Problem Set #4

Observational Techniques of Modern Astrophysics — PHYS 641

Fall 2018

Bridget Andersen

November 1, 2018

The code script for each of these problems is located in "./prob0/prob0.py" where "0" is replaced by the problem number $(1, 2, 3)$. Here we present any derivations, plots, and discussion necessary to answer the questions.

Problem 1

In this problem we have been given an example spherical harmonic power spectrum in the form of $\ell(\ell+1)C_\ell/2\pi$.

a) *What should the variance in the map be, given the power spectrum? Don't forget that the $Y_{\ell m}$ also introduce a scaling.*

Solution: Given the power spectrum, we can come up with an analytical expression for the variance of the corresponding map. First, let's define a few things. When studying 1D timeseries or flat 2D maps, we can glean information by expanding our data in terms of coefficients $F(k_x, k_y)$ dependent on the wavenumber in each dimension multiplied by sinusoidal basis functions. In the 2D case we have:

$$T(x,y) = \frac{1}{MN}\sum_{k_x=0}^{N-1}\sum_{k_y=0}^{M-1} F(k_x,k_y)\, e^{2\pi i\left(\frac{k_x x}{N}+\frac{k_y y}{M}\right)} \tag{1}$$

where the basis functions are normalized:

$$\int_0^N dx \int_0^M dy \quad e^{2\pi i\left(\frac{k_x x}{N}+\frac{k_y y}{M}\right)} e^{-2\pi i\left(\frac{k'_x x}{N}+\frac{k'_y y}{M}\right)} = MN\delta\left(k_x-k'_x\right)\delta\left(k_y-k'_y\right) \tag{2}$$

To simplify things, we often refer to the magnitude of the k-vector $(\vec{\mathbf{k}} = k_x\hat{\mathbf{x}} + k_y\hat{\mathbf{y}})$ as $k = \sqrt{k_x^2+k_y^2}$, instead of specifying both k_x and k_y. Within this framework, we can calculate a series of values that describe the distribution of power into the different k-components of the expansion: $C_k = \langle |F(k)|^2\rangle$. This series of values is the power spectrum of our data.

However, when exploring the cosmic microwave background, we often want to study temperature fluctuations over the entire sky. To do this, we can complete a similar sort of expansion over the spherical harmonic basis functions:

$$T(\theta,\phi) = \sum_{\ell=0}^{\ell_{max}}\sum_{m=-\ell}^{\ell} a_{\ell m} Y_{\ell,m} \tag{3}$$

Here, the $a_{\ell m}$ are equivalent to the $F\left(k_x, k_y\right)$ coefficients in the flat 2D expansion, and the power spectrum of this expansion is given by: $C_\ell = \langle |a_{\ell m}|^2 \rangle$. The basis functions are given by:

$$Y_{\ell,m} = \sqrt{\frac{2\ell+1}{4\pi}\frac{(\ell-m)!}{(\ell+m)!}} e^{im\phi} P_\ell^m\left(\cos\theta\right) \tag{4}$$

where $P_\ell^m\left(\cos\theta\right)$ are the associated Legendre functions and ℓ, m are integers such that $|m| \leq \ell$. Like before, these basis functions are normalized:

$$\iint d\Omega \quad Y_{\ell,m} Y^*_{\ell',m'} = \delta\left(\ell - \ell'\right)\delta\left(m - m'\right) \tag{5}$$

Note that when dealing practically with spherical harmonic expansions, since $Y_{\ell,-m} = Y^*_{\ell,m}$, we usually only keep track of $a_{\ell m}$ values for positive m's.

Now, back to the question at hand. With the mathematical machinery above, we can calculate what the variance of a generated map T should be, given the spherical harmonic power spectrum C_ℓ:

$$\begin{aligned}
\langle T^2 \rangle &= \left\langle \sum_{\ell,m} a_{\ell m} Y_{\ell,m} \sum_{\ell',m'} a^*_{\ell' m'} Y^*_{\ell',m'} \right\rangle = \sum_{\ell,\ell'} \sum_{m,m'} \langle Y_{\ell,m} Y^*_{\ell',m'} \rangle \langle a_{\ell m} a^*_{\ell' m'} \rangle \\
&= \sum_{\ell,\ell'} \sum_{m,m'} \frac{\iint d\Omega \quad Y_{\ell,m} Y^*_{\ell',m'}}{\iint d\Omega} \cdot \langle |a_{\ell m}|^2 \rangle = \sum_{\ell,\ell'} \sum_{m,m'} \frac{\delta\left(\ell-\ell'\right)\delta\left(m-m'\right)}{\iint d\Omega} \cdot C_\ell \\
&= \sum_\ell \frac{C_\ell}{4\pi} \sum_m 1
\end{aligned}$$

