Title

Structure, Symmetry, and Singularity in Learning Systems

ISBN

9798270213121

Author

Li, Jiayi

Publication Date

2025-12-12

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Structure, Symmetry, and Singularity in Learning Systems

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Statistics

by

Jiayi Li

2025

ABSTRACT OF THE DISSERTATION

Structure, Symmetry, and Singularity in Learning Systems

by

Jiayi Li

Doctor of Philosophy in Statistics

University of California, Los Angeles, 2025

Professor Guido F. Montúfar Cuartas, Chair

We study structural, algebraic, and statistical aspects of learning systems, with an emphasis on understanding how structure, symmetry, and singularity influence their behavior. Part I examines models whose parameterizations or loss landscapes have polynomial or rational form. Using tools from algebraic geometry and numerical algebraic geometry, we investigate the geometry of their critical sets, the role of degeneracies and singularities, and the symmetries that arise from overparameterization or model design. This part also considers rational neural networks, for which we introduce algebraic regularization schemes aimed at improving trainability and analyze the resulting optimization landscapes through a combination of theoretical and numerical methods.

Part II focuses on learning systems motivated by applications in statistics and engineering. Here we study procedures for estimating means of bounded random variables based on betting strategies, with an emphasis on statistical guarantees and practical performance. We also explore models of resilience and recovery in artificial and biological neural networks and investigate machine learning components used in digital twins for manufacturing systems, where structural assumptions play a central role in inference and control.

The application of algebraic and numerical algebraic tools to machine learning theory is still at an early stage, and many questions remain open. A deeper understanding of how

algebraic structure aligns with learning dynamics, how singularities arise in parameterized models, and how these features relate to implicit bias and other emergent phenomena may provide valuable insight. The mathematical ideas explored in this thesis reflect only a small part of what may be possible, but working with these tools has been a source of enjoyment due to the elegance they bring to complex systems. I hope that readers will find the methods and perspectives presented here accessible and motivating for future exploration.

The dissertation of Jiayi Li is approved.

Arash Ali Amini

Xiaowu Dai

Quanquan Gu

Guido F. Montúfar Cuartas, Committee Chair

University of California, Los Angeles

2025

To my family

Contents

List of Figures

Acknowledgements

As I look back on my doctoral journey, I am filled with gratitude for the many people and communities who made this path not only possible but deeply meaningful.

I begin with my advisor, whose joint appointment in the Departments of Mathematics and Statistics and Data Science reflects the same breadth and generosity of vision that shaped my work. His expertise in mathematical machine learning and applied algebraic methods formed the foundation on which this dissertation was built, enabling me to bridge my background in algebraic geometry with contemporary questions in machine learning. His guidance opened doors to new ideas and directions.

I am deeply grateful to my committee and to many faculty members in the Departments of Statistics and Data Science, Mathematics, and Computer Science at UCLA. Their patience and encouragement supported me as I transitioned from a primarily theoretical background into the world of statistics and machine learning. Their advice at key moments helped me navigate uncertainty, and their confidence in me sustained my growth as a researcher.

Collaboration has been one of the great joys of my PhD. I was fortunate to work with practitioners, where my role was to develop mathematical models for concrete problems, and with mathematicians, where we explored theoretical questions in applied mathematics and statistics. These collaborations were more than academic exchanges—they were opportunities to learn new ways of thinking, to share curiosity, and to see how mathematics takes shape across disciplines. The generosity of my collaborators made this journey far more rewarding than I could have imagined.

Equally important were the friends and classmates I met along the way—at the University of Hong Kong, SUNY Stony Brook, UCLA, the Max Planck Institutes, and through research programs at IMSI, IPAM, and the Simons Institute. Their presence provided strength during the most challenging stretches of my PhD. Their friendship, encouragement, and humor made difficult days bearable and turned good days into cherished memories.

I am also grateful to the Institute of Pure and Applied Mathematics, the Institute of Mathematical and Statistical Innovation, and the Simons Institute for the Theory of Computing, whose long research programs broadened my horizons and introduced me to vibrant communities of scholars. My journey was made possible as well by generous support from fellowships and scholarships, including those from the National Science Foundation,

Cathay Bank, UCLA, the Association for Computing Machinery, and the Dimitris N. Chorafas Foundation.

Teaching was another meaningful thread throughout my PhD. Working with the MSOL program allowed me to engage with students who brought their own perspectives and curiosity into the classroom. These experiences enabled me to explore ideas beyond the bounds of my dissertation while also providing the practical support that made this work possible. I am grateful to the MSOL faculty, to the students I had the honor to teach and mentor, and to the administrative staff in the Department of Statistics and Data Science, the MSOL program, and the Samueli School of Engineering, whose dedication and kindness supported me throughout my time at UCLA.

I also had the privilege of serving as Editor-in-Chief of *XRDS*, the student magazine of the Association for Computing Machinery. This role was a source of joy and growth, connecting me with a vibrant community of staff, editors, authors, and readers. I am grateful to them, and to the broader ACM community, for the opportunity to engage in conversations that extended far beyond my own research and for reminding me of the power of collective effort in shaping ideas.

Above all, I am indebted to my family. Their unwavering love and support have been the ground beneath my feet, giving me the courage to take risks and the resilience to persevere. I am grateful to my mother, the most humorous and intelligent person I know, who has always been like an elder sister to me—lifting my spirits in difficult times and reminding me to laugh even in the hardest moments. I am grateful to my father, whose quiet strength and steady confidence have been a constant source of encouragement, even when words were few. I am grateful to my husband, who has grown alongside me through these years, challenging me to take on difficult tasks, reminding me of my potential, and always believing in me even when I doubted myself. And finally, I must acknowledge Hama—the cat—who kept vigil by my keyboard and, in his quiet way, became the first and most faithful reviewer of this dissertation.

Vita

Education

University of California, Los Angeles — expected 2025

Ph.D. in Statistics

Thesis advisor: Guido Montúfar

State University of New York, Stony Brook — 2018

B.Sc. in Mathematics

Thesis advisor: Robert Lazarsfeld

Research Visits

Mathematics of Intelligences, — Sept.–Dec. 2024

Institute for Pure and Applied Mathematics (IPAM) — Los Angeles, USA

Algebraic Statistics and Our Changing World, — Sept.–Dec. 2023

Institute for Mathematical and Statistical Innovation (IMSI) — Chicago, USA

Awards

Dimitris N. Chorafas Foundation Award — 2024

Dissertation Year Fellowship — 2024

Summer Mentored Research Fellowship — 2022

ACM-W Scholarship, Association for Computing Machinery — 2020

Cathay Bank Scholarship — 2020

Chapter 1

Introduction

Machine learning has advanced quickly in recent years, yet the theoretical understanding of modern models still lags behind their empirical performance. Phenomena such as implicit bias, over-parameterization, the appearance or disappearance of spurious critical points, and varying degrees of robustness suggest that learning systems possess rich internal geometric structure. In this dissertation we adopt a geometric viewpoint and examine learning systems through three lenses: *structure*, *symmetry*, and *singularity*. These themes appear throughout both the mathematical analysis in Part I and the applications in Part II.

By *structure* we refer to the algebraic and semi-algebraic patterns that arise from model architectures and loss functions. These include polynomial and rational parameterizations, determinantal constraints, polyhedral decompositions induced by piecewise-linear activations, and algebraic relations among intermediate features or outputs. By *symmetry* we refer to invariances and equivariances, ranging from permutation and rotational symmetry to the group actions that arise naturally in engineered systems. By *singularity* we mean non-generic behavior such as nonidentifiable parameterizations, degenerate Fisher information, or critical points lying on singular strata of the parameter-to-function map. Algebraic geometry offers a natural framework for articulating all three aspects, providing tools to describe how these phenomena arise and how they interact with optimization and inference.

Many learning architectures admit algebraic descriptions, either exactly (for polynomial or rational networks) or piecewise (for ReLU or max-pooling networks). Linear layers and polynomial activations yield polynomial parameterizations, while rational activations yield maps of the form

$$R_{l,j}(t) = \frac{P_{l,j}(t)}{Q_{l,j}(t)}, \qquad Q_{l,j} \not\equiv 0,$$

defined on domains where denominators remain nonvanishing. Piecewise-linear networks induce semi-algebraic partitions of input space: they are polynomial on polyhedral regions and glue along semi-algebraic boundaries. The resulting families of realized functions often form algebraic or semi-algebraic subsets of function space—sometimes referred to as *neurovarieties* or *neuromanifolds*. Although these sets are generally nonconvex and stratified, they possess algebraic regularities that are not readily visible through classical analytic techniques.

To articulate these regularities, it is useful to recall basic algebraic and semi-algebraic notions. Let $\mathbb{K}$ be a field and let $\mathbb{A}^n(\mathbb{K})$ denote affine n-space. For a set of polynomials $S \subseteq \mathbb{K}[x_1, \ldots, x_n]$, the *vanishing set*

$$\mathcal{V}(S) = \{a \in \mathbb{A}^n(\mathbb{K}) : f(a) = 0 \text{ for all } f \in S\}$$

is an algebraic set. Over algebraically closed fields, the Weak and Strong Nullstellensatz establish a precise correspondence between radical ideals and algebraic sets. This correspondence allows one to convert geometric statements—such as the characterization of critical points, degeneracies, or feasible regions—into algebraic ones. Over $\mathbb{R}$, many relevant sets are semi-algebraic, defined by finitely many polynomial equations and inequalities. Two classical theorems play an important role for results presented in later chapters of this thesis:

- **Chevalley's Constructibility Theorem**, which states that the image of an algebraic set under a polynomial map is a constructible set. In our context, this implies that the sets obtained by applying polynomial representation layers remain within a well-behaved

geometric class, even though they need not be algebraic.

- **Tarski–Seidenberg Theorem**, which states that the image of a semi-algebraic set under a coordinate projection is again semi-algebraic. This closure property under elimination of variables ensures that many sets arising in learning—such as feasible regions, decision boundaries, and regularized sublevel sets—retain a finite semi-algebraic description after projection.

Combined with Hilbert's basis theorem and standard conventions for projective space, these results justify treating many objects in learning—the structure of loss landscapes, the constraints imposed by architectures, and the domains of intermediate activations—as algebraic or semi-algebraic sets.

This viewpoint influences several aspects of learning. First, *optimization* is governed by the geometry of the parameterization map. Flat directions arise when parameters vary along fibers that correspond to the same underlying function, while spurious critical points may lie on singular strata where the Jacobian of the parameter-to-function map loses rank. Second, *statistical properties* such as identifiability, sample complexity, and the behavior of likelihood-based estimators often admit algebraic formulations: moment constraints define algebraic varieties, injectivity of polynomial or rational maps characterizes identifiability, and the maximum likelihood degree quantifies the number of complex critical points of likelihood equations. Third, *robustness and certification* questions—such as bounding Lipschitz constants, verifying margins, or ensuring the absence of poles in rational networks—can be expressed as verifying polynomial inequalities on semi-algebraic sets and addressed through sum-of-squares relaxations and Positivstellensatz certificates.

One approach to understand these phenomena is through the parameter-to-function map

$$\Phi : \Theta \longrightarrow \mathcal{F}, \qquad \theta \mapsto f_\theta.$$

The expressive power, redundancies, symmetries, and singularities of an architecture are encoded in the geometry of $\Phi(\Theta)$ (the set of realizable functions), the fibers $\Phi^{-1}(f)$, and the singularities of Φ itself. Many geometric approaches that appear distinct—function-space analyses, algebraic statistics, tropical geometry, invariant theory, and numerical algebraic geometry—can be viewed as tools for examining these images, fibers, and singularities.

From the perspective of the image $\Phi(\Theta)$, neural architectures define structured subsets of function space. For linear and convolutional networks, these sets often admit explicit algebraic or semi-algebraic descriptions: determinantal varieties for linear architectures and architecture-dependent semi-algebraic families for convolutional ones [Trager et al., 2020, Kohn et al., 2022, Kohn et al., 2024]. Restricting the loss to $\Phi(\Theta)$ shows that many practical difficulties in parameter space—such as flat directions or spurious local minima—are artifacts of redundancies or singularities in the parameterization. This aligns with geometric phenomena observed during training, such as NTK linearization [Jacot et al., 2018], neural collapse [Papyan et al., 2020], and the simplicity that emerges in interpolation regimes [Belkin et al., 2019].

The structure of the fibers of Φ also plays a central role in statistical analysis. Probabilistic models such as graphical models, mixture models, and latent-variable models correspond to algebraic varieties in the space of observable distributions [Drton et al., 2009, Sullivant, 2018]. Identifiability issues often reduce to algebraic questions involving secant varieties or tensor decompositions [Allman et al., 2009, Landsberg, 2012], and estimation complexity is reflected in the maximum likelihood degree [Catanese et al., 2006].

Architectures with piecewise-linear activations introduce polyhedral geometry. ReLU networks partition input space into polyhedral cells and act as polynomial maps on each region. Tropical geometry provides a compact algebraic representation of these maps [Maclagan and Sturmfels, 2015], enabling quantitative analysis of expressivity, decision boundaries, and depth–complexity tradeoffs [Zhang et al., 2018, Montúfar et al., 2014]. These results illuminate how architectural choices shape the functional complexity of learned models.

