Master in High Performance Computing

Computational Challenges in Cosmological Tests of Gravity

Supervisors:
Prof. Luca Heltai,
Prof. Carlo Baccigalupi

Candidate:
Marco Raveri

1st edition
2014–2015

Contents

Chapter 1

Collaborations

This thesis contains some of the result of the research done during my four years of Ph.D. and work done within the Master in High Performance Computing. The codes developed and improved during the Master in High Performance Computing was used in the following papers published in refereed Journals:

1. Bin Hu, Marco Raveri, Matteo Rizzato, Alessandra Silvestri,
   "Testing Hu-Sawicki $f(R)$ gravity with the Effective Field Theory approach",
   Monthly Notices of the Royal Astronomical Society (in press) [arXiv:1601.07536 [astro-ph.CO]] [1].

2. Noemi Frusciante, Marco Raveri, Daniele Vernieri, Bin Hu, Alessandra Silvestri,
   "Horava Gravity in the Effective Field Theory formalism: from cosmology to observational constraints",
   Phys. Dark Univ. **13**, 7 (2016) [arXiv:1508.01787 [astro-ph.CO]] [2].

3. Bin Hu, Marco Raveri, *"Can modified gravity models reconcile the tension between CMB anisotropy and lensing maps in Planck-like observations?"*,
   PRD **91**, (2015) 123515 [arXiv:1502.06599 [astro-ph.CO]] [3].

4. Bin Hu, Marco Raveri, Alessandra Silvestri, Noemi Frusciante, *"EFT-CAMB/EFTCosmoMC: massive neutrinos in dark cosmologies"*,
   PRD **91**, (2015) 063524 [arXiv:1410.5807 [astro-ph.CO]] [4].

5. Marco Raveri, Bin Hu, Noemi Frusciante, Alessandra Silvestri,
   "Effective Field Theory of Cosmic Acceleration: constraining dark energy with CMB data",
   PRD **90** (2014) 043513 [arXiv:1405.1022 [astro-ph.CO]] [5].

6. Bin Hu, Marco Raveri, Noemi Frusciante, Alessandra Silvestri,
   "Effective Field Theory of Cosmic Acceleration: an implementation in CAMB",
   PRD **89** (2014) 103530 [arXiv:1312.5742 [astro-ph.CO]] [6].

In addition it was used in the following papers in consideration for publication in refereed Journals:

1. CMB-S4 collaboration,
   "CMB-S4 Science Book, First Edition",
   arXiv:1610.02743 [astro-ph.CO] [7].

2. Simone Peirone, Marco Raveri, Matteo Viel, Stefano Borgani and Stefano Ansoldi,
   "Constraining f(R) Gravity with Planck Sunyaev-Zel'dovich Clusters",
   arXiv:1607.07863 [astro-ph.CO] [8].

3. Marco Raveri, Matteo Martinelli, Gongbo Zhao, and Yuting Wang,
   "Information Gain in Cosmology: From the Discovery of Expansion to Future Surveys",
   arXiv:1606.06273 [astro-ph.CO] [9].

Supplementary material includes:

- EFTCAMB/EFTCosmoMC codes available at `http://eftcamb.org`
- Bin Hu, Marco Raveri, Noemi Frusciante, Alessandra Silvestri
  "EFTCAMB/EFTCosmoMC: Numerical Notes v2.0",
  arXiv:1405.3590 [astro-ph.IM], (2014) [10].

Chapter 2

Abstract

Explaining the physical origin of cosmic acceleration still poses a challenge to modern cosmology. On one hand, observational evidence corroborating this phenomenon is compelling and continuously becoming stronger and stronger. On the other hand a physical explanation for it is still missing.

To address this issue many models have been proposed and testing their phenomenology against the growing amount of precise cosmological data has become a challenge.

In this thesis we develop the efficient, high performance, computational tools to do that. We implement the Effective Field Theory approach to cosmic acceleration in the public Einstein-Boltzmann solver CAMB and we dub the resulting code EFTCAMB.

We discuss its architecture and performances and overall we show that, in its present version, EFTCAMB stands as a powerful and versatile tool that can be used to study both model independent and model dependent approaches to the problem of cosmic acceleration.

Chapter 3

Introduction

We are witnessing the dawn of a golden era in cosmology. Ongoing and upcoming experiments will map cosmic structures over a significant fraction of our Universe, providing us with extremely accurate measurements. Observational collaborations are driving the community toward this objective, working relentlessly to reach such accuracy. On the theoretical side, many questions still need to be answered and these measurements will offer a unique opportunity to tackle them.

In this thesis we focus on the computational aspects of one of these questions.

A universe described by General Relativity and filled with ordinary matter is naturally expected to decelerate, after the initial phase of rapid expansion following the Big Bang. In this respect, an important breakthrough occurred in the late 90s. Measurements of Type Ia Supernovae by the Supernova Cosmology Project [11] and the High-Z Supernova Search Team [12] provided strong evidence that the Universe has recently entered a phase of accelerated expansion. This discovery was later corroborated by many independent probes. All these observations converged in confirming this description of our Universe, and nowadays cosmic acceleration is a well established fact and a cornerstone of our standard cosmological model. Yet theoretically we are struggling to find an explanation for its physical origin. To make the expansion of the Universe accelerate, we need an unnaturally small amount of vacuum energy, or an additional ingredient whose nature is unknown, Dark Energy; or we might have to change our theory of gravity on cosmological scales.

To address the problem of cosmic acceleration, in both these directions, models have been proposed in a number that far exceeds our capability of practically testing them. From this perspective, hope comes from unifying frameworks that allow to enclose and study many of these models with the same set of tools.

In this thesis we discuss the implementation of one of these frameworks in the numerical tools that are used to test models against cosmological data. Furthermore we focus on the computational challenges that we faced in optimizing these tools to allow systematic and massive explorations of the space of models that aim at addressing the problem of cosmic acceleration.

Chapter 4

Efficient Prediction of Cosmological Observables

Anticipating a wealth of high precision large scale structure data from ongoing and upcoming cosmological surveys it is important to identify a model-independent way of testing theories of gravity against observations.

To this extent, over the past years there has been a lot of activity in the cosmological community to construct frameworks [13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58] that would allow model-independent tests of gravity. These are generally based on parametrizations of the dynamics of linear scalar perturbations, either at the level of the equations of motion, e.g. [46], of solutions of the equations, e.g. [18, 28, 43], or of the action, e.g. [59, 60, 61], with the general aim of striking a delicate balance among theoretical consistency, versatility and feasibility of the parametrization.

In this thesis we focus on a recent proposal which applies the effective field theory (EFT) formalism to the phenomenon of cosmic acceleration [60, 61, 62], inspired by the EFT of Inflation and Quintessence [63, 64, 65, 59, 66, 67, 68, 69, 70].

We refer the reader to [60, 71, 72] for a detailed discussion of the assumptions and limitations of this framework.

Despite of the inherent limitations, the EFT framework includes most of the viable approaches to the phenomenon of cosmic acceleration that will undergo scrutiny with upcoming cosmological surveys. We shall mention, among others, the Horndeski class which includes quintessence, k-essence, $f(R)$, covariant Galileon, the effective $4D$ limit of DGP [73], Hořava gravity and more. From this perspective the EFT formalism can be used as a general, model-independent, framework to efficiently test theories of gravity with cosmological observations: studying the phenomenology of different models and the constraints that can be put by data on their model parameters.

4.1 Computational challenges

The development of unifying frameworks for theories of gravity and the availability of large, very precise, data sets present us with several computational challenges.

To test a model against data one has to compute the predictions for what the data would look like within the given model. Such cosmological predictions are then compared to data to estimate the likelihood of the model. This process is then repeated several times, for different choices of the model parameters, to perform parameter estimation or model selection [74].

The parameter space of a model is sampled using Markov Chain Monte Carlo (MCMC) algorithms [75, 76, 77, 78] that require several thousands computations of cosmological predictions. Even when using refined sampling techniques, that allow to sample the parameter space of a model more efficiently [79], getting cosmological predictions in more than ten seconds, by all means necessary, makes it impossible to practically test a model against data.

The computational challenge, within the framework of cosmological tests of gravity, is then that of computing all relevant cosmological predictions, for as many models as possible, in the fastest possible way that has not to exceed the timings dictated by the practical feasibility of data comparison.

For the fiducial cosmological model, and some variations of it, the standard community tool to compute cosmological observables is the *Code for Anisotropies in the Microwave Background* (CAMB) [80]. This can compute the expected values of all the observables of interest in an efficient and fast manner. This code is written in *Fortran* and exploits shared memory multiprocessing programming to achieve good performance boosts benefitting from parallelism.

To sample the parameter space of the models covered by the CAMB code this is interfaced with an optimized MCMC sampler, *Cosmological Monte Carlo* (CosmoMC). In its default settings CosmoMC uses the Metropolis algorithm [75] or an optimized fast-slow sampling method [79] that allows to efficiently explore many nuisance parameters related to data likelihoods. The CosmoMC code exploits MPI (Message Passing Interface) parallelism to speed up computations. If the CosmoMC code is compiled and run with MPI there is the possibility to dynamically learn the sampling proposal matrix from the covariance of the post-burn-in samples so far. This results in a dramatic increase in sampling performances and is only possible with parallel calculations as they provide a way of estimating the length of the burn-in phase.

To analyze the results of the parameter space sampling the CosmoMC code provides a Python package, *GetDist*, to estimate parameters bounds and plot the parameters probability distributions.

4.2 EFTCAMB

We implemented the EFT approach to DE/MG in the public Einstein-Boltzmann solver CAMB. The resulting code, which we dub EFTCAMB, is a powerful and versatile tool that can be used for several objectives. It can be employed to evolve the full dynamics of linear scalar perturbations of a broad range of mod-

els, once the model of interest is mapped into the EFT formalism. It offers a numerical implementation of EFT as a model-independent framework to test gravity on cosmological scales.

In this section we briefly review the key features of the implementation of the EFT formalism in CAMB.

4.2.1 Code structure

The structure of the EFTCAMB code resembles the logical structure of the different families of models implemented in the code. With code updates the structure of the code changed, growing in complexity while model coverage increased.

In this section we shall briefly outline the structure of EFTCAMB V3.0 as shown in Figure A.3.

By acting on a series of numerical flags the user selects the model that is being tested and a number is associated to each model selection flag. Such number is reported in Figure A.3 and controls the behavior of the code. The main code flag is `EFTflag`, which is the starting point after which all the other sub-flags, can be chosen according to the user interests.

- The number `EFTflag` = 0 corresponds to the standard CAMB code. Every EFT modification to the code is automatically excluded by this choice.

- The number `EFTflag` = 1 corresponds to *pure* EFT models. In these models the EFT framework is used to perform parametrized tests. The user needs to select a model for the various functions that are parametrized in the code by acting on the `EFTwDE`, `PureEFTmodelOmega` and `PureEFTmodelGammai` (with i = $1, .., 6$) flags. Various common parametrizations for such functions are natively included in the code. After setting these flags the user has to define the values of the EFT model parameters for the chosen model. Every other value of parameter and flag which do not concern the chosen model is automatically ignored.

- The number `EFTflag` = 2 corresponds to the designer *mapping* EFT procedure. In this case portions of the model are treated in a parametrized way and other aspects are fixed by the model itself. For the *mapping* case the user can investigate a particular DE/MG model once the matching with the EFT functions is provided and the background evolution has been implemented in the EFT code.
  The model selection flag for the *mapping* EFT procedure is `DesignerEFTmodel`. Various models are already included in the code, see the blue lines in the Flowchart A.3). Models corresponding to the grey lines in the Flowchart A.3 are some of the models that can be cast into the EFT formalism and that will be gradually implemented in future code releases.

- The number `EFTflag` = 3 corresponds to the implementation of alternative model-independent parametrizations in terms of EFT functions. A lot of alternative parametrizations already present in literature can be completely described by using the versatility of the EFT approach allowing to

preserve all the advantages of EFTCAMB. The first reparametrization implemented is the one proposed in [58], which is a built-in feature of the v2.0 release. Hereafter we will refer to this parametrization as ReParametrized Horndeski (RPH) as it is a model-independent parametrization of Horndeski theory in terms of five functions of time defined in such a way that they correspond to specific physical properties of the scalar d.o.f..

- The number `EFTflag` = 4 corresponds to the full *mapping* EFT procedure. In this case a model is fully mapped to the EFT framework and the code produces exactly the cosmological predictions of the given model. Low energy Hořava gravity has been included as the first example of the implementation of full mapping models. More models will be gradually filled in the near future.

For further details about the different running modes of the code we refer the reader to the appropriate sections of [10].

4.2.2 Code documentation

In order to implement the EFT formalism in the CAMB code we had to add and modify several code files. In addition the code structure of EFTCAMB v3.0 is such that when adding a new model only one code file has to be added to the distribution. To allow a comprehensive and always up to date documentation of the EFT part of the code we implemented, starting from v3.0, an automatic documentation generated with Doxygen `http://www.stack.nl/~dimitri/doxygen/`.
All the functions, types and modules in the EFTCAMB code are documented and the code documentation is available on the EFTCAMB website for the stable release. The documentation of the EFTCAMB developer's version is built every time the code successfully passes continuous integration tests and available at `https://eftcamb.github.io/EFTCAMB/`.

