Learning Bounds on Computational Values in Iterative Methods using Reachability Analysis

Mukund Verma, Ian McInerney, and Ludovic Renson

Department of Mechanical Engineering, Imperial College London,
London, UK, SW7 2AZ.
`mukund.verma20@imperial.ac.uk`, `i.mcinerney17@imperial.ac.uk`,
`l.renson@imperial.ac.uk`

Abstract. In this extended abstract, we present our initial work on how the reachable sets of the Koopman operator for an iterative method can be approximated and then used to make decisions about the number formats required for implementations. We present our initial framework for learning the Koopman operator and performing reachability analysis on it, followed by an illustrative example on the Gauss-Seidel stationary iterative method, where the reachability analysis can inform the decision to use signed/unsigned variables.

Keywords: iterative methods, Koopman operator, reachability analysis

1 Introduction

The importance of understanding the numerical behavior of iterative methods is growing as the proliferation of accelerators such as GPUs and FPGAs (Field Programmable Gate Arrays) increases. Each new generation of accelerators brings new numerical formats — from novel floating-point formats like TensorFloat32, BFloat16 and the recently announced 8/4-bit MX-Floats, to the resurgence of 8/4-bit integer formats, there has been a growing trend of shrinking the number of bits used to represent numbers. Additionally, custom hardware accelerators, such as FPGAs, allow engineers to implement their own custom floating-point formats, integer data types, or even newer non-standard formats such as Posits.

Using these new smaller number formats for algorithms may lead to problems though, since these formats have a smaller *representable range* (the largest and smallest value that can be represented) leading to overflow problems, and larger *unit round-off* (the distance between two adjacent representable numbers). Analysing how algorithms will behave with these smaller formats is essential, since algorithms that previously worked in higher precision may become inaccurate, or even fail to converge to the correct result when using these smaller number formats.

In general, the majority of the prior work in the numerical analysis of algorithms in floating-point has ignored the representable range, since the double/single precision formats found in modern computers have a representable range on the order of $\pm 10^{308}$ and $\pm 10^{38}$, respectively, which is large enough to not affect most computations. Instead, the focus was on the effect of round-off error on iterative methods, building frameworks such as backwards error analysis to model the propagation of round-off errors through the computations in algorithms. Designers of custom hardware for control have examined the overflow behavior of algorithms, deriving analytical guarantees for algorithms such as linear system solvers [13] and optimization solvers [14]. Recently, however, the availability of the smaller number formats, such as Float16, in modern devices has brought the subject of overflow/underflow for numerical algorithms in floating point back into more mainstream focus [16].

In this work, we propose analyzing the representable range necessary to implement specific algorithms by performing reachability analysis on a Koopman operator linearization of the algorithm. We propose using data-driven Koopman operator estimation for this analysis to allow for building the Koopman operator of complex, or even black-box, numerical codes by only capturing the necessary algorithm state information at each iteration. The learned Koopman operator is then used to compute the reachable sets of the iterative method, which then provide the relevant information to determine what data types will have a suitable representable range.

2 Background

We begin with background on several elements and ideas we use in this work, including the Koopman operator and its estimation, and the idea of viewing algorithms as dynamic systems for analysis.

2.1 Dynamics of Algorithms

Iterative methods, such as linear system solvers and optimization solvers, can be viewed as a nonlinear mapping $\mathcal{F}$ from the algorithm state at iteration k, called s^k, to the algorithm state at iteration $k{+}1$, called s^{k+1}, of the form

$$s^{k+1} = \mathcal{F}(s^k). \tag{1}$$

The actual state update operations done by the mapping $\mathcal{F}$ can take any form, ranging from the simple linear update of the Jacobi method solving $As = b$, through the single-equation nonlinear update of a simple Newton's method, to the complicated multi-step update in solvers such as the Generalized Minimum Residual (GMRES) method for solving linear systems. The nonlinear mapping (1) can be naturally associated with a nonlinear discrete-time system

$$x^{k+1} = F(x^k) \tag{2}$$

that relates the current states at time sample k to the next states at time sample $k{+}1$ through a map $F : \chi \to \chi$.

