{\sum_{j} D^{-1}_{ij}}{\sum_{ij} D^{-1}_{ij}} \tag{3.8}$$

where,

$$D_{ij} \equiv \sum_{p} T(i, p)T(j, p) \tag{3.9}$$

and where the sum over the pixels is done for pixels in a single cluster.

3.1.1 Combining information from maps with different resolutions

Different resolution maps, has different information. In order to effectively use out algorithm, We need to find a way to combine these information effectively. [9] Since the noise properties depend on the resolution of the map used, We need to rebeam all the maps to a common resolution. But, It is not straightforward to see which resolution map is useful for performing the ILC. Rebeaming all the maps to a lower resolution means, We loose all the large ℓ signals. Whereas, rebeaming all the maps to a higher resolution means, the noise in the lower resolution gets boosted significantly, and these noise dominated maps will contribute less to the final solution, effectively loosing any information contained in these maps.

We would effectively like a technique, where all the channels with sufficient signal to noise ratio, provide information. We first rebeam all the maps to gaussian beams with different resolutions. We then adopt the following technique.

- Rebeam all maps to lowest resolution map and apply FC-ILC algorithm. This solution will have the lowest foregrounds at that resolution since it uses the information from the maximum number of channels. Let's label the ILC solutions by index a, i.e. $s^a(p)$, with $a = 1$ corresponding to the lowest resolution, and $a > b$ implying resolution of map a is higher than map b. We will denote the corresponding (Gaussian) beams with b^a_ℓ.
- We, leave the low resolution channels and perform ILC after rebeaming to the next higher resolution.
- Repeat the above step until not enough frequency channels remain to get a viable ILC solution
- combine the solution in harmonic space, to get the final solution $s^f(p)$, or in harmonic space $a^f_{\ell m}$, with the resolution corresponding to the highest resolution channel with beam b^n_ℓ, as follows:

$$a^f_{\ell m} = b^n_\ell \left[a^1_{\ell m} + (1 - b^1_\ell) \left(a^2_{\ell m} + (1 - b^2_\ell) \left(a^3_{\ell m} + ... + \left(1 - b^{n-1}_\ell\right) \frac{a^n_{\ell m}}{b^n_\ell} \right) \right) \right] . \tag{3.10}$$

  In the last term in nested brackets in the above equation, we correct the highest resolution solution with its beam and then use the resulting $a_{\ell m}$ to fill in only the information not present in the next lower resolution map, hence the factor of $(1 - b^{n-1}_\ell)$. We keep repeating this process to fill-in the information missing from the previous iterations.

3.2 Quick Introduction to Machine Learning

Machine learning is a type of an algorithm which, rather than explicitly coding an algorithm for a particular function/behaviour, gives the computer the ability to *learn* that function. Machine learning involves a training set, from which the computer *learns* the function of interest, and a validation set where the computer applies whatever it has learned from the training set and gives us results. Whenever, We say that the computer *learns*, we mean a minimization of some cost function, by varying the parameters of the function the computer is trying to learn. This cost function captures the deviation from the desired behavior of the algorithm after learning.

We would be particularly interested in a classifier, which can be defined as a function ($f : \mathcal{D} \to [0, 1]^n$) which divides the input data (from the domain $\mathcal{D}$ which is usually $\mathbb{R}^m$) into n different classes, by assigning a probability for each of the classes.

We would also be only implementing hard clustering, where the final result is a single cluster value instead of a set of probabilities for each cluster. This is implemented by assigning the value of the cluster to be the one with maximum probability.

In our use case, We will be dividing the sky into different classes based on their foreground properties, while using the different frequencies as input.

Based on our training set, machine learning can be divided into two categories.

- Supervised Machine Learning
- Unsupervised Machine Learning

3.2.1 Supervised Machine Learning

A machine learning algorithm is said to perform supervised learning, when the training set is different from the validation set, and we have the expected learning outcomes as part of our training set. These labeled training sets consist of a input and an expected output value as part of the examples, and the algorithm learns the function which relates the input value to the output value based on these training examples. In this case, the cost function is some sort of deviation from the output value compared to the given output value in the training examples.

There have been attempts [10] to classify the foregrounds in the data using this approach by using simulations to generate the training set. This method involves using the simulations to generate the training data for different classes based on the different foreground models used in the simulation. We then expect the machine learning algorithm to *learn* from these simulations and help us classify the real sky into regions based on it's foreground properties.

This method suffers from being model dependent, similar to the parametric models of component separation as the algorithm tends to learn only based on the foreground models fed during the simulation. In order to avoid this, We use unsupervised machine learning which gives us the advantage of clustering the data in a model independent manner.