For rational networks, the domain of Φ acquires geometric significance. Fixing degree bounds places (P, Q) in quasi-affine parameter spaces, and modding out by $\mathbb{G}_m$-scaling places coefficients in products of projective spaces. This quasi-projective viewpoint clarifies how singularities—poles, vanishing denominators, rank-deficient Jacobians—affect optimization and motivates the regularization and reparametrization strategies developed in Part I.

Symmetries of the map Φ arise through group actions on data or model parameters. Algebraic invariant theory provides systematic descriptions of invariant and equivariant polynomial functions [Derksen and Kemper, 2002], forming the theoretical basis for modern equivariant architectures [Bronstein et al., 2021]. Symmetry also features prominently in the engineering applications examined in Part II.

Overparameterized models often exhibit intrinsic singularities: positive-dimensional fibers, degenerate Fisher information, and nonidentifiable directions. Singular learning theory relates generalization behavior in such settings to invariants of singularities such as the real log canonical threshold (RLCT) [Watanabe, 2009, Watanabe, 2018]. This perspective provides quantitative insight into deep models that fall far outside the classical regular setting.

Finally, numerical algebraic geometry provides computational tools that complement these geometric perspectives. Homotopy-continuation methods make it possible to solve polynomial systems arising from critical point equations, track solutions as hyperparameters vary, and study bifurcations in polynomial and rational networks [Sommese and Wampler, 2005, Bates et al., 2013]. These tools will be used in Part I to examine optimization landscapes in concrete examples.

Taken together, these perspectives suggest that the behavior of modern learning systems can be understood through the structure, symmetry, and singularity of the parameter-to-function map. The algebraic and geometric viewpoints developed in this chapter provide the conceptual framework for the analyses that follow in the remainder of the dissertation.

Bibliography

[Allman et al., 2009] Allman, E. S., Matias, C., and Rhodes, J. A. (2009). Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, 37(6A):3099–3132.

[Bates et al., 2013] Bates, D. J., Hauenstein, J. D., Sommese, A. J., and Wampler, C. W. (2013). *Numerically Solving Polynomial Systems with Bertini*. SIAM.

[Belkin et al., 2019] Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2019). Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences (PNAS)*, 116(32):15849–15854.

[Bronstein et al., 2021] Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *Proceedings of the National Academy of Sciences (PNAS)*, 118(47):e2020186118.

[Catanese et al., 2006] Catanese, F., Hoşten, S., Khetan, A., and Sturmfels, B. (2006). The maximum likelihood degree. *American Journal of Mathematics*, 128(3):671–697.

[Derksen and Kemper, 2002] Derksen, H. and Kemper, G. (2002). *Computational Invariant Theory*. Springer.

[Drton et al., 2009] Drton, M., Sturmfels, B., and Sullivant, S. (2009). *Lectures on Algebraic Statistics*. Birkhäuser.

[Jacot et al., 2018] Jacot, A., Gabriel, F., and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[Kohn et al., 2022] Kohn, K., Merkh, T., Montúfar, G., and Trager, M. (2022). Geometry of linear convolutional networks. *SIAM Journal on Applied Algebra and Geometry*, 6(3):368–406.

[Kohn et al., 2024] Kohn, K., Montúfar, G., Shahverdi, V., and Trager, M. (2024). Function space and critical points of linear convolutional networks. *SIAM Journal on Applied Algebra and Geometry*, 8(2):333–362.

[Landsberg, 2012] Landsberg, J. M. (2012). *Tensors: Geometry and Applications*. American Mathematical Society.

[Maclagan and Sturmfels, 2015] Maclagan, D. and Sturmfels, B. (2015). *Introduction to Tropical Geometry*, volume 161 of *Graduate Studies in Mathematics*. American Mathematical Society.

[Montúfar et al., 2014] Montúfar, G. F., Pascanu, R., Cho, K., and Bengio, Y. (2014). On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[Papyan et al., 2020] Papyan, V., Han, X. Y., and Donoho, D. L. (2020). Prevalence of neural collapse during the terminal phase of deep learning. *Proceedings of the National Academy of Sciences (PNAS)*, 117(40):24652–24663.

[Sommese and Wampler, 2005] Sommese, A. J. and Wampler, C. W. (2005). *The Numerical Solution of Systems of Polynomials Arising in Engineering and Science*. World Scientific.

[Sullivant, 2018] Sullivant, S. (2018). *Algebraic Statistics*. Cambridge University Press.

[Trager et al., 2020] Trager, M., Kohn, K., and Bruna, J. (2020). Pure and spurious critical points: A geometric study of linear networks. In *International Conference on Learning Representations (ICLR)*.

[Watanabe, 2009] Watanabe, S. (2009). *Algebraic Geometry and Statistical Learning Theory*. Cambridge University Press.

[Watanabe, 2018] Watanabe, S. (2018). *Mathematical Theory of Bayesian Statistics*. CRC Press.

[Zhang et al., 2018] Zhang, L., Naitzat, G., and Lim, L.-H. (2018). Tropical geometry of deep neural networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 5824–5832.

Chapter 2

Geometry of Polynomial Neural Networks

We study the expressivity and learning process for polynomial neural networks (PNNs) with monomial activation functions. The weights of the network parametrize the neuromanifold. In this paper, we study certain neuromanifolds using tools from algebraic geometry: we give explicit descriptions as semialgebraic sets and characterize their Zariski closures, called neurovarieties. We study their dimension and associate an algebraic degree, the learning degree, to the neurovariety. The dimension serves as a geometric measure for the expressivity of the network, the learning degree is a measure for the complexity of training the network and provides upper bounds on the number of learnable functions. These theoretical results are accompanied with experiments.

2.1 Introduction

Over the past decade, neural networks have achieved remarkable success, primarily driven by advancements in deep learning. With the increased computational power, availability of large datasets, and algorithmic innovations, deep learning has led to groundbreaking achievements in areas like image and speech recognition, natural language processing, and autonomous systems. In addition, deep neural networks outperform many traditional statistical models,

significantly impacting fields such as healthcare and finance. Despite the empirical triumph, the underlying theoretical behaviors of neural networks remain an open and active field of research.

In particular, researchers study various activation functions to understand their role in introducing non-linearity, affecting gradient propagation, and influencing computational efficiency. Polynomial activation functions have gained interest for their ability to introduce higher-order interactions between inputs, allowing networks to model complex, non-linear phenomena more effectively [Oh et al., 2003]. Although feedforward neural networks with polynomial activation functions are well-known to be non-universal approximators [Hornik et al., 1989], there are many practical tasks where architectures with polynomial activations have outperformed other ones, especially in environments where data relationships are polynomial in nature. In particular, polynomial neural networks have led to state-of-the-art results in engineering tasks like face detection from cluttered images [Huang et al., 2003], image generation [Chrysos et al., 2020], 3D shape recognition [Yavartanoo et al., 2021], as well as financial applications like forecasting trading signals [Ghazali et al., 2011], uncertain natural frequency quantification [Dey et al., 2016], and estimating stock closing indices [Nayak and Misra, 2018]. However, the choice of degree and the potential for overfitting are critical considerations in applying polynomial neural networks to individual practical tasks.

Compared to other low-degree activation functions, polynomial functions capture intricate patterns within data without the need for additional layers, potentially reducing model complexity and computational costs. In addition, common neural network activation functions, including sigmoid and ReLU, can be effectively approximated using polynomial ratios. Recent work also shows that fully-connected feedforward neural networks using ratios of polynomials as activation functions approximate smooth functions more efficiently than ReLU networks [Boullé et al., 2020]. Our exploration in the realm of polynomial networks lays the groundwork for further investigation of other activation functions.

In this paper, we perform an algebro-geometric study of neuromanifolds[1] and their Zariski closures, neurovarieties, following the previous work by [Kileel et al., 2019b]. In §2.2, we give the background on polynomial neural networks, neuromanifolds and neurovarieties. Neuromanifolds provide a natural generalization of the set of symmetric tensors of bounded symmetric rank. In §2.3, we recall the connection between homogeneous polynomials and symmetric tensor decompositions, and characterize neuromanifolds for some shallow polynomial neural networks. In §2.4, we describe different approaches for studying neurovarieties. The dimension of the neurovariety provides a measure for the expressivity of the network. In §2.5, we study these dimensions. For a shallow polynomial neural network with a single output unit, this corresponds to the Alexander–Hirschowitz Theorem in neural network settings. In the deep case, we present conjectures supported by empirical evidence. Finally, in §2.6, we study the optimization process of a polynomial neural network: from a static perspective, we describe the complexity of the optimization landscape by its *learning degree* and compute it for a family of architectures; for the dynamic optimization process, we review the backpropagation algorithm and summarize known results on linear neural networks from previous literature in both the machine learning and algebraic statistics communities.

We provide code for computations and experiments at [Kubjas et al., 2024].

Main contributions. For those new to the topic, we recommend coming back to this paragraph on the main contributions after reading Section 2.2. We characterize the neuromanifold $\mathcal{M}_{(d_0,d_1,1),2}$ in Lemma 2.3.3, $\mathcal{M}_{(d_0,1,d_2),2}$ in Lemma 2.3.4, $\mathcal{M}_{(d_0,1,d_2),r}$ in Lemma 2.3.6, and $\mathcal{M}_{(d_0,d_1,d_2),r}$ with $d_1 \geq \binom{r+d_0-1}{r}$ in Proposition 2.3.7. We characterize the neurovariety $\mathcal{V}_{(2,2,d_2),2}$ in Proposition 2.4.1 and partially the neuromanifold $\mathcal{M}_{(2,2,d_2),2}$ in Proposition 2.4.2 by studying the fibers of the PNN, the neurovariety $\mathcal{V}_{(3,3,3),2}$ in Example 2.4.4 by using Grassmannians and the neurovariety $\mathcal{V}_{(2,2,2,2),2}$ in Example 2.4.6 by applying the Hilbert–Burch Theorem. In Proposition 2.5.4, we show that if a PNN has a layer of width one, then the

[1]By an abuse of terminology we use the term *neuromanifold* to denote the functional space of a neural network, although this space is almost never a smooth manifold.

widths of layers after the width-1 layer except for the output layer do not influence the neuromanifold. In Corollary 2.5.9, we show that if a PNN has a defective subnetwork, then it is itself defective unless it is filling. In Section 2.5.2, we describe a family of non-trivial symmetries for the neuromanifold $\mathcal{V}_{(5,7,1),3}$ that does not have expected dimension. We introduce the learning degree, a characteristic of the optimization landscape, in Definition 2.6.1. In Theorem 2.6.3, we show that learning degree of a PNN with respect to the Euclidean loss function exists and it is bounded above by the generic Euclidean distance degree of its neurovariety. In Theorem 2.6.5, we prove that the learning degree of $\mathcal{V}_{(2,2,k),2}$ for $k \geq 2$ is at most $8k^2 - 12k + 3$.

Acknowledgments. This paper was written in connection with the "Apprenticeship Week: Varieties from Statistics" which took place at IMSI as part of the long program "Algebraic Statistics and Our Changing World". We would like to thank Miles Bakenhus and Maksym Zubkov who participated in the work of this group part of the time. Lemma 2.3.5 is due to Maksym Zubkov. We would like to thank Elizabeth Gross, Leonie Kayser, Joe Kileel, Mark Kong, Guido Montúfar, Alessandro Oneto, Bernd Sturmfels, and Matthew Trager for useful discussions and various suggestions. We thank Lisa Seccia and Nils Sturma for careful reviews of an earlier version of this paper. We also thank Giovanni Luca Marchetti and Vahid Shahverdi for pointing out a mistake in Section 2.6.1 in a previous version of this article. Part of this research was performed while the authors were visiting the Institute for Mathematical and Statistical Innovation (IMSI), which is supported by the National Science Foundation (Grant No. DMS-1929348).