Viewing algorithms as discrete-time systems allows for tools from dynamical systems theory and control theory to be used to both analyze and design iterative algorithms [9]. Algorithm convergence can be studied using popular tools in the analysis of dynamical systems, like Lyapunov methods [23], dissipativity theory [18], and input-to-state stability in the presence of disturbances [11], while tools like Integral Quadratic Constraints [21] and methods like PID controllers [5] can be used to design stable and robust algorithms.

Prior works have analytically examined the reachability and controllability of some numerical linear algebra methods, such as showing the QR decomposition is stable when used with Hessenburg matrices [15], or determining the classes of matrices that lead to a stable Rayleigh iteration in eigenvalue methods [12].

2.2 Koopman Operator

In this work, we focus on discrete-time dynamical systems, since the iterations of an iterative method can be mapped to the discrete time steps of the discrete-time system (2). When F is a linear map $F(x) = Ax$, the system (2) simplifies to the linear state update equation

$$x^{k+1} = Ax^k. \tag{3}$$

While the analysis of the linear system (3) is well-studied, analyzing the properties of the nonlinear system F in (2) is a more difficult task, and many techniques resort to analyzing a linearization of the nonlinear system instead.

One such linearization is the *Koopman operator*, which is an infinite-dimensional linear operator $\mathcal{K} : \mathcal{G}(\chi) \to \mathcal{G}(\chi)$ that operates in a space of *observables* formed by the set of measurement functions $\mathcal{G}(\chi)$ composed of individual functions $g : \chi \to \mathbb{R}$ [7]. This space of observables follows the underlying state space χ, so that

$$\mathcal{K}g(x) = g(F(x)). \tag{4}$$

The Koopman operator $\mathcal{K}$ can, in theory, perfectly represent the nonlinear dynamics of F as an infinite-dimensional linear operator. However, this infinite-dimensional operator is computationally intractable, and so using the Koopman operator to analyze a system is done using a finite-dimensional approximation of $\mathcal{K}$ using a finite number of observable functions, creating a *truncated Koopman operator*.

The choice of these observable functions is crucial to the accuracy of the truncated Koopman operator, and how to choose them is still an active area of research [7]. Common observable functions used include radial basis functions and monomials [4], while recent work has investigated using machine learning techniques to learn the observable functions alongside the truncated Koopman operator [10]. However, in cases where the structure/equations of the underlying dynamics is known, a suitable dictionary of observable functions can be analytically derived, possibly leading to the truncated Koopman operator being a perfect approximation of the original nonlinear system [6].

The use of the Koopman operator to approximate dynamical systems has found applications in both the analysis and control of nonlinear systems. For instance, it is routinely used as an analysis tool to locate the regions of attraction for nonlinear systems and determine their stability [19], and to compute the reachability of nonlinear systems. [17, 2, 1].

Koopman operators of algorithms Combining data-based/learning methods with a system-theoretic view of algorithms has recently appeared as a new research area. The authors of [8] proposed the construction of Koopman operators for algorithms, and then used them to compute the regions of stability for the original algorithm. The spectrum of the Koopman operator was used in [20] to identify commonalities and relations between algorithms to build taxonomies without requiring access to the underlying equations. Finally, recent work in [22] built Koopman operators for cryptographic algorithms such as the Diffie-Hellman key exchange and Rivest-Shamir-Adleman cryptosystems, approximating these highly nonlinear methods with a linear Koopman operator that was then used to study the methods and derive a (computationally-intractable) approach for decrypting the systems.

3 Learning Algorithm Reachability

We propose learning a Koopman operator representation for numerical methods and then using dynamical systems reachability analysis to analyze the reachability of this operator to determine the representable range needed for a numerical method.

3.1 Koopman Approximation

The Koopman operator for the algorithm is computed using the Extended Dynamic Mode Decomposition (eDMD) on generated datasets containing the algorithm state trajectories for various initial conditions and algorithm parameters. The choice of what quantities are included in the algorithm state is crucial to the accuracy of the learned operator, and generally should contain more than just the current iterate and may need to include “hidden states” internal to the method. For example, algorithms with adaptive step-sizes should include the step-size as a state to ensure the operator can learn how the step-size affects the growth/shrinkage of the iterates.

The observables used when learning the Koopman operator also play an important role. In this work, we currently combine two sets of observables: Gaussian Radial Basis Functions (RBFs) and the original system states. The Gaussian RBF was chosen since it has generally been used in other works as an observable when there was no prior information available to guide the choosing of the observable functions. For the RBFs, the centers used were placed randomly in the algorithm state space, however other methods, such as clustering, could be used to place the centers. In addition to the RBFs, we place the original states as observables to aid in the reconstruction of the state trajectories from the Koopman operator [7].