3.2.2 Unsupervised Machine Learning

Unsupervised learning is when a machine learning algorithm has no prior information about the data set in terms of a training set and the data learns and validates from the same set. In this method only the raw data is given to the algorithm, and it tries to *learn* patterns inherent in the data, without any labels to learn from.

This would be really useful for our purposes since it allows us to partition the sky based on the frequency without any model dependent inputs.
We would specifically be using clustering algorithms and dimensionality reduction algorithms.

Clustering Algorithms Partition the data into different classes based on how every data point is located with respect to it's neighbours. These algorithms have a varying point in the *data space*[1] which is representative of the cluster and assign the cluster to the available data based on it's relationship to this point.

Dimensionality Reduction Clustering algorithms in higher dimensions tend to be heavily dependent on the initial seed for the clusters. It is therefore, incredibly useful to reduce the dimentionality of the data by summarizing the information in fewer number of variables and then performing the clustering in a lower dimensional space.

3.2.3 Neural Networks

A neural network is a special case of a machine learning algorithm, which has a specific functional form. The advantage a neural network provides is based on a mathematical theorem that this specific functional form can approximate any continuous function [11].
A single layer neural network is defined by the following function.

$$\vec{c} = \sigma(M_1\vec{v} + \vec{b_1}) \tag{3.11}$$

Where,

- $\vec{v}$is the Input Data with the dimensionality of our Input variables space
- $\vec{c}$ is the Output data with the dimensionality of our number of classes
- $\vec{b}$ is the bias vector with the dimensionality of our number of classes,
- σ is a sigmoidal[2] function which acts component wise. Also known as the activation function
- M is a matrix whose elements we will tune as parameters

We can now, tune the parameters (the elements of the matrix M) based on a cost function, which after training will help us get the final matrix, which we can use to classify our data. A multi-layer neural network can be constructed by composition of this, an arbitary number of times. We use ReLU ($\max(0, x)$) as our activation function.

[1]The space of all possible data

[2]A sigmoid function is a bounded, differentiable, real function that is defined for all real input values and has a non-negative derivative at each point.

3.3 New tSZ maps with unsupervised Machine Learning

In unsupervised machine learning, we take the hard clustering approach, where each data belongs only to a single class. In order to smooth the boundaries of the partitions like the current ILC algorithms, we repeat the hard clustering multiple times with different seeds. Since various seeds produce different partitions, and there is no a-priori reason to consider one partition over the other, we consider all of them with equal probability. We see that this approach essentially smoothens the boundary and no additional smoothing across the cluster boundaries are necessary.

3.3.1 'k-means clustering' method

k-means clustering partitions n data points into k clusters, by associating each point to the nearest centroid, which serves as a representation of that cluster. By minimising the inertia, We partition the space into k-regions, which serve as the partition for our clusters to be used in ILC.

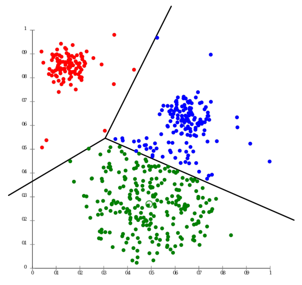

Figure 3.1: K-Means Clustering Example

The initial positions are initialized using `k-means++` algorithm in order to avoid the local minimas, which the random initializations can get into. And the data is rescaled with quartile scaling, to prevent the outliers from influencing the scaling. k-means clustering was implemented using `scikit-learn` [12]

Choice of Variables for k-means clustering

Let the frequency maps in K_{CMB} units be denoted by, T_i, where $i \in \{30, 44, 70, 100, 143, 217, 353, 545, 857\}$Hz. In order to model the foregrounds, We first subtract the CMB. $A_i = T_i - T_{100Hz}$. Then we subtract out the tSZ, by changing to the corresponding y units. The maps we get are mostly dominated by foregrounds. Apart from clustering these foreground maps separately (labeled as *raw k-means*), we divide out the amplitudes of these *raw maps* to make different measures, which capture how fast the foreground is increasing or decreasing with frequency similar to the measure defined in the FC-ILC (labelled as *kmeans with m*) [9].

$$\begin{aligned} y_1 = A_{857} - 24.371A_{143} \quad y_2 = A_{857} - 7.199A_{217} \quad y_3 = A_{545} - 14.836A_{143} \\ y_4 = A_{545} - 4.3826A_{217} \quad y_5 = A_{353} - 8.215A_{143} \quad y_6 = A_{353} - 2.4267A_{217} \\ y_7 = A_{30} - 1.7339A_{70} \quad y_8 = A_{44} - 1.541A_{70} \quad y_9 = A_{217} + 3.671A_{44} \\ y_{10} = A_{217} + 5.657A_{70} \quad y_{11} = A_{353} + 13.73A_{70} \quad y_{12} = A_{30} - 1.125A_{44} \end{aligned} \tag{3.12}$$