To further help the user in understanding our part of code and/or applying the EFT modification to an already modified version of CAMB we enclosed every modification that we made inside the following commented code lines:

```
! EFTCAMB MOD START
...
! EFTCAMB MOD END
```

In order to help the user in understanding the physics at the basis of EFTCAMB we provide a guide to the physical and technical details of the code with [10]. We shall refer to this guide as the *EFTCAMB Numerical Notes*. In the *EFTCAMB Numerical Notes* we reproduce, as they appear in the code, the complete set of the modified equations and the expressions for all the other relevant quantities used to construct the EFT modification of CAMB. We submit these notes to the arXiv to grant full and permanent access to this material which provides very useful guidance to the numerical implementation of the EFT framework. The arXiv version of the *EFTCAMB Numerical Notes* is updated at every relevant update of the stable version of the code. The version of

the *EFTCAMB Numerical Notes* following the developer's version is available at `https://github.com/EFTCAMB/NumericalNotes`.

4.2.3 Scalability in model space

One of the most challenging phases in the program of scaling by orders of magnitude the number of gravity models tested against data is that of model implementation. Even if the flexibility of the EFT approach is such that the effort to get the phenomenology of a model is minimal, it still requires the user to write several properties of the model into the code. In addition different models might suffer from different numerical problems. For example, in Hořava gravity, the coefficient of the scalar field equation suffer from cancellation errors. This makes the whole code unstable and the only way out is to have these coefficients rewritten for the specific model.

In the v2.0 version of EFTCAMB implementing a new model was a demanding task. Several files had to be modified, in different places, and this required the user to have a global view of the structure of the code. Moreover most of the model specific choices, i.e. compute the scalar field equation coefficients from ad-hoc expressions, were hard coded, resulting in complicated and nested logical structures, within the code. All these factors were limiting the scalability of the EFTCAMB code in model space. Further implementing models was becoming more complicated as the number of model grew.

To address this problem we completely rewrote the code changing its structure. In doing so we used Object Oriented Fortran constructs to implement several layers of abstraction.
The first, and most abstract, layer is represented by EFTCAMB itself. One object is responsible for holding all the EFTCAMB model specification, to initialize the model choice and provide the implementation of all EFTCAMB specific procedures. The second layer of abstraction is represented by models. All the calculations that EFTCAMB is modifying, within CAMB, are written for an abstract model, using the EFT expressions. Different models are inherited from the abstract one and have to override and implement a small subset of the model specific procedures. The implementation of some of the procedures, the one defining the specific model, is forced at compile time to ensure strict compliance with the EFTCAMB structure. In case of some numerical problems arising in one of the computations, and detected by means of the EFTCAMB debug tools, the user can override the procedure responsible for the calculation, implementing a numerically stable version of that same calculation. The main EFTCAMB object contains a model object that is allocated at run time. With this structure, if the user wants to implement a new model, he/she has to write a single file with the model object, inheriting it from the abstract model object, and does not have to be aware of the full structure of the EFTCAMB code. In addition the user has to include its new model in the logical structure of the code. The third layer of abstraction is represented by parametrized functions. In the EFT framework several functions can be parametrized in different ways. In v2.0 every time a parametrization was used it had to be written explicitly, resulting in code duplication. In v3.0 all parametrized functions are inherited from an abstract object and, when present in a model, allocated at run time. To implement a new parametrization in a model only requires the effort of imple-

menting the functional form, as inherited from abstract parametrized functions, and modify the allocation phase of a model to include this choice.

All these modifications included in v3.0 are meant to simplify the process of model implementation so that its difficulty does not depend on the number of models included in the code and, as such, scales well in model space. The reliability of the structure was tested when all the model included in the 2.0 version of the code had to be included in v3.0.

4.2.4 Test suite

To aid the safe development of the EFTCAMB code, starting from v2.0, we implemented a test suite and in v3.0 we improved many of its aspects. The aim of the test suite is twofold. First help developers in discovering errors in the code that they are developing, second ensure that results do not change over time and that the code maintains consistency.

Since the EFTCAMB code is a modification of another code that is itself long we could not implement complete unit testing. This was also not useful in case of the EFTCAMB specific functions as all core algorithm come from legacy, well tested, code. The test suite was then designed to compute all cosmological observables of interest for many different models, store the results and compare them with previous trusted ones. The comparison phase is based on *numdiff* `http://www.nongnu.org/numdiff/`, with relative and absolute accuracy requirements set by the output format of the EFTCAMB code results. If the *nudiff* test fails a python plotter compares the two results and provides the user with a handy way of interpreting discrepancies between the new version of the code and the trusted one. Plotting the difference between the two results usually helps developers in identifying the physical effect that is responsible for the unexpected behavior. An example of the output of the test suite, for a model not passing the test, is shown in figure A.4.

New tests can be easily added by adding a new parameter file in the parameters directory. Trusted results are stored in a separate *git* repository, available at `https://github.com/EFTCAMB/EFTCAMB_legacy`, to ensure that the main EFTCAMB distribution remains lightweight.

The test suite is used by continuous integration services to monitor the working status of the code.

4.2.5 Benchmarks and performances

Code performances are a limiting factor to the scaling in model space of cosmological test of gravity and better performances directly translate into more models that we can test.

To evaluate the performances of the EFTCAMB code, starting from v2.0, we introduced a quality benchmarker and two profiler. The benchmark program behaves as the normal EFTCAMB program and accepts all input options. The difference is that after reading the input parameters the code runs all calculations that it is requested to perform, a number of times (by default ten and can be changed by command line), measures the time taken, with an *OpenMP* compatible timer, and computes the average time of execution and its variance.

Since in all practical applications the output phase is not the limiting factor in performances it is not included in the time estimate.
The default number of calculation execution is chosen to have reliable estimates of the execution time variance that in turn is needed to quantify whether fluctuations in performances are statistically significant.
When running full benchmarks, the code is set up to evaluate the performances for all the models included in the test suite. A series of python scripts is then used to interpret the benchmark results.
Since the EFTCAMB code is a modification of the CAMB code, its performances are measured in units of the standard code run times. This makes the benchmark results stable against multiplicative biases, i.e. running benchmark on a different machine, with different performances but similar architecture, will likely result in statistically compatible results. In the full benchmark run the code evaluates the performances of the CAMB code at the beginning and at the end. This allows the python analysis programs to asses whether there has been a significant drift in performances during benchmark execution. This situation commonly arises when running benchmarks on portable computers that do not guarantee constant machine performances due to energy management considerations. If a statistically significant drift in performances is detected across a single benchmark run the benchmark results are discarded as not meeting quality requirements.

The profiling part of EFTCAMB is a slight modification of the benchmarking one. Both profiler consist in the benchmarking application compiled with different flags to allow different profilers to interpret the results. The first one is compiled to interface with the *gprof* `https://sourceware.org/binutils/docs/gprof/` profiler while the second works with the *vtune* profiler `https://software.intel.com/en-us/intel-vtune-amplifier-xe`. When running systematic profiling, the code would run the profiler and *gprof* for all the models in the test suite.

In EFTCAMB v3.0 we introduced continuous performance monitoring of the EFTCAMB developer's version. Full code benchmarks are run daily on a cluster node, that provides a controlled environment, and the results are saved in the EFTCAMB legacy repository with their time stamp. A set of dedicated python scripts then takes care of plotting the results and compare the performances of different versions of the code. Full code profiling, with *gprof*, is instead run weekly as it is much more time consuming. Also in this case the results are saved in the EFTCAMB legacy repository. If the git version of the code did not change since the last results were generated, the code is not run. In the near future we shall include the automatic generation of a web-page summary of the performance of the code.

In the remainder of this section we shall describe the evolution of the code performances and the optimization steps that we followed in going from v2.0 to v3.0. The benchmark results and the speed comparison of all the steps in the process of code optimization are reported in Appendix A.0.3. We highlight that the date of the results do not reflect the actual time interval between different revisions. A global review of the evolution of code performances is shown in figure 4.1.

The first results correspond to the v2.0 of the EFTCAMB code. As we can see in figure A.5 performances are subdivided in three groups: *Pure EFT*

models, labeled with a 2 in front of the name, are roughly two times slower than the standard CAMB code; Reparametrized models, indicated with a label 3 are a factor three to seven slower than CAMB; Designer f(R) models, indicated with 4 are a factor three slower.
This behavior is due to the fact that the implementation of these models differs in many respects. In all models the physical calculations that the code has to perform are more complicated, with respect to the CAMB code, and so the code is slower. *Pure EFT* models are the closest to the EFT framework, all equations are simpler in this family of models, and, as such, benefit more from compiler optimization. Reparametrizations need an additional step to be mapped to the EFT framework and are the slowest. In designer models instead the functions specifying the model are precomputed and interpolated rather than computed at run time and the code is slower than the *Pure EFT* case. In addition in designer f(R) models the physical equations that the code is solving are badly behaved and thus require more calculations and time to achieve a solution. Optimizing this step would imply deeply modifying the architecture of the CAMB code and this is beyond the scope of this work.

We then implemented the changes defining the third version of EFTCAMB and discussed in the previous section. In addition, to improve performances we introduced caching mechanisms to save intermediate results to be exploited at later stages of computations. After these changes the code was on average slower than its previous version, as can be seen in figure A.6. This average degradation in performances is due to the fact that equations were previously hard coded in the CAMB code and could be optimized by the compiler. In the latest version they are hidden by the layers of abstraction discussed in the previous section and thus the compiler optimization capabilities are limited. Reparametrized Horndeski models are the exception to this trend as, exploiting caching mechanisms, have now performances that are in line with *Pure EFT* models.

We started the optimization phase by profiling the code with *gprof*. This highlighted some time consuming functions that we optimized. In doing so we reshuffled the mathematical expressions of the EFT formalism to perform the least number of operations in two different functions. This resulted in an average five percent improvement, as shown in figure A.17, and ten percent improvement, as shown in figure A.18. The statistical significance of these improvements is however limited by the benchmarker errors. In this phase we also faced some of the limitations of the *gprof* profiler. The result of the profiling runs was, in fact, showing that most of the time was spent on a long function of the original CAMB code (*derivs*), without any indication of the time consuming regions inside that function.

We turned to the *vtune* profiler that allows to explore the performances of single code lines evaluating the quality of the time spent there. This located two main bottleneck in the calculations, that could be easily solved, and that resulted in an average fifteen to twenty percent performance improvement, as shown in figure A.21. Only designer f(R) models did not benefit from this improvement and we run dedicated profiling, with *vtune*, for these models only. This allowed to optimize a time consuming part of interpolation, by means of caching, resulting in ten to twenty percent performance improvements, as shown in figure A.22.

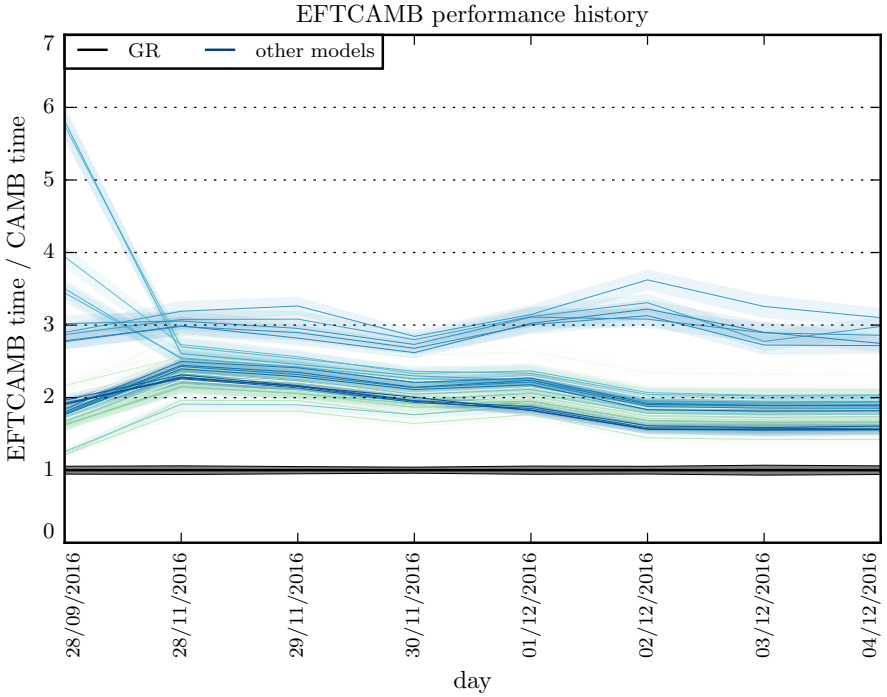

Figure 4.1: EFTCAMB performance evolution. The continuous black line correspond to the CAMB code that sets the scale. All the other colors correspond to the models included in the test suite. For all models the shaded area represent the 68% confidence region.

At last we tested the dependence of the code on some compiler optimization flags and found no statistically significant difference, as shown in figure A.24.

To have a measurements of the overall performance evolution between EFTCAMB v2.0 and v3.0 we compare the benchmark results of the last v2.0 version and the last revision of v3.0 as shown in figure 4.2. As we can see overall most of the models have constant performances while Reparametrizations have a significant boost in performances. Full mapping models like Hořava (indicated with the number 5) are twenty percent faster than the previous version.