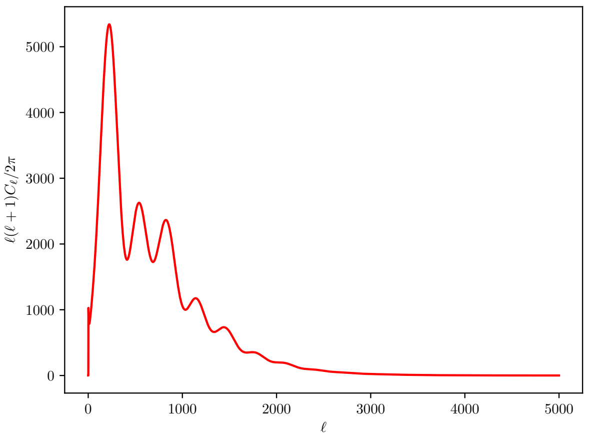
Now for each ℓ value, there are $2\ell+1$ corresponding m values ($m = -\ell, -\ell+1, ..., 0, ..., \ell-1, \ell$). Taking this into account, we have:

$$\boxed{\langle T^2_{pred} \rangle = \frac{C_\ell}{4\pi} \sum_\ell \left(2\ell + 1\right)} \tag{6}$$

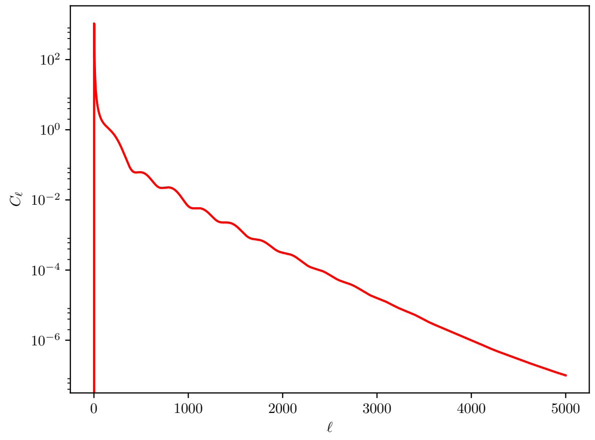
Notice that this variance is independent of any m values. The $1/4\pi$ comes from the $Y_{\ell,m}$ scaling. In **prob1.py**, we use equation 6 in function `calc_map_variance()` to calculate the predicted map variance based on the provided example $\ell\left(\ell+1\right)C_\ell/2\pi$ values (see Figures 1a and 1b for plots of the example power spectrum). The result is $\boxed{\langle T^2_{pred} \rangle = 12077\,\mu\text{K}^2}$. In our code, we set the $\ell = 0$ and $\ell = 1$ power spectrum values to zero: $C_0, C_1 = 0$. We can do this since the $\ell = 0$ mode is the monopole mode and is not very well-detected. The $\ell = 1$ dipole mode only contains information about the peculiar velocity of the Earth, and so it is not relevant to our studies of the cosmic microwave background.

b) *Generate a set of $a_{\ell m}$ from this power spectrum yourself. You can turn this into a map with the healpix command `alm2map`, as shown in class. Make an image of the map (`healpy.mollview`). Does the map look sensible? Does the variance of the map agree with your answer from part a)?*

Solution: We can generate $a_{\ell m}$ values from our power spectrum by using the relation $C_\ell = \langle |a_{\ell m}|^2 \rangle$, which tells us that the variance of a given $a_{\ell m}$ is simply given by C_ℓ, independent of the m value. Thus, we can simulate $a_{\ell m}$ values by grabbing values from a normal Gaussian distribution with a mean of zero and a standard deviation of $\sqrt{C_\ell}$. We do



(a) A plot the provided example power spectrum in terms of $\ell\,(\ell+1)\,C_\ell/2\pi$.



(b) A plot the provided example power spectrum in terms of C_ℓ.

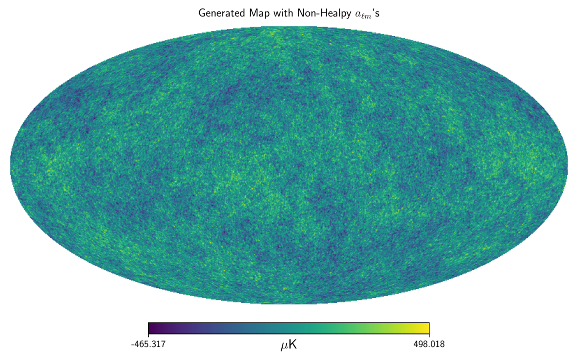
Figure 1

this in the `generate_alms()` function to generate both the real and imaginary components for each $a_{\ell m}$ value, and we store our resulting complex $a_{\ell m}$'s in a matrix called `alm_matrix`. This matrix is organized with each row representing the $a_{\ell m}$ values for a given ℓ, ordered by increasing m with increasing column index. Since each m value only goes up to the ℓ value for each row, we simply pad the rest of each row with filler values of `-10`. We go through all this trouble because the `healpy.alm2map()` function takes in $a_{\ell m}$ values in order of increasing m (first, all $a_{\ell 0}$ values, then all $a_{\ell 1}$ values, etc). This way, once we have constructed our matrix, we can simply slice down each column, trim off the filler values, and add them to the `input_alms` array that we will eventually feed into the `healpy.alm2map()` function. Yes, our method of generating $a_{\ell m}$'s is memory inefficient and there is a small chance that `-10` is not a good padding value, but this method is much faster than a double for-loop and it works well for our particular situation.