Apart from clustering these y-maps separately. We divide out the amplitudes to make different measures, which capture how fast the foreground is increasing or decreasing with frequency.

$$\begin{aligned} m_1 = y_1/y_3 \quad m_2 = y_2/y_4 \quad m_3 = y_3/y_5 \\ m_4 = y_4/y_6 \quad m_5 = y_7/y_{12} \quad m_6 = y_{12}/y_5 \\ m_7 = y_7/y_5 \quad m_8 = y_7/y_8 \quad m_9 = y_8/y_5 \end{aligned} \tag{3.13}$$

By visually inspecting the maps, We choose the 3 measures. m_1, m_3, m_7, and perform k-means clustering (Appendix:A) We also masked the galactic center along with the point sources and clustered the masked regions separately.

3.3.2 Preprocessing

Before we do machine learning, We need to pre-process the data to remove outliers and heavily contaminated regions. For this, We first mask the sky and seperate the masked and the un-masked regions. We then

perform machine learning on these two regions seperately. Since all our cost functions are metric based, We need to scale all the different frequency maps into the same range. This makes sure that all the information in the different frequency maps are used to the same extent.
While there are many ways to perform this scaling, We specifically use quartile scaling, which is robust to the presence of outliers. Quartile scaling, scales using the percentile values. In our case, We scale the region between 15-percentile and 85-percentile to a range of 0 to 1. This makes sure that all the frequency maps are prioritized equally.

Masks

In order to mask the regions which will be dominated by foregrounds, We create masks with different skyfractions. The use the Linearized Iterative Least squares method [4] to fit the spectra to a foreground model and mask the region based on the χ^2 value. [13]
Along side this mask, For all the algorithms, We also mask the outliers (Top and Bottom 1-percentile) in the m_1, m_3, m_7 maps[3].

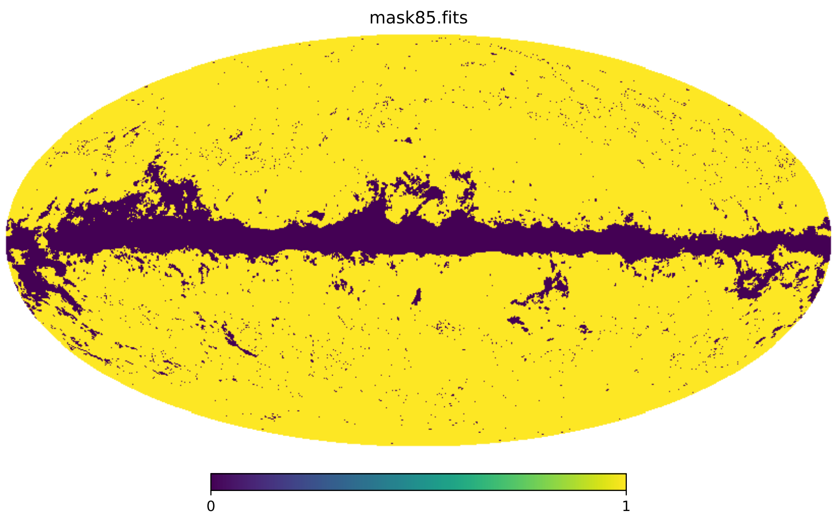

Figure 3.2: The final Mask

While, Calculating the power spectrum, We use a stricter condition on the masks, revealing a lower fraction of the sky. We apodise the masks by replacing the 1s in the mask by $1 - \exp(-9\theta^2/(2\theta_{ap}^2))$ for $\theta < \theta_{ap} = 30'$, the apodization angle, where θ is the distance from the nearest masked pixel.

3.3.3 Dimensionaliy Reduction

Since, the choice of parameters is up to us in the previous method, we use non-linear dimensionality reduction algorithms in order to reduce the 12 dimensional input data (CMB and tSZ subtracted maps) to a lower dimension in order to apply the k-means algorithm on it. Unlike the previous method, we use neural networks to determine the parameters which encode the maximum information about the data. We use two non-linear dimensionality reduction algorithms.
The two algorithms used are,

- Auto Encoders
- Self Organising Maps

[3]This is done in order to mask radio sources and other point sources

3.3.4 Auto Encoders

Auto Encoder is a neural network where, the output dimensionality is the same as input dimentionality. With a hidden layer having a smaller dimension. We train the network by use the same data for both input and output. The idea is to make the network learn to reduce the dimentionality to a lower dimension and reconstruct the input data from that (known as encoder and decoder respectively).
We use a denoising auto encoder, made from an fully connected neural network, using L^2-norm as the cost. There are two intermediate layers for both the encoder and the decoder containing seven and three neurons each.
Once the network has been trained we can use the encoder part for dimensionality reduction. And then the k-means algorithm is used in this lower dimensional space to cluster the pixels. Auto encoders were implemented using both `tensorflow` and `pytorch` to test out different architectures [14, 15].