2.2 Background

A general L-layer *(feedforward) neural network* is a composition of L affine-linear maps with coordinatewise nonlinearity in between [Haykin, 1998]. Let $F_{\boldsymbol{\theta}}$ be a feedforward neural

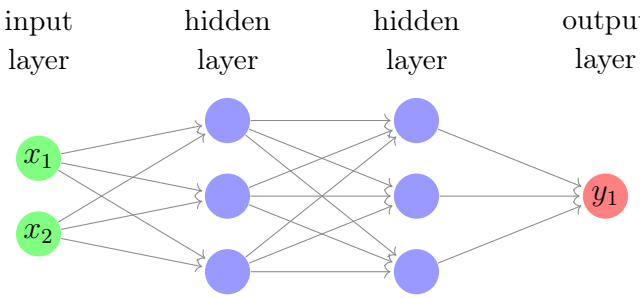


Figure 2.1: A neural network architecture with widths $\mathbf{d} = (2, 3, 3, 1)$, input $\mathbf{x} = (x_1, x_2)^T$ and output y_1.

network with parameters $\boldsymbol{\theta}$,

$$F_{\boldsymbol{\theta}}(\mathbf{x}) = f_L \circ \sigma_{L-1} \circ f_{L-1} \circ \cdots \circ f_2 \circ \sigma_1 \circ f_1(\mathbf{x}),$$

where

$$f_l(\mathbf{x}) \colon \mathbb{R}^{d_{l-1}} \to \mathbb{R}^{d_l}, \quad \mathbf{x} \mapsto W_l \mathbf{x} + \mathbf{b}_l$$

and the function $\sigma_l \colon \mathbb{R}^{d_l} \to \mathbb{R}^{d_l}$ is the *activation function*, typically a real function that is applied coordinatewise, i.e., $\sigma_l = (\sigma_{l,1}, \ldots, \sigma_{l,d_l})$ with $\sigma_{l,j} \colon \mathbb{R} \to \mathbb{R}$. The matrices $W_1, \ldots, W_L$ are the *weights* of the neural network, and $b_1, \ldots, b_L$ are referred to as the *biases*. Together, they constitute the parameter set $\boldsymbol{\theta}$.

Pictorially, a neural network architecture can be represented as in Figure 2.1. Each node is called a *neuron* and each column of neurons forms a *layer* of the network. The first layer is called *input layer*, the last layer is called *output layer* and the remaining layers are referred to as *hidden layers*. A neural network with exactly one hidden layer is called *shallow*. The number of neurons in the i^{th} layer is the i^{th} *width*, denoted d_i. The vector of widths $\mathbf{d} = (d_0, d_1, \ldots, d_L)$ together with a choice of activation functions $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_{L-1})$ constitute the *architecture* of the network.Arrows between layers represent the maps f_l. Note that all architectures in this paper are considered to be fully-connected feedforward neural networks.

The goal of deep learning is to approximate a target function $f\colon \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ with a neural network $F_{\boldsymbol{\theta}}$ of a chosen architecture $(\mathbf{d} = (d_0, d_1, \ldots, d_L), \boldsymbol{\sigma})$. This amounts to an optimization task over the space of parameters, the "learning" or "training" process. For more details, see §2.6.

The *parameter map*

$$\Psi_{\mathbf{d},\boldsymbol{\sigma}}\colon \mathbb{R}^N \to \mathrm{Fun}(\mathbb{R}^{d_0}, \mathbb{R}^{d_L}), \quad \boldsymbol{\theta} \mapsto F_{\boldsymbol{\theta}}$$

associates a tuple of parameters $\boldsymbol{\theta}$ with the corresponding neural network $F_{\boldsymbol{\theta}}$. Its image is called the *neuromanifold* $\mathcal{M}_{\mathbf{d},\boldsymbol{\sigma}}$ and consists of all functions a network with architecture $(\mathbf{d}, \boldsymbol{\sigma})$ can learn. The neuromanifold is also referred to as *functional space* in the literature. Note that oftentimes $\mathcal{M}_{\mathbf{d},\boldsymbol{\sigma}}$ is not a smooth manifold.

In practice, typical choices of activation functions are *ReLU* or *sigmoid* functions

$$\sigma_{\mathrm{ReLU}}(x) = \max\{0, x\}, \quad \sigma_{\mathrm{sigmoid}}(x) = \frac{e^x}{1 + e^x}.$$

In this paper, we focus on *monomial* activations $x \mapsto x^r$. This turns the network $F_{\boldsymbol{\theta}}$ into a polynomial map and makes the study of these networks amenable to techniques from algebraic geometry.

The map f_l can be written as $f_l(\mathbf{x}) = W_l\mathbf{x} + \mathbf{b}_l$, where $W_l \in \mathbb{R}^{d_l \times d_{l-1}}$ is a linear map and $\mathbf{b}_l \in \mathbb{R}^{d_i}$. We will be considering networks without biases to make the polynomial map $F_{\boldsymbol{\theta}}$ homogeneous. Let us summarize the setup in the following.

Definition 2.2.1. A *polynomial neural network (PNN)* $p_{\mathbf{w}}$ with architecture $(\mathbf{d} = (d_0, d_1, \ldots, d_L), r)$ is a feedforward neural network

$$p_{\mathbf{w}} = W_L \circ \sigma_{L-1} \circ W_{L-1} \circ \sigma_{L-2} \circ \cdots \circ \sigma_1 \circ W_1\colon \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$$

where $W_i \in \mathbb{R}^{d_i \times d_{i-1}}$ are linear maps and the activation functions

$$\sigma_i(\mathbf{x}) = \rho_r(\mathbf{x}) \coloneqq (x_1^r, \ldots, x_n^r)$$

are monomial. The number r is called the *activation degree* of $p_{\mathbf{w}}$. The parameters $\mathbf{w}$ are given by the entries of the matrices W_i, i.e., $\mathbf{w} = (W_1, W_2, \ldots, W_L)$.

A PNN $p_{\mathbf{w}}$ with architecture $(\mathbf{d}, r)$ is a homogeneous polynomial map of degree r^{L-1}. Hence, the associated parameter map $\Psi_{\mathbf{d},r}$ maps an L-tuple of matrices $(W_1, W_2, \ldots, W_L)$ to a d_L-tuple of homogeneous polynomials of degree r^{L-1} in d_0 variables, i.e.,

$$\Psi_{\mathbf{d},r} : \mathbb{R}^{d_1 \times d_0} \times \cdots \times \mathbb{R}^{d_L \times d_{L-1}} \to (\mathrm{Sym}_{r^{L-1}}(\mathbb{R}^{d_0}))^{d_L}, \quad \mathbf{w} \mapsto p_{\mathbf{w}} = \begin{pmatrix} p_{\mathbf{w}}^{(1)} \\ \vdots \\ p_{\mathbf{w}}^{(d_L)} \end{pmatrix}.$$

We can identify elements in the image with their vectors of coefficients in

$$\mathbb{R}^{d_L \binom{r^{L-1}+d_0-1}{d_0-1}}.$$

Coordinates of this space will be denoted by $c_I^{(j)}$ so that $c_I^{(j)}$ is the coefficient of the monomial $\mathbf{x}^I$ in the polynomial $p_{\mathbf{w}}^{(j)}(\mathbf{x})$; here, I is a multiindex in

$$\binom{[d_0]}{r^{L-1}} = \left\{ r^{L-1} \text{ element subsets in } \{1, 2, \ldots, d_0\} \right\}.$$

Definition 2.2.2. The image of $\Psi_{\mathbf{d},r}$ is the *neuromanifold* $\mathcal{M}_{\mathbf{d},r}$. Its Zariski closure is the *neurovariety* $\mathcal{V}_{\mathbf{d},r}$, an affine variety inside $(\mathrm{Sym}_{r^{L-1}}(\mathbb{R}^{d_0}))^{d_L}$.

The neuromanifold $\Psi_{\mathbf{d},r}$ is a semialgebraic set inside $(\mathrm{Sym}_{r^{L-1}}(\mathbb{R}^{d_0}))^{d_L}$, which means that it is a finite union of sets that can be defined by polynomial equalities and inequalities. The observation that the neuromanifold $\Psi_{\mathbf{d},r}$ is a semialgebraic set follows from the Tarski–

Seidenberg theorem which states that a polynomial image of a semialgebraic set is again semialgebraic.

Increasing the widths of hidden layers gives a containment of neuromanifolds as shown in the following proposition.

Proposition 2.2.3. *Let* $\mathbf{d} = (d_0, \ldots, d_i, \ldots, d_L)$ *and let* $\mathbf{d}' = (d_0, \ldots, d_i', \ldots, d_L)$ *be a tuple which differs from* $\mathbf{d}$ *precisely in the* i^{th} *entry for* $0 < i < L$ *and assume* $d_i' \geq d_i$*. Then* $\mathcal{M}_{\mathbf{d},r} \subseteq \mathcal{M}_{\mathbf{d}',r}$*, in particular* $\mathcal{V}_{\mathbf{d},r} \subseteq \mathcal{V}_{\mathbf{d}',r}$*.*

Proof. Let $\mathbf{w} = (W_1, \ldots, W_L)$ be a parameter vector for the architecture $\mathbf{d}$ and $p_{\mathbf{w}} \in \mathcal{M}_{\mathbf{d},r}$ the corresponding polynomial network. Let $\mathbf{w}' = (W_1', \ldots, W_L')$ be the parameter vector for the architecture $\mathbf{d}'$ such that each W_i' has W_i as left-top submatrix and zeros elsewhere. Then $p_{\mathbf{w}} = p_{\mathbf{w}'} \in \mathcal{M}_{\mathbf{d}',r}$. □

As $\mathcal{M}_{\mathbf{d},r}$ is semialgebraic, it has the same dimension as its Zariski closure, $\mathcal{V}_{\mathbf{d},r}$. In [Kileel et al., 2019b], the dimension of $\mathcal{V}_{\mathbf{d},r}$ was proposed as a measure for the *expressivity* of the network architecture $(\mathbf{d}, r)$.

Definition 2.2.4. An architecture $(\mathbf{d}, r)$ is *filling* if $\mathcal{V}_{\mathbf{d},r} = (\mathrm{Sym}_{r^{L-1}}(\mathbb{R}^{d_0}))^{d_L}$. In this case, we say that $\mathcal{M}_{\mathbf{d},r}$ is *thick*, i.e., it has positive Lebesgue measure.

The case of filling architectures is particularly interesting from a machine learning perspective as filling networks have the most expressive power: if $\mathcal{M}_{\mathbf{d},r} = (\mathrm{Sym}_{r^{L-1}}(\mathbb{R}^{d_0}))^{d_L}$, any target function in $(\mathrm{Sym}_{r^{L-1}}(\mathbb{R}^{d_0}))^{d_L}$ can potentially be learned *exactly* by the network. In the case of non-filling architectures, a general target function can only be approximated by the network. For more details on the learning process, see §2.6. Moreover, from an optimization perspective it is advantageous to work with thick neuromanifolds as non-thick neuromanifolds are known to be non-convex, see [Kileel et al., 2019b, Proposition 7].

Example 2.2.5. For architecture $\mathbf{d} = (2, 1, 1)$, $r = 2$ with input $\mathbf{x} = (x_1, \ x_2)^T$ and

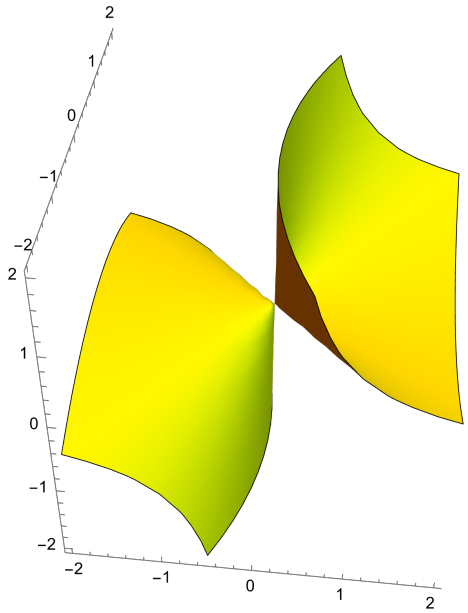


Figure 2.2: The neuromanifold $\mathcal{M}_{(2,1,1),2}$ in $\mathrm{Sym}_2(\mathbb{R}^2) \cong \mathbb{R}^3$.

parameters

$$W_1 = \begin{pmatrix} w_{111} & w_{112} \end{pmatrix}, \; W_2 = \begin{pmatrix} w_{211} \end{pmatrix},$$

the network is

$$p_{\mathbf{w}}(\mathbf{x}) = W_2 \rho_2 W_1 \mathbf{x} = \begin{pmatrix} w_{211}(w_{111}x_1 + w_{112}x_2)^2 \end{pmatrix}.$$

Example 2.2.6 ([Kileel et al., 2019b], Example 3)**.** For architecture $\mathbf{d} = (2, 2, 3),\ r = 2$ with input $\mathbf{x} = (x_1,\ x_2)^T$ and parameters

$$W_1 = \begin{pmatrix} w_{111} & w_{112} \\ w_{121} & w_{122} \end{pmatrix}, \; W_2 = \begin{pmatrix} w_{211} & w_{212} \\ w_{221} & w_{222} \\ w_{231} & w_{232} \end{pmatrix},$$

the network is

$$p_{\mathbf{w}}(\mathbf{x}) = W_2\rho_2 W_1\mathbf{x} = \begin{pmatrix} w_{211}(w_{111}x_1 + w_{112}x_2)^2 + w_{212}(w_{121}x_1 + w_{122}x_2)^2 \\ w_{221}(w_{111}x_1 + w_{112}x_2)^2 + w_{222}(w_{121}x_1 + w_{122}x_2)^2 \\ w_{231}(w_{111}x_1 + w_{112}x_2)^2 + w_{232}(w_{121}x_1 + w_{122}x_2)^2 \end{pmatrix}.$$

There are ten parameters in W_1 and W_2 which $\Psi_{(2,2,3),2}$ maps to the triple of quadrics above with $\dim(\mathrm{Sym}_2(\mathbb{R}^2)^3) = 9$ coefficients. The neurovariety is a hypersurface defined by the cubic equation

$$\det \begin{pmatrix} c_{11}^{(1)} & c_{12}^{(1)} & c_{22}^{(1)} \\ c_{11}^{(2)} & c_{12}^{(2)} & c_{22}^{(2)} \\ c_{11}^{(3)} & c_{12}^{(3)} & c_{22}^{(3)} \end{pmatrix} = 0.$$

This variety has dimension eight, implying the architecture $\mathbf{d} = (2, 2, 3), r = 2$ is not filling. For more details, see Proposition 2.4.1 below. Note that in this case $\mathcal{M}_{\mathbf{d},r} \subsetneq \mathcal{V}_{\mathbf{d},r}$, see Proposition 2.4.2.