We stopped optimizing the code after profiling it with the *vtune* profiler that showed that the performance penalty of EFTCAMB with respect to CAMB is now only due to physical reasons whose optimization would require significant architecture changes in the CAMB code.

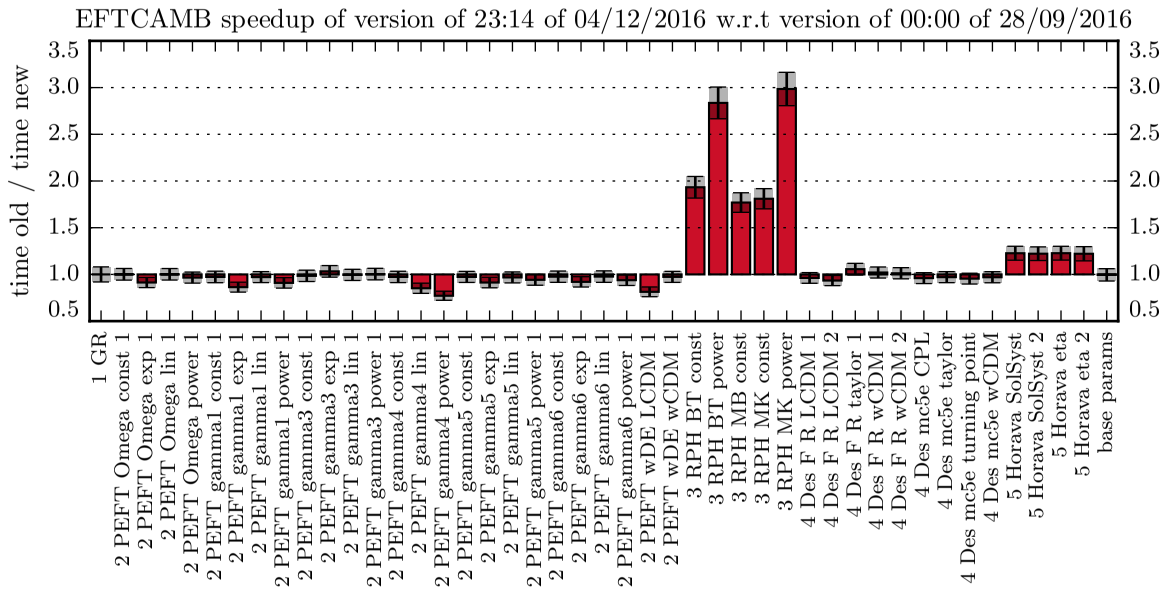

Figure 4.2: EFTCAMB speedup comparison. The timings of the EFTCAMB v2.0, relative to CAMB, are compared with timings of the last revision of EFTCAMB v3.0. Values above one mean that v3.0, for the given model, is faster than v2.0. The red bars correspond to the models included in the test suite and the shaded area represent the 68% confidence region.

4.2.6 Parallel scalability

To improve the time to solution of CAMB, the code has been parallelized with shared memory multiprocessing programming and with the *OpenMP* library. In developing EFTCAMB we did not change this architectural choice and tried to respect it as much as possible.

We use the EFTCAMB benchmarker, discussed in the previous section, to study the parallel scalability of the code.
In figure 4.3 we show the strong scalability of the CAMB and EFTCAMB codes and in figure 4.4 the weak one. We studied the scalability of the code in the regime that is commonly used in parameter estimation. Usually the CAMB code is rarely used on more than four cores and almost never used on more than eight. The reason for this is simple. Testing many models or testing different experimental combinations has perfect parallel scaling and can be easily distributed on many machines. The parallel configuration with four to eight cores per CAMB run ensures that the final results of parameter estimation can be achieved in one or two days per model or experimental configuration.
For this reason the code has been optimized to scale well in this regime. As we can see, however, it loses some efficiency for high number of cores. This loss in performance is due to serial segments of the code. These take care of initializing the calculations and performing some preparatory ones. Within EFTCAMB we need to perform additional initialization to allocate the model of choice, study its theoretical viability and understand whether it is very close to the GR limit. These potentially pose a threat to the parallel scalability of the code: they depend on each other and have to be executed one after the other in a specific order. For this reason, in EFTCAMB v3.0, we optimized them in such a way that they are much faster than the rest of serial initialization and thus do not impact the scalability of the code. As we can see from figures 4.3 and 4.4 we achieved this goal and all the EFTCAMB model tested are distributed around, and within the error bars, of the CAMB code. As we can see in the two figures, there is a class of models that scales better. This correspond to the family of the slowest models. There the parallel region takes longer to execute so that the serial part has less impact on the scalability of the code.

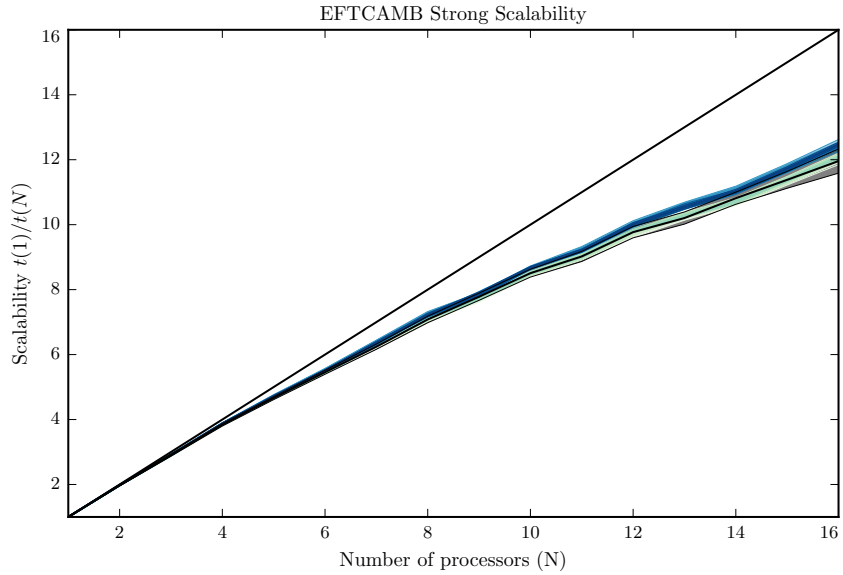

Figure 4.3: EFTCAMB strong scalability results. The dashed black line corresponds to perfect scalability. The continuous black line correspond to the scalability of the CAMB code. All the other colors correspond to the models included in the test suite. For all models the shaded area represent the 68% confidence region.

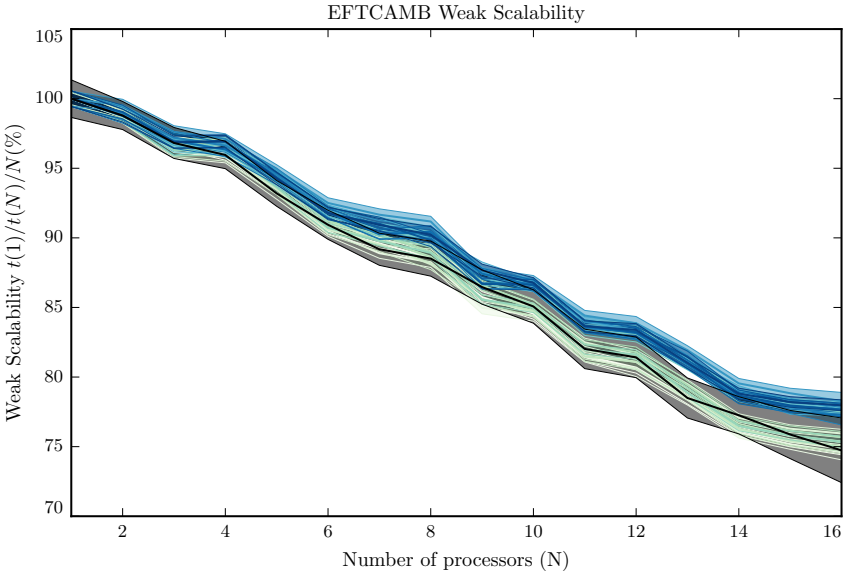

Figure 4.4: EFTCAMB weak scalability results. The continuous black line correspond to the weak scalability of the CAMB code. All the other colors correspond to the models included in the test suite. For all models the shaded area represent the 68% confidence region.

Chapter 5

Conclusions and outlook

Cosmic acceleration still poses a challenge for modern cosmology. While current cosmological data strikingly present observational evidence for this phenomenon its theoretical understanding is still lacking. To explain this effect one might resort to a cosmological constant, as it is done in the standard cosmological model; this, however, still does not have a deep theoretical motivation. On the other hand, one might want to add other dark fluids to the cosmic budget or modify the laws of gravity on large scales to drive this accelerated expansion.

In this thesis we presented the numerical implementation of a framework that unifies Dark Energy and Modified Gravity models, the Effective Field Theory of Cosmic Acceleration, into the tools that are used to compare cosmological models against data. We called the resulting code EFTCAMB.
This code allows to test many models with the same set of tools allowing efficient exploration of the space of models for Cosmic Acceleration. After presenting the structure of the code and its documentation we discussed the problem of scalability in model space. We commented on how this triggered an update of the EFTCAMB code architecture, how this problem is now solved and how the code structure can now easily accommodate many more models. We then introduced the EFTCAMB test suite and how we use it to measure the performances of the code. We further discussed the code performances and showed that the changes introduced to solve scalability in model space introduced a thirty percent performance penalty that was successively reabsorbed by means of profiling driven optimization. Finally we commented on the scalability of the code on multiprocessor machines to show that the EFTCAMB code does not introduce a significant scalability penalty.

The work presented in this thesis can be extended in several ways, that will be directly relevant and useful for the interpretation of data from contemporary and future experiments. The first extension of this work consists in implementing and studying new models, exploiting the new flexible code structure. The second relevant extension is to use the expertise and tools developed in this thesis to optimize the CAMB code to improve its performances.

On the long run both aspects will be the key to successfully pursuing the program of systematic investigations of the physical origin of cosmic acceleration.

Appendix A

A.0.1 Structure and Evolution of the EFTCAMB code

In parallel with the flexibility of the EFT approach to DE and MG models we need a complicated logical structure to control its numerical implementation. This structure changed from version to version growing in complexity while model coverage increased. The structure corresponding to different releases is shown in Figures A.1, A.2 and A.3.

A.0.2 EFTCAMB test suite sample output

In this Appendix we present a sample output of the EFTCAMB test suite, shown in figure A.4. For every model in the test suite the EFTCAMB code computes cosmological observables and compares them with the results of the latest version of the code. If significant differences are found the test suite plots the results to allow visual inspections and evaluation of the relevance of differences between outputs.

A.0.3 EFTCAMB performance evolution

In this Appendix we present the full evolution of the performances of EFTCAMB v3.0. The benchmark results at different stages of developments are shown in figures A.8, A.12 and A.15. The difference in speed of different versions of the code is instead shown in figures A.19, A.23 and A.25. All results are thoroughly discussed in Section 4.2.5.

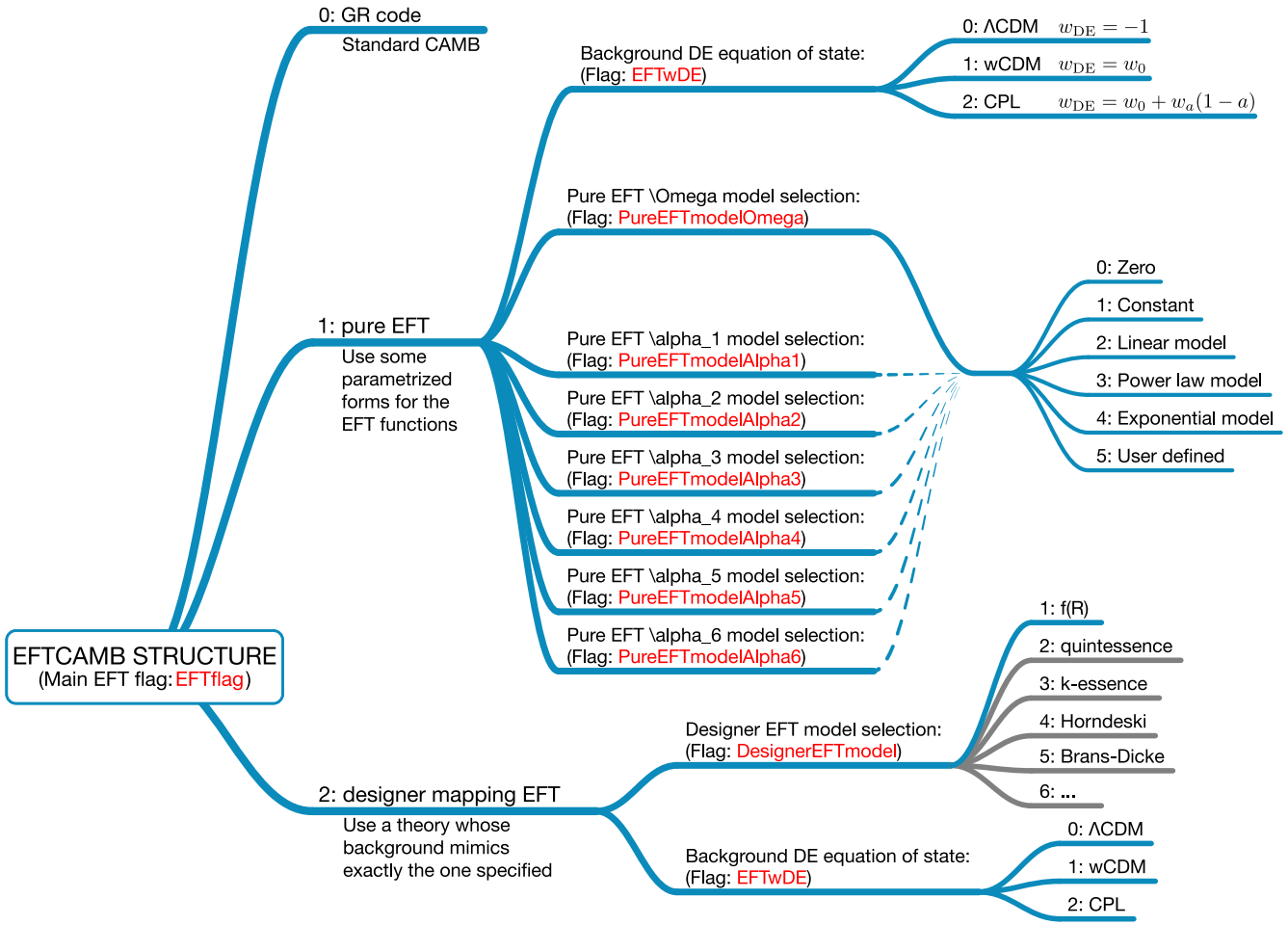

Figure A.1: Flowchart of the structure of EFTCAMB V1.0: blue lines correspond to flags that are already present in the code, while grey lines are a sample of the models that will be implemented gradually in future code releases.