We vary the learning rate of the neural network by decaying the learning rate, exponentially. The rate of decay is also considered a hyperparameter. The hyperparameters were tuned by performing a random search trying to minimise the cost as much as possible.

3.3.5 Self Organising Maps

Self organising map reduces the input data to two dimensions by fitting a discrete two dimensional manifold on the data and then use k-means algorithm on this manifold. Self Organising Maps was implemented using the `somoclu` Library [16]. We used a 200x200 grid for testing the performance on FFP6 data, and a 300x300 grid for the clustering using the actual Planck Data.

3.3.6 Results

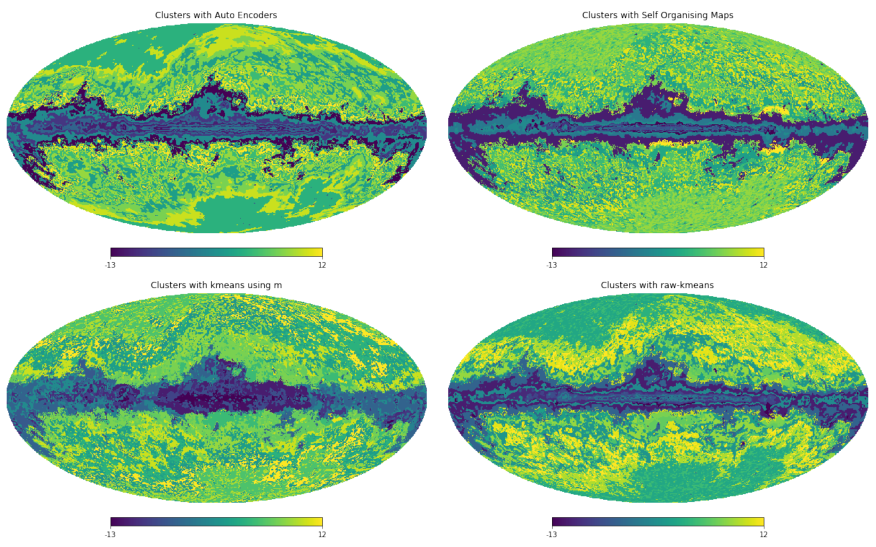
Clusters with Auto Encoders
-13
12
Clusters with Self Organising Maps
-13
12
Clusters with kmeans using m
-13
12
Clusters with raw-kmeans
-13
12

Figure 3.3: A single instance Clustering of the Sky based on foregrounds using various methods (negative values indicate masked regions)

Chapter 4

Thermal-SZ Weak Lensing Cross Correlation

4.1 Calculation from Data

Once the tSZ maps are found using the above method, we cross-correlate the the tangential shear component using the Weak-Lensing datasets available. We compute the cross correlation using the k-D trees algorithm. k-D trees perform efficiently in nearest neighbour searches and help speedup calculating the correlation functions significantly. We use the `treecorr` library for this [17]. We perform the cross correlation using the Red Cluster Sequence Lensing Survey (RCSLens) dataset [18] and the Planck tSZ skymaps [19], initially, to verify our cross correlations by comparing the results with [1]. We repeat the same with the Kilo Degree Survey (KiDS) dataset [20] and our own tSZ skymaps. Following this, we will compare with different theoretical predictions will help us constrain the cosmology and baryon models. Especially, We are interested in looking at non-gravitational feedback.

For $y - \gamma_T$, we compute the two point correlation function as,

$$\xi^{y-\gamma_T}(\theta) = \frac{\sum\limits_{ij} y^i e_t^{ij} w^j \Delta_{ij}(\theta)}{\sum\limits_{ij} w^j \Delta_{ij}(\theta)} \tag{4.1}$$

where, y^i is the y-value from the tSZ maps in pixel i. And e^{ij} is the tangential ellipticity of the galaxy j in the catalogue with respect to pixel i. The tangential ellipticity is corrected for both multiplicative and additive bias. $\Delta_{ij}(\theta)$ imposes our binning scheme. It is one if the angular seperation between i and j is θ and zero otherwise.

4.2 k-D Trees Algorithm

k-D trees is a space partitioning algorithm which stores the coordinates in a tree data structure, the algorithm constructs the tree based on the proximity of the points without explicitly computing the distances between the points. Finding the nearest neighbours then becomes merely the act of traversing the trees. This immensely speeds up the computation of correlation functions.

4.3 Systematic Tests

In order to test for systematics in the data, We perform 2 systematic tests.