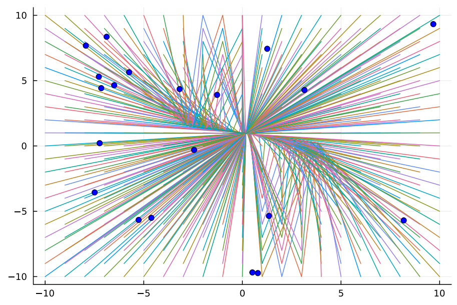


Fig. 1. Trajectories used for learning the Koopman operator. The centers of the RBFs are marked as blue circles.

3.2 Reachability Analysis

After computing the linear Koopman operator, we use it to perform a forward reachability analysis for the original iterative method. In this work, we perform the reachability analysis using Monte-Carlo methods – generating initial points inside a desired starting set, lifting them to the Koopman space, and then propagating them through the linear operator.

While other methods for linear reachability could be applied to the Koopman operator, care must be taken to ensure the starting set is properly constructed in the lifted space. One such method proposed in [1] builds an interval over-approximation of the initial set in the lifted space by solving a satisfiability problem, and then solves a linear program with the sets as constraints. Another proposed method uses Taylor model arithmetic to lift the initial set and then converts the set to a zonotope representation to apply standard linear system reachability techniques [2, 3].

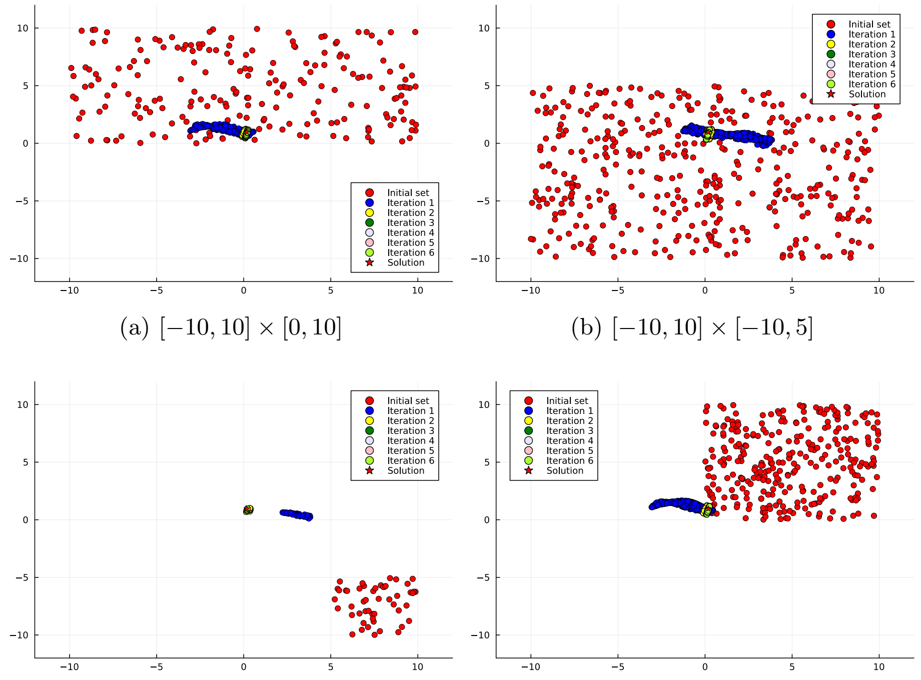
3.3 Examples

As this is early work, we focus on a simple example that can help explain the proposed method, leaving more complex examples for future work.

The algorithm we examine is the Gauss-Seidel stationary iterative method for computing the solution of a linear system $Ax = b$ of dimension 2 with $A = \left[\begin{smallmatrix} 6 & 2 \\ 1 & 5 \end{smallmatrix}\right]$ and $b = \begin{bmatrix} 3 \ 5 \end{bmatrix}^{\mathrm{T}}$. The algorithm state used to learn the Koopman operator is simply the value of x at each iteration. We sample the algorithm's trajectories on a uniform grid in the region $\mathcal{I} := [-10, 10] \times [-10, 10]$, and then place 20 RBFs randomly into the region $\mathcal{I}$ as the observable functions. The trajectories used to learn the Koopman operator, and the centers for the RBF observables, can be seen in Figure 1.