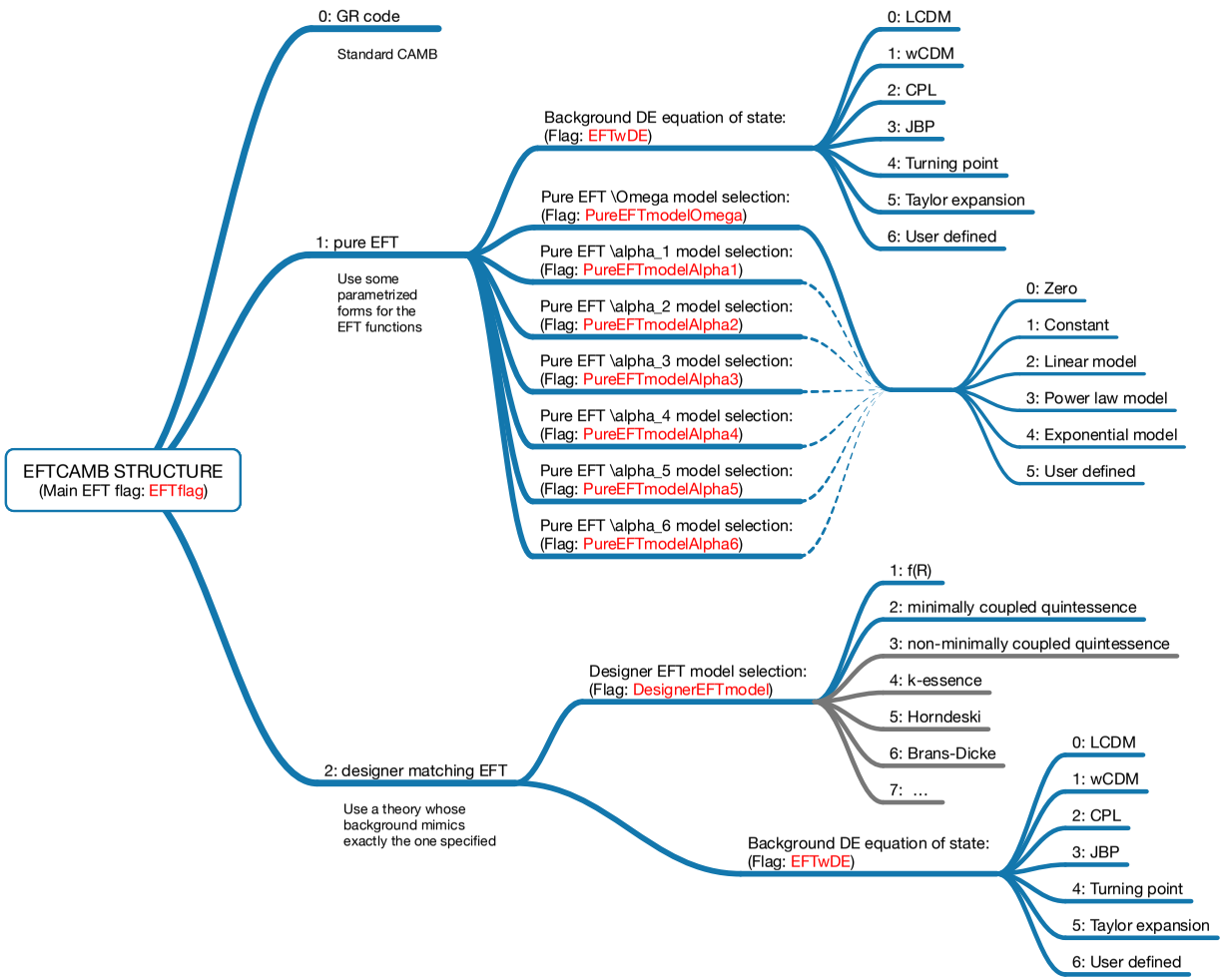

Figure A.2: Flowchart of the structure of EFTCAMB V1.1: blue lines correspond to flags that are already present in the code, while grey lines are a sample of the models that will be implemented gradually in future code releases.

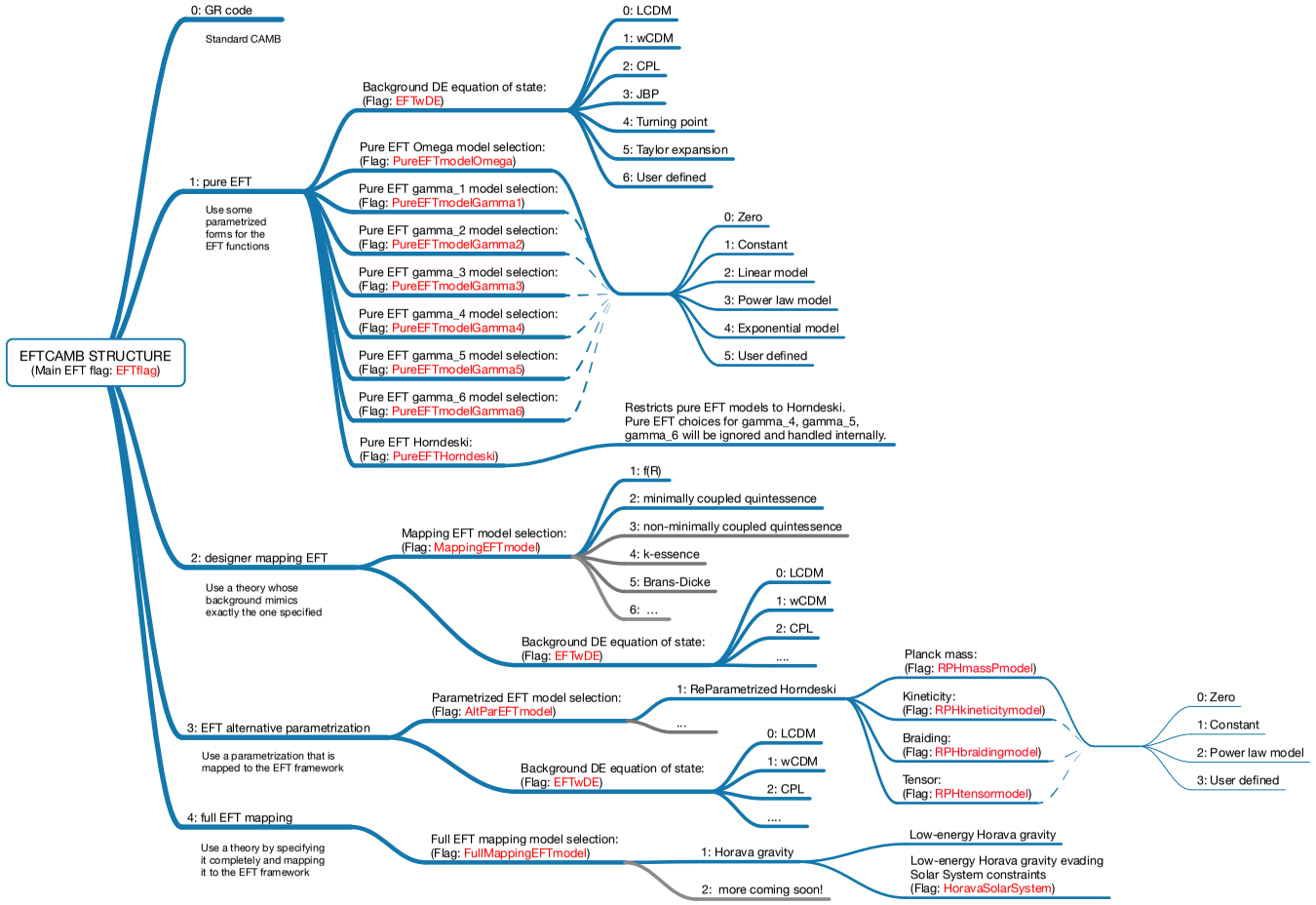

Figure A.3: Flowchart of the structure of EFTCAMB V2.0 and EFTCAMB V3.0: blue lines correspond to flags that are already present in the code, while grey lines are a sample of the models that will be implemented gradually in future code releases.

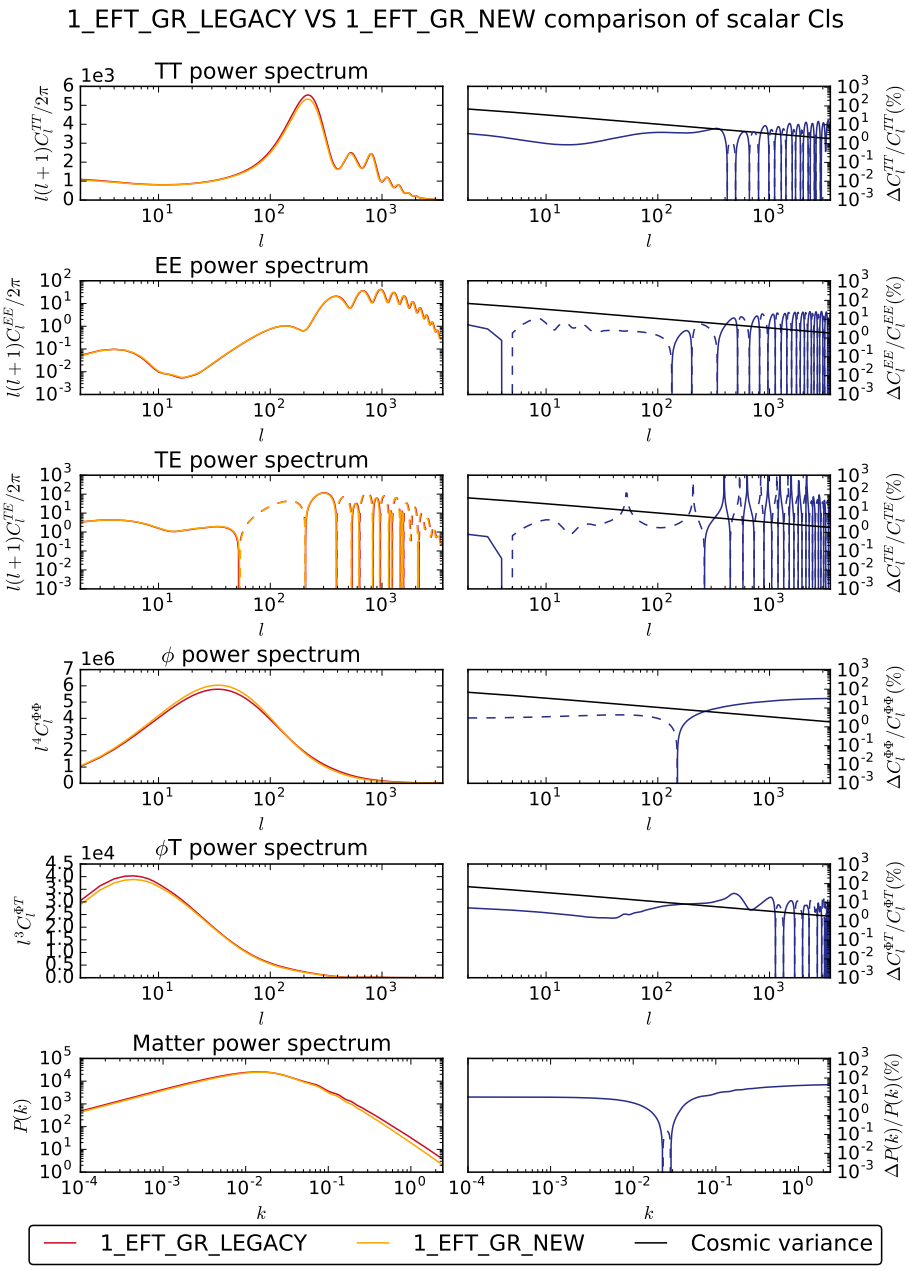

Figure A.4: EFTCAMB test suite sample output. Different colors represent different versions of the code, as shown in legend.