There is a naïve expectation for the dimension of the neurovariety, namely the number of parameters. However, one immediately observes a symmetry in the parameters for any network architecture, called *multi-homogeneity*.

Lemma 2.2.7 ([Kileel et al., 2019b, Lemma 13]). *For all diagonal matrices $D_i \in \mathbb{R}^{d_i \times d_i}$ and permutation matrices $P_i \in \mathbb{Z}^{d_i \times d_i}$ $(i = 1, \ldots, L-1)$, the parameter map $\Psi_{\mathbf{d},r}$ returns the same network under the replacement:*

$$\begin{aligned} W_1 &\leftarrow P_1 D_1 W_1 \\ W_2 &\leftarrow P_2 D_2 W_2 D_1^{-r} P_1^T \\ W_3 &\leftarrow P_3 D_3 W_3 D_2^{-r} P_2^T \\ &\vdots \\ W_L &\leftarrow W_L D_{L-1}^{-r} P_{L-1}^T. \end{aligned}$$

Hence, a generic fiber of $\Psi_{\mathbf{d},r}$ has dimension at least $\sum_{i=1}^{L-1} d_i$. We call the number of parameters subtracted by this dimension the expected dimension of the neurovariety.

Definition 2.2.8. We define the *expected dimension* of the neurovariety $\mathcal{V}_{\mathbf{d},r}$ to be

$$\operatorname{edim}(\mathcal{V}_{\mathbf{d},r}) \coloneqq \min\left\{ d_L + \sum_{i=0}^{L-1}(d_i d_{i+1} - d_{i+1}), d_L \binom{d_0 + r^{L-1} - 1}{r^{L-1}} \right\}.$$

By Lemma 2.2.7, $\dim(\mathcal{V}_{\mathbf{d},r}) \leq \operatorname{edim}(\mathcal{V}_{\mathbf{d},r})$. The difference $\operatorname{edim}(\mathcal{V}_{\mathbf{d},r}) - \dim(\mathcal{V}_{\mathbf{d},r})$ is the *defect* of $\mathcal{V}_{\mathbf{d},r}$. If the defect is nonzero $\mathcal{V}_{\mathbf{d},r}$ is called *defective*. We refer to

$$\dim((\operatorname{Sym}_{r^{L-1}}(\mathbb{R}^{d_0}))^{d_L}) = d_L \binom{d_0 + r^{L-1} - 1}{r^{L-1}}$$

as the *ambient dimension* of $\mathcal{V}_{\mathbf{d},r}$.

In Table 2.1 we compute the ideal of the neurovariety and its dimension for shallow polynomial neural networks with $d_i \in [3]$ and $r = 2$. We also compare the neurovariety with the neuromanifold.

Most ideals of neurovarieties in Table 2.1 are described by results in the upcoming sections. The ideals for all but two cases ($\mathbf{d} = (3,2,3)$ and $\mathbf{d} = (3,3,3)$) in the table are determinantal. In general, we expect most neurovarieties to be non-determinantal. Among the table, only the neurovariety for the architecture $\mathbf{d} = (3,2,1)$ does not have expected dimension. The following example describes the cases that are not explained by any results in the rest of the paper.

Example 2.2.9. The ideal of the neurovariety for the architecture $\mathbf{d} = (3,2,2), r = 2$ is determinantal and it is generated by the 3×3 minors of

$$\begin{pmatrix} c_{11}^{(1)} & \frac{1}{2}c_{12}^{(1)} & \frac{1}{2}c_{13}^{(1)} & c_{11}^{(2)} & \frac{1}{2}c_{12}^{(2)} & \frac{1}{2}c_{13}^{(2)} \\ \frac{1}{2}c_{12}^{(1)} & c_{22}^{(1)} & \frac{1}{2}c_{23}^{(1)} & \frac{1}{2}c_{12}^{(2)} & c_{22}^{(2)} & \frac{1}{2}c_{23}^{(2)} \\ \frac{1}{2}c_{13}^{(1)} & \frac{1}{2}c_{23}^{(1)} & c_{33}^{(1)} & \frac{1}{2}c_{13}^{(2)} & \frac{1}{2}c_{23}^{(2)} & c_{33}^{(2)} \end{pmatrix}.$$

$\mathbf{d}$	dim	edim	amb dim	ideal	$\mathcal{M}_{\mathbf{d},2} = \mathcal{V}_{\mathbf{d},2}$?
(1,1,1)	1	1	1	$\langle 0 \rangle$	yes
(1,1,2)	2	2	2	$\langle 0 \rangle$	yes
(1,1,3)	3	3	3	$\langle 0 \rangle$	yes
(1,2,1)	1	1	1	$\langle 0 \rangle$	yes
(1,2,2)	2	2	2	$\langle 0 \rangle$	yes
(1,2,3)	3	3	3	$\langle 0 \rangle$	yes
(1,3,1)	1	1	1	$\langle 0 \rangle$	yes
(1,3,2)	2	2	2	$\langle 0 \rangle$	yes
(1,3,3)	3	3	3	$\langle 0 \rangle$	yes
(2,1,1)	2	2	3	determinantal, Lemma 2.3.4	yes, Lemma 2.3.4
(2,1,2)	3	3	6	determinantal, Lemma 2.3.4	yes, Lemma 2.3.4
(2,1,3)	4	4	9	determinantal, Lemma 2.3.4	yes, Lemma 2.3.4
(2,2,1)	3	3	3	$\langle 0 \rangle$	yes, Lemma 2.3.3
(2,2,2)	6	6	6	$\langle 0 \rangle$	no, Proposition 2.4.2
(2,2,3)	8	8	9	determinantal, Proposition 2.4.1	no, Proposition 2.4.2
(2,3,1)	3	3	3	$\langle 0 \rangle$	yes, Lemma 2.3.3
(2,3,2)	6	6	6	$\langle 0 \rangle$	yes, Proposition 2.3.7
(2,3,3)	9	9	9	$\langle 0 \rangle$	yes, Proposition 2.3.7
(3,1,1)	3	3	6	determinantal, Lemma 2.3.4	yes, Lemma 2.3.4
(3,1,2)	4	4	12	determinantal, Lemma 2.3.4	yes, Lemma 2.3.4
(3,1,3)	5	5	18	determinantal, Lemma 2.3.4	yes, Lemma 2.3.4
(3,2,1)	5	6	6	determinantal, Lemma 2.3.3	yes, Lemma 2.3.3
(3,2,2)	8	8	12	determinantal, Example 2.2.9	no, Remark 2.4.3
(3,2,3)	10	10	18	Example 2.2.9	no, Remark 2.4.3
(3,3,1)	6	6	6	$\langle 0 \rangle$	yes, Lemma 2.3.4
(3,3,2)	12	12	12	$\langle 0 \rangle$	no, Lemma 2.3.5
(3,3,3)	15	15	18	Example 2.4.4	no, Lemma 2.3.5

Table 2.1: Properties of neurovarieties for shallow polynomial neural networks with $d_i \in [3]$ and $r = 2$.

The ideal of the neurovariety for the architecture $\mathbf{d} = (3, 2, 3), r = 2$ contains the ideal generated by the 3×3 minors of the following 3×9 matrix:

$$\begin{pmatrix} c_{11}^{(1)} & \frac{1}{2}c_{12}^{(1)} & \frac{1}{2}c_{13}^{(1)} & c_{11}^{(2)} & \frac{1}{2}c_{12}^{(2)} & \frac{1}{2}c_{13}^{(2)} & c_{11}^{(3)} & \frac{1}{2}c_{12}^{(3)} & \frac{1}{2}c_{13}^{(3)} \\ \frac{1}{2}c_{12}^{(1)} & c_{22}^{(1)} & \frac{1}{2}c_{23}^{(1)} & \frac{1}{2}c_{12}^{(2)} & c_{22}^{(2)} & \frac{1}{2}c_{23}^{(2)} & \frac{1}{2}c_{12}^{(3)} & c_{22}^{(3)} & \frac{1}{2}c_{23}^{(3)} \\ \frac{1}{2}c_{13}^{(1)} & \frac{1}{2}c_{23}^{(1)} & c_{33}^{(1)} & \frac{1}{2}c_{13}^{(2)} & \frac{1}{2}c_{23}^{(2)} & c_{33}^{(2)} & \frac{1}{2}c_{13}^{(3)} & \frac{1}{2}c_{23}^{(3)} & c_{33}^{(3)} \end{pmatrix}.$$

However, it is not equal to this determinantal ideal. It is minimally generated by 94 polynomials. The dimension of the determinantal variety is eleven while the neurovariety has dimension ten.

2.3 Neuromanifolds and Symmetric Tensor Decompositions

In this section, we investigate the relationship between shallow PNNs and symmetric tensor decompositions. Building on this connection, we derive results for neuromanifolds for some shallow PNNs. This section has three subsections: §2.3.1 on symmetric tensors and homogeneous polynomials, §2.3.2 on results for neuromanifolds for $r = 2$ and §2.3.3 on results for neuromanifolds for general $r \in \mathbb{N}$. An overview of the results in §2.3.2 and §2.3.3 can be found in Table 2.2.

Consider a PNN with architecture $\mathbf{d} = (d_0, d_1, 1)$. Then $\mathcal{M}_{\mathbf{d},r}$ consists of homogeneous polynomials of degree r in d_0 many variables that can be represented as a linear combination of d_1 many linear forms raised to the r^{th} power. We should emphasize here that the linear combinations are taken over the *real* numbers. There is a bijection between the set of homogeneous polynomials of degree r in d_0 many variables and the set of order-r symmetric tensors of format $d_0 \times d_0 \times \cdots \times d_0$ as we will explain below. Formulated in the language of tensors, the neuromanifold $\mathcal{M}_{\mathbf{d},r}$ consists of order-r symmetric tensors of format

$\mathbf{d}$	r	Assumptions	Result
$(d_0, d_1, 1)$	2		Lemma 2.3.3
$(d_0, 1, d_2)$	2		Lemma 2.3.4
(d_0, d_0, d_2)	2	$d_0, d_2 \geq 2$	Lemma 2.3.5
$(d_0, 1, d_2)$	$r \in \mathbb{N}$		Lemma 2.3.6
(d_0, d_1, d_2)	$r \in \mathbb{N}$	$d_1 \geq \binom{r+d_0-1}{r}$	Proposition 2.3.7
$(2, d_1, 1)$	3	$d_1 \geq 2$	Corollary 2.3.11
$(2, d_1, 1)$	4	$d_1 \geq 3$	Corollary 2.3.11
$(2, d_1, 1)$	5	$d_1 \geq 3$	Corollary 2.3.11
$(3, d_1, 1)$	4	$d_1 \geq 6$	Corollary 2.3.12
$(3, d_1, 1)$	5	$d_1 \geq 7$	Corollary 2.3.12
$(4, d_1, 1)$	3	$d_1 \geq 5$	Corollary 2.3.12

Table 2.2: An overview of architectures for which we give results about their neuromanifolds in §2.3.

$d_0 \times d_0 \times \cdots \times d_0$ with *real symmetric rank* $\leq d_1$.

2.3.1 Symmetric Tensor Decompositions

An order-r tensor is a multidimensional array in $\mathbb{K}^{n_1 \times \ldots \times n_r}$, where $\mathbb{K}$ is a field. In this paper, we take $\mathbb{K} = \mathbb{R}$. A tensor $T = (T_{j_1 \ldots j_r}) \in \mathbb{R}^{d_0 \times \ldots \times d_0}$ is symmetric if $T_{j_1 \ldots j_r} = T_{\sigma(j_1) \ldots \sigma(j_r)}$ for all permuations $\sigma \in S_{d_0}$. Given an order-r symmetric tensor $T = (T_{j_1 \ldots j_r})$ of format $d_0 \times d_0 \times \cdots \times d_0$, one can associate the following homogeneous polynomial of degree r in d_0 variables to the tensor T:

$$F(\mathbf{x}) = \sum_{1 \leq j_1, j_2, \ldots, j_r \leq d_0} T_{j_1 j_2 \ldots j_r} x_{j_1} x_{j_2} \cdots x_{j_r}. \tag{2.3.1}$$

Monomials generally appear more than once in the above sum.