After learning the Koopman operator, we use the following steps to compute reachable sets:

1. Define the non-lifted set.
2. Sample points in the non-lifted set.



(a) $[-10, 10] \times [0, 10]$ (b) $[-10, 10] \times [-10, 5]$

(c) $[5, 10] \times [-10, -5]$ (d) $[0, 10] \times [0, 10]$

Fig. 2. Simulated reachability from initial conditions using the Koopman operator

3. Lift the sampled points to the observable space.
4. Apply the Koopman operator to each sampled point.
5. Extract the reachable set using the state observables.

The results of the reachability analysis can be seen in Figure 2. Using these sets, we can start to make qualitative observations about the behavior of the Gauss-Seidel method applied to this problem. We can observe the regions that the trajectories are in, with those that start in the upper-half of the space staying inside that region, as shown in Figure 2a, and those starting in the region $[-10, 10] \times [-10, 5]$ staying inside that region (shown in Figure 2b). We can then use this information to inform our choice of number systems to use when storing the x and y values in Gauss-Seidel - with Figures 2a and 2d showing that y will be positive if it starts positive but x can go negative if it starts positive - implying we can use unsigned numbers for y but must use signed numbers for x.

4 Future work & Discussion

In this work, we presented the initial framework for using the reachability of the Koopman operator to determine required properties of the number formats used to implement algorithms. This work merely scratches the surface of this field, though, and there are still many avenues and open questions that we plan to explore.

The next stages of this work will explore how to perform the Koopman lifting for more complicated algorithms, such as conjugate gradient methods and optimization solvers. This will require determining better methods for observable function selection (e.g., possibly using learned observables as discussed in [7]), exploring how the "hidden" states affect the approximation and subsequent reachability analysis, and generalizing the method to handle classes of problems instead of individual problem instances.

After this, the key question is how to perform the reachability analysis and produce a meaningful result for implementers/hardware designers. In this work, we simply performed a Monte-Carlo reachability analysis, however that does not necessarily provide a certifiable statement about the values the algorithm will obtain. Instead, set preserving methods for reachability analysis using the Koopman operator should be explored to provide a certifiable analysis of the algorithm's necessary representable range. This will require a more thorough investigation of the various reachability methods available, such as looking at techniques in the literature for performing a set-preserving lifting into zonotopes in the observable space that are then propagated through the Koopman operator [1, 3] or doing reachability analysis directly during the computation of the Koopman operator [17].

Following this, we would like to build an optimization-based framework that can be used to find the smallest number formats that can be used in numerical methods for various problem classes.

References

1. Bak, S., Bogomolov, S., Duggirala, P.S., Gerlach, A.R., Potomkin, K.: Reachability of Black-Box Nonlinear Systems after Koopman Operator Linearization. IFAC-PapersOnLine **54**(5), 253–258 (2021)
2. Bak, S., Bogomolov, S., Hekal, A., Kochdumper, N., Lew, E., Mata, A., Rahmati, A.: Falsification using Reachability of Surrogate Koopman Models. In: Proceedings of the 27th ACM International Conference on Hybrid Systems: Computation and Control, HSCC '24, pp. 1–13. Association for Computing Machinery, Hong Kong, China (2024)
3. Bak, S., Bogomolov, S., Hencey, B., Kochdumper, N., Lew, E., Potomkin, K.: Reachability of Koopman Linearized Systems Using Random Fourier Feature Observables and Polynomial Zonotope Refinement. In: S. Shoham, Y. Vizel (eds.) Computer Aided Verification, 34th International Conference, CAV 2022, CAV: International Conference on Computer Aided Verification, pp. 490–510. Springer International Publishing, Cham (2022)
4. Bevanda, P., Sosnowski, S., Hirche, S.: Koopman operator dynamical models: Learning, analysis and control. Annual Reviews in Control **52**, 197–212 (2021)
5. Bhaya, A., Kaszkurewicz, E.: Iterative methods as dynamical systems with feedback control. In: 42nd IEEE Conference on Decision and Control (CDC), vol. 3, pp. 2374–2380. IEEE, Maui, HI (2003)

