1 Summary of Mspec

Mspec is a pseudo-C_ℓ method for producing a multi-frquency signal estimate and covariance from *Planck* maps. The covariance is calculated analytically directly from the data. We do galactic cleaning via linear combinations at the powerspectrum level in such a way as to avoid assumptions about galactic foreground gaussianity. Parameter estimation proceeds via MCMC for cosmological, extragalactic foreground, and frequency calibration parameters.

Aside from dependencies on Healpix and CAMB, the code is only a few hundred lines of Python. The code is parallelized with MPI and takes anywhere from a few CPU-minutes to a few CPU-hours to run from maps to parameters, depending on the number of input maps and cosmological parameterization. We avoid complicated indexing schemes by using Python dictionaries to call powerpsectra by name.

2 More Detail

2.1 Conventions

- We will use (a,b,c,d) to index individual detectors and $(\alpha, \beta, \gamma, \delta)$ to index frequencies.
- The mode-coupling matrix is,

$$M_{\ell_1\ell_2} = \frac{2\ell_2+1}{4\pi}\sum_{\ell_3}(2\ell_3+1)W_{\ell_3}\begin{pmatrix}\ell_1 & \ell_2 & \ell_3\\ 0 & 0 & 0\end{pmatrix}^2 \tag{1}$$

- We will also refer to the symmetric mode-coupling matrix $G_{\ell_1\ell_2} = M_{\ell_1\ell_2}/(2\ell_2+2)$ and a superscript of (2) means its for the square of the mask.
- The pseudo-powerspectra are assumed to be related to the signal, noise, and beams by,

$$\widetilde{C}_\ell^{ab} = \sum_{\ell'} M_{\ell\ell'}\left(B^a_{\ell'}B^b_{\ell'}C_{\ell'}^{\nu(a)\nu(b)} + \delta_{ab}N^a_{\ell'}\right) \tag{2}$$

- The mask deconvolved pseudo-powerspectra are,

$$\mathring{C}_\ell^{ab} = \sum_{\ell'} M^{-1}_{\ell\ell'}\widetilde{C}^{ab}_{\ell'} \tag{3}$$

- The unbiased estimator of the signal is,

$$\widehat{C}_\ell^{ab} = \frac{1}{B^a_\ell B^b_\ell}\left(\sum_{\ell'} M^{-1}_{\ell\ell'}\widetilde{C}^{ab}_{\ell'} - \delta_{ab}N^a_{\ell'}\right) \tag{4}$$

$$\tag{5}$$

2.2 The pseudo-C_ℓ's

We use `anafast` to compute pseudo-powerspectra from all possible 143, 217, and 353 detector combinations. We ignore hitcounts and use spatially uniform weighting in regions which are not masked, so as to not complicate the B_ℓ calculation. The covariance of the pseudo-powerspectra can be approximated with a straight-forward generalization of Efstathiou 2004 to multiple frequencies/detectors, given by,

$$\widetilde{\Sigma}_{(ab,\ell_1)(cd,\ell_2)} \quad = \quad \left[\langle\mathring{C}^{ac}_{\ell_1}\rangle\langle\mathring{C}^{bd}_{\ell_2}\rangle + \langle\mathring{C}^{ad}_{\ell_1}\rangle\langle\mathring{C}^{bc}_{\ell_2}\rangle + [\ell_1 \leftrightarrow \ell_1]\right] G^{(2)}_{\ell_1\ell_2}/2 \tag{6}$$

This approximation holds for ℓ larger than the width of the mode-coupling kernel. With a highly apodized mask, this is roughly $\ell = 70$.

The powerspectra terms on the RHS require knowledge of the true signal and noise. Since we aim to use only the data itself to estimate the covariance, we approximate these terms with the following procedure. We isolate the noise by looking at the difference between auto and cross spectra, then heavily smooth with a gaussian kernel in ℓ. This works because of the strong assumption that the noise is flat.

For the signal, we similarly isolate it by looking at cross spectra, but instead of gaussian smoothing, we do a smoothing based on principal components. Using a suite of signal powerspectra sampled from some region of parameter space, we create a low dimensional description of the signal (in practice about 20 principal components). We then estimate $\langle \mathring{C}_\ell \rangle \approx P_{\ell\ell'} \mathring{C}_{\ell'}$ where P is the projection matrix onto the lower dimensional space. What we've done is essentially smooth the noisy $\mathring{C}_\ell$'s given some assumptions about possible signal spectra. Tests on simulations in the next section reveal minimal biases from this procedure.

