Second-order Boltzmann Code and CMB bispectrum

Zhiqi Huang
IPhT, CEA/Saclay
Collaborator: Filippo Vernizzi

September 11, 2012
COSMO12 @ Beijing

Outline

No Introduction (thanks to Christian)

Second-order Boltzmann Code
- Theory
- Code
- Testing the Code

CMB bispectrum
- Theory
- Code
- Testing the Code

Conclusions and Outlook

Second-order Boltzmann code

Perturbation Theory in General

1. Choose a **gauge**. *(Gauge transformation see e.g. Bruni et al 1997.)*
2. Choose a set of complete and independent **variables** and write down the dynamic **equations** for these variables. *(Choice of variables does not matter.)*
3. Formally **expand** each variable X as $X = X_0 + X_1\epsilon + X_2\epsilon^2 + \ldots$, where X_i is the i-th order perturbation ($i = 0, 1, 2, \ldots$) and ϵ is treated as an infinitesimal.
4. **Solve** the equations order by order in ϵ.

Theoretical works:

Bruni *et al* 97; Pitrou *et al* 09, 10; Beneke & Fidler 10; Christopherson *et al* 08, 09, 11; Bartolo 07, 11; Senatore 08; Nitta *et al* 09; Khatri *et al* 09; Creminelli, Pitrou, and Vernizzi 11; Lewis 12 (squeezed-limit, can be used to test the codes) ...

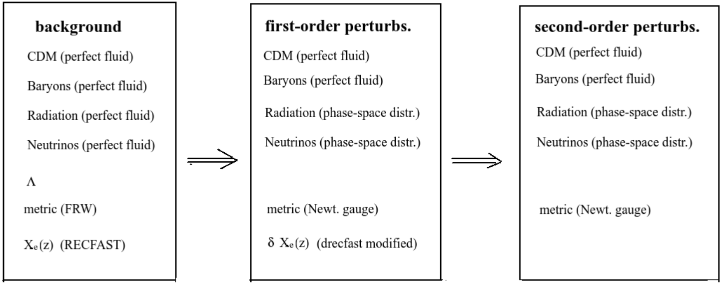
A little bit problematic? ...

- Equations are big. Typos and erros are in common. Discrepancies exist between different papers and different arXiv versions of each paper.
- So far only one public available code CMBquick (by Cyril Pitrou). Bugs have been found in V1.0, V2.0, V3.0 ...

The solution

- Use the idea of "machine proof": if the code passes all nontrivial tests with extremely high precision, the equations used in the code are less likely to be wrong.
- Write more codes and compare between each other.

The structure of the code: CosmoLib++

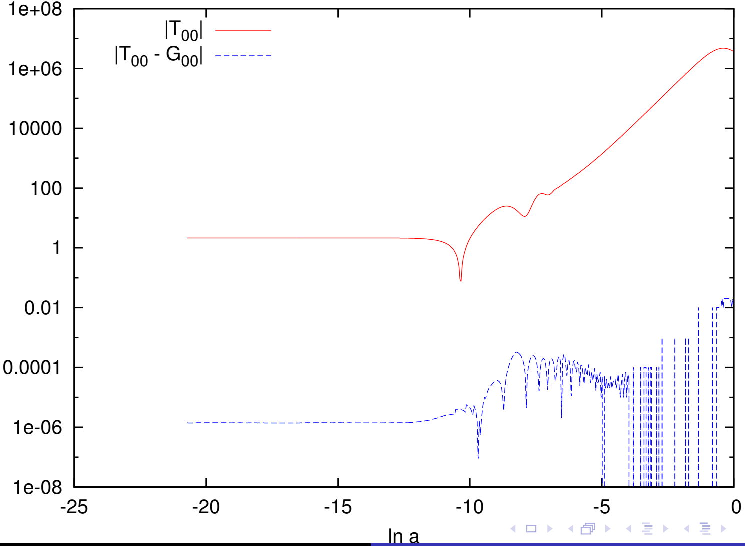


CPU time	~ 0.1 sec	~ 10 sec	~ 10 hrs
Memory cost	~ MB (saving everything)	~ GB (saving everything)	~ GB (saving only sources)

