Short description and comments for the **IGWsfSC** code

Albert Escriva

22 February

These short notes and comments correspond to the *IGWsfSC* (Induced Gravitational Waves from a Smooth Crossover), written in *Julia* [1]. The code aims to numerically compute the induced gravitational waves for the case of a general smooth crossover beyond Standard Model. The numerical code has been utilized to obtain the results presented in [2], calculating the corresponding signature of the induced gravitational waves associated with the existence of a crossover beyond the Standard Model. For theoretical details and some extra details of the simulations we refer the reader to [2] and references therein.

The numerical code is provided with sufficient comments to facilitate understanding, and additional comments are included in these notes. It's important to note that the code has been presented in a simplified manner to showcase the computation. Perhaps the first question to address is: Why Julia? Julia addresses the gap between two languages, transitioning from C to *Python*. We chose Julia as a language to strike a balance between the low-level efficiency of fast software like C/Fortran and the readability of *Python*.

Below, the reader will find some useful comments about the code.

- The code doesn't need installation itself; instead, install "Julia" on your computer. We refer the reader to [3] for details about the installation and the corresponding required numerical packages. In the case of using Ubuntu (where the code has been tested), it is simple to execute it using the command **julia IGWsfSC.jl**. In case of using parallel threads, you need to specify the number of threads with **julia -t 10 IGWsfSC.jl**, where 10 would be the number of cores used.

- We use **JLD2** package to allocate the different quantities needed for the final computation, and in particular for building the averaged square kernel matrix, given by $\bar{I}^2(k_1, k_2, k, \eta_{\rm sh})$ where $\eta_{\rm sh}$ is the time once the modes are well inside the horizon. Then, the averaged power spectrum $\bar{\mathcal{P}}_h(k, \eta_{\rm sh})$ can be computed as indicated by Eqs.(10,11) in [2]. Details of the package and installation can be found in [4]. Basically, **JLD2** saves and loads Julia data structures in a format comprising a subset of **HDF5**, without any dependency on the **HDF5** C library.

- The numerical solutions of the scalar transfer functions $\Phi_k(\eta)$ have been stored in auxiliary files using memory mapping of files (**mmap**). This not only facilitates potential post-processing of the scalar transfer functions (if the user is interested) but also, in terms of memory efficiency, allows the storage of all data, preventing memory exhaustion, especially when performing computations with high resolution and multi-threading across several cores.

- Notice the final output is a matrix $\bar{I}^2(k_1, k_2, k, \eta_{\rm sh})$ that can be evaluated for any power spectrum $\mathcal{P}_\zeta(k)$ (see Eq.(11) in [2]). For the case of [2] the computation is considered for a flat spectrum.

- For a practical computation, it is important to notice that the most challenging case corresponds to the flat spectrum (the one given in the code as an example), since involve the computation of many combinations k_1, k_2 values. In this case, to achieve the analytical solution for a flat spectrum given by $\Omega_{\rm GWs} \approx 0.822/\mathcal{A}^2$ [5] it is necessary to sufficiently increase the resolution of the computation. In particular, we recommend the user to always check the convergence of the numerical results, increasing sufficiently the number of points in the numerical convolution, the final evaluation time $\eta_{\rm sh}$, decrease the tolerance of the ODE solver and increase the number of points in the grid.

- It should be noted that for an extensively prolonged crossover period (corresponding to the scenario of $\sigma \gg 1$ for the crossover template outlined in [2]), the computation becomes significantly more challenging numerically. This is due to an extended duration where the Universe deviates from being exactly radiation-dominated.

References

[1] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, "Julia: A fresh approach to numerical computing," *SIAM review*, vol. 59, no. 1, pp. 65–98, 2017.

[2] A. Escrivà, Y. Tada, and C.-M. Yoo, "Primordial Black Holes and Induced Gravitational Waves from a Smooth Crossover beyond Standard Model," *arXiv e-prints*, p. arXiv:2311.17760, Nov. 2023.

[3] "Julia code installation." `https://julialang.org/downloads/`.

[4] "Jld2 package." `https://github.com/JuliaIO/JLD2.jl`.

[5] K. Kohri and T. Terada, "Semianalytic calculation of gravitational wave spectrum nonlinearly induced from primordial curvature perturbations," *Phys. Rev. D*, vol. 97, p. 123532, Jun 2018.