# Supplemental material to "Ability of error correlations to improve the performance of variational quantum algorithms"

Joris Kattemölle and Guido Burkard
*Department of Physics, University of Konstanz, D-78457 Konstanz, Germany*

We introduce the raw data and the code that generated it.

The code for this work is based on the code used in Ref. [1]. Major additions are the implementation of quantum-classical mixed-state registers, used to model spatiotemporally correlated errors, and code for running QAOA with said errors. The most important files and folders of `HQAOA.zip` are as follows.

- `qem.py` This script contains the code for the quantum simulator used by `HQAOA.py`. QEM stands for Quantum EMulator. i
- `HQAOA.py` Run QAOA from the command line. In a Unix shell, go to the correct folder and run `\$ python3 HQAOA.py -h` for detailed information on the command line input options. This script generated all data in the data folder and in the paper. HQAOA stands for HeisenbergQAOA.
- `data/` Folder containing all generated data. The data pertaining to the plots in the paper are in `data/SK/SWAP-network/6/<spatial or temporal >/<instance bit string>/output.txt`. That is, it contains the data pertaining to Sherrington-Kirkpatrick (SK) problem instances, solved by the SWAP-network implementation of QAOA on 6 qubits, with either spatially or temporally correlated errors, for the instance with the given instance bit string. The error correlation type (spatial or temporal) is ambiguous for uncorrelated errors. Without loss of generality, we chose to categorize uncorrelated errors as spatially coddrrelated errors with vanishing correlation strength.
- `analysis/` Folder containing scripts for the analysis of the data. The script `instance_properties.py` computes the instance properties listed in Appendix C of the main text, `derivatives.py` computes the cost function landscape noise susceptibility [Eq. (23) of the main text, Figs. 4 and 5 of the main text], and `pmerrors.py` computes the probability that, for given $p$, a single fluctuator causes $m$ errors, as a function of $m$ (not reported in the paper). The plots in the main text were generated by `kappa_plot.py` and `p_plot.py`. The plots in this Supplemental Material were generated by `plots.py`.
- `environment.yml` To make sure Python and all the needed packages are installed, install Miniconda [2] or Anaconda [3] and set up a Python environment [4] from `environment.yml`. Alternatively, run `\$ python3 test_qem.py` and `\$ python3 test_HQAOA.py` until there are no unknown packages.
- `test/` Folder containing all tests. Change directory to `test/` and run `\$ python3 test_qem.py` and `\$ python3 test_HQAOA.py` to test (your installation of) `qem.py` and `HQAOA.py`, respectively.

Figures 1 to 16 visualize all generated data. The problem instances occur in the order of Table I of the main text. In the main text, plots pertaining to the instance 0100101001111110 are shown. For visual clarity, the plots in the main text show fewer (but regularly distributed) data points than the plots pertaining to that same instance in the current Supplemental Material. This lower density of data points has no effect on the trends that the data indicate.

The data indicate that the global optimization routine (Sec. IV of the main text) consistently found the global optimum of the cost function landscape. The global routine employs inherent randomness, as 32 independent optimization runs are started per data point at random initial points. Furthermore, each of these 32 independent optimization routines is itself a basin-hopping routine with added randomness and the ability to traverse large parts of the cost function landscape [5, 6]. The global optimization routine demonstrates the ability to explore the *entire* cost function landscape, as observed from the large variation of the raw optimal parameters (i.e., parameters that are unconstrained during optimization and not considered modulo the symmetries) for data points close to each other in $\kappa$. The variations are larger than the periods of the parameters arising from the symmetries of the cost function landscape (Appendix B of the main text).

Despite the randomness and reach of the optimization routine, the outcomes for different data points (Figs. 1–16) are highly regular. If the routine had not found the global optimum consistently, larger jumps in the optimal cost function values would be expected. Furthermore, considering the raw optimal parameters modulo the symmetries of the cost function landscape (Appendix B 2 of the main text) collapses these parameters to the same parameter representatives for various ranges of $\kappa, p$, showing additional regularity in the (processed) optimal parameters. Therefore, the data in our work are independent of the optimization routine being used, as long as the optimization routine consistently finds the global optimum of the cost function landscape.

There are two instances, 011100011010011 and 111100111011001, for which the data do not show a purely monotonic increase in the AR as a function of $\kappa$ due to a sudden and localized drop in the AR. Because of the otherwise fully regular behavior, and because a similar drop is not observed for other instances in the same cost function landscape class, we attribute these drops to a rare instance of imperfect global optimization. The instance 011101001011100 is the only instance for which there is a $p$ such that the AR for fully spatially correlated errors is higher than the AR for fully temporally correlated errors. We have not investigated this slight deviation from the apparent general qualitative behavior of the other instances further because the deviation occurs only at extremely high error probabilities.

The Euclidean distance $D$ [Eq. (B6) of the main text] between the noise-unaware and the noise-aware parameters is shown for all data points. Without the removal of redundant freedom in the parameters by Algorithm I, $D$ fluctuates wildly as a function of $p$ and $\kappa$. The high stability shown for small $p$ indicates that the generators listed in Appendix B of the main text generate the whole symmetry group of the QAOA SK cost function landscape. We attribute the finding that $D$ starts to fluctuate for larger $p$ to the fact that with increasing $p$, the cost function becomes increasingly shallow, causing many competing local minima with a practically identical value of the cost function landscape for large $p$.

FIGS. 1–16. Performance of the SWAP-network implementation of QAOA with $r = 3$ cycles, for 16 random SK instances, under the influence of the spatial and temporal fluctuator models. Per instance, the cost function landscape class $\tilde{c}$, the instance bit string $\tilde{\omega}$, and the following plots are shown.

(a) Top: the approximation ratio (AR) as a function of the error probability $p$, for fully ($\kappa = 1$) temporally correlated errors (orange), fully spatially correlated errors (green) and uncorrelated errors (blue). Open circles show the noise-aware AR, whereas the filled circles show the noise-unaware AR.
Bottom: the Euclidean distance $D$ [Eq. (B6) of the main text] between the noise-unaware parameters and the noise-aware parameters. Note that the noise-unaware parameters do not depend on $p$. Parameter symmetries are removed using Algorithm I of the main text. The color coding is as in the upper plot.

(b) As (a), but restricted to the domain $p \in [0, 0.15]$ for more detail for low error probabilities.