For a vector $\mathbf{i} = (i_1, \ldots, i_{d_0}) \in \mathbb{N}^{d_0}$ with $i_1 + \ldots + i_{d_0} = r$, the multinominal coefficient is defined as

$$\binom{r}{i_1, \ldots, i_{d_0}} = \frac{r!}{i_1! \cdots i_{d_0}!}.$$

Using multinomial coefficients, the polynomial (2.3.1) can be rewritten as

$$F(\mathbf{x}) = \sum_{i_1+i_2+\ldots+i_{d_0}=r} \binom{r}{i_1,\ldots,i_{d_0}} a_{i_1 i_2 \ldots i_{d_0}} x_1^{i_1} x_2^{i_2} \cdots x_{d_0}^{i_{d_0}} \tag{2.3.2}$$

such that each monomial appears precisely once in the sum.

To obtain the converse map from homogeneous polynomials to symmetric tensors, we note that there is a bijection between the monomials of degree r in d_0 variables and unique entries of a general order-r symmetric tensor of format $d_0 \times d_0 \times \cdots \times d_0$. The bijection is given by the map $f : \mathbb{N}^{d_0} \to \mathbb{N}^r, \mathbf{i} \mapsto \mathbf{j}$, where i_k denotes the number of appearances of k in $\mathbf{j}$ and the entries of $\mathbf{j}$ are in ascending order. The homogeneous polynomial (2.3.2) maps to the symmetric tensor T with the entries

$$T_{\mathbf{j}} = a_{f^{-1}(\mathbf{j})},$$

where the entries of $\mathbf{j}$ are in ascending order. The rest of the entries of T are obtained from the symmetry property. For more details on the bijection, we refer the reader to [Comon and Mourrain, 1996, Comon et al., 2008].

The outer product of r vectors $v_i \in \mathbb{R}^{n_i}$ is an order-r tensor defined as

$$v_1 \otimes \ldots \otimes v_r = (v_{1,i_1} \cdots v_{r,i_r})_{i_1,\ldots,i_r=1}^{n_1,\ldots,n_r}.$$

Definition 2.3.1. Let $T \in \mathbb{K}^{d_0 \times \ldots \times d_0}$ be a symmetric tensor. The *symmetric rank* of T over a field $\mathbb{K}$ is the smallest $k \in \mathbb{N}$ such that

$$T = \sum_{i=1}^{k} \lambda_i v_i \otimes \ldots \otimes v_i,$$

where $\lambda_i \in \mathbb{K}$ and $v_i \in \mathbb{K}^{d_0}$ for $i \in [k]$. If $\mathbb{K} = \mathbb{R}$ (resp. $\mathbb{K} = \mathbb{C}$), then we call it the *real* (resp. *complex*) *symmetric tensor rank*.

Note that for symmetric matrices, the symmetric rank is equal to the rank.

Let $I_1 \cup I_2 = [r]$ be a partition of the set $[r]$. Let $D_j = \prod_{i \in I_j} n_i$ for $j = 1, 2$. Every partition $I_1 \cup I_2 = [r]$ induces a *flattening* of a tensor $T \in \mathbb{K}^{n_1 \times \ldots \times n_r}$ to a matrix in $\mathbb{K}^{D_1 \times D_2}$. The importance of tensor flattenings is that their minors give relations for tensor rank. A non-zero tensor has rank one if and only if all 2×2 minors of all its flattenings vanish [Landsberg, 2011, §3.4].

Example 2.3.2. Let $d_0 = 2$ and $r = 3$. Then we consider cubic forms in two variables or, equivalently, order-3 $2 \times 2 \times 2$ symmetric tensors. The polynomial $x_1^3 + 3x_1x_2^2 + 3x_2^3$ corresponds to the symmetric tensor whose flattening corresponding to the partition $\{1, 2\} \cup \{3\}$ is $\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 3 \end{pmatrix}^T$.

2.3.2 Neuromanifolds for $r = 2$

Lemma 2.3.3. *The neuromanifold for the architecture* $\mathbf{d} = (d_0, d_1, 1), r = 2$ *consists of* $(d_0 \times d_0)$ *symmetric matrices*

$$\begin{pmatrix} c_{11} & \frac{1}{2}c_{12} & \cdots & \frac{1}{2}c_{1d_0} \\ \frac{1}{2}c_{12} & c_{22} & \cdots & \frac{1}{2}c_{2d_0} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{2}c_{1d_0} & \frac{1}{2}c_{2d_0} & \cdots & c_{d_0d_0} \end{pmatrix} \tag{2.3.3}$$

of rank at most d_1*. It is equal to its neurovariety and the ideal of the neurovariety is generated by all* $d_1 + 1$ *minors of the symmetric matrix. The architecture is filling if and only if* $d_1 \geq d_0$*.*

Proof. We have $p_{\mathbf{w}}(x) = w_{211}(w_{111}x_1 + \ldots + w_{11d_0}x_{d_0})^2 + \ldots + w_{21d_1}(w_{1d_11}x_1 + \ldots + w_{1d_1d_0}x_{d_0})^2$. Each of the summands of $p_{\mathbf{w}}(x)$ corresponds to a rank-1 symmetric matrix. Since there are d_1 summands, the sum corresponds to a symmetric matrix (2.3.3) of rank at most d_1. □

The next lemma is a special case of Lemma 2.3.6 and we state it here separately as the

special case is used in Table 2.1.

Lemma 2.3.4. *The neuromanifold for the architecture* $\mathbf{d} = (d_0, 1, d_2), r = 2$ *consists of* d_2 *tuples of* $d_0 \times d_0$ *symmetric matrices of rank at most one such that all the rank-1 matrices are multiples of each other. The neuromanifold is equal to the neurovariety and the ideal of the neurovariety is generated by the* 2×2 *minors of*

$$\begin{pmatrix} c_{11}^{(1)} & \frac{1}{2}c_{12}^{(1)} & \cdots & \frac{1}{2}c_{1d_0}^{(1)} & \cdots & c_{11}^{(d_2)} & \frac{1}{2}c_{12}^{(d_2)} & \cdots & \frac{1}{2}c_{1d_0}^{(d_2)} \\ \frac{1}{2}c_{12}^{(1)} & c_{22}^{(1)} & \cdots & \frac{1}{2}c_{2d_0}^{(1)} & \cdots & \frac{1}{2}c_{12}^{(d_2)} & c_{22}^{(d_2)} & \cdots & \frac{1}{2}c_{2d_0}^{(d_2)} \\ \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots & \ddots & \vdots \\ \frac{1}{2}c_{1d_0}^{(1)} & \frac{1}{2}c_{2d_0}^{(1)} & \cdots & c_{d_0d_0}^{(1)} & \cdots & \frac{1}{2}c_{1d_0}^{(d_2)} & \frac{1}{2}c_{2d_0}^{(d_2)} & \cdots & c_{d_0d_0}^{(d_2)} \end{pmatrix}.$$

The next lemma is due to Maksym Zubkov.

Lemma 2.3.5. *Let* $d_0, d_2 \geq 2$*, then for* $\mathbf{d} = (d_0, d_0, d_2), r = 2$*, the parameter map* $\Psi_{\mathbf{d},r}$ *is not surjective.*

Proof. We will first prove the statement for $\mathbf{d} = (d_0, d_0, 2), r = 2$. Assume to the contrary that the map $\Psi_{\mathbf{d},r}$ is surjective. For $i \in [d_0]$, we write $\ell_i = (W_1\mathbf{x})_i = w_{1i1}x_1 + \cdots + w_{1id_0}x_{d_0}$. Then for any pair of quadratic forms $(q_1, q_2) \in \mathrm{Sym}_2(\mathbb{R}^{d_0}) \times \mathrm{Sym}_2(\mathbb{R}^{d_0})$, we have some $\mathbf{w}$ such that

$$\Psi_{\mathbf{d},r}(\mathbf{w}) = \begin{pmatrix} w_{211}\ell_1^2 + \cdots + w_{21d_0}\ell_{d_0}^2 \\ w_{221}\ell_1^2 + \cdots + w_{22d_0}\ell_{d_0}^2 \end{pmatrix} = \begin{pmatrix} \ell_1 w_{211}\ell_1 + \cdots + \ell_{d_0} w_{21d_0}\ell_{d_0} \\ \ell_1 w_{221}\ell_1 + \cdots + \ell_{d_0} w_{22d_0}\ell_{d_0} \end{pmatrix} = \begin{pmatrix} \mathbf{x}^T W_1^T D_1 W_1 \mathbf{x} \\ \mathbf{x}^T W_1^T D_2 W_1 \mathbf{x} \end{pmatrix}$$

where D_i is the diagonal matrix with w_{2ij} for $i = 1, 2$ and $j = 1, \ldots, d_0$. So, if A_1 and A_2 are the corresponding matrices for quadratic forms q_1 and q_2, then we would have

$$A_1 = W_1^T D_1 W_1, \quad A_2 = W_1^T D_2 W_1.$$

This tells us that the two symmetric matrices A_1 and A_2 can be diagonalized simultaneously

which is in general not true as A_1 and A_2 do not commute in general for $d_0 \geq 2$. Therefore, $\Psi_{\mathbf{d},r}$ is not surjective.

The proof can be extended to the architectures (d_0, d_0, d_2) for any $d_2 \geq 2$ as it corresponds to simultaneous diagonalization of d_2 matrices which is impossible for an arbitrary choice of d_2 matrices. □

2.3.3 Neuromanifolds for $r \in \mathbb{N}$

Lemma 2.3.6. *The neuromanifold for the architecture* $\mathbf{d} = (d_0, 1, d_2), r \in \mathbb{N}$ *consists of* d_2 *order-*r $(d_0 \times d_0 \times \cdots \times d_0)$ *symmetric tensors of rank at most one such that all the rank-1 tensors are multiples of each other. The neuromanifold is equal to the neurovariety and the ideal of the neurovariety is generated by the* 2×2 *minors of the flattenings of the* $(d_0 \times \cdots \times d_0 \times d_0 d_2)$ *tensor that is obtained from combining the* d_2 $(d_0 \times d_0 \times \cdots \times d_0)$ *symmetric tensors along the last index.*

Proof. We have

$$p_{\mathbf{w}}(x) = \begin{pmatrix} w_{211}(w_{111}x_1 + \ldots + w_{11d_0}x_{d_0})^r \\ w_{221}(w_{111}x_1 + \ldots + w_{11d_0}x_{d_0})^r \\ \vdots \\ w_{2d_21}(w_{111}x_1 + \ldots + w_{11d_0}x_{d_0})^r \end{pmatrix}.$$

The form $(w_{111}x_1 + \ldots + w_{11d_0}x_{d_0})^r$ corresponds to an order-r rank-1 symmetric tensor. The i^{th} component of $p_{\mathbf{w}}(x)$ is the symmetric tensor multiplied by the constant w_{2i1}. This proves the first statement. For the second statement, we note that combining the rank-1 tensors to a $d_0 \times \cdots \times d_0 \times d_0 d_2$ tensor along the last index gives another rank-1 tensor. Moreover, any $d_0 \times \cdots \times d_0 \times d_0 d_2$ tensor of rank one such that each of the d_2 $(d_0 \times d_0 \times \cdots \times d_0)$-subtensors is symmetric and can be obtained from symmetric $(d_0 \times d_0 \times \cdots \times d_0)$ rank-1 tensors that are multiples of each other. The second statement follows from the result that the ideal of the tensors of rank at most one is generated by 2×2 minors of flattenings [Landsberg,

2011, §3.4]. □

Proposition 2.3.7. *Let* $\mathbf{d} = (d_0, d_1, d_2)$ *with* $d_1 \geq \binom{r+d_0-1}{r}$. *Then the neuromanifold itself is filling, i.e.,* $\mathcal{M}_{\mathbf{d},r} = (\mathrm{Sym}_r(\mathbb{R}^{d_0}))^{d_2}$.

Proof. Let $N \coloneqq \binom{r+d_0-1}{r}$. Write the coefficients of $p_{\mathbf{w}} = \Psi_{\mathbf{d},r}(\mathbf{w})$ as

$$C = \begin{pmatrix} (c_I^{(1)})_I \\ \vdots \\ (c_I^{(d_2)})_I \end{pmatrix} = \begin{pmatrix} w_{211} & w_{212} & \dots & w_{21d_1} \\ \vdots & \vdots & \ddots & \vdots \\ w_{2d_21} & w_{2d_22} & \dots & w_{2d_2d_1} \end{pmatrix} \begin{pmatrix} (W_{1,1})^I \\ \vdots \\ (W_{1,d_1})^I \end{pmatrix},$$

where I ranges over all multiindices in $\binom{[d_0]}{r}$ and $(W_{1,i})^I$ denotes the row consisting of all degree-r monomials formed by entries of the i^{th} row of W_1 with the corresponding multinomial coefficients. We want to show that any real $d_2 \times N$ matrix C can be written in this form. If $d_1 \geq N$ then for a general choice of parameters W_1 this matrix has a left-inverse with real entries and hence the system above can be solved over the reals. □

Definition 2.3.8. Let $\daleth$ be $\mathbb{R}$ or $\mathbb{C}$ and let

$$S^n_{d,k}(\daleth) = \{T \in \mathrm{Sym}_d(\daleth^n) : \text{ symmetric rank of } T \text{ is } k\}.$$

If $S^n_{d,k}(\daleth) \subseteq \mathrm{Sym}_d(\daleth^n)$ has non-empty interior with respect to the Euclidean topology, k is called a *typical symmetric rank* for order-d tensors of dimension n.