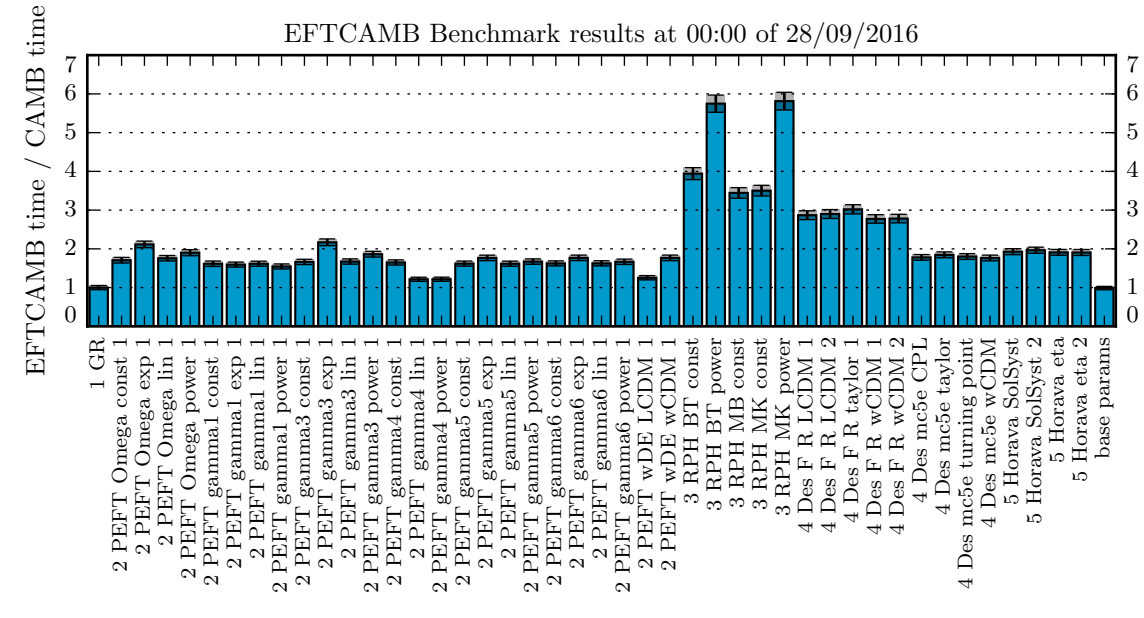

Figure A.5: (a) EFTCAMB v2.0

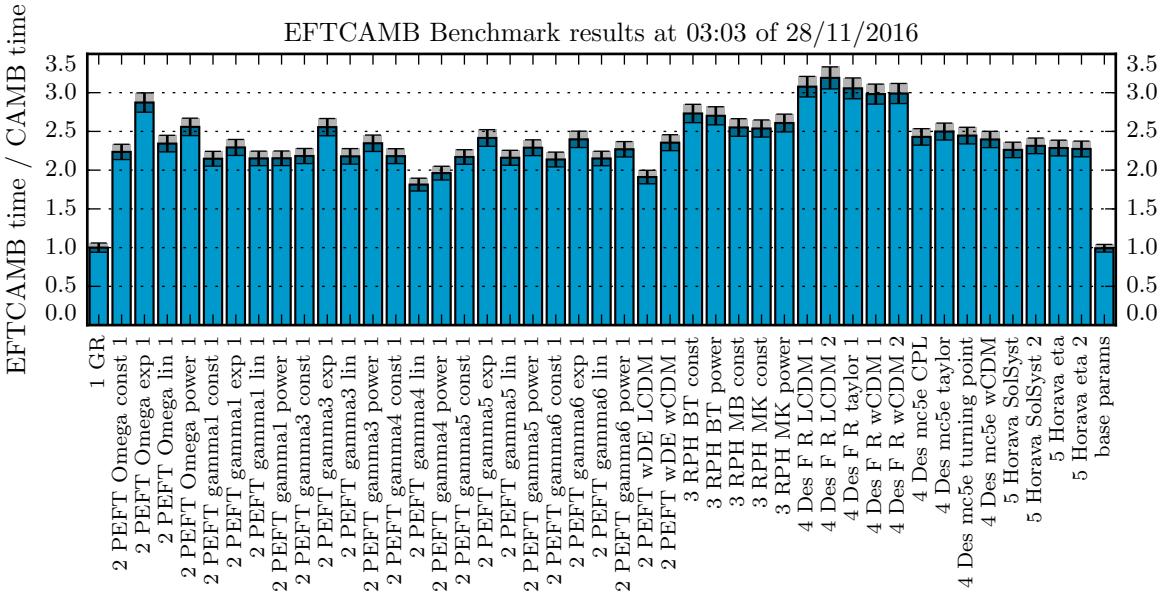

Figure A.6: (b) EFTCAMB v3.0

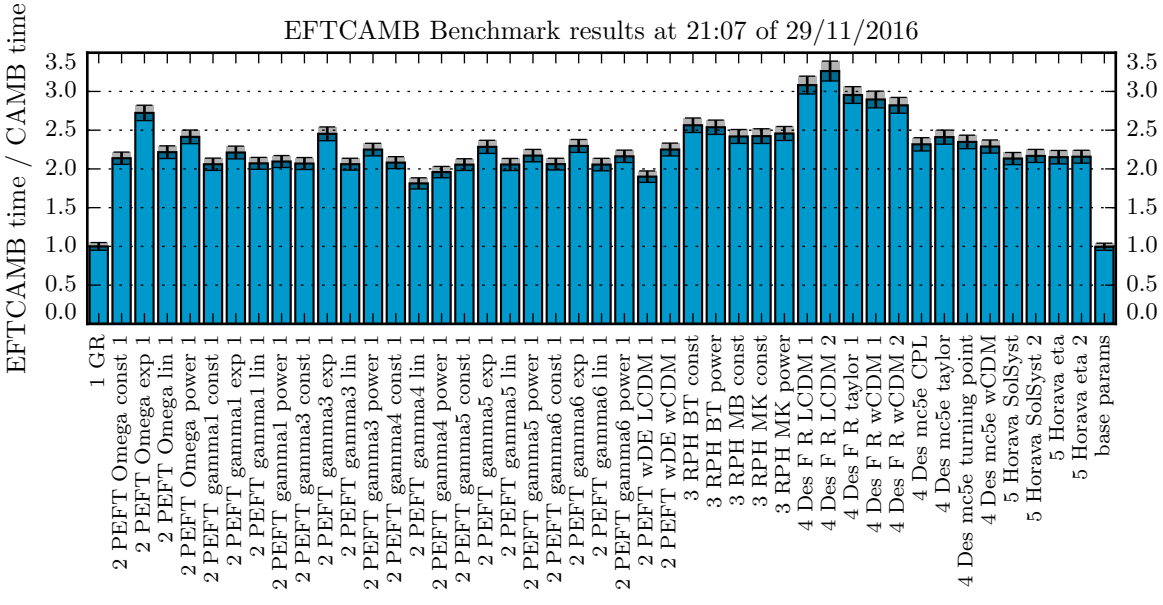

Figure A.7: (c) EFTCAMB v3.0

Figure A.8: Performances of different versions of the EFTCAMB code.

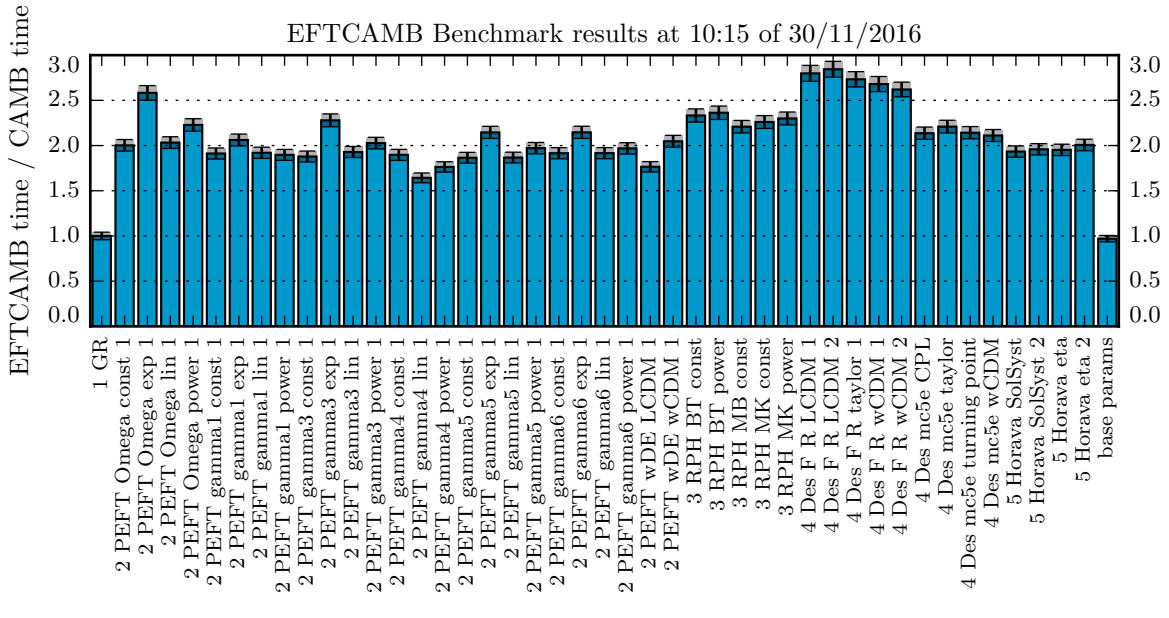

Figure A.9: (a) EFTCAMB v3.0

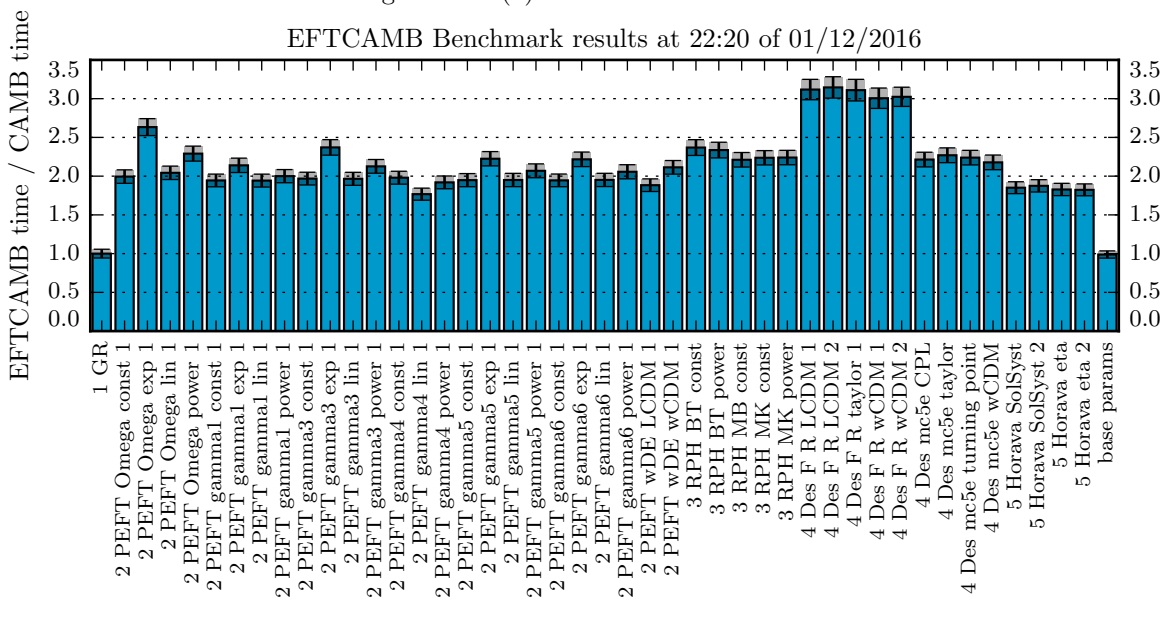

Figure A.10: (b) EFTCAMB v3.0

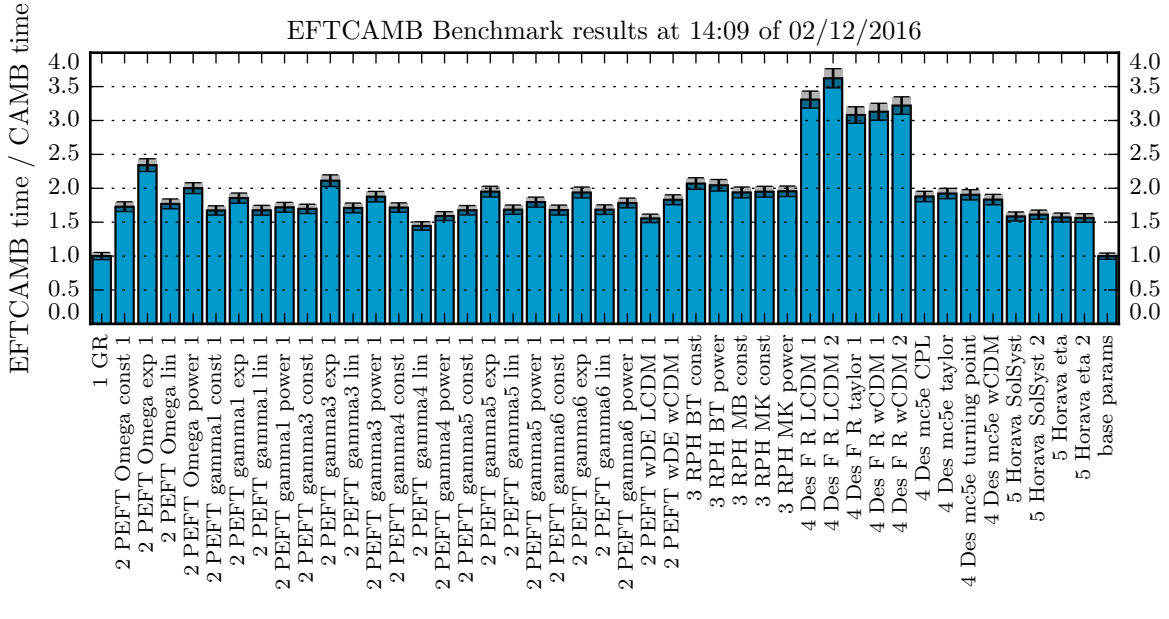

Figure A.11: (c) EFTCAMB v3.0

Figure A.12: Performances of different versions of the EFTCAMB code.