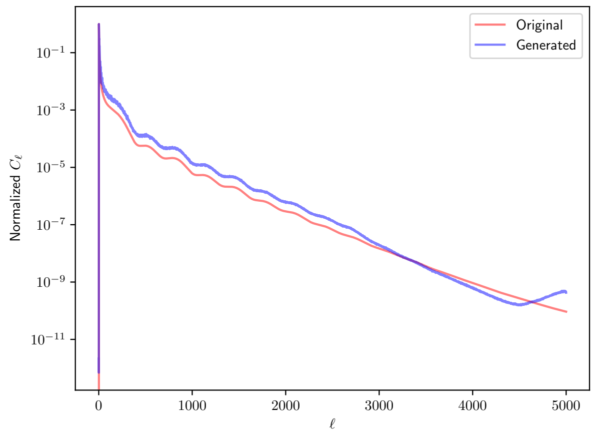
An example of the resulting map from our homemade $a_{\ell m}$'s is shown in Figure 2a. The map looks roughly sensible, with mK-level variations of about a degree in size (zoom in to see the variations more clearly). Although this map will change every time we generate it, its variance is always within 10% of the variance that we predicted in part a). For example, for our most recent run, the variance of the generated map was $\langle T_{gen}^2\rangle = 11438\,\mu\text{K}^2$, which is within 5.44% of $\langle T_{pred}^2\rangle = 12077\,\mu\text{K}^2$. If you run our **prob1.py** script, then it will print out each of the variances and the percent difference.

c) *You could turn your map back into $a_{\ell m}$'s with* ***`healpy.map2alm`***. *However, you can do that, plus average the squared $a_{\ell m}$'s with* ***`healpy.anafast`***. *Plot the power spectrum you get here against the input model. Do they agree?*
Solution: Exactly as proposed in the question, we use `healpy.anafast()` to convert our map back into a power spectrum ($C_\ell = \langle |a_{\ell m}|^2\rangle$). Figure 2b shows the original input power spectrum and the newly generated power spectrum, normalized and plotted on the same axes. As we can see, their shapes roughly agree.

d) ***`Healpy`*** *also provides functionality to generate $a_{\ell m}$'s and maps directly, via* ***`healpy.synalm`***

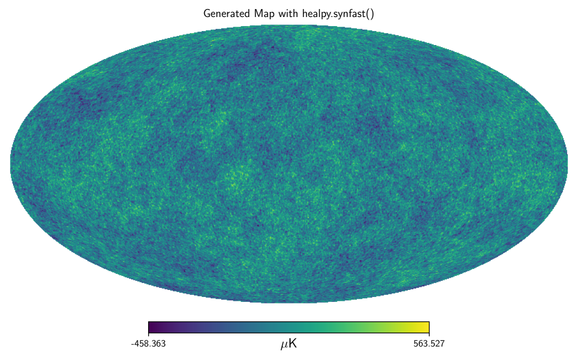


(a) An example of the map generated from our homemade $a_{\ell m}$'s.

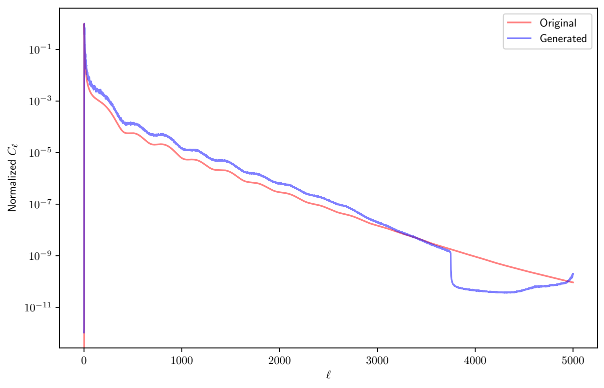


(b) An example of the power spectrum reverse-engineered from our homemade $a_{\ell m}$'s generated map. The original input power spectrum is also plotted.

Figure 2



(a) An example of the map generated from `healpy.synfast()`.



(b) An example of the power spectrum reverse-engineered from our `healpy.synfast()` generated map. The original input power spectrum is also plotted.

Figure 3

and ***healpy.synfast****. Use one of these to repeat the problem, generating a map that you analyze with* ***anafast*** *and compare to the input.*

Solution: Again, we do exactly as the question suggests and use `healpy.synfast()` to generate a map from the input C_ℓ values. The resulting map is shown in Figure 3a. It's variance is $\langle T^2_{gen} \rangle = 11953\,\mu\text{K}^2$, which is within 1.04% of $\langle T^2_{pred} \rangle = 12077\,\mu\text{K}^2$. Then we use `healpy.anafast()` to convert our map back into a power spectrum. Figure 3b shows the original input power spectrum and the newly generated power spectrum, normalized and plotted on the same axes. As we can see, their shapes roughly agree.