Over the complex numbers, there is a unique typical symmetric rank which is called the generic symmetric rank. However, over the real numbers there can be multiple typical symmetric ranks. In the following we will write $S^n_{d,k}$ for $S^n_{d,k}(\mathbb{R})$. A neuromanifold $\mathcal{M}_{\mathbf{d},r}$ having non-empty interior with respect to the Euclidean topology implies that the architecture $(\mathbf{d}, r)$ is filling. The importance of typical symmetric ranks for PNNs comes from the following fact which is a consequence of the discussions above and the inclusion $S^n_{d,k} \subseteq \mathcal{M}_{(n,k,1),d}$.

Theorem 2.3.9. *If k is a typical symmetric rank for order-d tensors of dimension n then the architecture $\mathbf{d} = (n, k, 1)$, $r = d$ is filling, i.e., $\mathcal{V}_{\mathbf{d},r} = \mathrm{Sym}_d(\mathbb{R}^n)$. Moreover, if there are two typical symmetric ranks $k < k'$ then $\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(n,k,1),d}) \subsetneq \mathrm{Sym}_d(\mathbb{R}^n)$ where $\mathrm{cl}_{\mathrm{Eucl}}$ denotes the closure with respect to the Euclidean topology.*

Let us consider the case $n = 2$ first; this is the setting of the paper [Comon and Ottaviani, 2012]. We summarize their findings below.

Theorem 2.3.10 ([Comon and Ottaviani, 2012, Proposition 2.2 & Main Theorem])**.**

1. *$S^2_{3,k}$ has non-empty interior only for $k \in \{2, 3\}$.*

2. *$S^2_{4,k}$ has non-empty interior only for $k \in \{3, 4\}$.*

3. *$S^2_{5,k}$ has non-empty interior only for $k \in \{3, 4, 5\}$.*

Using this result we obtain the following.

Corollary 2.3.11.

1. *$\mathbf{d} = (2, d_1, 1)$, $r = 3$ is filling for $d_1 \geq 2$ and*

$$\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(2,2,1),3}) \subsetneq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(2,3,1),3}) = \mathrm{Sym}_3(\mathbb{R}^2).$$

2. *$\mathbf{d} = (2, d_1, 1)$, $r = 4$ is filling for $d_1 \geq 3$ and*

$$\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(2,3,1),4}) \subsetneq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(2,4,1),4}) = \mathrm{Sym}_4(\mathbb{R}^2).$$

3. *$\mathbf{d} = (2, d_1, 1)$, $r = 5$ is filling for $d_1 \geq 3$ and*

$$\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(2,3,1),5}) \subsetneq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(2,4,1),5}) \subsetneq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(2,5,1),5}) = \mathrm{Sym}_3(\mathbb{R}^2).$$

Some similar results on typical symmetric ranks are also known for higher dimensional tensors though the difficulty increases quickly. We summarize some consequences below.

Corollary 2.3.12.

1. $\mathbf{d} = (3, d_1, 1),\ r = 4$ *is filling for* $d_1 \geq 6$ *and*

$$\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(3,6,1),4}) \subsetneq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(3,7,1),4}) \subseteq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(3,8,1),4}) = \mathrm{Sym}_4(\mathbb{R}^3).$$

2. $\mathbf{d} = (3, d_1, 1),\ r = 5$ *is filling for* $d_1 \geq 7$ *and*

$$\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(3,7,1),5}) \subsetneq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(3,8,1),5}) \subseteq \cdots \subseteq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(3,13,1),5}) = \mathrm{Sym}_5(\mathbb{R}^3).$$

3. $\mathbf{d} = (4, d_1, 1),\ r = 3$ *is filling for* $d_1 \geq 5$ *and*

$$\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(4,5,1),3}) \subsetneq \mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{(4,6,1),3}) = \mathrm{Sym}_3(\mathbb{R}^4)$$

Proof. The statements follow from Theorem 2.3.9 and [Bernardi et al., 2018, Theorem 1.2]. □

The algebraic boundary $\partial_{\mathrm{alg}}(\mathcal{M})$ of a set $\mathcal{M}$ is the Zariski closure of its topological boundary $\partial(\mathcal{M})$. Little is known about the algebraic boundaries in the cases considered above. Theorem 4.1 in [Michałek et al., 2016] implies that the algebraic boundary $\partial_{\mathrm{alg}}(\mathcal{M}_{(3,6,1),4})$ has an irreducible component of degree 51.

2.4 Neurovarieties

In this section, we will describe different approaches to studying neurovarieties. We start by studying the fibers of a neurovariety to characterize the neurovariety $\mathcal{V}_{(2,2,d_2),2}$ (Proposition 2.4.1) and partially the neuromanifold $\mathcal{M}_{(2,2,d_2),2}$ (Proposition 2.4.2). We use Grass-

mannians to describe the neurovariety $\mathcal{V}_{(3,3,3),2}$ (Example 2.4.4) and apply the Hilbert–Burch Theorem to study $\mathcal{V}_{(2,2,2,2),2}$ (Example 2.4.6).

Proposition 2.4.1. *The neurovariety $\mathcal{V}_{(2,2,d_2),2}$ is the vanishing locus of all 3×3 minors of*

$$C_{d_2} = \begin{pmatrix} c_{11}^{(1)} & c_{12}^{(1)} & c_{22}^{(1)} \\ \vdots & \vdots & \vdots \\ c_{11}^{(d_2)} & c_{12}^{(d_2)} & c_{22}^{(d_2)} \end{pmatrix} = \begin{pmatrix} w_{211} & w_{212} \\ \vdots & \vdots \\ w_{2d_21} & w_{2d_22} \end{pmatrix} \begin{pmatrix} w_{111}^2 & 2w_{111}w_{112} & w_{112}^2 \\ w_{121}^2 & 2w_{121}w_{122} & w_{122}^2 \end{pmatrix}.$$

Proof. Clearly $\mathcal{V}_{(2,2,d_2),2} \subseteq \mathcal{V}(\langle 3 \times 3 \text{ minors of } C_{d_2} \rangle)$ as $\mathrm{rk}(C_{d_2}) \leq 2$. $\mathcal{V}(\langle 3 \times 3 \text{ minors of } C_{d_2} \rangle)$ is irreducible and $\dim(\mathcal{V}(\langle 3 \times 3 \text{ minors of } C_{d_2} \rangle)) = 2d_2 + 2$, see e.g. [Bruns and Vetter, 2006, Proposition 1.1]. Consider the parameter map and the corresponding neurovariety

$$\Psi_{(2,2,d_2),2} \colon \underbrace{\mathbb{R}^{2\times 2} \times \mathbb{R}^{d_2 \times 2}}_{\dim 2d_2+4} \to (\mathrm{Sym}_2(\mathbb{R}^2))^{d_2}, \quad \mathcal{V}_{(2,2,d_2),2} = \overline{\mathrm{im}(\Psi_{(2,2,d_2),2})}.$$

The fibers of $\Psi_{(2,2,d_2),2}$ over $\mathcal{V}_{(2,2,d_2),2}$ are at least 2-dimensional by Lemma 2.2.7. For $d_2 = 2$, one can computationally check that over a generic matrix C_{d_2} the fiber is 2-dimensional. Since generic fibers are at least 2-dimensional, they are exactly 2-dimensional. For the case $d_2 > 2$, note that by setting the parameters $w_{211}, w_{212}, \ldots, w_{2(d_2-1)1}, w_{2(d_2-1)2}$ to zero and varying w_{2d_21}, w_{2d_22}, we obtain a 2-dimensional family not contained in the embedding of $\mathcal{M}_{(2,2,d_2-1),2}$ into $\mathcal{M}_{(2,2,d_2),2}$. As the dimension of the parameter space increases by two when passing from $d_2 - 1$ to d_2, the generic fiber is 2-dimensional by induction. Thus $\dim(\mathcal{V}_{(2,2,d_2),2}) = 2d_2 + 2$ and moreover $\mathcal{V}_{(2,2,d_2),2}$ is irreducible, hence $\mathcal{V}_{(2,2,d_2),2} = \mathcal{V}(\langle 3 \times 3 \text{ minors of } C_{d_2} \rangle))$. □

Proposition 2.4.2. *For $\mathbf{d} = (2, 2, d_2)$ ($d_2 \geq 2$) and $r = 2$, the Euclidean closure of $\mathcal{M}_{\mathbf{d},r}$ is strictly included in $\mathcal{V}_{\mathbf{d},r}$. In the case $\mathbf{d} = (2, 2, 2)$, we have*

$$C = \begin{pmatrix} c_{11}^{(1)} & c_{12}^{(1)} & c_{22}^{(1)} \\ c_{11}^{(2)} & c_{12}^{(2)} & c_{22}^{(2)} \end{pmatrix} \in \mathcal{M}_{(2,2,2),2}$$

if and only if

$$M_{1,3}(C)^2 \geq M_{1,2}(C)M_{2,3}(C),$$

where $M_{i,j}(C)$ *is the* 2×2 *minor of* C *obtained by taking the determinant of the* i^{th} *and* j^{th} *column. In particular, the algebraic boundary of* $\mathcal{M}_{(2,2,2),2}$ *is given by*

$$\partial_{alg}\mathcal{M}_{(2,2,2),2} = \mathcal{V}(M_{1,3}(C)^2 - M_{1,2}(C)M_{2,3}(C)).$$

Proof. First consider the case $\mathbf{d} = (2,2,2)$. We are looking for conditions on a matrix $A \in \mathbb{R}^{2\times 3}$ such that there exists $G \in GL_2(\mathbb{R})$ with

$$\begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{pmatrix} = \begin{pmatrix} w_{111}^2 & 2w_{111}w_{112} & w_{112}^2 \\ w_{121}^2 & 2w_{121}w_{122} & w_{122}^2 \end{pmatrix}. \tag{2.4.1}$$

We consider two cases: when both w_{111}, w_{121} are non-zero and when at least one of them is zero. If w_{111} and w_{121} are non-zero, then after rescaling, w.l.o.g. we can assume $w_{111} = w_{121} = 1$. Each matrix entry in (2.4.1) yields a polynomial equation; these generate an ideal I in the polynomial ring $\mathbb{R}[g_{11}, \ldots, g_{22}, w_{112}, w_{122}, a_{11}, \ldots, a_{23}]$. Two generators in a Gröbner basis of I with respect to lexicographic ordering are

$$w_{1i2}^2 a_{11}a_{22} - w_{1i2}^2 a_{12}a_{21} - 2w_{1i2}a_{11}a_{23} + 2w_{1i2}a_{13}a_{21} + a_{12}a_{23} - a_{13}a_{22}, \tag{2.4.2}$$

where $i \in \{1,2\}$. These quadratic equations in w_{1i2} have discriminant

$$\Delta = M_{1,3}(A)^2 - M_{1,2}(A)M_{2,3}(A),$$

where A is the 2×3 matrix with entries a_{ij} from (2.4.1). The denominator of the quadratic formula for (2.4.2) is $M_{1,2}(A)$. Therefore, real solutions to (2.4.1) with w_{111}, w_{121} non-zero exist if and only if $M_{1,3}(A)^2 \geq M_{1,2}(A)M_{2,3}(A)$ and $M_{1,2}(A) \neq 0$.

When at least one $w_{1i1} = 0$, then possibly after rescaling, the Gröbner basis of I with respect to the lexicographic ordering contains $w_{i12}^2 a_{11} a_{22} - w_{i12}^2 a_{12} a_{21}$. This implies $w_{i12} = 0$ or $M_{1,2}(A) = 0$. The former condition implies $M_{1,2}(A) = 0$, so we do not have to consider it separately. When $M_{1,2}(A) = 0$, the inequality $M_{1,3}(A)^2 \geq M_{1,2}(A) M_{2,3}(A)$ is automatically satisfied. Therefore, real solutions to (2.4.1) with at least one of w_{111}, w_{121} zero exist if and only if $M_{1,3}(A)^2 \geq M_{1,2}(A) M_{2,3}(A)$ and $M_{1,2}(A) = 0$. Combining the two cases gives that real solutions to (2.4.1) exist if and only if $M_{1,3}(A)^2 \geq M_{1,2}(A) M_{2,3}(A)$.

As this inequality remains unchanged after multiplying A with a 2×2 matrix (both sides get multiplied by the squared determinant of the matrix which is positive), we conclude $C \in \mathcal{M}_{(2,2,2),2}$ if and only if $M_{1,3}(C)^2 \geq M_{1,2}(C) M_{2,3}(C)$.