(c) Top: the AR as a function of $\kappa$ at $p = 0.001$ fixed, for temporally correlated errors (orange) and spatially correlated errors (green).
Bottom: as in the bottom plot of (a), now as a function of $\kappa$ at $p = 0.001$ fixed.

(d) As (c), but at $p = 0.01$.

FIG. 1. $\tilde{c} = 0$ $\tilde{\omega} = 000100100011110$

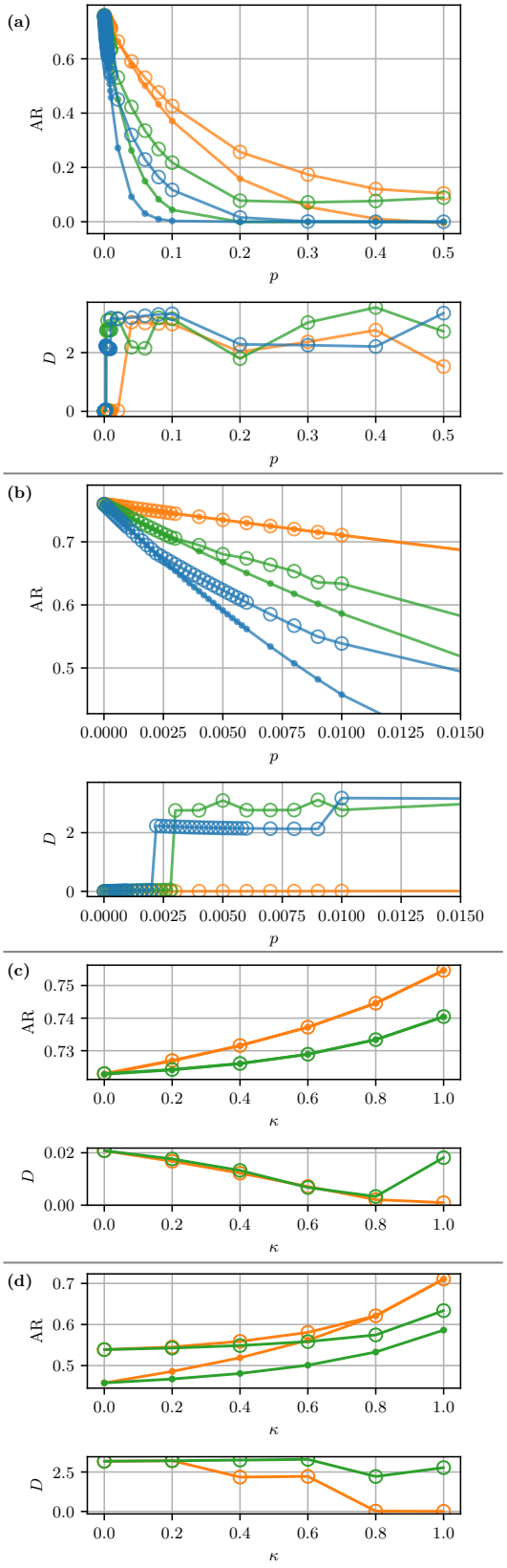