- We calculate $\langle y\gamma_x \rangle$
- We randomise the catalogue and find the correlation once again.

in the absence of systematic errors, in our calculation. Both of these tests should give us values close to zero.

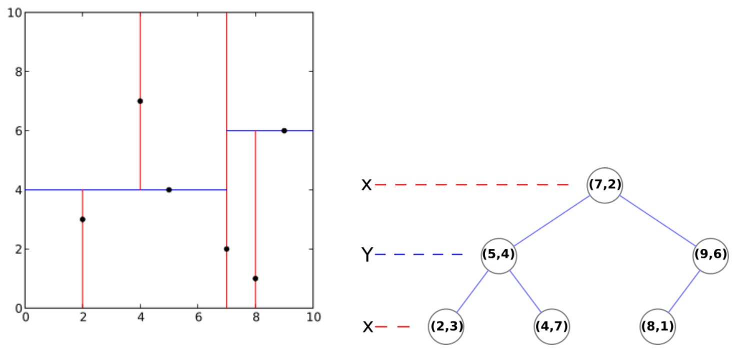

Figure 4.1: Example k-D Trees partitioning and data structure

4.4 Comparison with Theory

We like to compare our observations with theoretical predicitions based on Halo models, closely following the method developed by [21]. Currently we are concentrating on computing the real space cross correlation $\xi = \langle \gamma_T - y \rangle$, By using

$$\xi^{y-\gamma_T}(\theta) = \int \frac{d^2\vec{l}}{2\pi^2} C_l^{y-k} \cos(2(\phi-\psi)) \exp(i\theta\cos(\phi-\psi)) \tag{4.2}$$

Where, ϕ is the Polar angle with respect to the coordinate system and ψ is the angle between $\vec{l}$ and the coordinate.

In order to compute the $y-k$ cross correlation power spectra found in Equation 4.2, we use the 1-halo term as defined in [22].

$$C_l^{y-k,1h} = \int_0^{z_{max}} dz \frac{dV}{dz d\Omega} \int_{M_{min}}^{M_{max}} dM \frac{dn}{dM} y_l(M,z) k_l(M,z) \tag{4.3}$$

For the halo mass function, We use the form suggested by Sheth and Tormen [23] For the convergence profile in fourier space, We use

$$k_l = \frac{W^k(z)}{\xi^2(z)} \frac{1}{\rho_m} 4\pi \int_0^{r_{vir}} dr r^2 \frac{\sin(lr/\xi)}{lr/\xi} \rho(r;M,z) \tag{4.4}$$

And for the fourier transform of the projected gas pressure.

$$y_l = \frac{4\pi r_s}{l_s^2} \frac{\sigma_T}{m_e c^2} \int dx x^2 \frac{\sin(lx/\xi)}{lx/\xi} P_e(x;M,z) \tag{4.5}$$

For electron pressure, We use the *universal pressure profile* and the NFW model. We plan to use the best fit parameters for the Pressure profile from the Planck Collaboration and then compare the halo model predictions from both the Planck and WMAP-7yr cosmologies. To look for any non-gravitational feedback, we plan to use the method developed by [24], applied to tSZ C_ls.

Chapter 5

Main results

The result of using the various algoritms FFP6 simulations is shown in Fig(5). We see that our clustering algorithm is better than the one dimensional version [9]. We hope to compare it with the existing algorithms using FFP6 simulations. We also see that the neural network method like self organising map with a 200x200 grid performs, very close to the k-means algorithm where the measures are chosen by hand. It is interesting to note that *raw-kmeans* performs the same as a self organising map.

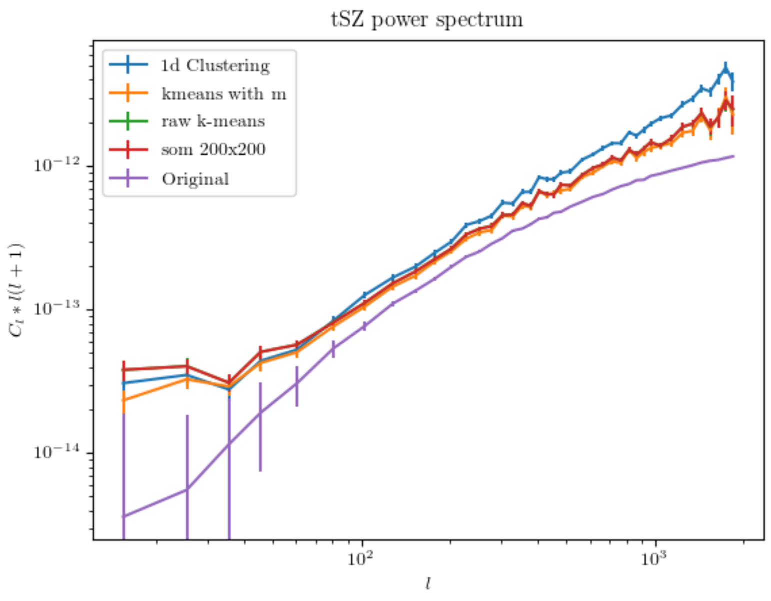

Figure 5.1: Comparison of the tSZ power spectrum for various methods

Seeing that the k-means algorithm using the measures, performs the best out of the various algorithms. We use that on the frequency maps available on the planck data.