In the case $d_2 > 2$, it is clear that at least for any two rows i, j of C, the inequality

$$M_{1,3}(C_{i,j})^2 \geq M_{1,2}(C_{i,j}) M_{2,3}(C_{i,j})$$

needs to hold. Thus we get a full-dimensional set of points in $\mathcal{V}_{\mathbf{d},r} \setminus \mathcal{M}_{\mathbf{d},r}$, hence $\mathrm{cl}_{\mathrm{Eucl}}(\mathcal{M}_{\mathbf{d},r}) \subsetneq \mathcal{V}_{\mathbf{d},r}$. □

Remark 2.4.3. For example, if $A = \begin{pmatrix} a & \star & -a \\ b & \star & -b \end{pmatrix}$ in the proof of Proposition 2.4.2, then the equation

$$\begin{pmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{pmatrix} \begin{pmatrix} a & \star & -a \\ b & \star & -b \end{pmatrix} = \begin{pmatrix} w_{111}^2 & 2w_{111}w_{112} & w_{112}^2 \\ w_{121}^2 & 2w_{121}w_{122} & w_{122}^2 \end{pmatrix}$$

does not have a solution. This example can be extended to any architecture $\mathbf{d} = (d_0, 2, d_2)$ with $d_0, d_2 \geq 2$ and activation degree $r = 2$ to show that the neuromanifold $\mathcal{M}_{\mathbf{d},r}$ is not equal to $\mathcal{V}_{\mathbf{d},r}$.

In the rest of the section, we use more advanced tools from algebraic geometry to compute neurovarieties. A naïve elimination approach fails to compute these varieties.

Example 2.4.4. Consider the architecture $\mathbf{d} = (3, 3, 3)$, $r = 2$; the image of $\Psi_{\mathbf{d},r}$ consists of three linear combinations of three squares of linear forms in three variables. Let us take a 3×6 matrix representing the embedding of $\mathbb{P}^2_x \times \mathbb{P}^2_y \times \mathbb{P}^2_z$ via the degree two Veronese map (with corresponding scaling) into $\mathbb{P}^5 \times \mathbb{P}^5 \times \mathbb{P}^5$:

$$\begin{pmatrix} x_0^2 & 2x_0x_1 & 2x_0x_2 & x_1^2 & 2x_1x_2 & x_2^2 \\ y_0^2 & 2y_0y_1 & 2y_0y_2 & y_1^2 & 2y_1y_2 & y_2^2 \\ z_0^2 & 2z_0z_1 & 2z_0z_2 & z_1^2 & 2z_1z_2 & z_2^2 \end{pmatrix}$$

By taking all the 3×3 minors (20 in total) of this matrix, we get an embedding into the Grassmannian $\mathrm{Gr}(3, 6)$ in its Plücker coordinates. The ideal of this variety in Plücker coordinates can be found using elimination. Its dimension is six and its degree is 57. Now for each of the 20 Plücker coordinates we substitute a 3×3 minor of a 3×6 matrix of unknowns, where each unknown represents a coefficient of a degree two monomial (we view $\mathbb{P}^5$ as the space of all homogeneous quadrics in three variables):

$$C = \begin{pmatrix} c_{11}^{(1)} & c_{12}^{(1)} & \dots & c_{33}^{(1)} \\ c_{11}^{(2)} & c_{12}^{(2)} & \dots & c_{33}^{(2)} \\ c_{11}^{(3)} & c_{12}^{(3)} & \dots & c_{33}^{(3)} \end{pmatrix}$$

The resulting ideal I gives rise to a variety $X = \mathcal{V}(I)$. This variety has two irreducible components: one is given by the vanishing of all 3×3 minors of C, the other is the neurovariety $\mathcal{V}_{(3,3,3),2}$ embedded into $\mathbb{P}^5_{c^{(1)}} \times \mathbb{P}^5_{c^{(2)}} \times \mathbb{P}^5_{c^{(3)}}$. The codimension of $\mathcal{V}_{(3,3,3),2} \subseteq \mathbb{P}^5_{c^{(1)}} \times \mathbb{P}^5_{c^{(2)}} \times \mathbb{P}^5_{c^{(3)}}$ is three, hence the affine dimension is $\dim(\mathcal{V}_{\mathbf{d},r}) = 18 - 3 = 15$.

The first part of the computation can be found in [Brodsky and Sturmfels, 2011]; we implemented the whole procedure and made it available at [Kubjas et al., 2024].

We would like to thank Bernd Sturmfels for telling us about the following example; the computation is motivated by the Hilbert–Burch Theorem, see for instance [Eisenbud,

2013, Theorem 20.15].

Theorem 2.4.5 (Hilbert–Burch)**.** *Let R be a local ring with an ideal $I \subset R$ such that*

$$R^n \xrightarrow{f} R^m \to R \to R/I \to 0$$

is a free resolution of R/I, then $m = n - 1$ and $I = aJ$ where a is a regular element of R and J is a depth-two ideal generated by all $m \times m$ minors of the matrix representing f.

Consider the architecture ($\mathbf{d} = (2, 2, 2, 2), r = 2$). We expect that $\mathcal{V}_{\mathbf{d},r}$ has dimension eight and thus has codimension two in $(\mathrm{Sym}_4(\mathbb{R}^2))^2$. Therefore, we can hope that by the Hilbert–Burch Theorem we can find a matrix whose minors generate the ideal of $\mathcal{V}_{\mathbf{d},r}$. Again, we work in projective space, i.e., we consider $\mathcal{V}_{\mathbf{d},r}$ as a projective variety inside $\mathbb{P}^4_{c^{(1)}} \times \mathbb{P}^4_{c^{(2)}}$.

Example 2.4.6. One finds that the ideal cutting out $\mathcal{V}_{(2,2,2,2),2} \subseteq \mathbb{P}^4_{c^{(1)}} \times \mathbb{P}^4_{c^{(2)}}$ is generated by the 5×5 minors of the following matrix

$$\begin{pmatrix} c^{(2)}_{1112} & -c^{(2)}_{1111} & 0 & c^{(1)}_{1112} & -c^{(1)}_{1111} & 0 \\ 4c^{(2)}_{1122} & -c^{(2)}_{1112} & 8c^{(2)}_{1111} & 4c^{(1)}_{1122} & -c^{(1)}_{1112} & 8c^{(1)}_{1111} \\ 8c^{(2)}_{1222} & 0 & 8c^{(2)}_{1112} & 8c^{(1)}_{1222} & 0 & 8c^{(1)}_{1112} \\ 8c^{(2)}_{2222} & c^{(2)}_{1222} & 4c^{(2)}_{1122} & 8c^{(1)}_{2222} & c^{(1)}_{1222} & 4c^{(1)}_{1122} \\ 0 & c^{(2)}_{2222} & c^{(2)}_{1222} & 0 & c^{(1)}_{2222} & c^{(1)}_{1222} \end{pmatrix}.$$

2.5 Dimension

2.5.1 A plethora of conjectures

Determining dimensions of neurovarieties is a difficult task and there exists a wide range of open conjectures in this direction. A classical result is the Alexander–Hirschowitz Theorem; already conjectured in the late 19^{th} century, the proof was completed about a century later in 1995. Here is a formulation in the language of PNNs.

Theorem 2.5.1 (Alexander–Hirschowitz [Alexander and Hirschowitz, 1995]). *If* $\mathbf{d} = (d_0, d_1, 1)$, $\mathcal{V}_{\mathbf{d},r}$ *attains the expected dimension* $\min\{d_0 d_1, \binom{d_0+r-1}{r}\}$, *except for the following cases:*

1. $r = 2,\ 2 \leq d_1 < d_0$,
2. $r = 3,\ d_0 = 5,\ d_1 = 7$ *where* $\dim(\mathcal{V}_{(5,7,1),3}) = 34$ *(defect 1),*
3. $r = 4,\ d_0 = 3,\ d_1 = 5$ *where* $\dim(\mathcal{V}_{(3,5,1),4}) = 14$ *(defect 1),*
4. $r = 4,\ d_0 = 4,\ d_1 = 9$ *where* $\dim(\mathcal{V}_{(4,9,1),4}) = 34$ *(defect 1),*
5. $r = 4,\ d_0 = 5,\ d_1 = 14$ *where* $\dim(\mathcal{V}_{(5,14,1),4}) = 69$ *(defect 1).*

It is conjectured that the neurovariety attains the expected dimension for large activation degree. Compare this for example with the Alexander–Hirschowitz Theorem which tells us there exist no defective single-output two-layer neurovarieties with activation degree $r \geq 5$. The conjecture appeared as a theorem in [Kileel et al., 2019b], but a mistake in the proof was pointed out by Theo Elenius in his BSc thesis. After the present article first appeared as a preprint on arXiv, the conjecture has been proven in [Finkel et al., 2024, Theorem 12].

Conjecture 2.5.2 ([Kileel et al., 2019b], Theorem 14). *For any fixed widths* $\mathbf{d} = (d_0, \ldots, d_L)$ *with* $d_i > 1$ *for* $i = 1, \ldots, L-1$, *there exists* $\tilde{r} = \tilde{r}(\mathbf{d})$ *such that whenever* $r > \tilde{r}$, *the neurovariety* $\mathcal{V}_{\mathbf{d},r}$ *attains the expected dimension.*

It is, however, not true that $\mathcal{V}_{\mathbf{d},r}$ being defective implies that $\mathcal{V}_{\mathbf{d},r'}$ is defective for all $r' < r$. For example, $\mathcal{V}_{(5,7,1),3}$ has defect one whereas $\mathcal{V}_{(5,7,1),2}$ is non-defective.

Also note that the assumption $d_i > 1$ for $i = 1, \ldots, L-1$ is necessary as is shown by the following example.

Example 2.5.3. Consider the widths $\mathbf{d} = (2, 1, 2, 1)$. We claim that for any $r > 1$, $\mathcal{V}_{\mathbf{d},r}$ has

defect one, i.e., $\dim(\mathcal{V}_{\mathbf{d},r}) = 2$. Indeed, observe that the parameter map $\Psi_{\mathbf{d},r}$ is given by

$$\begin{aligned}\mathbf{x} \mapsto (w_{111}x_1 + w_{112}x_2)^r &\mapsto \begin{pmatrix} w_{211}^r(w_{111}x_1 + w_{112}x_2)^{r^2} \\ w_{221}^r(w_{111}x_1 + w_{112}x_2)^{r^2} \end{pmatrix} \\ &\mapsto (w_{311}w_{211}^r + w_{312}w_{221}^r)(w_{111}x_1 + w_{112}x_2)^{r^2}.\end{aligned}$$

As all weights coming from the last two layers can be factored out, we get that $\mathcal{V}_{\mathbf{d},r} \cong \mathcal{V}_{(2,1,1,1),r}$. The latter neurovariety is immediately seen to have dimension two. This generalizes to the following statement.

Proposition 2.5.4. *Let $\mathbf{d}_0 \in \mathbb{N}^{n_0}$ and $\mathbf{d}_2 \in \mathbb{N}^{n_2}$. For $\mathbf{d} = (\mathbf{d}_0, 1, \mathbf{d}_2, d_3)$, the neuromanifold $\mathcal{M}_{\mathbf{d},r}$ is equal to $\mathcal{M}_{(\mathbf{d}_0,1,\mathbf{1},d_3),r}$ for any r. In particular, for $d_3 = 1$, $\dim(\mathcal{M}_{\mathbf{d},r}) = \dim(\mathcal{M}_{(\mathbf{d}_0,1),r})$.*

Proof. Fix parameters $\mathbf{w} = (W_1, W_2, \ldots, W_L)$. Let $p = \Psi_{(\mathbf{d}_0,1),r}(\pi_1(\mathbf{w}))$, $q = \Psi_{(1,\mathbf{d}_2,1)}(\pi_2(\mathbf{w}))$, where π_1 denotes the projection to the parameter space corresponding to the first $n_0 + 1$ layers, and π_2 denotes the projection to the remaining parameters. Then $p_{\mathbf{w}}(\mathbf{x})$ factors as a composition $p_{\mathbf{w}}(\mathbf{x}) = (q \circ p)(\mathbf{x})$. The polynomial q is a single-input neural network, hence we can write $(q \circ p)_i(\mathbf{x})$ as a product $\alpha_i(\pi_2(\mathbf{w})) \cdot p(\mathbf{x})^{r^{n_2}}$ where $\alpha_i(\pi_2(\mathbf{w}))$ only depends on weights corresponding to the last $n_2 + 2$ layers and thus

$$\mathcal{M}_{\mathbf{d},r} = \{(\alpha_i \cdot p^{r^{n_2}})_i \colon \alpha_i \in \mathbb{R}, p \in \mathcal{M}_{(\mathbf{d}_0,1),r}\}.$$

The latter set is by definition equal to $\mathcal{M}_{(\mathbf{d}_0,1,\mathbf{1},d_3),r}$. In the case $d_3 = 1$, its dimension is equal to $\dim(\mathcal{M}_{(\mathbf{d}_0,1),r})$. □

Remark 2.5.5. This allows to give a strong bound on the dimension of $\mathcal{M}_{\mathbf{d},r}$ for $\mathbf{d} = (\mathbf{d}_0, 1, \mathbf{d}_2, d_3)$:

$$\dim(\mathcal{M}_{\mathbf{d},r}) = \dim(\mathcal{M}_{\mathbf{d}_0,1,\mathbf{1},d_3}, r) \leq \operatorname{edim}(\mathcal{M}_{(\mathbf{d}_0,1,\mathbf{1},d_3),r}) = \dim(\mathcal{M}_{\mathbf{d}_0,r}) + d_3 - 1.$$

If $d_0 > 1$, this is strictly less than the ambient dimension which is $d_3\binom{r^{L-1}+d_0-1}{r^{L-1}}$, where d_0 is the width of the first layer. Hence the width one at the $(n_0+1)^{\text{st}}$ layer is an instance of an *asymptotic bottleneck*, i.e., the architecture is not filling regardless of adding more layers or changing the widths in all other layers except the input and $(n_0+1)^{\text{st}}$ layer [Kileel et al., 2019b, Def. 18].