FIG. 2. $\tilde{c} = 1$ $\tilde{\omega} = 001100110101110$

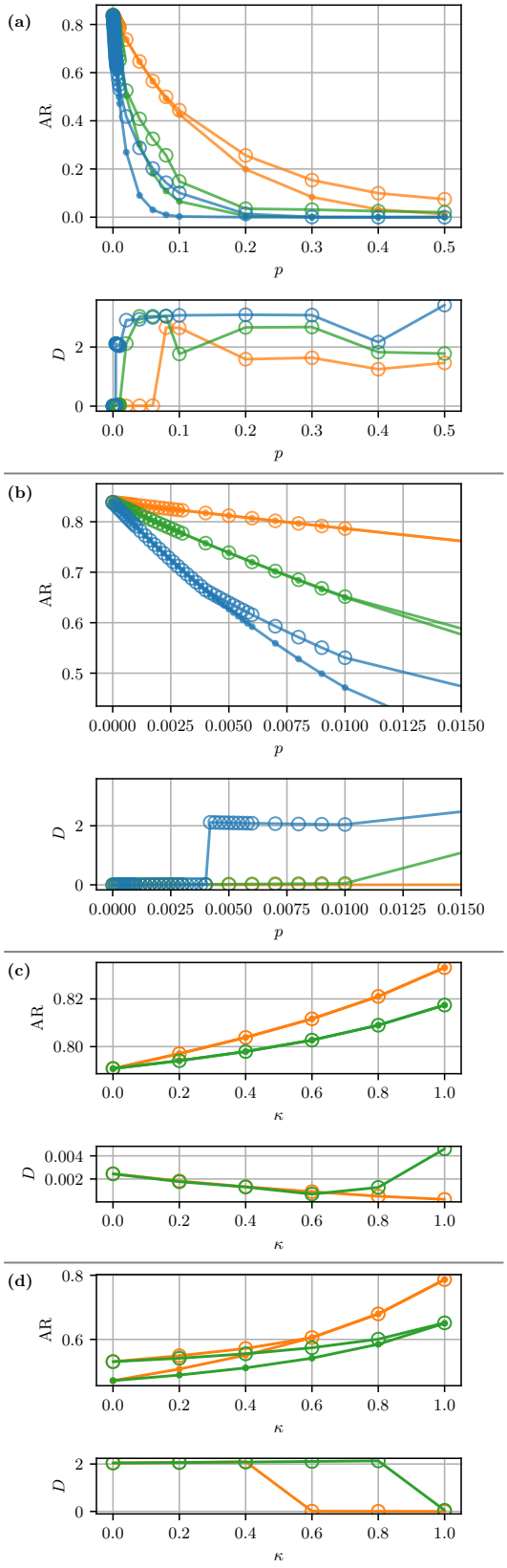



FIG. 3. $\tilde{c} = 1$ $\tilde{\omega} = 001111110000010$

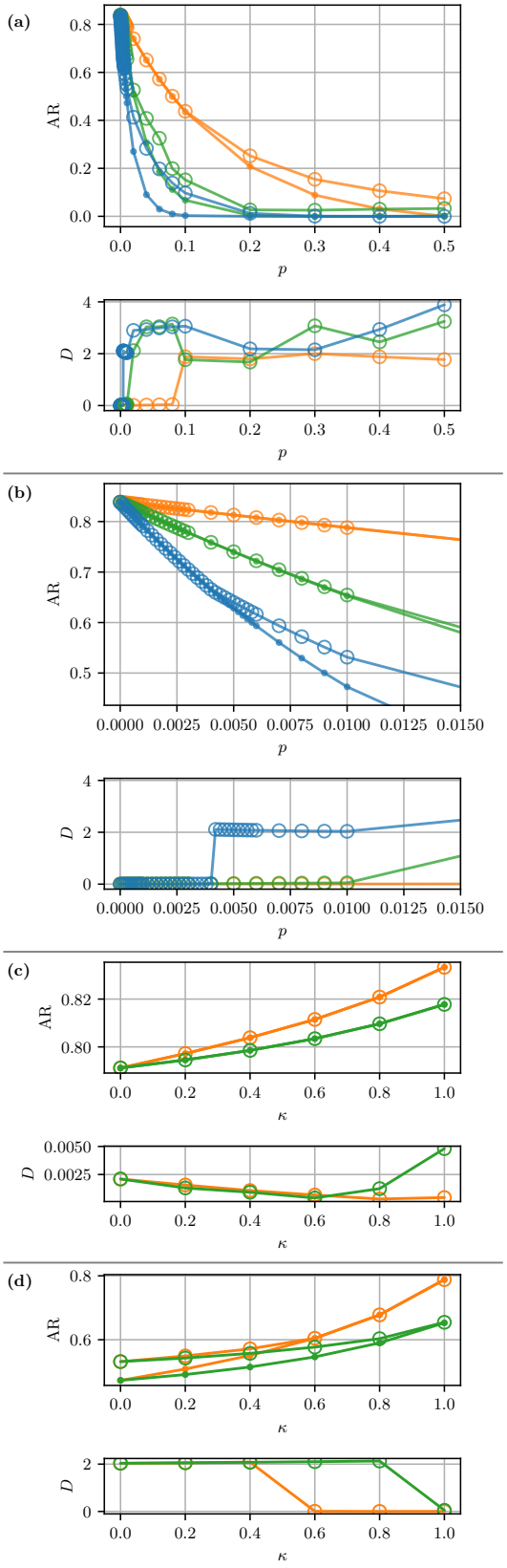


FIG. 4. $\tilde{c} = 1$ $\tilde{\omega} = 010011011101010$

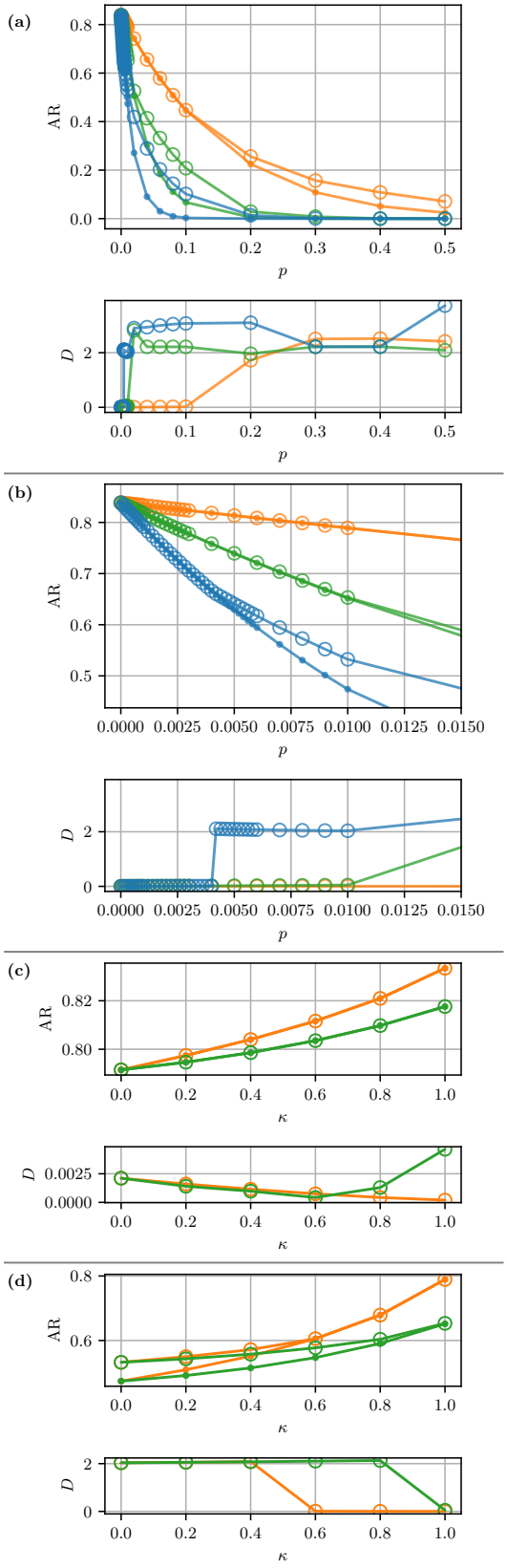



FIG. 5. $\tilde{c} = 2$ $\tilde{\omega} = 001111011010001$