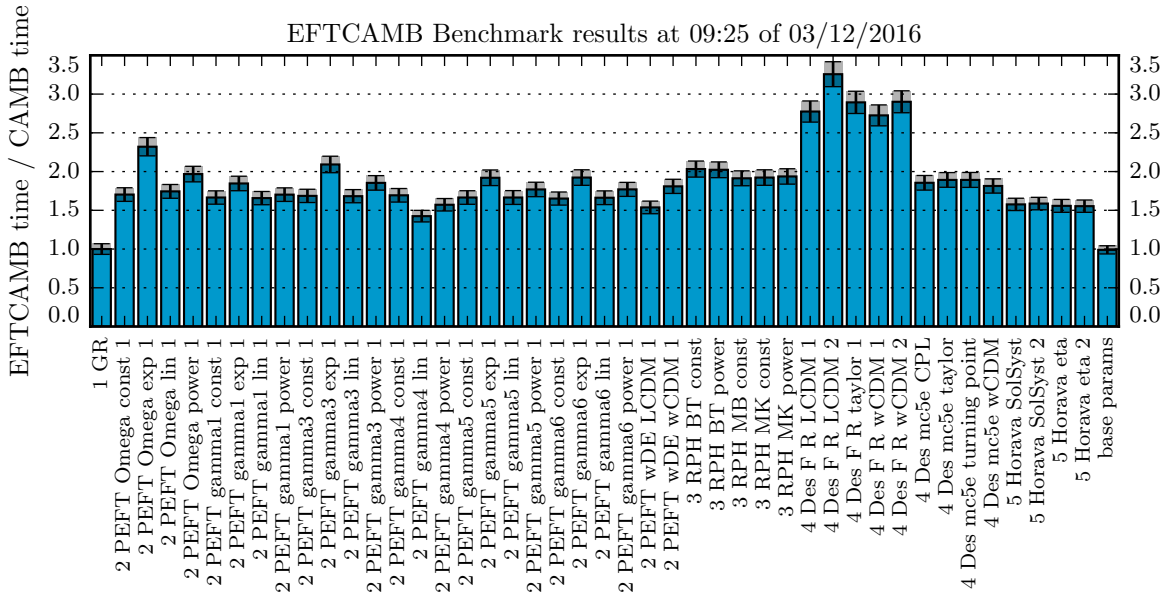

Figure A.13: (a) EFTCAMB v3.0

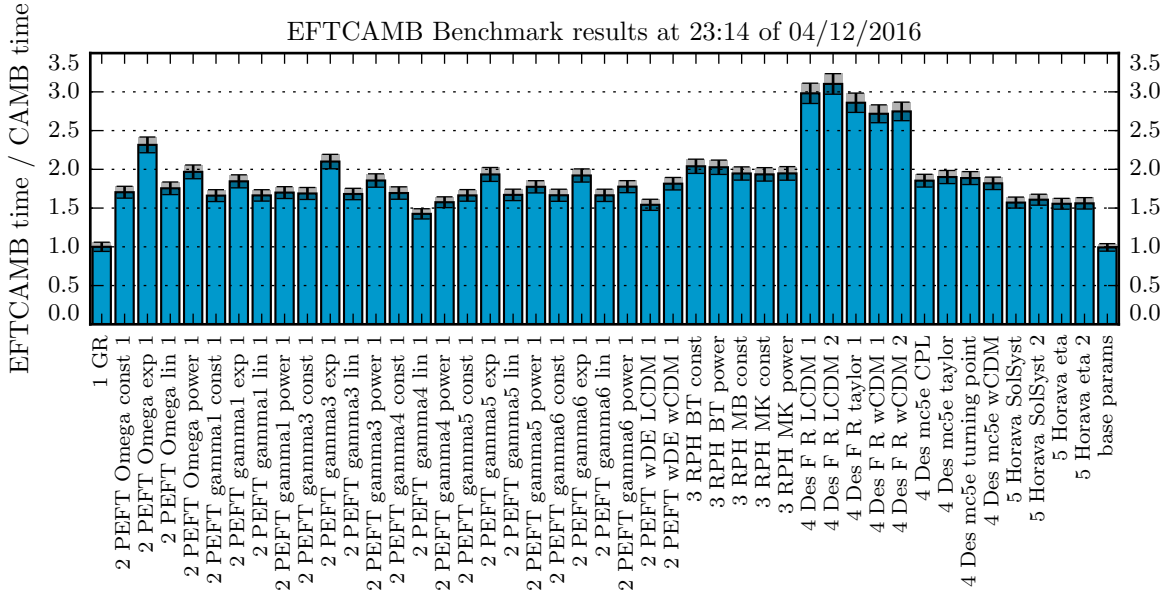

Figure A.14: (b) EFTCAMB v3.0

Figure A.15: Performances of different versions of the EFTCAMB code.

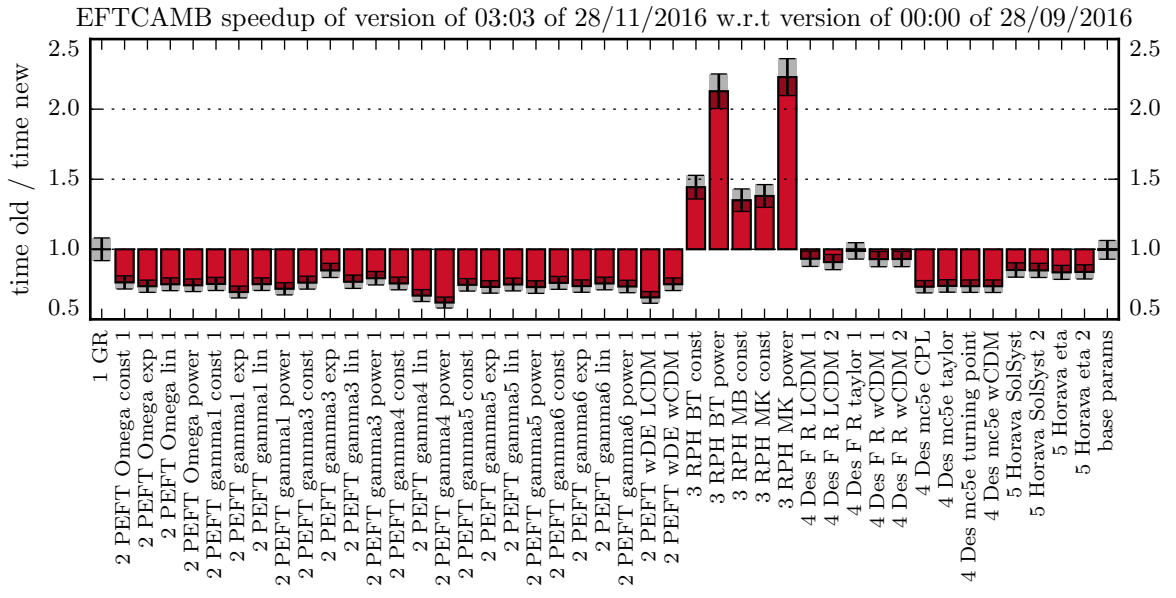

Figure A.16: (a) EFTCAMB v2.0 vs EFTCAMB v3.0

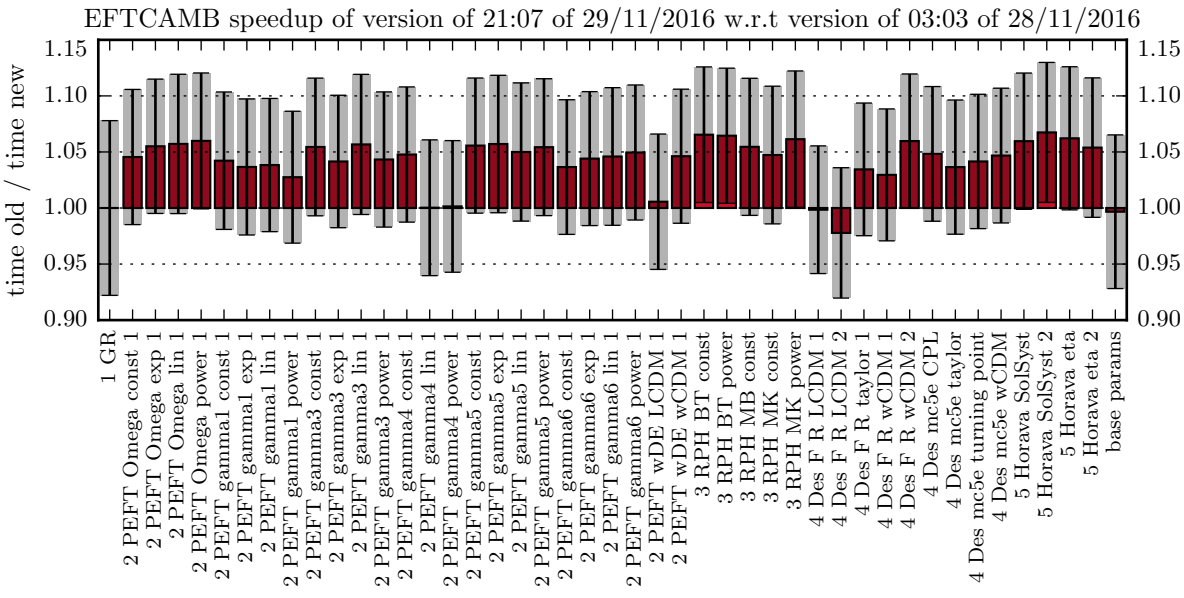

Figure A.17: (b) EFTCAMB v3.0

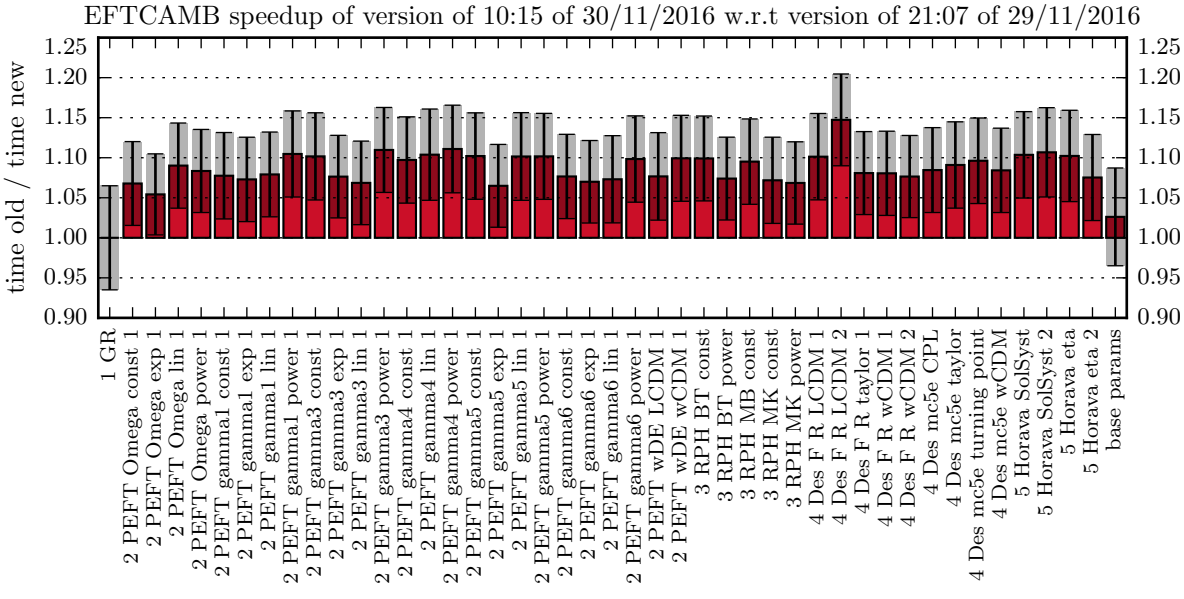

Figure A.18: (c) EFTCAMB v3.0

Figure A.19: Comparison of the performances of the EFTCAMB code.