Problem 2

In this problem, we'll repeat the essentials of Problem 1, but on a flat sky. A perfectly sensible patch size is 20° by 20°, large enough to be well away from the first peak in the power spectrum, but small enough to be reasonably flat.

a) *What is the (rough) conversion between k and ℓ for this map? What is the lowest ℓ you could think of going to?*

Solution: In this flat sky approximation, we will use the 2D Fourier transform to decompose our sky rather than the spherical harmonic transform that we used in Problem 1. Since our patch of sky is square, we have that $M = N$ in the definition given in equation 1. In the 2D Fourier transform, the important mode number is k rather than ℓ, where the k-vector is defined as $\vec{\mathbf{k}} = k_x\hat{\mathbf{x}}+k_y\hat{\mathbf{y}}$. k_x and k_y are the mode numbers in each of the coordinate directions on the 2D plane, and so we can write $k = \sqrt{k_x^2 + k_y^2}$.

However, our input power spectrum is given in terms of ℓ modes and C_ℓ values. Thus, we need a way to convert between ℓ (defined over the whole sky) and k (defined over our 20° by 20° patch of the sky). This will allow us to pick out values of C_ℓ from our input spectrum that roughly correspond to the values of C_k that we will need to simulate our sky patch later in this question.

Each k mode will have an integral number of wavelengths over the sky patch (the $k = 1$ mode will have 1 period, $k = 2$ will have 2 periods, etc). $Y_{\ell\ell}$ has ℓ periods across the 360° equator. We must pick out the closest ℓ that would also fit k wavelengths over the 20° by 20° sky patch. Essentially, we want ℓ to be like an angular frequency ω. If ω has a period P (in degrees), then we have the relation:

$$\omega = \frac{360^\circ}{P} \tag{7}$$

In our case, the $\omega = \ell$ corresponding to the kth mode will have k periods over 20°, so $P = 20^\circ/k$ and we find the relation:

$$\boxed{\ell = \frac{360^\circ \cdot k}{20^\circ}} \tag{8}$$

It follows that the lowest ℓ that we could possibly go to would be the ℓ that fits one wavelength over 20° (corresponding to $k = 1$).

b) *Up to some ℓ_{max} and corresponding k_{max}, how many $a_{\ell m}$'s are there in the sky? How many modes up to k_{max}? From this, show how you would convert the usual CMB power spectrum to a flat-sky one.*

Solution: First, for a given ℓ_{max}, we can calculate how many $a_{\ell m}$'s there are there in the sky by first considering the simple case of $\ell_{max} = 3$. For a given ℓ_{max} there will be $(\ell_{max} + 1)$ possible m values (considering only $a_{\ell m}$'s with positive m's). For our simple case, that includes $m = 0, 1, 2, 3$. For each m value, there will be a certain number of $a_{\ell m}$'s corresponding to ℓ values from $\ell = 0$ to $\ell = \ell_{max}$. We can write out the number of $a_{\ell m}$ values per m (n_m) for our simple case:

$$\begin{aligned}
m = 0 &\implies n_0 = 4 = \ell_{max} + 1 \text{ values}\\
m = 1 &\implies n_1 = 3 = \ell_{max}\\
m = 2 &\implies n_2 = 2 = \ell_{max} - 1\\
m = 3 &\implies n_3 = 1 = \ell_{max} - 2
\end{aligned}$$

From this we can see that there are $\ell_{max} - (m-1)$ values per m, except for the $m = 0$ case, for which there are $\ell_{max} + 1$ values. Thus, we can find the total number of $a_{\ell m}$'s over the whole sky like so:

$$\begin{aligned}
n_{\ell_{max} m} &= \ell_{max} + 1 + \sum_{m=1}^{\ell_{max}} \ell_{max} - (m-1) \\
&= \ell_{max} + 1 + \ell_{max}(\ell_{max}+1) - \frac{(\ell_{max}+1)(\ell_{max}+2)}{2} + \ell_{max} + 1 \\
&= \ell_{max}^2 + 3\ell_{max} + 2 - \frac{(\ell_{max}+1)(\ell_{max}+2)}{2} \\
&= (\ell_{max}+1)(\ell_{max}+2) - \frac{(\ell_{max}+1)(\ell_{max}+2)}{2} \\
\implies & \boxed{n_{\ell_{max} m} = \frac{(\ell_{max}+1)(\ell_{max}+2)}{2}}
\end{aligned}$$

A given k_{max} will have maximum mode values in both the $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$ directions, given by $k_{x,max}$ and $k_{y,max}$, such that $k_{max} = \sqrt{k_{x,max}^2 + k_{y,max}^2}$. Since our patch of sky is square, we have that $k_{x,max} = k_{y,max} = k_{u,max}$. So if we define the number of modes in one of the coordinate directions to be given by $k_{u,max} = k_{max}/\sqrt{2}$, then:

$$\sqrt{k_{x,max}^2 + k_{y,max}^2} = \sqrt{2 \cdot k_{u,max}^2} = \sqrt{2\left(\frac{k_{max}}{\sqrt{2}}\right)^2} = \sqrt{k_{max}^2} = k_{max} \tag{9}$$