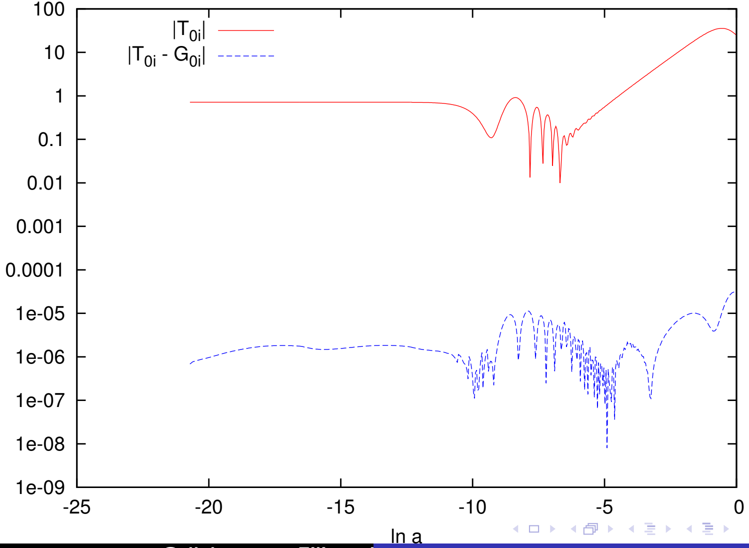
Comparison with CMBQuick by Cyril Pitrou

	CMBquick	CosmoLib++
language	Mathematica	Fortran (mixed with C)
status	released (v3.0)	coming by the end of this yr.
2nd-order Boltzmann code	scalar + vector + tensor	scalar + vector + tensor
2nd-order Einstein equations check	relative error < 0.1	relative error $< 10^{-6}$
Boltzmann code squeezed limit check	$\surd$	$\surd$
bispectrum ignoring lensing	flatsky approx.	fullsky exact
bispectrum squeezed-limit check	$\surd$	$\surd$
lensing bispectrum	N.A.	N.A.

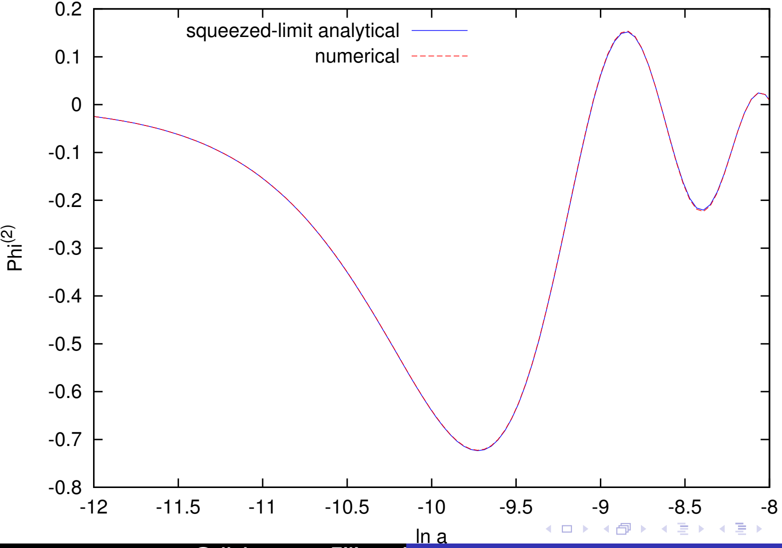
Einstein equations: energy constraint



Einstein equations: momentum constraint

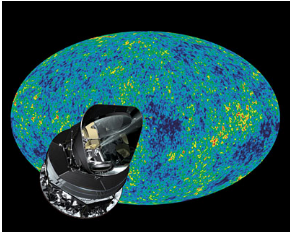


Squeezed limit



CMB bispectrum

Bispectrum due to GR nonlinearity



- Current limit $-10 < f_{NL} < 74$ (WMAP7)
- Planck is expected to measure $\Delta f_{NL} \sim 3 - 5$.
- Naively expect an effective $f_{NL} \sim O(1)$ – predictions in the literature vary from -3.5 to 1.
- In general case need an accurate code to do a full calculation.

The bispectrum: definition

$$\frac{\Delta T}{T}(\mathbf{n}) = \sum_{l,m} a_{lm} Y_{lm}(\mathbf{n}) \,.$$

$$\langle a_{l_1 m_1} a_{l_2 m_2} a_{l_3 m_3} \rangle = B_{l_1 l_2 l_3} \begin{pmatrix} l_1 & l_2 & l_3 \\ m_1 & m_2 & m_3 \end{pmatrix} \,.$$

$$B_{l_1 l_2 l_3} = b_{l_1 l_2 l_3} \sqrt{\frac{(2l_1+1)(2l_2+1)(2l_3+1)}{4\pi}} \begin{pmatrix} l_1 & l_2 & l_3 \\ 0 & 0 & 0 \end{pmatrix}$$

A difficult integral

Primordial bispectrum $\Rightarrow$ CMB angular bispectrum

$$b_{l_1 l_2 l_3} = \int dx dk_1 dk_2 dk_3 (x k_1 k_2 k_3)^2 B_{\rm prim}(k_1, k_2, k_3) \Delta_{l_1}(k_1) \Delta_{l_2}(k_2) \Delta_{l_3}(k_3) j_{l_1}(k_1 x) j_{l_2}(k_2 x) j_{l_3}(k_3 x)$$

Solution: Reduce the 4D integral into products of 1D integrals by factorizing the primordial bispectrum $B_{\rm prim}(k_1, k_2, k_3) = \sum_i X_i(k_1) Y_i(k_2) Z_i(k_3)$.