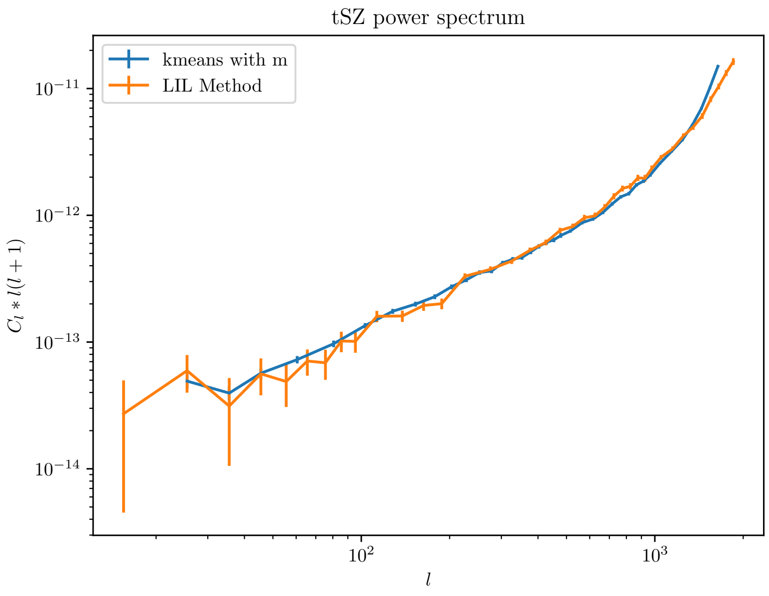

Figure 5.2: Comparison of the tSZ power spectrum of Planck Data using various methods

Now, Computing the cross-correlation using the Planck SkyMaps and our own sky maps, We get the following results.

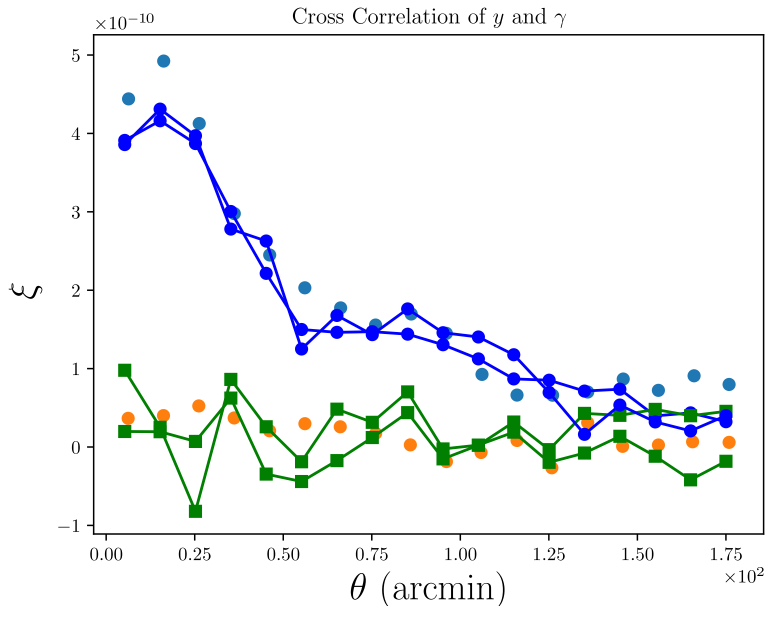

Figure 5.3: Cross Correlation between tSZ and Weak Lensing maps for RCSLens fields