2.3 The signal estimator

Once we have $\widetilde{\Sigma}$ we simply use Equation 4 to compute the covariance of the estimator. We compute it directly in ℓ-bins, using the binning matrix Q, to speed up computation and conserve disk space.

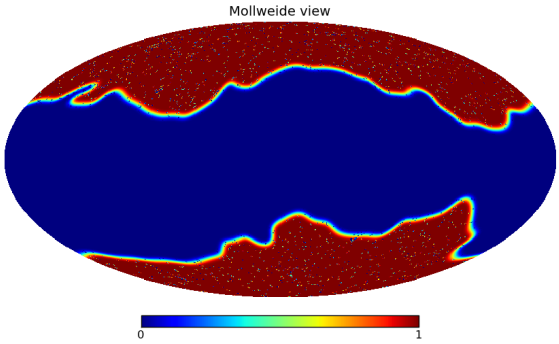
$$\widehat{\Sigma}_{(ab,b_1)(cd,b_2)} = \sum_{\ell_1,\ell_2} Q_{b_1\ell_1} Q_{b_2\ell_2} \frac{1}{B^a_{\ell_1} B^b_{\ell_1} B^c_{\ell_2} B^d_{\ell_2}} \sum_{\ell_3,\ell_4} M^{-1}_{\ell_1\ell_3} M^{-1}_{\ell_2\ell_4} \widetilde{\Sigma}_{(ab,\ell_1)(cd,\ell_2)} \tag{7}$$

Finally, we average over detectors which have the same frequency using the weight function W^{ab}_ℓ which is 0 if $a = b$, meaning only noise-bias-free powerspectra contribute to the estimate of the signal. There may be a clever way to find the optimal W, or we may stick with some simple approximation. Thus the per-frequency signal estimator and its covariance is,

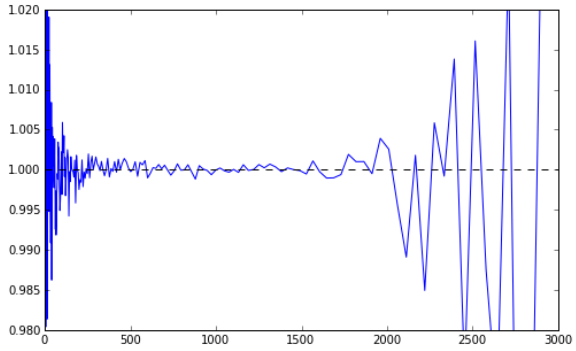
$$\widehat{C}^{\alpha\beta}_{b_1} = \sum_{\substack{a\in\alpha \\ b\in\beta}} W^{ab}_{b_1} \widehat{C}^{ab}_{b_1} \tag{8}$$

$$\widehat{\Sigma}_{(\alpha\beta,b_1)(\gamma\delta,b_2)} = \sum_{\substack{a\in\alpha \\ \cdots}} W^{ab}_{b_1} W^{cd}_{b_2} \widehat{\Sigma}_{(ab,b_1)(cd,b_2)} \tag{9}$$

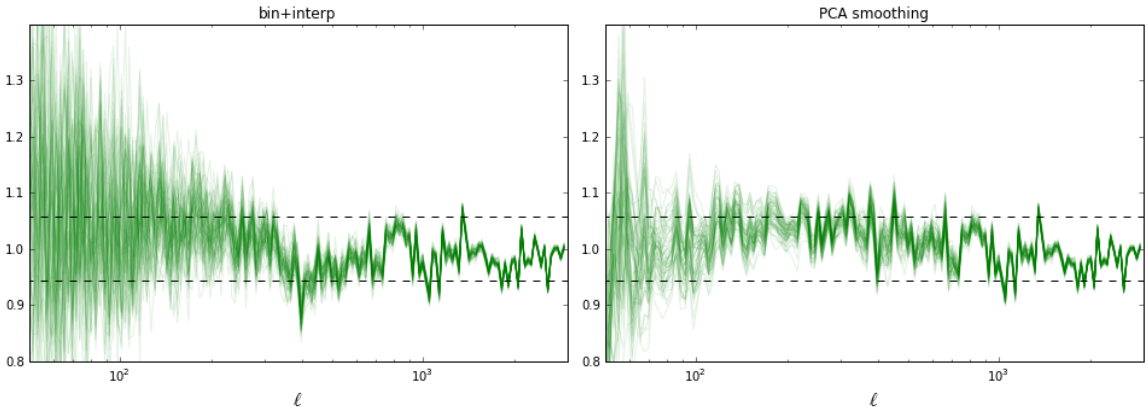
3 Mspec CTP3 Simulations Results



Because Mspec is an analytic likelihood which needs to invert the mode coupling matrix, its essential to have an apodized mask. Thus, instead of the "cb6_gal" mask provided with the CTP3 sims, we use this custom mask which is an apodized version of George's "mask_1.fits" and Mathiew Tristram's apodized point source mask.