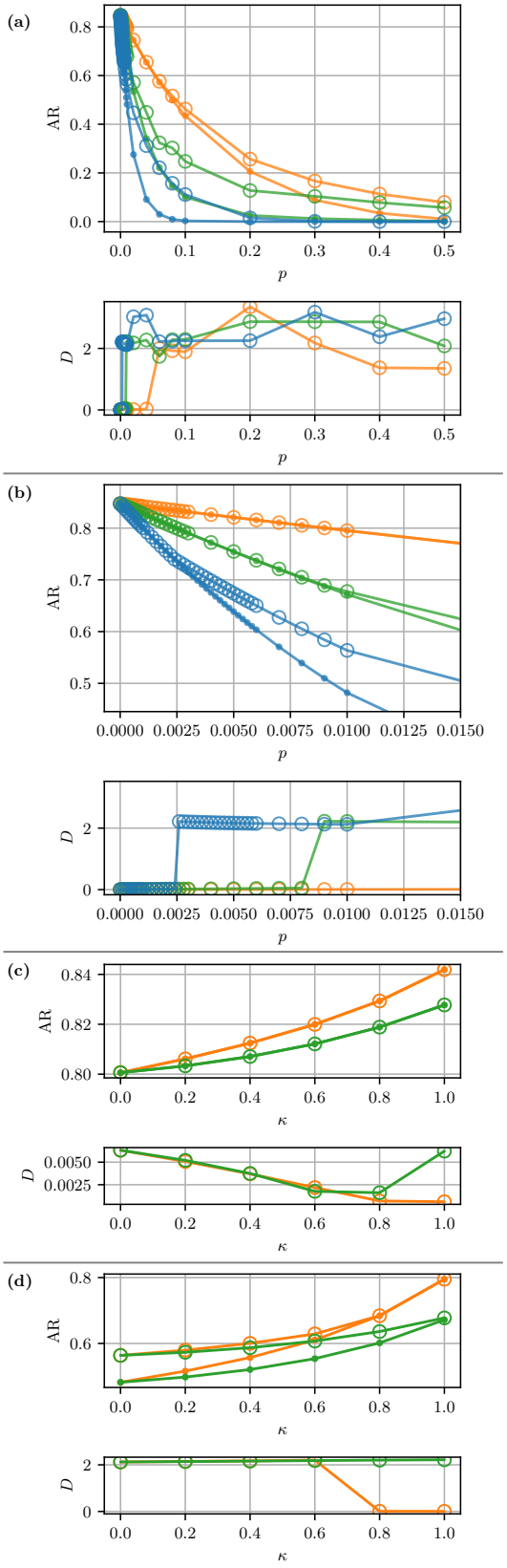


FIG. 6. $\tilde{c} = 2$ $\tilde{\omega} = 010100101011100$

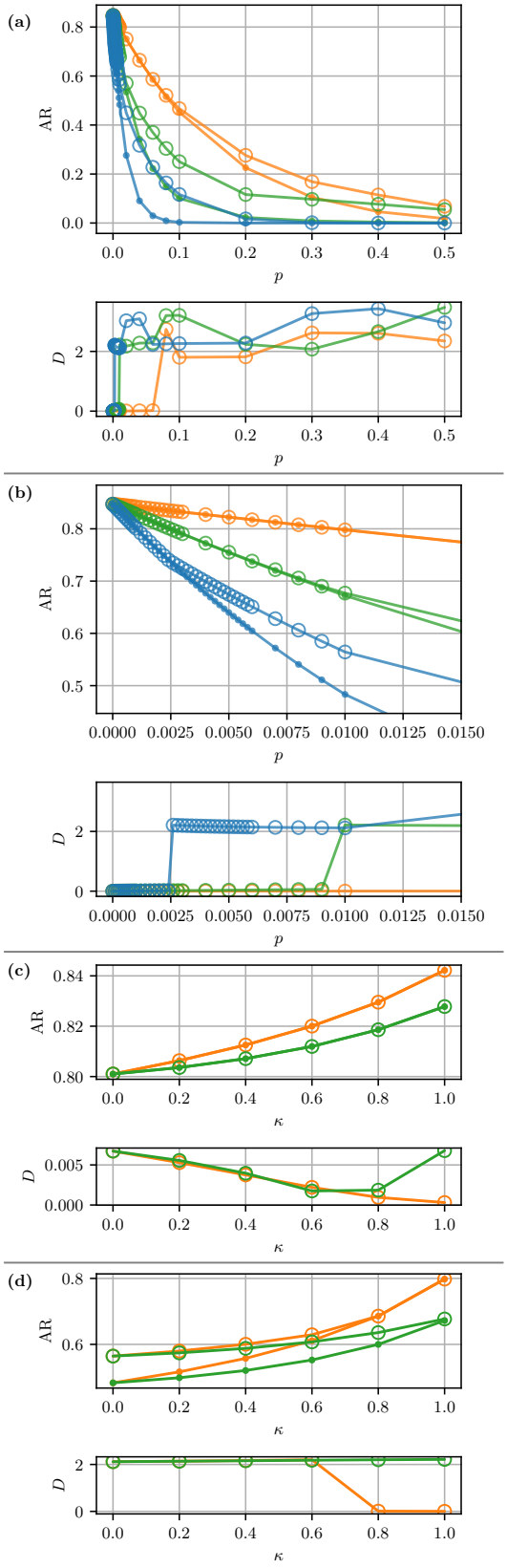



FIG. 7. $\tilde{c} = 2$ $\tilde{\omega} = 100010101000100$

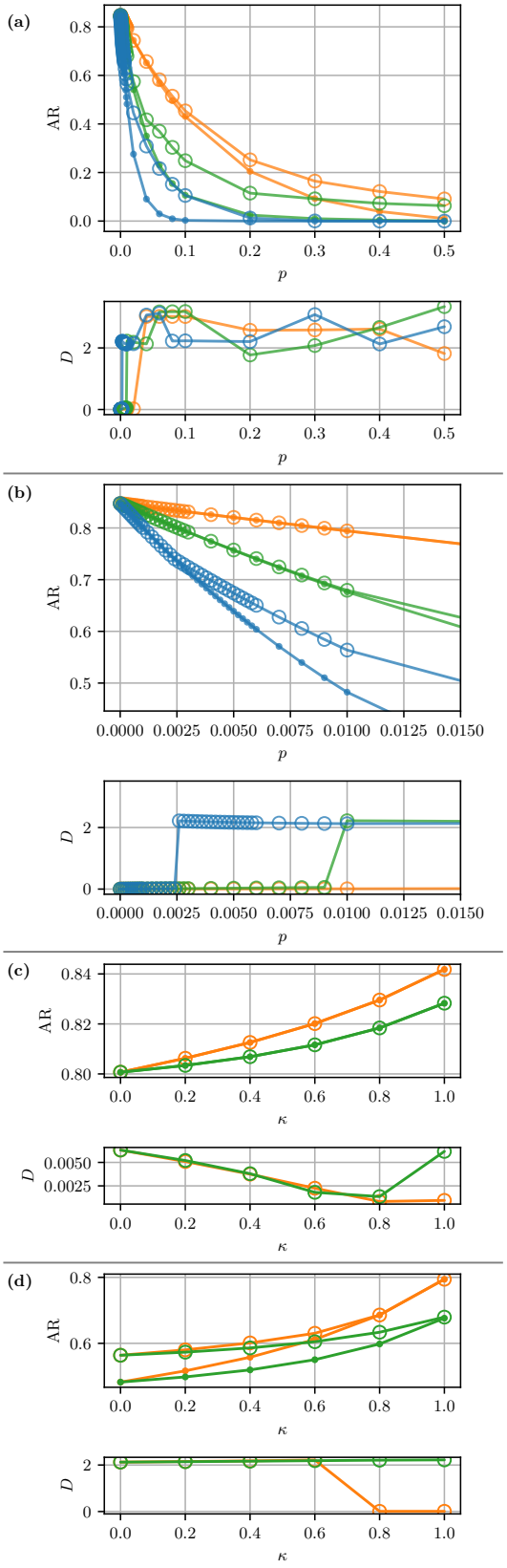


FIG. 8. $\tilde{c} = 2$ $\tilde{\omega} = 110010010101000$

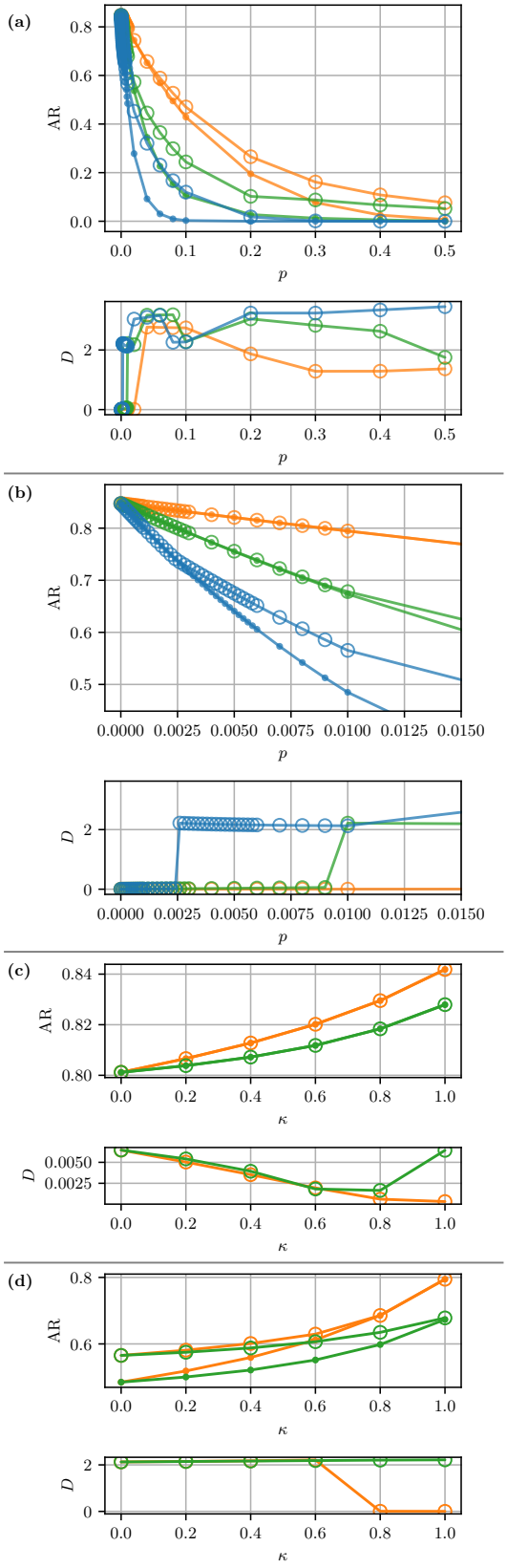



FIG. 9. $\tilde{c} = 3$ $\tilde{\omega} = 010010100111110$