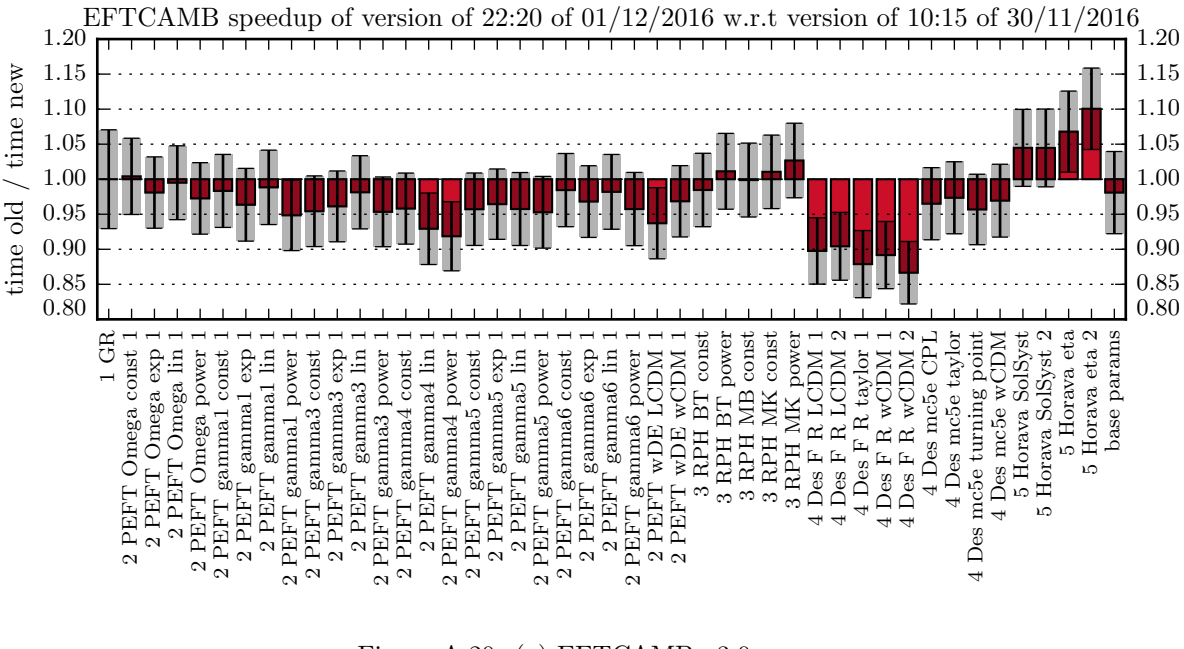

Figure A.20: (a) EFTCAMB v3.0

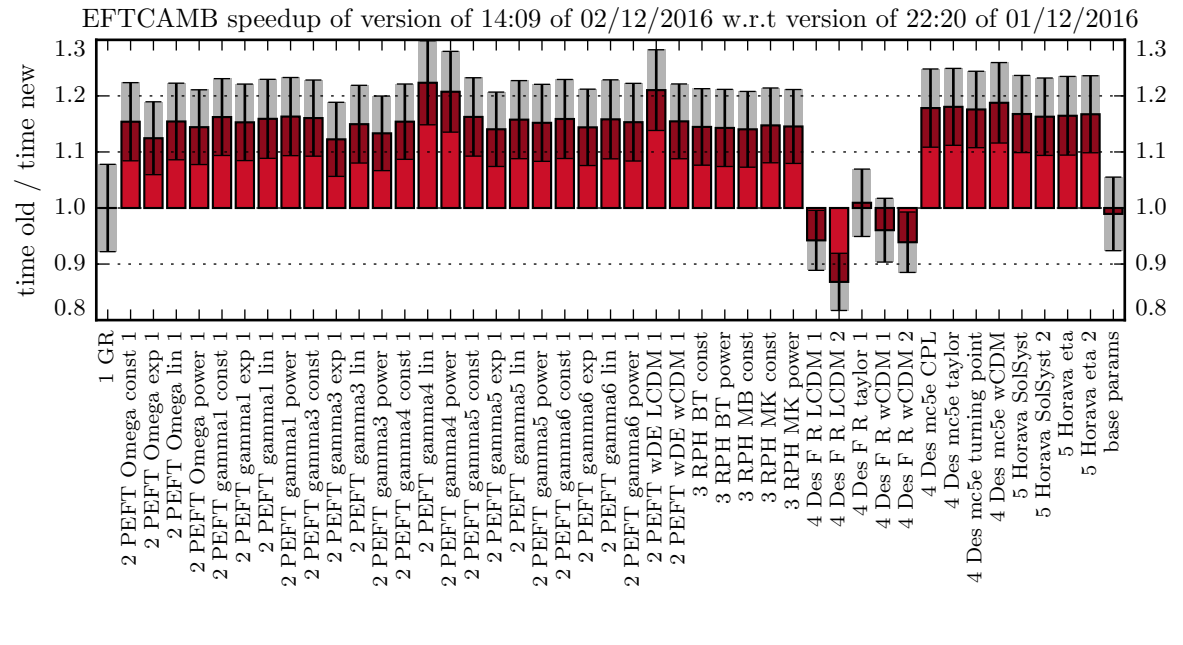

Figure A.21: (b) EFTCAMB v3.0

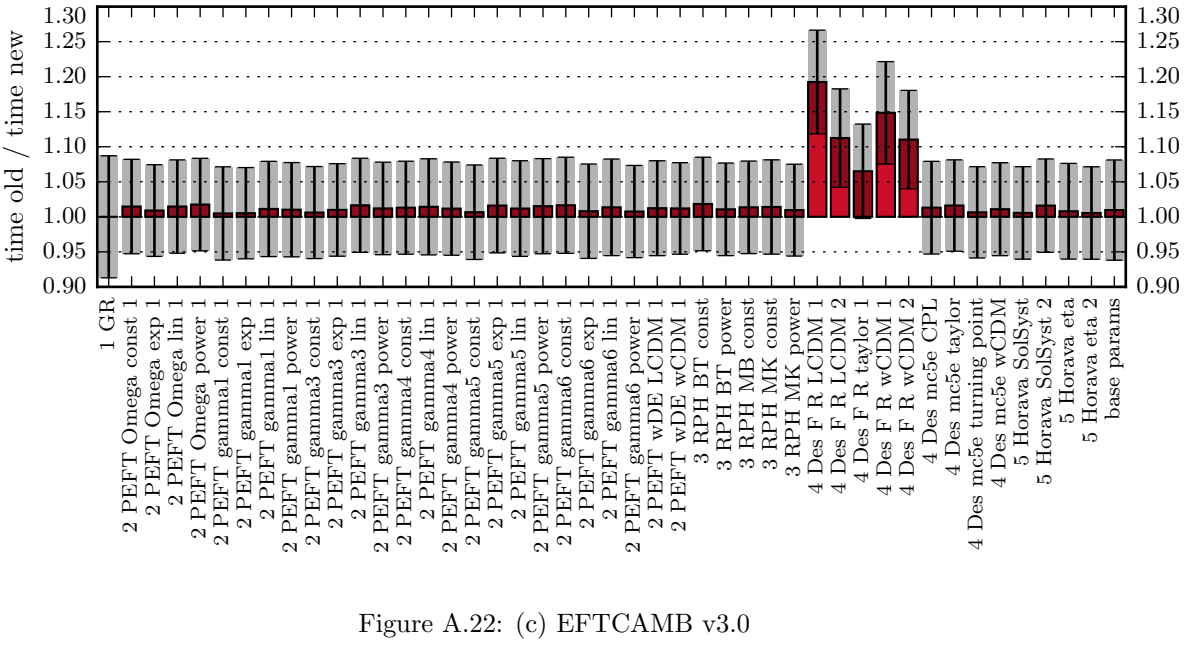

Figure A.22: (c) EFTCAMB v3.0

Figure A.23: Comparison of the performances of the EFTCAMB code.

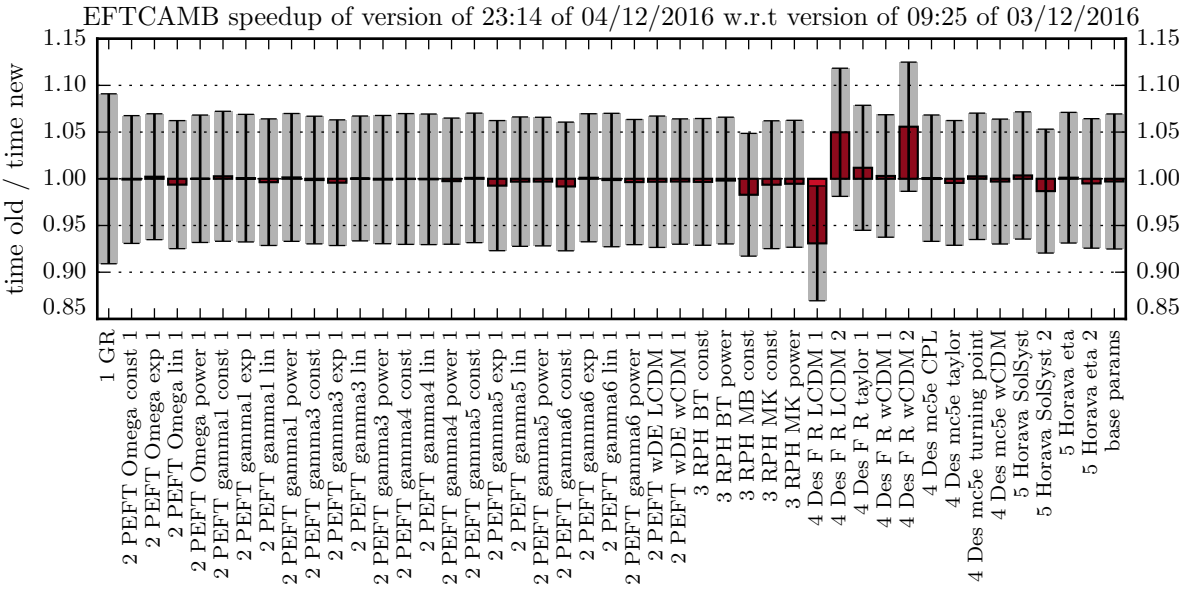

Figure A.24: (a) EFTCAMB v3.0

Figure A.25: Comparison of the performances of the EFTCAMB code.

Acknowledgments

I wish to thank...

...my supervisors Luca Heltai and Carlo Baccigalupi for wisely guiding me through the master. I have learned a great deal about numerical calculations and their application to physical problems and I will be forever grateful.

...my collaborators for their constant support and the great amount of work that we did together. None of this would have been possible without you.

...my fellows students in the master and Giuseppe Piero Brandino, Nicola Cavallini and Alberto Sartori, for making it enjoyable and for contributing to such an interesting environment.

Finally, my deepest gratitude goes to all the important people in my life outside scientific research.

Bibliography

[1] Bin Hu, Marco Raveri, Matteo Rizzato, and Alessandra Silvestri. Testing Hu-Sawicki f(R) gravity with the Effective Field Theory approach. 2016.

[2] Noemi Frusciante, Marco Raveri, Daniele Vernieri, Bin Hu, and Alessandra Silvestri. Hoava Gravity in the Effective Field Theory formalism: From cosmology to observational constraints. *Phys. Dark Univ.*, 13:7–24, 2016.

[3] Bin Hu and Marco Raveri. Can modified gravity models reconcile the tension between the CMB anisotropy and lensing maps in Planck-like observations? *Phys. Rev.*, D91(12):123515, 2015.

[4] Bin Hu, Marco Raveri, Alessandra Silvestri, and Noemi Frusciante. Exploring massive neutrinos in dark cosmologies with $EFTCAMB/$ EFTCosmoMC. *Phys. Rev.*, D91(6):063524, 2015.

[5] Marco Raveri, Bin Hu, Noemi Frusciante, and Alessandra Silvestri. Effective Field Theory of Cosmic Acceleration: constraining dark energy with CMB data. *Phys. Rev.*, D90(4):043513, 2014.

[6] Bin Hu, Marco Raveri, Noemi Frusciante, and Alessandra Silvestri. Effective Field Theory of Cosmic Acceleration: an implementation in CAMB. *Phys. Rev.*, D89(10):103530, 2014.

[7] Kevork N. Abazajian et al. CMB-S4 Science Book, First Edition. 2016.

[8] Simone Peirone, Marco Raveri, Matteo Viel, Stefano Borgani, and Stefano Ansoldi. Constraining f(R) Gravity with Planck Sunyaev-Zel'dovich Clusters. 2016.

[9] Marco Raveri, Matteo Martinelli, Gongbo Zhao, and Yuting Wang. Information Gain in Cosmology: From the Discovery of Expansion to Future Surveys. 2016.

[10] Bin Hu, Marco Raveri, Noemi Frusciante, and Alessandra Silvestri. EFTCAMB/EFTCosmoMC: Numerical Notes v2.0. 2014.