6. Brunton, S.L., Brunton, B.W., Proctor, J.L., Kutz, J.N.: Koopman invariant subspaces and finite linear representations of nonlinear dynamical systems for control. PLoS ONE **11**(2), 1–19 (2016)
7. Brunton, S.L., Budišić, M., Kaiser, E., Kutz, J.N.: Modern Koopman Theory for Dynamical Systems. SIAM Review **64**(2), 229–340 (2022)
8. Dietrich, F., Thiem, T.N., Kevrekidis, I.G.: On the Koopman Operator of Algorithms. SIAM Journal on Applied Dynamical Systems **19**(2), 860–885 (2020)
9. Dörfler, F., He, Z., Belgioioso, G., Bolognani, S., Lygeros, J., Muehlebach, M.: Towards a Systems Theory of Algorithms. arXiv **2401.14029** (2024)
10. Hao, W., Huang, B., Pan, W., Wu, D., Mou, S.: Deep Koopman learning of nonlinear time-varying systems. Automatica **159**, 111,372 (2024)
11. Hasan, A., Kerrigan, E.C., Constantinides, G.A.: Control-Theoretic Forward Error Analysis of Iterative Numerical Algorithms. IEEE Transactions on Automatic Control **58**(6), 1524–1529 (2013)
12. Helmke, U., Wirth, F.: On controllability of the real shifted inverse power iteration. Systems & Control Letters **43**(1), 9–23 (2001)
13. Jerez, J.L., Constantinides, G.A., Kerrigan, E.C.: A Low Complexity Scaling Method for the Lanczos Kernel in Fixed-Point Arithmetic. IEEE Transactions on Computers **64**(2), 303–315 (2015)
14. Jerez, J.L., Goulart, P.J., Richter, S., Constantinides, G.A., Kerrigan, E.C., Morari, M.: Embedded Online Optimization for Model Predictive Control at Megahertz Rates. IEEE Transactions on Automatic Control **59**(12), 3238–3251 (2014)
15. Jordan, J., Helmke, U.: Controllability of the QR Algorithm on Hessenberg Flags. In: Fifteenth International Symposium on Mathematical Theory of Networks and Systems. Notre Dame, IN, USA (2002)
16. Klöwer, M., Hatfield, S., Croci, M., Düben, P.D., Palmer, T.N.: Fluid Simulations Accelerated With 16 Bits: Approaching 4x Speedup on A64FX by Squeezing ShallowWaters.jl Into Float16. Journal of Advances in Modeling Earth Systems **14**(2), e2021MS002,684 (2022)
17. Kumar, A., Umathe, B., Vaidya, U., Kelkar, A.: Reachability Analysis in Ground Vehicle System using Koopman Operator Theory. In: 2023 15th IEEE International Conference on Industry Applications (INDUSCON), pp. 493–498. IEEE, São Bernardo do Campo, Brazil (2023)
18. Lessard, L., Recht, B., Packard, A.: Analysis and Design of Optimization Algorithms via Integral Quadratic Constraints. SIAM Journal on Optimization **26**(1), 57–95 (2016)
19. Mauroy, A., Sootla, A.: Estimation of Regions of Attraction with Formal Certificates in a Purely Data-Driven Setting. In: 2023 62nd IEEE Conference on Decision and Control (CDC), pp. 4682–4687. IEEE, Singapore, Singapore (2023)
20. Redman, W.T., Fonoberova, M., Mohr, R., Kevrekidis, I.G., Mezić, I.: Algorithmic (Semi-)Conjugacy via Koopman Operator Theory. In: 2022 IEEE 61st Conference on Decision and Control (CDC), pp. 6006–6011. IEEE, Cancun, Mexico (2022)
21. Scherer, C., Ebenbauer, C.: Convex Synthesis of Accelerated Gradient Algorithms. SIAM Journal on Control and Optimization **59**(6), 4615–4645 (2021)
22. Schlor, S., Strässer, R., Allgöwer, F.: Koopman interpretation and analysis of a public-key cryptosystem: Diffie-Hellman key exchange. IFAC-PapersOnLine **56**(2), 984–990 (2023)
23. Wilson, A.C.: A Lyapunov Analysis of Accelerated Methods in Optimization. Journal of Machine Learning Research **22**(113), 34 (2021)