For k's, the number of modes in one direction is equal to the maximum mode value in that direction plus one for the zeroth mode. Also note that each k_x mode can pair up with each k_y mode to form a new k mode (so we must square of the number of modes in one direction to get the total number of modes). Therefore, the total number of modes for a given k_{max} is given by:

$$\boxed{n_{k_{max}} = \left(\left\lfloor \frac{k_{max}}{\sqrt{2}} \right\rfloor + 1\right)^2} \tag{10}$$

From all of this, we can create a method to convert the usual CMB power spectrum to a flat-sky one by finding the C_ℓ values from our input spectrum that correspond with the C_k values of the flat-sky spectrum. First, we use equation 8 to convert from the ℓ_{max} of our input spectrum to the k_{max} of our resulting flat-sky spectrum. Then, we can use the square root of equation 10 to find the number of k modes in each coordinate direction (k_x, k_y), and use that number to set up a matrix of resulting k values. Once we have this matrix of k values, we can again use equation 8 to convert the whole matrix into a matrix of ℓ's, rounding to the nearest integer to find the closest ℓ value to a given k. Finally, we can input each closest ℓ value into the C_ℓ spectrum to pick out the C_k for each corresponding k value. We coded this whole procedure in **prob2.py**. The resulting C_k power spectrum is shown in Figure 4. Note that we also multiply the power spectrum amplitude by a factor of ℓ_{max}/k_{max} to convert between the ℓ and k space.

c) *Given the max ℓ in the sample power spectrum, there should be a maximum pixel size you can pick to capture all the possible information in the power spectrum (for instance, if your power spectrum goes out to the first peak, 2° pixels are clearly too large). What is it? Pick a pixel*

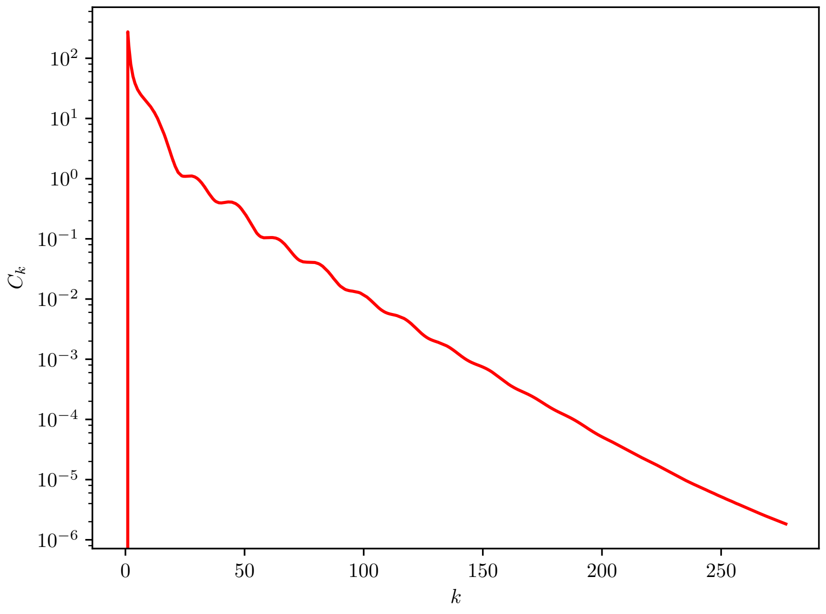


Figure 4: The C_k power spectrum converted from the input C_ℓ spectrum.

size much smaller than this, and simulate flat-sky CMB maps.

Solution: We cannot have pixels larger than a single wavelength of ℓ_{max}, otherwise we lose information (the ℓ_{max} mode will contain finer structure than we can display). Since $Y_{\ell\ell}$ has ℓ wavelengths across the 360° equator, then the pixel size must be less than or equal to the angular size of a wavelength of the ℓ_{max} mode:

$$\boxed{\texttt{pix\_size} \leq \frac{360^\circ}{\ell_{max}}} \tag{11}$$

From this, the number of pixels along one dimension of our 20° by 20° sky patch should follow the relation:

$$n_{pix} = \frac{20^\circ}{\texttt{pix\_size}} \geq \frac{20^\circ \cdot \ell_{max}}{360^\circ} \tag{12}$$

Note that if we take the equality, then this is just equation 8 with $\ell = \ell_{max}$. In this case, we would have that $n_{pix} = k_{max}$.

For our particular problem, the ℓ_{max} of our input spectrum is 5000. Using equations 11 and 12, the maximum possible pixel size that we could use would be $\texttt{pix\_size} = 0.072^\circ$ and the minimum number of pixels in one dimension would be $n_{pix} = k_{max} = 278$. When we actually simulate our flat-sky CMB map in **prob2.py**, we use a number of pixels along each dimension given by:

$$\boxed{n_{pix} = 2 \cdot \left(\left\lfloor \frac{k_{max}}{\sqrt{2}} \right\rfloor + 1 \right)} \tag{13}$$

This ends up being $n_{pix} = 394$, which is more pixels than the minimum value dictated by equation 12. We chose this number of pixels so that we could make our k_x and k_y values symmetric about the origin $(0,0)$. Then we can make k_x vary from $-n_{pix}/2$ to $n_{pix}/2$ (and same for k_y). The fact that our k_x, k_y's are symmetric makes the 2D Fourier transform behave nicely later on when we are generating our map. At the same time, our k_x, k_y's reach high enough modes to completely sample the highest k_{max} possible given our ℓ_{max} input information (see the derivation of equation 10).