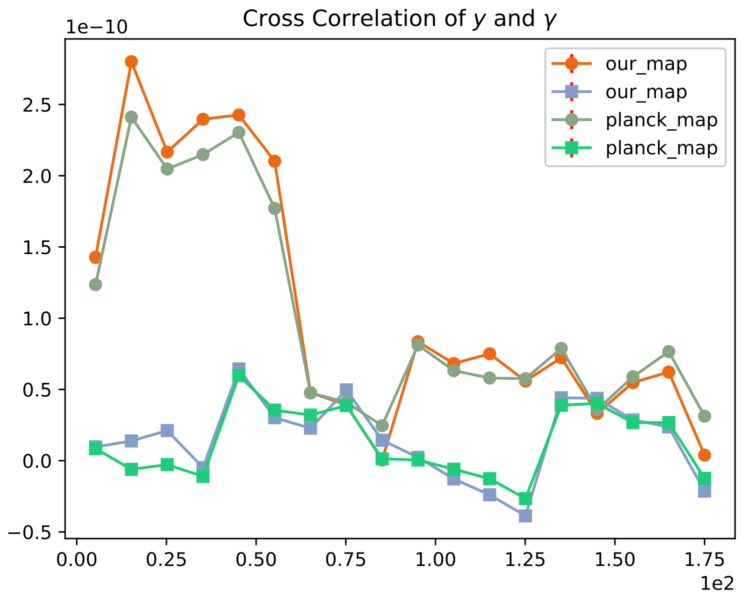

Figure 5.4: Cross Correlation between tSZ and Weak Lensing maps for KiDs fields

Chapter 6

Conclusions

To summarize, We saw that doing foreground clustering using a single measure already performed on par with the Planck collaboration's pipelines [9]. We see now that the various machine learning algorithms used for this improves upon the FC-ILC algorithm using a single measure. We hope that using a independent algorithm to produce maps with different residuals, would be useful in testing for the effect of foregrounds and any biases in the existing algorithms for the estimates of the cosmological parameters. We would also like to point out that in future experiments with more channels using machine learning techniques with allow for more accuracy than single parameter clustering.

Appendix A

m-maps

Looking at the m-maps we can say that, m_5 and m_8 are very noisy. m1,m2,m4 have similar information and m_6, m_7, m_9 have similar information. And m_9 is also more noisier than m_7. Therefore, We choose m_1, m_3 and m_7.

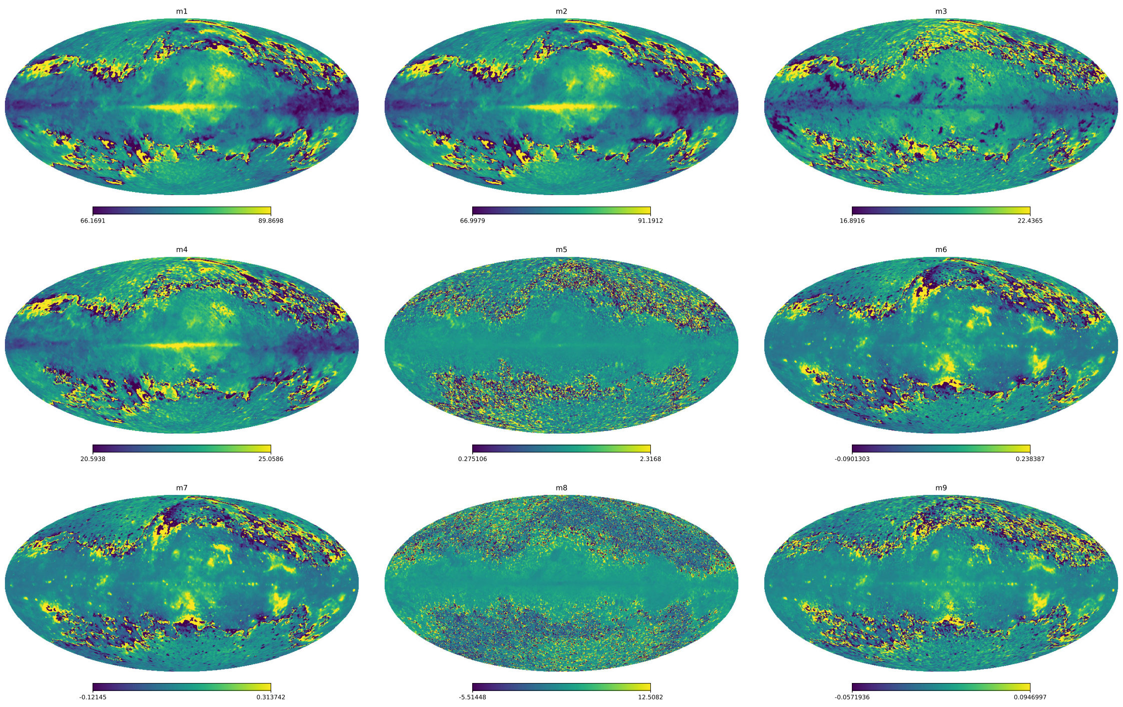

Figure A.1: The m-maps for planck data (these maps are plotted only from 5 to 95 percentile to avoid the outliers from scaling the plot)

Bibliography

[1] A. Hojjati et al., *Cross-correlating Planck tSZ with RCSLenS weak lensing: Implications for cosmology and AGN feedback*, *Mon. Not. Roy. Astron. Soc.* **471** (2017) 1565 [1608.07581].