[11] S. Perlmutter et al. Measurements of Omega and Lambda from 42 high redshift supernovae. *Astrophys. J.*, 517:565–586, 1999.

[12] Adam G. Riess et al. Observational evidence from supernovae for an accelerating universe and a cosmological constant. *Astron. J.*, 116:1009–1038, 1998.

[13] Edmund Bertschinger. On the Growth of Perturbations as a Test of Dark Energy. *Astrophys. J.*, 648:797–806, 2006.

[14] Eric V. Linder and Robert N. Cahn. Parameterized Beyond-Einstein Growth. *Astropart. Phys.*, 28:481–488, 2007.

[15] Luca Amendola, Martin Kunz, and Domenico Sapone. Measuring the dark side (with weak lensing). *JCAP*, 0804:013, 2008.

[16] Pengjie Zhang, Michele Liguori, Rachel Bean, and Scott Dodelson. Probing Gravity at Cosmological Scales by Measurements which Test the Relationship between Gravitational Lensing and Matter Overdensity. *Phys. Rev. Lett.*, 99:141302, 2007.

[17] Wayne Hu and Ignacy Sawicki. A Parameterized Post-Friedmann Framework for Modified Gravity. *Phys. Rev.*, D76:104043, 2007.

[18] Edmund Bertschinger and Phillip Zukin. Distinguishing Modified Gravity from Dark Energy. *Phys. Rev.*, D78:024015, 2008.

[19] Scott F. Daniel, Robert R. Caldwell, Asantha Cooray, and Alessandro Melchiorri. Large Scale Structure as a Probe of Gravitational Slip. *Phys. Rev.*, D77:103513, 2008.

[20] Yong-Seon Song and Kazuya Koyama. Consistency test of general relativity from large scale structure of the Universe. *JCAP*, 0901:048, 2009.

[21] Constantinos Skordis. Consistent cosmological modifications to the Einstein equations. *Phys. Rev.*, D79:123527, 2009.

[22] Yong-Seon Song and Olivier Dore. A step towards testing general relativity using weak gravitational lensing and redshift surveys. *JCAP*, 0903:025, 2009.

[23] Gong-Bo Zhao, Levon Pogosian, Alessandra Silvestri, and Joel Zylberberg. Cosmological Tests of General Relativity with Future Tomographic Surveys. *Phys. Rev. Lett.*, 103:241301, 2009.

[24] Gong-Bo Zhao, Tommaso Giannantonio, Levon Pogosian, Alessandra Silvestri, David J. Bacon, Kazuya Koyama, Robert C. Nichol, and Yong-Seon Song. Probing modifications of General Relativity using current cosmological observations. *Phys. Rev.*, D81:103510, 2010.

[25] Jason Dossett, Mustapha Ishak, Jacob Moldenhauer, Yungui Gong, Anzhong Wang, and Yungui Gong. Constraints on growth index parameters from current and future observations. *JCAP*, 1004:022, 2010.

[26] Yong-Seon Song, Lukas Hollenstein, Gabriela Caldera-Cabral, and Kazuya Koyama. Theoretical Priors On Modified Growth Parametrisations. *JCAP*, 1004:018, 2010.

[27] Scott F. Daniel, Eric V. Linder, Tristan L. Smith, Robert R. Caldwell, Asantha Cooray, Alexie Leauthaud, and Lucas Lombriser. Testing General Relativity with Current Cosmological Data. *Phys. Rev.*, D81:123508, 2010.

[28] Levon Pogosian, Alessandra Silvestri, Kazuya Koyama, and Gong-Bo Zhao. How to optimally parametrize deviations from General Relativity in the evolution of cosmological perturbations? *Phys. Rev.*, D81:104023, 2010.

[29] Rachel Bean and Matipon Tangmatitham. Current constraints on the cosmic growth history. *Phys. Rev.*, D81:083534, 2010.

[30] Shaun A. Thomas, Stephen A. Appleby, and Jochen Weller. Modified Gravity: the CMB, Weak Lensing and General Parameterisations. *JCAP*, 1103:036, 2011.

[31] Tessa Baker, Pedro G. Ferreira, Constantinos Skordis, and Joe Zuntz. Towards a fully consistent parameterization of modified gravity. *Phys. Rev.*, D84:124018, 2011.

[32] Daniel B. Thomas and Carlo R. Contaldi. Testing model independent modified gravity with future large scale surveys. *JCAP*, 1112:013, 2011.

[33] Jason N. Dossett, Mustapha Ishak, and Jacob Moldenhauer. Testing General Relativity at Cosmological Scales: Implementation and Parameter Correlations. *Phys. Rev.*, D84:123001, 2011.

[34] Gong-Bo Zhao, Hong Li, Eric V. Linder, Kazuya Koyama, David J. Bacon, and Xinmin Zhang. Testing Einstein Gravity with Cosmic Growth and Expansion. *Phys. Rev.*, D85:123546, 2012.

[35] Alireza Hojjati, Gong-Bo Zhao, Levon Pogosian, Alessandra Silvestri, Robert Crittenden, and Kazuya Koyama. Cosmological tests of General Relativity: a principal component analysis. *Phys. Rev.*, D85:043508, 2012.

[36] Philippe Brax, Anne-Christine Davis, and Baojiu Li. Modified Gravity Tomography. *Phys. Lett.*, B715:38–43, 2012.

[37] Jason Dossett and Mustapha Ishak. Spatial Curvature and Cosmological Tests of General Relativity. *Phys. Rev.*, D86:103008, 2012.

[38] Philippe Brax, Anne-Christine Davis, Baojiu Li, and Hans A. Winther. A Unified Description of Screened Modified Gravity. *Phys. Rev.*, D86:044015, 2012.

[39] Ignacy Sawicki, Ippocratis D. Saltas, Luca Amendola, and Martin Kunz. Consistent perturbations in an imperfect fluid. *JCAP*, 1301:004, 2013.

[40] Tessa Baker, Pedro G. Ferreira, and Constantinos Skordis. The Parameterized Post-Friedmann framework for theories of modified gravity: concepts, formalism and examples. *Phys. Rev.*, D87(2):024015, 2013.

[41] Luca Amendola, Martin Kunz, Mariele Motta, Ippocratis D. Saltas, and Ignacy Sawicki. Observables and unobservables in dark energy cosmologies. *Phys. Rev.*, D87(2):023501, 2013.

[42] Alireza Hojjati. Degeneracies in parametrized modified gravity models. *JCAP*, 1301:009, 2013.

[43] Alessandra Silvestri, Levon Pogosian, and Roman V. Buniy. Practical approach to cosmological perturbations in modified gravity. *Phys. Rev.*, D87(10):104015, 2013.

[44] Mariele Motta, Ignacy Sawicki, Ippocratis D. Saltas, Luca Amendola, and Martin Kunz. Probing Dark Energy through Scale Dependence. *Phys. Rev.*, D88(12):124035, 2013.

[45] Shinsuke Asaba, Chiaki Hikage, Kazuya Koyama, Gong-Bo Zhao, Alireza Hojjati, and Levon Pogosian. Principal Component Analysis of Modified Gravity using Weak Lensing and Peculiar Velocity Measurements. *JCAP*, 1308:029, 2013.

[46] Tessa Baker, Pedro G. Ferreira, and Constantinos Skordis. A Fast Route to Modified Gravitational Growth. *Phys. Rev.*, D89(2):024026, 2014.

[47] Jason Dossett and Mustapha Ishak. Effects of Dark Energy Perturbations on Cosmological Tests of General Relativity. *Phys. Rev.*, D88(10):103008, 2013.

[48] Ayumu Terukina, Lucas Lombriser, Kazuhiro Yamamoto, David Bacon, Kazuya Koyama, and Robert C. Nichol. Testing chameleon gravity with the Coma cluster. *JCAP*, 1404:013, 2014.

[49] Federico Piazza, Heinrich Steigerwald, and Christian Marinoni. Phenomenology of dark energy: exploring the space of theories with future redshift surveys. *JCAP*, 1405:043, 2014.

[50] Lszl . Gergely and Shinji Tsujikawa. Effective field theory of modified gravity with two scalar fields: dark energy and dark matter. *Phys. Rev.*, D89(6):064059, 2014.

[51] Bin Hu, Michele Liguori, Nicola Bartolo, and Sabino Matarrese. Future CMB integrated-Sachs-Wolfe-lensing bispectrum constraints on modified gravity in the parametrized post-Friedmann formalism. *Phys. Rev.*, D88(2):024012, 2013.

[52] D. Munshi, B. Hu, A. Renzi, A. Heavens, and P. Coles. Probing Modified Gravity Theories with ISW and CMB Lensing. *Mon. Not. Roy. Astron. Soc.*, 442(1):821–837, 2014.

[53] Bin Hu, Michele Liguori, Nicola Bartolo, and Sabino Matarrese. Parametrized modified gravity constraints after Planck. *Phys. Rev.*, D88(12):123514, 2013.

[54] Heinrich Steigerwald, Julien Bel, and Christian Marinoni. Probing non-standard gravity with the growth index: a background independent analysis. *JCAP*, 1405:042, 2014.

[55] Qing-Guo Huang. An analytic calculation of the growth index for $f(R)$ dark energy model. *Eur. Phys. J.*, C74:2964, 2014.

[56] Lucas Lombriser. Constraining chameleon models with cosmology. *Annalen Phys.*, 526:259–282, 2014.

[57] Shinji Tsujikawa. The effective field theory of inflation/dark energy and the Horndeski theory. *Lect. Notes Phys.*, 892:97–136, 2015.

[58] Emilio Bellini and Ignacy Sawicki. Maximal freedom at minimum cost: linear large-scale structure in general modifications of gravity. *JCAP*, 1407:050, 2014.

[59] Richard A. Battye and Jonathan A. Pearson. Effective action approach to cosmological perturbations in dark energy and modified gravity. *JCAP*, 1207:019, 2012.

[60] Giulia Gubitosi, Federico Piazza, and Filippo Vernizzi. The Effective Field Theory of Dark Energy. *JCAP*, 1302:032, 2013. [JCAP1302,032(2013)].

[61] Jolyon K. Bloomfield, anna . Flanagan, Minjoon Park, and Scott Watson. Dark energy or modified gravity? An effective field theory approach. *JCAP*, 1308:010, 2013.

[62] Federico Piazza and Filippo Vernizzi. Effective Field Theory of Cosmological Perturbations. *Class. Quant. Grav.*, 30:214007, 2013.

[63] Paolo Creminelli, Markus A. Luty, Alberto Nicolis, and Leonardo Senatore. Starting the Universe: Stable Violation of the Null Energy Condition and Non-standard Cosmologies. *JHEP*, 12:080, 2006.

[64] Paolo Creminelli, Guido D'Amico, Jorge Norena, and Filippo Vernizzi. The Effective Theory of Quintessence: the w¡-1 Side Unveiled. *JCAP*, 0902:018, 2009.

[65] Minjoon Park, Kathryn M. Zurek, and Scott Watson. A Unified Approach to Cosmic Acceleration. *Phys. Rev.*, D81:124008, 2010.

[66] Clifford Cheung, Paolo Creminelli, A. Liam Fitzpatrick, Jared Kaplan, and Leonardo Senatore. The Effective Field Theory of Inflation. *JHEP*, 03:014, 2008.

[67] Steven Weinberg. Effective Field Theory for Inflation. *Phys. Rev.*, D77:123541, 2008.

[68] Raul Jimenez, P. Talavera, and Licia Verde. An effective theory of accelerated expansion. *Int. J. Mod. Phys.*, A27:1250174, 2012.

[69] John Joseph M. Carrasco, Mark P. Hertzberg, and Leonardo Senatore. The Effective Field Theory of Cosmological Large Scale Structures. *JHEP*, 09:082, 2012.

[70] Mark P. Hertzberg. Effective field theory of dark matter and structure formation: Semianalytical results. *Phys. Rev.*, D89(4):043521, 2014.

[71] Jerome Gleyzes, David Langlois, Federico Piazza, and Filippo Vernizzi. Essential Building Blocks of Dark Energy. *JCAP*, 1308:025, 2013.

[72] Jolyon Bloomfield. A Simplified Approach to General Scalar-Tensor Theories. *JCAP*, 1312:044, 2013.

[73] Alberto Nicolis, Riccardo Rattazzi, and Enrico Trincherini. The Galileon as a local modification of gravity. *Phys. Rev.*, D79:064036, 2009.

[74] Alan F. Heavens, T. D. Kitching, and L. Verde. On model selection forecasting, Dark Energy and modified gravity. *Mon. Not. Roy. Astron. Soc.*, 380:1029–1035, 2007.

[75] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[76] W.K. Hastings. Monte carlo samping methods using markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[77] W. R. Gilks. *Markov Chain Monte Carlo In Practice*. Chapman and Hall/CRC, 1999.

[78] Antony Lewis and Sarah Bridle. CosmoMC Notes. `http://cosmologist.info/notes/CosmoMC.pdf`.

[79] Radford. M. Neal. Mcmc using ensembles of states for problems with fast and slow variables such as gaussian process regression. 2011.

[80] Antony Lewis, Anthony Challinor, and Anthony Lasenby. Efficient computation of CMB anisotropies in closed FRW models. *Astrophys. J.*, 538:473–476, 2000.