See e.g. Fergusson *et al.* 09.

A more difficult integral

GR nonlinearity ⇒ CMB angular bispectrum

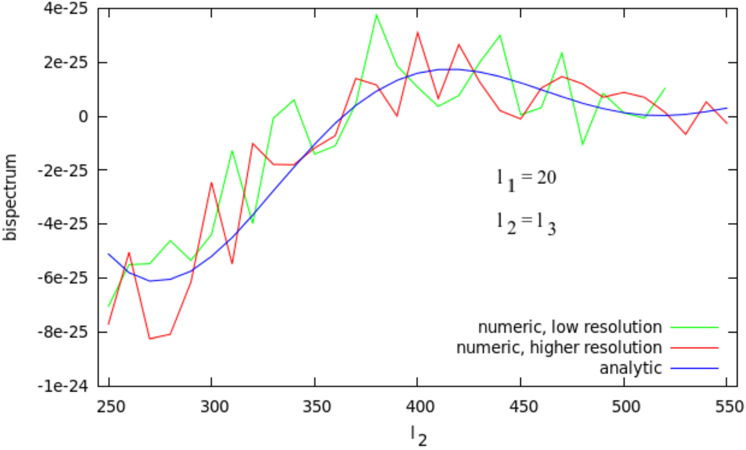
$$b_{l_1 l_2 l_3} = \sum_{m_1, m_2, m_3} \int dk_1 dk_2 d\mu d\eta Y^*_{l_1 m_1}(\theta_1, \pi) Y^*_{l_2 m_2}(\theta_2, 0) S_{l_3' m_3}(k_1, k_2, \mu, \eta) j_{l_3}^{l_3' m_3}[k(\eta_0 - \eta)] \ldots + \text{perms.}$$

- The factorization method is not necessarily a winner (need to factorize at each time step).
- Evaluation of billions of spherical harmonics and a lot of 3-j symbols.
- Evaluation of all $j_{l_3}^{(l_3' m_3)}$ is numerically expensive.
- Too many lensing source terms (need to compute source up to $l_3' \sim l_3$.)

Bispectrum integrator

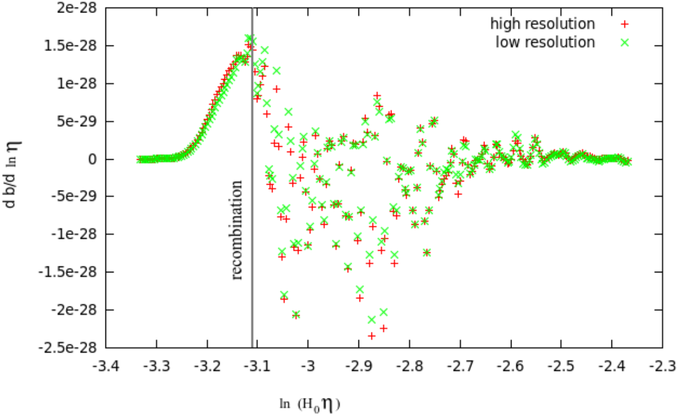
- Brute-force 4D integral
- Quick evaluations of Y_{lm} and 3-j symbols. Evaluation of $j_{l_3}^{(l_3' m_3)}$ is avoided by integrating by part.
- Computing and saving the line-of-sight sources takes ~ 10 CPU hours (parallelizable).
- Each evaluation of $b_{l_1 l_2 l_3}$ takes about 1 CPU hour (parallelizable).

Bispectrum: squeezed-limit test



Model: no DE, no reionization (easier to compare with theoretical predictions)

Understanding the integral better



- Main contribution is from last scattering surface (no DE, no reionization and ignoring lensing)
- Need to make the resolution (much?) higher to get more accurate results.

Conclusions and Outlook

- Second-order Boltzmann code $\surd$
- Bispectrum integrator ignoring lensing contribution $\surd$ (but need to be more accurate)
- Lensing (future work)

谢谢

Merci!

Thanks!