Once we have our matrix of C_k values that we obtain from the procedure described in part b), we can use them to generate the $F(k)$ coefficients of the 2D Fourier transform as described in equation 1. Note that since $\langle |F(k)|^2 \rangle = C_k$ just like $\langle |a_{\ell m}|^2 \rangle = C_\ell$, we can use the same procedure to generate $F(k)$'s as we did to generate $a_{\ell m}$'s in Problem #1. To generate complex numbers with appropriate phase relations to be the transform of a real field, we take the Fourier transform of a real n_{pix} by n_{pix} field of Gaussian random numbers. We then multiply this matrix by $\sqrt{C_k}$ and take the inverse Fourier transform to obtain our simulated field. An example simulated field is shown in Figure 5.

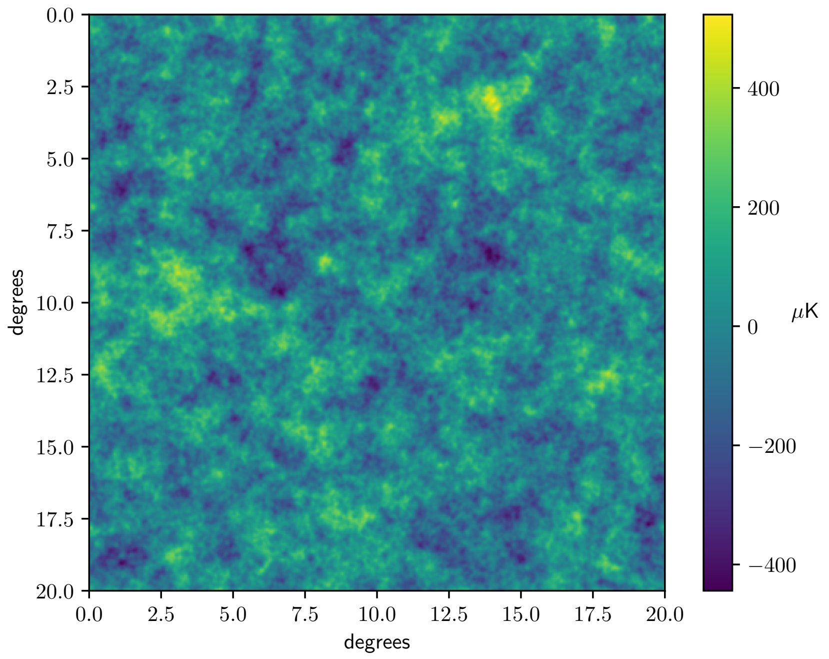


Figure 5: An example of the 20 by 20 degree map generated by our code.

d) *Show that the variance in your maps agrees with what you expect from the first problem.*
Solution: We can derive an analytical expression for the variance of the map generated from

a 2D Fourier transform, just as we did for the spherical harmonic transform:

$$
\begin{aligned}
\langle T^2 \rangle &= \left\langle \frac{1}{n_{pix}^2} \sum_{k_x,k_y} F(k) e^{2\pi i\left(\frac{k_x x}{n_{pix}}+\frac{k_y y}{n_{pix}}\right)} \cdot \frac{1}{n_{pix}^2} \sum_{k_x',k_y'} F^*(k) e^{-2\pi i\left(\frac{k_x' x}{n_{pix}}+\frac{k_y' y}{n_{pix}}\right)} \right\rangle \\
&= \frac{1}{n_{pix}^4} \sum_{k_x,k_y} \sum_{k_x',k_y'} \langle F(k) F^*(k) \rangle \left\langle e^{2\pi i\left(\frac{k_x x}{n_{pix}}+\frac{k_y y}{n_{pix}}\right)} e^{-2\pi i\left(\frac{k_x' x}{n_{pix}}+\frac{k_y' y}{n_{pix}}\right)} \right\rangle \\
&= \frac{1}{n_{pix}^4} \sum_{k_x,k_y} \sum_{k_x',k_y'} \langle F(k) F^*(k) \rangle \cdot \frac{\int dx \int dy \quad e^{2\pi i\left(\frac{k_x x}{n_{pix}}+\frac{k_y y}{n_{pix}}\right)} e^{-2\pi i\left(\frac{k_x' x}{n_{pix}}+\frac{k_y' y}{n_{pix}}\right)}}{\iint dx dy} \\
&= \frac{1}{n_{pix}^4} \sum_{k_x,k_y} \sum_{k_x',k_y'} C_k \cdot \frac{n_{pix}^2 \delta\left(k_x - k_x'\right) \delta\left(k_y - k_y'\right)}{n_{pix}^2} \\
\implies \langle T_{pred}^2 \rangle &= \frac{1}{n_{pix}^4} \sum_{k_x,k_y} C_k
\end{aligned}
$$

Using this formula, we find that the predicted variance of our simulated 20° by 20° given our derived C_k spectrum is $\langle T_{pred}^2 \rangle = 12776\,\mu\text{K}^2$. If we take the variance of the map that we derived in part c), we find a variance of $\langle T_{gen}^2 \rangle = 12768\,\mu\text{K}^2$, which is within 0.06% of our predicted map variance and within 5.55% of our predicted variance from Problem #1. If you run our **prob1.py** script, then it will print out each of the variances and the percent difference.