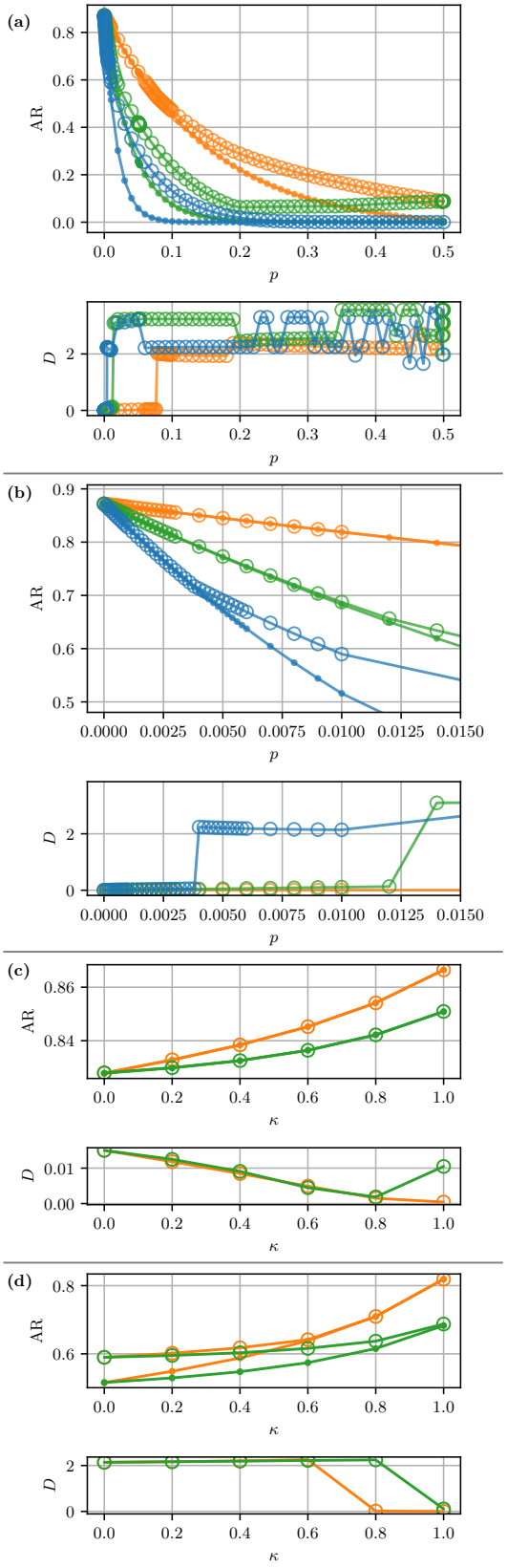


FIG. 10. $\tilde{c} = 3$ $\tilde{\omega} = 011100011010011$

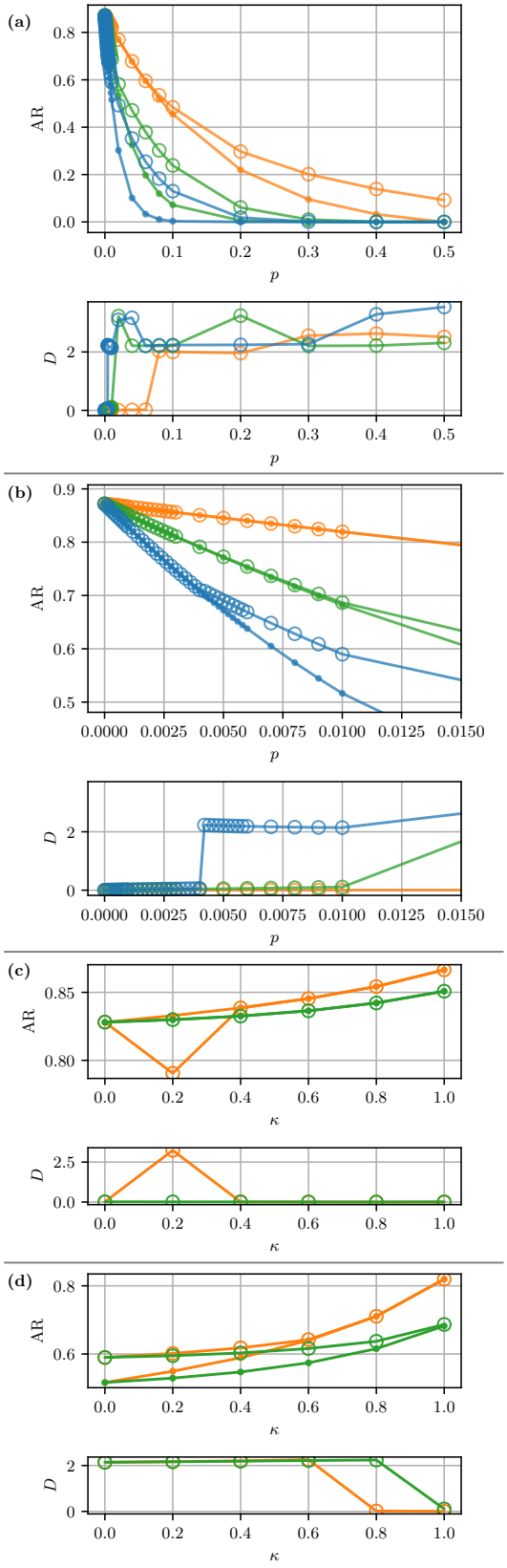



FIG. 11. $\tilde{c} = 3$ $\tilde{\omega} = 100110010001101$

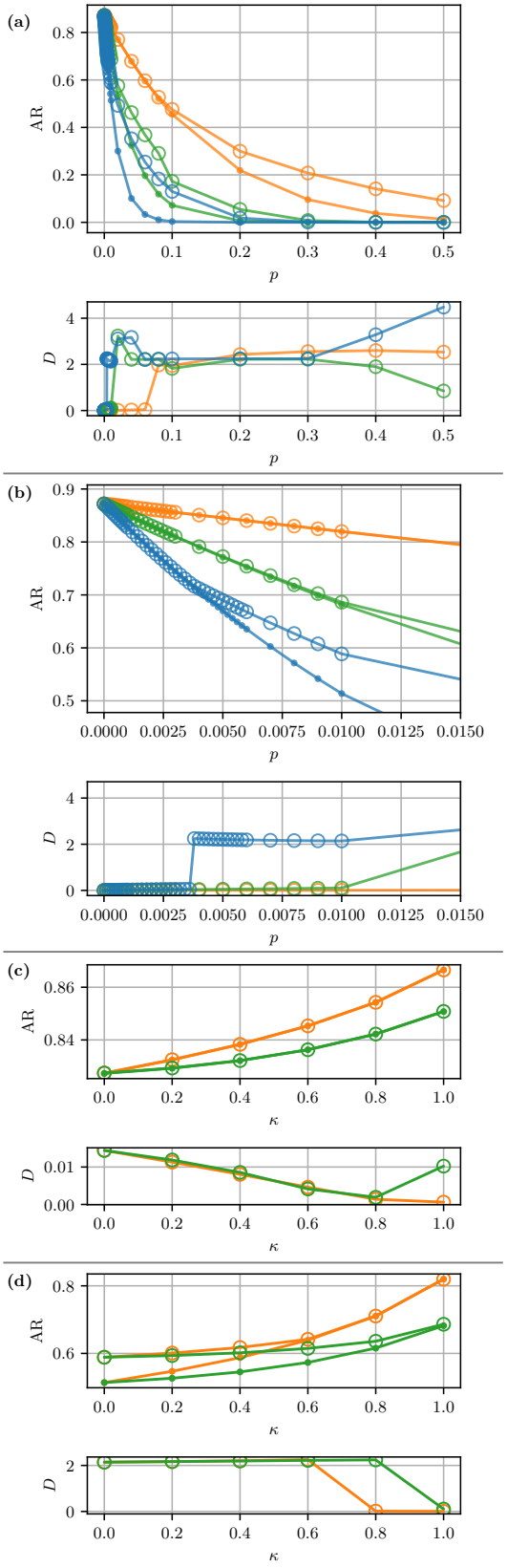


FIG. 12. $\tilde{c} = 3$ $\tilde{\omega} = 101100001010110$

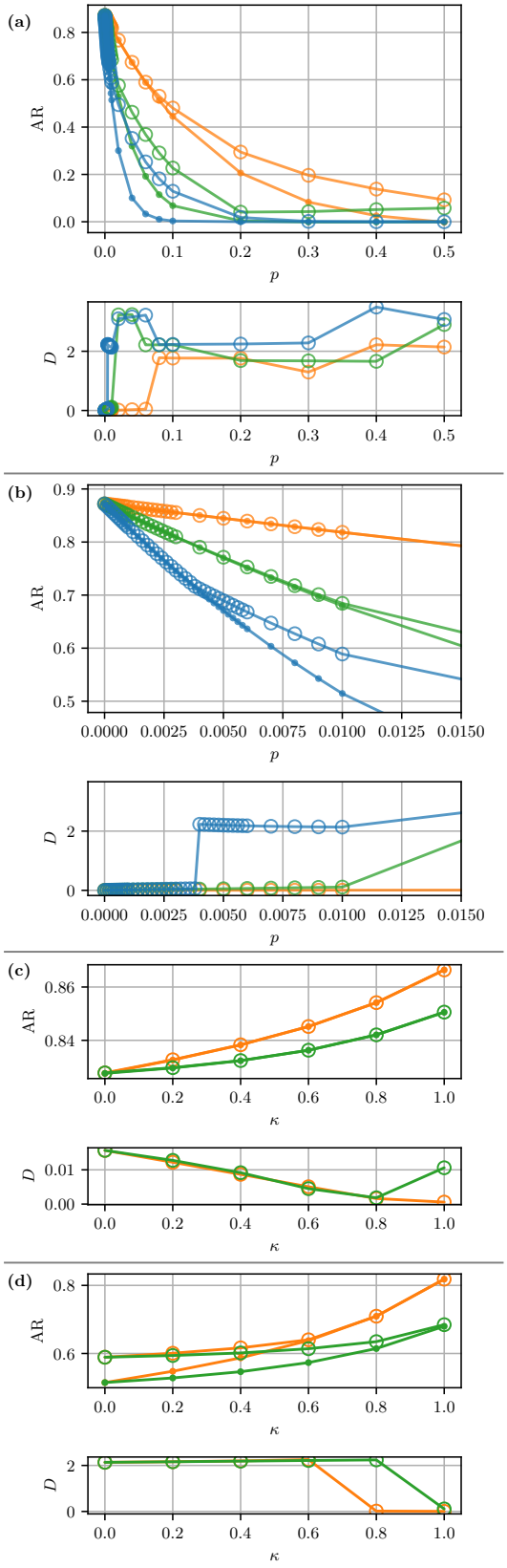