The mean of the 1000 signal estimates divided by the fiducial, confirming that the estimator is unbiased.



These plots show the ratio of the Mspec error bars to the std-dev of the estimator from simulations. Each green line shows the error bars as computed from one realization, since Mspec uses the data itself to compute the covariance. Dashed lines show the expected spread due to the finite number of Monte Carlo realizations. The two plots show two methods for computing the fiducial power spectrum which forms the covariance. On the left, we bin the data then interpolate between the bins, whereas on the right we use the PCA based smoothing. We expect the analytic error bar approximation to fail for low ℓ, and this shows that with the PCA smoothing, we can trust the error bars above roughly $\ell = 70$.

4 Documentation

Mspec is four part process, each a suggestively named Python script.

1. `mask_to_mll.py` - Given a fits mask, computes the coupling matrix and inverse of the mask and square of the mask.

2. `maps_to_pcl.py` - Given a folder full of fits maps, applies a mask and computes all the possible auto and cross power spectra and save them in some folder.

3. `pcl_to_signal.py` - Given the pseudo power spectra and the mode coupling matrix, computes the signal estimator and its covariance.

4. `signal_to_params.py` - Given the signal and covariance, run an MCMC chain for parameters.

Except for the first one, the scripts are parallelized and can be called with `mpiexec`. Each script takes one argument, a parameter file.

4.1 Parameter File

maps The folder which contains the map fits files. Filenames are matched by a regular expression specified in **map_regex**.

map_regex (optional) A regular expression which matches valid map filenames and returns as groups the frequency and dectector id. The default is `(143|217|353).*?((?:[0-9][abc]?)|(?:ds[0-9]))` and matches file names like `HFI_143-1a_2048_20120114.fits`

pcls The folder where to store/read the pseudo spectra, one powerspectrum per file.

mask The fits mask. The mode coupling kernel files will be stored/read from files with the same name as this but with different postfixes.

beams The folder which contains the beams. File names are matched using **map_regex**.

signal The filename root for the signal estimator and covariance.

noise (optional) If running Mspec with auto-spectra included, this folder contains noise power spectra, matched using **map_regex**.

freqs (optional) Used by `maps_to_pcl` and `pcl_to_signal.py` to select only certain frequencies. This should be a valid Python expression for a list, for example `['143','217']`.

lmin Minumum ℓ, used *only* by `signal_to_params.py` when computing the likelihood.

lmax Used by all the scripts as the maximum ℓ. Power spectra, mode coupling matrix, and/or signal estimates can be computed to higher ℓ, then **lmax** can be decreased.

binning Which binning to use. Only the signal and covariance are binned. Options are:

None

WMAP WMAP binning at low-ℓ, SPT binning at high-ℓ.

CTP CTP binning

flat(x) Uniform bins of width **x**

get_covariance Whether `pcl_to_signal.py` should compute the covariance.

det_calibration Whether `pcl_to_signal.py` should do intra-frequency calibration.

freq_calibration Whether `signal_to_params.py` should do inter-frequency calibration.

cleaning Used by `signal_to_params.py` to do galactic cleaning. This should be a valid Python expression for a dictionary which maps "cleaned" power spectra to linear combinations of frequency power spectra (automatically normalized for the CMB). For example, `{'217c': [('217',1),('353',-.14)], '143c': [('143',1),('353',-.038)]}`

4.2 Utility functions

Aside from the four scripts, Mspec provides a number of utility functions to load and manipulate the intermediate data. Their documentation is contained in the Python docstring which is accessed from Python by typing "?" then the name of the function.

To load the utility functions, type `import Mspec as M`. Some notable functions are,

- `load_pcl`
- `load_signal`
- `load_beams`
- `get_bin_func`
- `PowerSpectra`
- `Chain`