We thank all the reviewers for the careful and insightful review of our manuscript. We will of course correct all the minor suggestions and typos provided by all the reviewers. Most reviewers are concerned with handling larger networks and we provide a detailed view on this point at the end, in the response to AC.

Response to R1: • Regarding comparison with baselines, we build on reference [21] where detailed comparison with baseline methods were carried out. Accordingly, LipOpt provides superior results compared to concurent methods. We stick to comparison with LipOpt with these results in mind for simplicity. • We will work on the introduction and moderate the claim about L2 norm emphasizing that this mainly holds for the robustness certification problem.

Response to R2: • Regarding scaling for bigger network the reviewer can see the response to AC for a detailed account. • Regarding the validity of the upper bound, the SDP relaxation provides a guaranteed upper bound of the exact optimal value. Thus our heuristic relaxation does give valid upper bounds of the exact Lipschitz constant. • Regarding friendliness of the paper, we provided simple examples for the SDP relaxation in appendix B. We will make more explicit the presence of these examples starting from the introduction.

Response to R3: • We are aware of the importance of downstream applications but decided to stick the problem of Lipschitz constant estimation for several reasons: 1/ The main concurrent approach was [21] which did not perform such experiments, we decided to stick to the same evaluation metrics since the methods were designed for it. 2/ Technical and methodological improvements for the problem of Lipschitz constant estimation could possibly be translated to different certification problems. 3/ We stick to the intuition that better estimation of Lipschitz constants will be helpful for downstream applications. • Regarding the advantage of our method, we see in Table 1 that HR-2 actually gets better upper bounds than LipOpt-3 with less running time. Although the results of HR-2 is a little bit worse than LipOpt-4, the running time of LipOpt-4 is much bigger than HR-2. Shor's relaxation presented in the paper is actually also one of our methods. Since we use an exact model to describe the Lipschitz constant, which is different from [21]. The results in Table 2 show that our method (both HR-2 and Shor) works for the MNIST network while LipOpt doesn't. • We also would like to point out that references [1] and [3] proposed by the referee do not consider Lipschitz constants, but rather consider directly the problem of robustness certification. The reference [2] provides an algorithm to compute upper bounds on Lipschitz constants to derive a training procedure in order to obtain a robust network. These (rather coarse) bounds are recursively obtained for each network's component by relying on composition, addition and concatenation rules. We will add completeness experiments on the problem of robustness certification utilizing our method.

Response to R4: Indeed the relaxation applied to two layer networks is possible but requires slight modifications. Precisely, SHOR is the first order method for one layer networks, HR-1 is the first order method for two layer networks, and HR-2 stands for the second order method both for one and two layer networks. Pushing further the size of networks which can be considered is one of our middle term goals (see also response to AC).

Response to AC: **Theoretical contributions:** • Provide a semi-algebraic representation of the subgradient of the ReLU function. Prove that this can be used to formulate a polynomial optimization problem whose solutions provide certified upper bounds to neural network Lipschitz constants (as noted by one of the reviewers, this is not a direct consequence of the definition of gradients and stands on recent developments in nonsmooth analysis). Derive an adaptation of Lasserre's Hierarchy specifically tailored to the problem of Lipschitz constant estimation of ReLU networks. • Since our model is an exact description of the Lipschitz constant, one obtains convergence (only asymptotic) to the exact constant value with the dense hierarchy. We plan to derive similar guarantees for the proposed sparse hierarchy in future research. Thus the only problem is the scalability. The hierarchy is used here in a non-asymptotic regime, between the first and second orders. Obtaining approximation guarantees for the resulting upper bounds is difficult, most available bounds are very pessimistic, which is coherent with the fact that the hierarchy can solve NP-hard problems. **Empirical contribution:** • We provide results on two hidden layer networks in the appendix (up to size $50 \times 50 \times 10$, the relaxation has to be slightly modified). We run our method on a 784×500 dense single hidden layer MNIST network (results are reported in the main text). The methods take between 3 and 5 hours to run on a small personal laptop. • Generally speaking, an L hidden layer (fully-connected or convolutional) network results in a polynomial optimization problem whose objective is of degree $L+1$. And one needs to use the $\lceil (L+1)/2 \rceil$-th order SOS-hierarchy to solve such problems, which contains $O(n^{L+1})$ variables and PSD matrices of size $O(n^{\lceil (L+1)/2 \rceil})$ where n is the total number of variables in the problem. • Considering real networks, such as AlexNet, requires to treat 5 convolution layers and 3 fully-connected layers where in each layer there are more than 4000 nodes. This means that we will have $O(4000^9) \approx 10^{33}$ variables in the targeted optimization problem, which seems impossible. Even if we compute the Lipschitz constant recursively layer by layer, there are still $O(4000^2) \approx 10^7$ variables. Handling problems with million variables requires to handle the very sparse structure provided by these layers in a tailored version of the SOS-hierarchy. This constitutes an interesting venue for future research. To our knowledge, none of the concurrent methods is able, as of today, to handle such real scale networks with similar guarantees (i.e. valid upper bounds and asymptotic convergence). This would require considerable work on the implementation, much more performing hardware and a lot of engineering. • A key in lowering these numbers is to devise relaxation methods specifically tailored to the sparsity patterns of deep neural convolutional networks. Indeed, the associated polynomial optimization problems have a specific sparsity pattern, which can be exploited to significantly reduce the computational burden. This work constitutes a step in this direction and we will continue to push this idea further.