FIG. 13. $\tilde{c} = 3$ $\tilde{\omega} = 111011011000010$

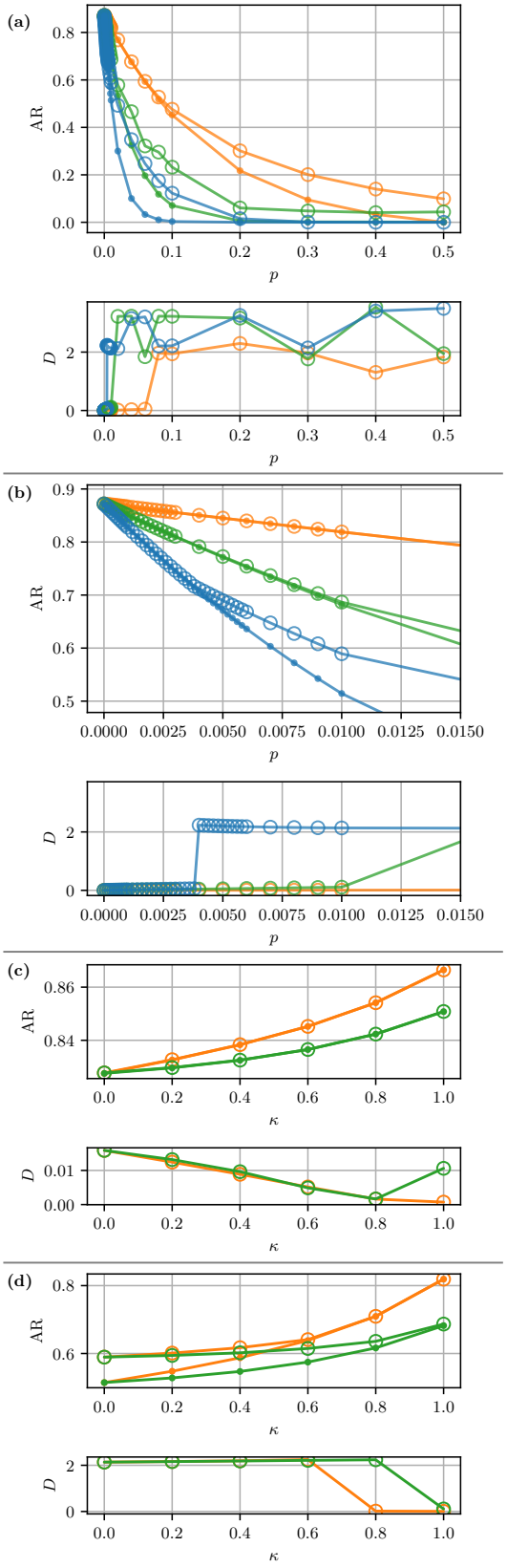


FIG. 14. $\tilde{c} = 3$ $\tilde{\omega} = 111100111011001$

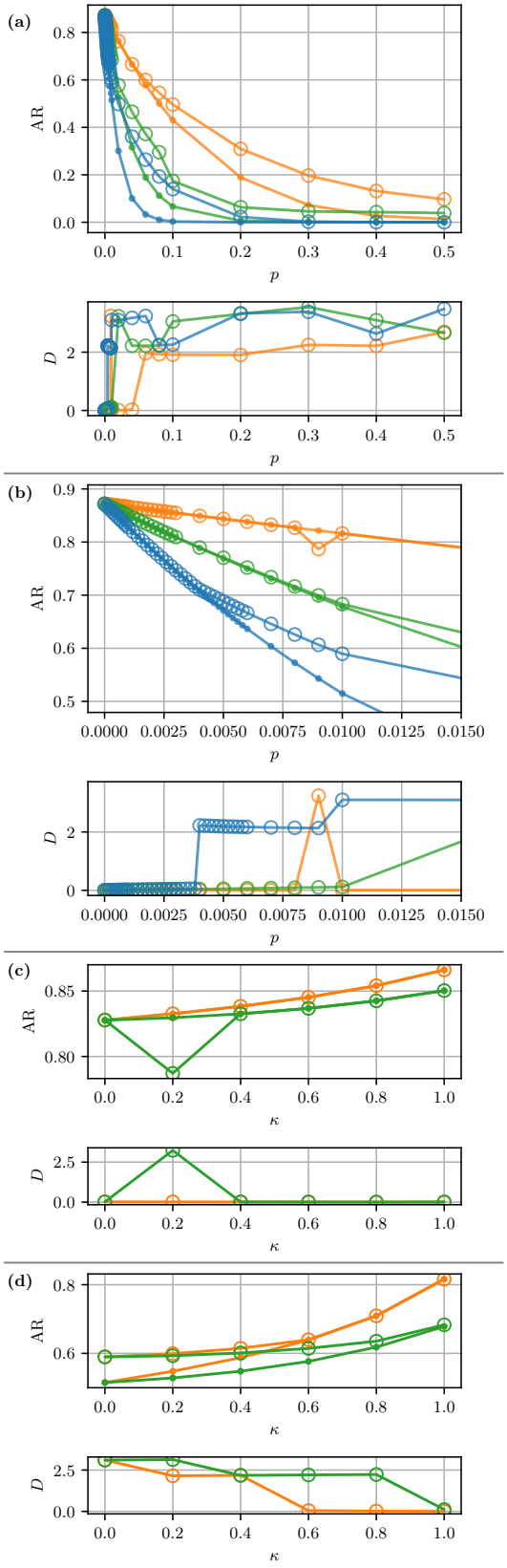



FIG. 15. $\tilde{c} = 4$ $\tilde{\omega} = 011101001011100$

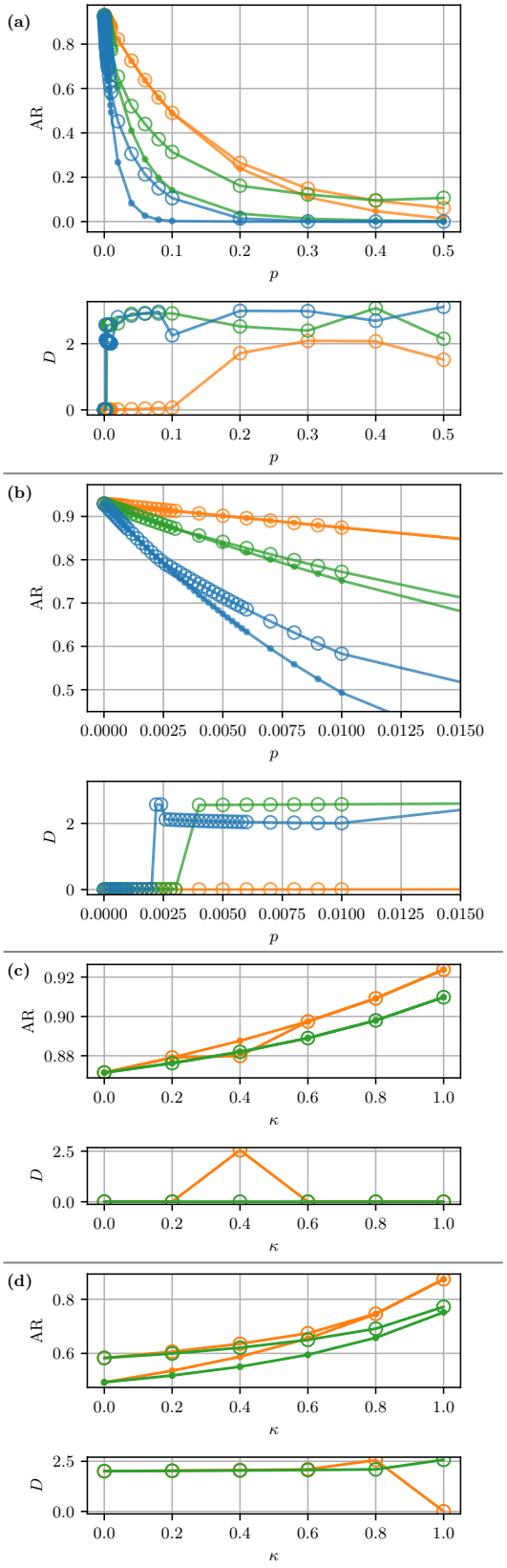
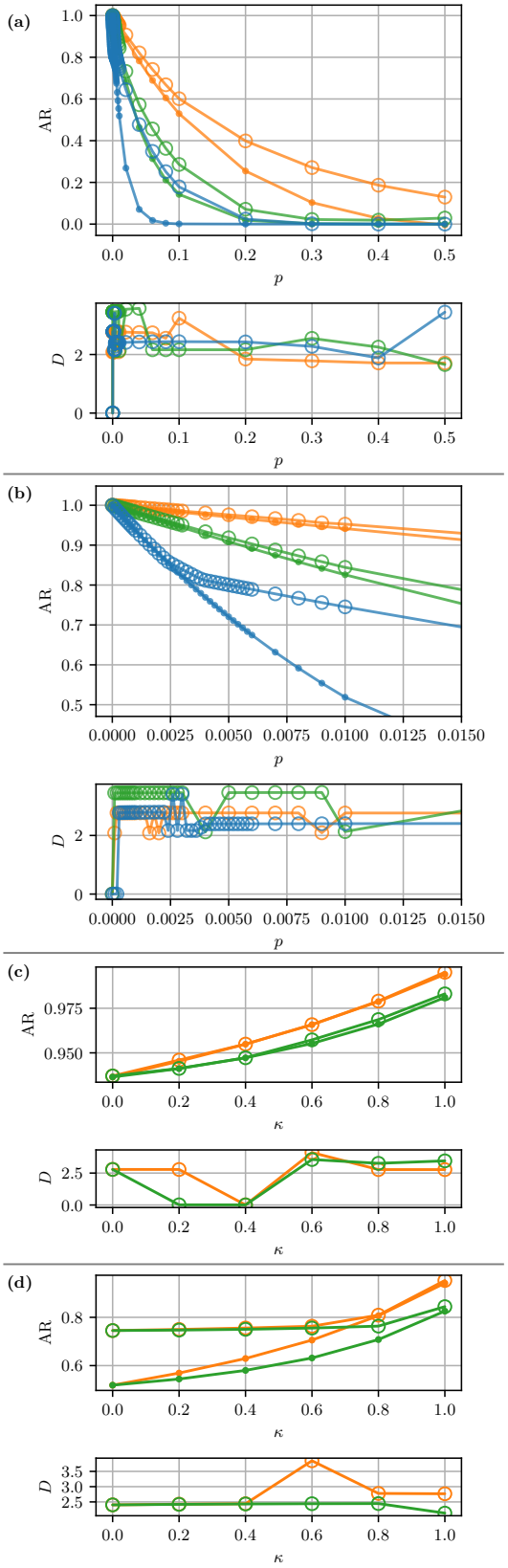


FIG. 16. $\tilde{c} = 5$ $\tilde{\omega} = 110101111100011$



[1] Joris Kattemölle and Jasper van Wezel. Variational quantum eigensolver for the Heisenberg antiferromagnet on the kagome lattice. *Phys. Rev. B*, 106:214429, Dec 2022.

[2] Miniconda. https://docs.conda.io/en/latest/miniconda.html, 2022.

[3] Anaconda. https://docs.anaconda.com/anaconda/install/, 2022.

[4] Conda documentation: creating an environment from an environment yml file. https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#creating-an-environment-from-an-environment-yml-file, 2022.

[5] David J. Wales and Jonathan P. K. Doye. Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *The Journal of Physical Chemistry A*, 101(28):5111–5116, jul 1997.

[6] Scipy 1.5.2 documentation, Accessed July 2022.