$\boxed{\text{Although the generated map changes, it is always within } 10\% \text{ of our predicted variance}}$

$\boxed{\text{and generated variance from Problem \#1.}}$

Note that throughout all of these variance calculations, we scale by factors of ℓ_{max}/k_{max} and n_{pix}^2 so that we can compare our values between ℓ and k space.

Problem 3

In this problem, we'll investigate the ultimate performance of a perfect 150 *GHz detector in space, with the only load coming from the CMB.*

a) *The units of the Planck function are ergs per second square cm per Hz per steradian. A typical detector will see a solid angle of around a steradian, with a collecting area of* λ^2. *How many ergs per second per Hz would hit a detector from the* 2.725 *K CMB?*
Solution: The Planck function is given by:

$$B_{\nu,\text{erg}}(T) = \frac{2h\nu^3}{c^2} \cdot \frac{1}{e^{h\nu/kT} - 1} \tag{14}$$

where ν is the frequency, T is the temperature, and the rest of the variables are fundamental constants. As stated in the question, the units of this function are:

$$[B_{\nu,\text{erg}}] = \frac{\text{erg}}{\text{s} \cdot \text{cm}^2 \cdot \text{Hz} \cdot \text{sr}} \tag{15}$$

We wish to derive something of the units:

$$\frac{\text{erg}}{\text{s} \cdot \text{Hz}} \tag{16}$$

To do this, we need to integrate the Planck function over the field of view of the detector ($\Omega = 1\,\mathrm{sr}$) and the collecting area of the detector ($A = \lambda^2 = c^2/\nu^2$) like so:

$$\int_0^{1\,\mathrm{sr}} d\Omega \int_0^{c^2/\nu^2} dA \quad \frac{2h\nu^3}{c^2} \cdot \frac{1}{e^{h\nu/kT}-1} = \frac{2h\nu^3}{c^2} \cdot \frac{c^2}{\nu^2} \cdot \frac{1}{e^{h\nu/kT}-1} = \frac{2h\nu}{e^{h\nu/kT}-1} \tag{17}$$

Plugging in numbers we find that there is:

$$= \frac{2\left(6.626 \cdot 10^{-27}\,\mathrm{erg\,s}\right)\left(150 \cdot 10^{9}\,\mathrm{Hz}\right)}{e^{(6.626\cdot 10^{-27}\,\mathrm{erg\,s})(150\cdot 10^{9}\,\mathrm{Hz})/[(1.38\cdot 10^{-16}\,\mathrm{erg/K})(2.725\,\mathrm{K})]}-1} = \boxed{1.523 \cdot 10^{-16}\,\frac{\mathrm{erg}}{\mathrm{s\,Hz}}} \tag{18}$$

hitting our detector from the 2.725 K CMB. That's really small.

b) *Let's assume we have a 30 GHz bandwidth for our 150 GHz detector. How many photons per second come into the detector (as a sanity check, I get several billion photons per second per detector)? Are we in the shot-noise or continuous signal regime?*
Solution: We can convert equation 14 from

$$[B_{\nu,\mathrm{erg}}] = \frac{\mathrm{erg}}{\mathrm{s}\cdot\mathrm{cm}^2\cdot\mathrm{Hz}\cdot\mathrm{sr}} \tag{19}$$

to

$$[B_{\nu,\mathrm{photon}}] = \frac{\mathrm{photons}}{\mathrm{s}\cdot\mathrm{cm}^2\cdot\mathrm{Hz}\cdot\mathrm{sr}} \tag{20}$$

by simply dividing our equation by the energy in each photon $E = h\nu$:

$$B_{\nu,\mathrm{photon}}(T) = \frac{B_{\nu,\mathrm{erg}}}{h\nu} = \frac{2h\nu^3}{h\nu c^2} \cdot \frac{1}{e^{h\nu/kT}-1} = \frac{2\nu^2}{c^2} \cdot \frac{1}{e^{h\nu/kT}-1} \tag{21}$$

Now, to find the number of photons per second hitting the detector, we need to integrate over the field of view ($\Omega = 1\,\mathrm{sr}$), the collecting area ($A = \lambda^2 = c^2/\nu^2$), and the bandwidth of the detector (from $\nu_\ell = 150 - 30/2\,\mathrm{GHz} = 135\,\mathrm{GHz}$ to $\nu_h = 150 + 30/2\,\mathrm{GHz} = 165\,\mathrm{GHz}$):

$$\begin{aligned} n_\gamma &= \int_{\nu_\ell}^{\nu_h} d\nu \int_0^{1\,\mathrm{sr}} d\Omega \int_0^{c^2/\nu^2} dA \quad \frac{2\nu^2}{c^2} \cdot \frac{1}{e^{h\nu/kT}-1} \\ &= \int_{\nu_\ell}^{\nu_h} d\nu \quad \frac{2\nu^2}{c^2} \cdot \frac{c^2}{\nu^2} \cdot \frac{1}{e^{h\nu/kT}-1} = \int_{\nu_\ell}^{\nu_h} d\nu \quad \frac{2}{e^{h\nu/kT}-1} \end{aligned}$$

