CCA New York Workshop: SONG Exercise

Christian Fidler

July 15, 2019

1 The CMB Bispectrum

1.1 Running the code

Copy the input files ini/intrinsic.ini and pre/quick_song_run.pre. Use the new .ini file to adjust the verbosity of the code. Set all verbose parameters to 2 (or more?). Run the code and study the shell-output.

1.2 Increasing the precision

The quick_song_run.pre file has low accuracy and only computes multipoles up to $l_{\max} = 100$. Increase the k-sampling, time-sampling, $l_{\max}$, the multipole-cuts in the Boltzmann hierarchy and in the line-of-sight sources. Which parameters are particularly expensive to increase? Find a setting that runs in acceptable times on your machine.

1.3 The intrinsic bispectrum

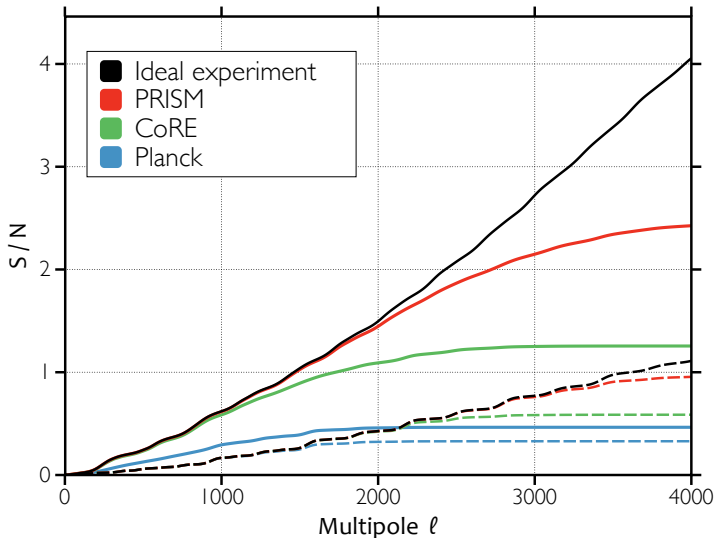
The code outputs the Fisher matrix. The intrinsic-intrinsic component is related to the signal-to-noise ratio of the intrinsic bispectrum $\frac{S}{N} = \sqrt{f_{ii}}$. Start computing the signal-to-noise using only the temperature bispectrum for scalar modes (modes_song = 0). Does the signal-to-noise improve when adding vectors and tensors? Does it increase when adding polarisation?

So far we were considering an ideal experiment that measures up to $l_{\max}$. How much does the signal-to-noise decrease for a PLANCK-like experiment.

The code also outputs the correlation matrix describing the similarity between the intrinsic bispectrum and various primordial templates. Add the orthogonal template to check if it is a good fit for the intrinsic bispectrum.

1.4 The Dependance on $l_{\max}$

In the output folder SONG creates the file fisher_lmax.dat. It contains the diagonal entries of the Fisher matrix as a function of $l_{\max}$. Visualise your results to analyse the dependance of the signal-to-noise on the maximum observed multipole.



1.5 Storing Runs

Running SONG on high precision setting can take considerable amounts of time. It can make sense to store intermediary results to be able speed up future runs. In your .ini file copy

```
store_run = yes
store_sources = yes
store_transfers = yes
store_bispectra = yes
run_directory = test_run
data_directory = \$SONG_RUN_DIR
append_date = yes
```

then run SONG again. In your SONG folder you should now have a folder called test_run with the current date appended. Inside are your ini- and pre-files plus binary files that store the sources, transfer functions and bispectra.

You can now repeat the run by executing

```
> ./song test_run_<date>/
```

The code will detect the stored files and finish significantly faster.

Deactivate storing the runs again, otherwise the code creates a lot of unneeded data (especially on higher precision settings).

2 The matter bispectrum

2.1 Visualising transfer functions

SONG can be used to compute and visualise any second-order transfer-function. We want to study the cold dark matter density for a given configuration of Fourier modes. In principle all the needed information is already contained in the binary files stored in the previous run. However, to make extracting it a little simpler SONG also contains options to write the needed data to file in human-readable format. In the .ini file add

```
k1_out =  0.05, 0.02
k2_out =  0.04, 0.02
k3_out =  0.03, 0.02
```

and run the code. Note that the values of k need to be given in descending order. The output folder now also contains the file perturbations_song_k000.txt and perturbations_song_k001.txt corresponding to the two selected configurations. Visualise the dark matter density as a function of time or redshift. What is the problem?

2.2 SONG in LSS-mode

The problem can be avoided by adding delta_cdm_bk to the requested outputs, or even better using the predefined matter.ini and matter.pre files that contain precision parameters optimised for quick runs related to matter perturbations. Can you think of other options to avoid the problem?

In addition to outputting a transfer-function as a function of time, SONG can also create fixed-time output as a function of k_3 by adding z_out or tau_out in addition to the k_out's. Visualise the data.