[2] S. Dodelson, *Gravitational Lensing*. Cambridge University Press, 2017, 10.1017/9781316424254.

[3] H. K. Eriksen et al., *CMB component separation by parameter estimation*, *Astrophys. J.* **641** (2006) 665 [astro-ph/0508268].

[4] R. Khatri, *Linearized iterative least-squares (LIL): a parameter-fitting algorithm for component separation in multifrequency cosmic microwave background experiments such as Planck*, *Mon. Not. Roy. Astron. Soc.* **451** (2015) 3321 [1410.7396].

[5] H. K. Eriksen, A. J. Banday, K. M. Gorski and P. B. Lilje, *Foreground removal by an internal linear combination method: Limitations and implications*, *Astrophys. J.* **612** (2004) 633 [astro-ph/0403098].

[6] M. Remazeilles, J. Delabrouille and J.-F. Cardoso, *Foreground component separation with generalised ILC*, *Mon. Not. Roy. Astron. Soc.* **418** (2011) 467 [1103.1166].

[7] J.-F. Cardoso, M. Martin, J. Delabrouille, M. Betoule and G. Patanchon, *Component separation with flexible models. Application to the separation of astrophysical emissions*, 0803.1814.

[8] D. Marinucci, D. Pietrobon, A. Balbi, P. Baldi, P. Cabella, G. Kerkyacharian et al., *Spherical Needlets for CMB Data Analysis*, *Mon. Not. Roy. Astron. Soc.* **383** (2008) 539 [0707.0844].

[9] R. Khatri, *Data driven foreground clustering approach to component separation in multifrequency CMB experiments: A new Planck CMB map*, *JCAP* **1902** (2019) 039 [1808.05224].

[10] H. U. Norgaard-Nielsen and H. E. Jorgensen, *Foreground removal from CMB temperature maps using an MLP neural network*, *Astrophys. Space Sci.* **318** (2008) 195 [0809.2914].

[11] K. Hornik, *Approximation capabilities of multilayer feedforward networks*, *Neural Networks* **4** (1991) 251 .

[12] F. Pedregosa et al., *Scikit-learn: Machine Learning in Python*, *J. Machine Learning Res.* **12** (2011) 2825 [1201.0490].

[13] R. Khatri, *An alternative validation strategy for the Planck cluster catalogue and y-distortion maps*, *Astron. Astrophys.* **592** (2016) A48 [1505.00778].

[14] M. Abadi et al., *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*, 1603.04467.

[15] A. Paszke, S. Gross et al., *Automatic differentiation in PyTorch*, in *NIPS Autodiff Workshop*, 2017.

[16] P. Wittek, S. Gao, I. Lim and L. Zhao, *somoclu: An efficient parallel library for self-organizing maps*, *Journal of Statistical Software, Articles* **78** (2017) 1.

[17] M. Jarvis, G. Bernstein and B. Jain, *The skewness of the aperture mass statistic*, *Mon. Not. Roy. Astron. Soc.* **352** (2004) 338 [astro-ph/0307393].

[18] H. Hildebrandt et al., *RCSLenS: The Red Cluster Sequence Lensing Survey*, *Mon. Not. Roy. Astron. Soc.* **463** (2016) 635 [1603.07722].

[19] Planck collaboration, N. Aghanim et al., *Planck 2015 results. XXII. A map of the thermal Sunyaev-Zeldovich effect*, *Astron. Astrophys.* **594** (2016) A22 [1502.01596].

[20] K. Kuijken et al., *The fourth data release of the Kilo-Degree Survey: ugri imaging and nine-band optical-IR photometry over 1000 square degrees*, *Astron. Astrophys.* **625** (2019) A2 [1902.11265].

[21] Y.-Z. Ma, L. Van Waerbeke, G. Hinshaw, A. Hojjati, D. Scott and J. Zuntz, *Probing the diffuse baryon distribution with the lensing-tSZ cross-correlation*, *JCAP* **1509** (2015) 046 [1404.4808].

[22] A. Cooray and R. K. Sheth, *Halo Models of Large Scale Structure*, *Phys. Rept.* **372** (2002) 1 [astro-ph/0206508].

[23] R. K. Sheth and G. Tormen, *An Excursion Set Model of Hierarchical Clustering : Ellipsoidal Collapse and the Moving Barrier*, *Mon. Not. Roy. Astron. Soc.* **329** (2002) 61 [astro-ph/0105113].

[24] A. Iqbal, S. Majumdar, B. B. Nath, S. Ettori, D. Eckert and M. A. Malik, *Excess entropy and energy feedback from within cluster cores up to r_{200}*, *Mon. Not. Roy. Astron. Soc.* **472** (2017) 713 [1703.00028].