Using a u-substitution, we can plug in $u = e^{h\nu/kT} - 1$ to get:

$$\begin{aligned} &= \int du \quad \frac{kT}{h} \cdot \frac{1}{u\,(u+1)} = \frac{kT}{h}\left[\log(u) - \log(u+1)\right] = \frac{kT}{h}\log\left(\frac{u}{u+1}\right) \\ &= \frac{kT}{h}\left[\log\left(\frac{e^{h\nu_h/kT}-1}{e^{h\nu_h/kT}}\right) - \log\left(\frac{e^{h\nu_\ell/kT}-1}{e^{h\nu_\ell/kT}}\right)\right] \end{aligned}$$

Plugging in values, we find that this is equal to $\boxed{n_\gamma = 2.33 \cdot 10^{9}\ \text{photons per second}}$ hitting our detector.

In the shot-noise limit, a small enough number of photons arriving at the detector that the noise can be treated as a Poisson process, and the noise is given by $\delta n = n/\sqrt{nt} \sim n^{1/2}$ where n is the number of photons and t is a given time period. In the continuous limit, photons are continually arriving at the detector, and the noise is given by $\delta T = T/\sqrt{Bt} \sim T$ where T is the brightness temperature and B is the bandwidth of the detector. Since the photon occupation number goes like $1/\left(\exp(h\nu/kT) - 1\right)$, then far to the left of the blackbody peak ν decreases and so the occupation number increases and we are in the continuous limit. Far to the right of peak, ν increases and so the occupation number decreases and we are in the shot-noise limit. However, if we look at the plot of the 2.725 K blackbody curve shown in Figure 6, our bandwidth is very near to the peak, $\boxed{\text{not in the continuous or shot-noise limit.}}$

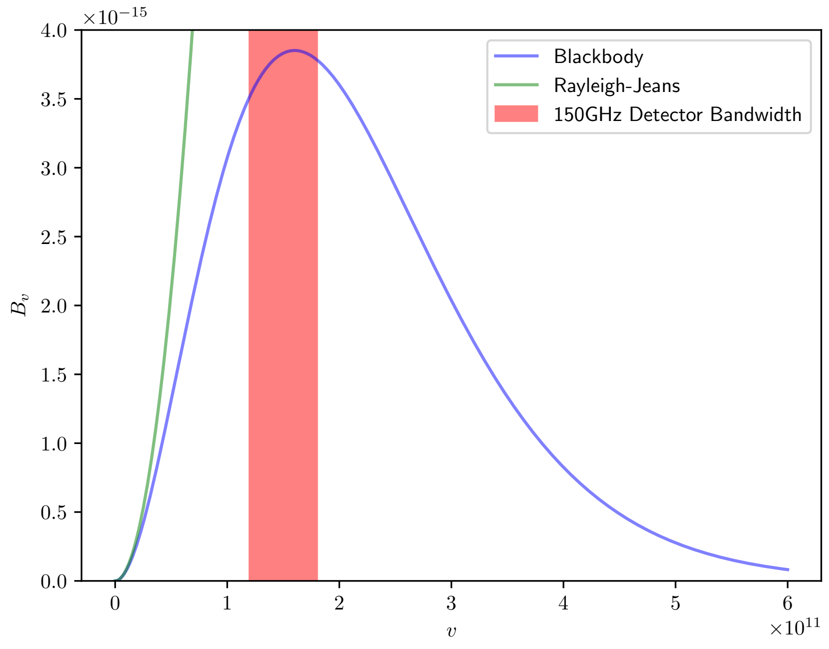


Figure 6: A plot of the 2.725 K blackbody function with the bandwidth of our detector labelled.

c) *From your answer to part b), work out the noise in μK in one second. How does this compare to the best Planck 143 GHz detectors, with noises in the 50 μKs$^{1/2}$ range? Could a next-generation CMB satellite get to much deeper maps just by improving detectors?*
Solution: Since our situation is not quite in either the continuous or shot-noise limit, we will not make either of the noise approximations available in those regimes. Instead, we note that the fractional error in the brightness temperature is given by:

$$\frac{\delta T}{T} = \frac{1}{\sqrt{n_\gamma t}} \tag{22}$$

where $n_\gamma t$ is the number of independent samples. In our case, this is the number of photons that reach our detector in time $t = 1\,\text{s}$. Therefore, our noise is given by:

$$\delta T = \frac{2.725 \cdot 10^6\,\mu\text{K}}{\sqrt{2.33 \cdot 10^9}} = \boxed{56\,\mu\text{K}} \tag{23}$$

This is about the same noise as the Planck 143 GHz detectors, so $\boxed{\text{no, a next-generation CMB satellite could not get to much deeper maps just by improving detectors. Bummer.}}$