Remark 2.5.6. For $\mathbf{d} = (d_0, 1, d_2, 1)$, the neuromanifold $\mathcal{M}_{\mathbf{d},r}$ is equal to $\mathcal{M}_{(d_0,1,1,1),r}$ for any r. In particular, $\dim(\mathcal{M}_{\mathbf{d},r}) = d_0$. This enables us to compute the dimension of many defective neurovarieties. For example, $\dim(\mathcal{V}_{(3,1,5,1),3}) = 3$ and therefore $\mathcal{V}_{(3,1,5,1),3}$ has defect 4.

In contrast to the asymptotic statement for large activation degree in Conjecture 2.5.2, we also conjecture the following.

Conjecture 2.5.7. *Let $\mathbf{d} = (d_0, d_1, \ldots, d_L)$ be a non-increasing sequence with $d_L > 1$. Then for any r, the neurovariety $\mathcal{V}_{\mathbf{d},r}$ attains the expected dimension.*

We verified this conjecture using Algorithm 1 described in §2.6.4 for four- and five-layer PNNs with widths up to three and activation degree up to five, see [Kubjas et al., 2024]. In the context of feedforward neural networks with ReLU activation this statement has been proved [Grigsby et al., 2022, Theorem 8.11].

Using Lemma 2.2.7, one can deduce the following recursive dimension bound.

Proposition 2.5.8 (Recursive bound, [Kileel et al., 2019b, Proposition 17])**.** *For any $r \in \mathbb{N}$, $\mathbf{d} = (d_0, \ldots, d_i, \ldots, d_L)$ and $i \in \{1, \ldots, L-1\}$, we have*

$$\dim(\mathcal{V}_{\mathbf{d},r}) \leq \dim(\mathcal{V}_{(d_0,\ldots,d_i),r}) + \dim(\mathcal{V}_{(d_i,\ldots,d_L),r}) - d_i.$$

Corollary 2.5.9. *Let $\tilde{\mathbf{d}} = (d_0, \ldots, d_{i-1}, \mathbf{d}, d_{j+1}, \ldots, d_L)$, where $\mathbf{d} = (d_i, d_{i+1}, \ldots, d_j)$ is an architecture such that $\mathcal{V}_{\mathbf{d},r}$ is defective for some $r > 0$. Here we allow $i = 0$ or $j = L$. Suppose $\mathcal{V}_{\tilde{\mathbf{d}},r}$ is not filling, then $\mathcal{V}_{\tilde{\mathbf{d}},r}$ is defective.*

Proof. It follows from Proposition 2.5.8 that

$$\dim(\mathcal{V}_{\tilde{\mathbf{d}},r}) \leq \dim(\mathcal{V}_{(d_0,\ldots,d_i),r}) + \dim(\mathcal{V}_{\mathbf{d},r}) + \dim(\mathcal{V}_{(d_j,\ldots,d_L),r}) - d_i - d_j \tag{2.5.1}$$

where $\dim(\mathcal{V}_{(d_0,\ldots,d_i),r})$ or $\dim(\mathcal{V}_{(d_j,\ldots,d_L),r})$ might not appear in the sum above. By Definition 2.2.8, the expected dimension is defined as

$$\operatorname{edim}(\mathcal{V}_{\mathbf{d},r}) = \min\left\{ d_L + \sum_{i=0}^{L-1}(d_i d_{i+1} - d_{i+1}), d_L \binom{d_0 + r^{L-1} - 1}{r^{L-1}} \right\}.$$

Since $\dim(\mathcal{V}_{\mathbf{d},r}) < \operatorname{edim}(\mathcal{V}_{\mathbf{d},r})$, it follows from equation (2.5.1) that

$$\begin{aligned}\dim(\mathcal{V}_{\tilde{\mathbf{d}},r}) &< d_L + \sum_{\alpha=0}^{i-1}(d_\alpha d_{\alpha+1} - d_{\alpha+1}) + \sum_{\beta=i}^{j-1}(d_\beta d_{\beta+1} - d_{\beta+1}) + \sum_{\gamma=j}^{L}(d_\gamma d_{\gamma+1} - d_{\gamma+1}) \\ &= d_L + \sum_{\alpha=0}^{L-1}(d_\alpha d_{\alpha+1} - d_{\alpha+1})\end{aligned}$$

where $\sum_{\alpha=0}^{i-1}(d_\alpha d_{\alpha+1} - d_{\alpha+1})$ or $\sum_{\gamma=j}^{L}(d_\gamma d_{\gamma+1} - d_{\gamma+1})$ might not appear in the sum above. Since $\mathcal{V}_{\tilde{\mathbf{d}},r}$ is non-filling, it follows directly that $\mathcal{V}_{\tilde{\mathbf{d}},r}$ is defective. □

Remark 2.5.10. The converse to this statement is not true. For example, consider $\mathbf{d} = (2,2,1,2),\ r = 2$. Then $\mathcal{V}_{\mathbf{d},r}$ has defect one, but both $\mathcal{V}_{(2,2,1),2}$ and $\mathcal{V}_{(2,1,2),2}$ are non-defective.

Let $\preccurlyeq$ be the partial order on the set of widths induced by coordinatewise comparison. The following conjecture suggests a unimodal distribution of layer widths within a neural network is efficient. In the realm of machine learning theory, this architectural pattern is commonly believed to allow the network to initially expand its capacity for feature representation, enabling the network to capture intricate patterns within the data. As the layers narrow, heuristically, the network would refine these features into more sophisticated representations suitable for making predictions. The conjecture agrees with this common heuristic.

Conjecture 2.5.11 ([Kileel et al., 2019b], Conjecture 12)**.** *Fix L, d_0, d_L and r; any minimal*

(with respect to $\preccurlyeq$) vector of widths $\mathbf{d} = (d_0, d_1, \ldots, d_L)$ such that the architecture $\mathcal{V}_{\mathbf{d},r}$ is filling, is unimodal, i.e. there exists $i \in \{0, 1, \ldots, L\}$ such that $(d_0, \ldots, d_i)$ is weakly increasing and $(d_i, \ldots, d_L)$ is weakly decreasing.

2.5.2 A look into the symmetries of an exceptional shallow network

We explicitly describe a family of symmetries beyond multi-homogeneity (Lemma 2.2.7) for the architecture $\mathbf{d} = (5, 7, 1),\ r = 3$, one of the exceptional cases of the Alexander–Hirschowitz Theorem. The expected dimension of $\mathcal{V}_{\mathbf{d},r}$ is 35, however, its actual dimension is 34. Therefore, there exists a one-dimensional family of symmetries in the parameter space not of the form as in Lemma 2.2.7.

In the following we will use an equivalent formulation of the Alexander–Hirschowitz Theorem as can be found in [Brambilla and Ottaviani, 2008, Theorem 1.1].

Theorem 2.5.12 (Alexander–Hirschowitz, geometric version)**.** *Let X be a general collection of k double points in $\mathbb{P}^n$ and let $I_X(d) \subseteq \mathrm{Sym}_d(\mathbb{C}^n)$ be the subspace of polynomials through X, i.e., with all first partial derivatives vanishing at the points of X. Then the subspace $I_X(d)$ has the expected codimension $\min\{(n+1)k, \binom{n+d}{n}\}$ except in the cases as in Theorem 2.5.1 (with the notation $d = r,\ n = d_0 - 1,\ k = d_1$).*

Then geometrically the situation depicts as follows: consider seven points $p_1, \ldots, p_7$ in $\mathbb{P}^4$. We are interested in cubics singular at these seven points, i.e., we are looking for $f \in \mathrm{Sym}_3(\mathbb{C}^5)$ such that $f(p_i) = \mathrm{d}f_{p_i} = 0$ for $i = 1, \ldots, 7$. One would expect that no such cubic exists: the space of cubics in five variables has dimension 35 and the seven points are expected to impose 35 independent conditions. But through seven points there exists a rational normal curve C

which after a suitable coordinate transformation might be given as

$$C = \left\{ (x_0 : \cdots : x_4) \in \mathbb{P}^4 \colon \operatorname{rk} \begin{pmatrix} x_0 & x_1 & x_2 \\ x_1 & x_2 & x_3 \\ x_2 & x_3 & x_4 \end{pmatrix} \leq 1 \right\}.$$

A different way to write the curve C is as follows. Assume the five points $p_1, \ldots, p_5$ are the points $(1:0:0:0:0), \ldots, (0:0:0:0:1)$; if $p_1, \ldots, p_7$ are in general position p_6 and p_7 have non-zero coordinates then. Consider the birational Cremona transformation $\tau \colon (x_0 : \cdots : x_5) \mapsto (x_0^{-1} : \cdots : x_5^{-1})$. Then C is the preimage under τ of the line $\overline{\tau(p_6)\tau(p_7)}$.

It is well-known that the secant variety of such a determinantal variety is given by

$$\sigma(C) = \left\{ (x_0 : \cdots : x_4) \in \mathbb{P}^4 \colon \det \begin{pmatrix} x_0 & x_1 & x_2 \\ x_1 & x_2 & x_3 \\ x_2 & x_3 & x_4 \end{pmatrix} = 0 \right\}.$$

This is a cubic hypersurface with singular locus C; in particualar, $\sigma(C)$ is singular at $p_1, \ldots, p_7$. See [Brambilla and Ottaviani, 2008, §3].

Now it is possible to vary p_7 along C such that the construction of $\sigma(C)$ remains unchanged. Consider the PNN

$$W_2 \circ \rho_3 \circ W_1 \mathbf{x}$$

where $\mathbf{x} \in \mathbb{R}^5$, $W_1 \in \mathbb{R}^{7\times 5}$, $W_2 \in \mathbb{R}^{1\times 7}$. Assume similar to above that the curve $\tau(C)$ is given by the 2×2 minors of

$$\begin{pmatrix} w_{161}^{-1} & \ldots & w_{165}^{-1} \\ w_{171}^{-1} & \ldots & w_{175}^{-1} \end{pmatrix}. \tag{2.5.2}$$

We are looking for a symmetric matrix $G \in \mathrm{GL}(7, \mathbb{R})$ such that GW_1 gives rise to the same curve $\tau(C)$. Again, w.l.o.g., we assume that the first five rows and columns of G are the

identity matrix. Thus, restricting to the last two rows and columns of G, we require that

$$\begin{pmatrix} g_1 & g_2 \\ g_2 & g_3 \end{pmatrix} \begin{pmatrix} w_{161}^{-1} & \dots & w_{165}^{-1} \\ w_{171}^{-1} & \dots & w_{175}^{-1} \end{pmatrix}$$

have the same 2×2 minors as (2.5.2). Normalizing G to have determinant one we can solve this to obtain

$$G = \begin{pmatrix} \mathrm{Id}_5 & 0 & 0 \\ 0 & g & \phi \\ 0 & \phi & \frac{1+\phi^2}{g} \end{pmatrix}$$

where $\phi = (\frac{1}{2}(-1+\sqrt{3}))^{1/2}$ and $g \in \mathbb{R}$ is a free parameter. Thus we have obtained a one-dimensional family of symmetries on the parameter space leaving the image of the parameter map $\Psi_{\mathbf{d},r}$ unchanged.

2.6 Optimization

In supervised machine learning tasks, a neural network is trained through the process of *empirical risk minimization (ERM)*. Consider a training dataset $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), ..., (\mathbf{x}_N, \mathbf{y}_N)\}$, where each $\mathbf{x}_i$ is an input vector and $\mathbf{y}_i$ is the corresponding output or label. The objective in ERM is to find a function f from a hypothesis space $\mathcal{H}$ that minimizes the empirical risk, which is defined as the average loss over the training data:

$$\hat{R}(f) = \frac{1}{N} \sum_{i=1}^{N} L(\mathbf{y}_i, f(\mathbf{x}_i)).$$

Here, $\hat{R}(f)$ represents the empirical risk of the function f, L is a loss function that measures the discrepancy between the predicted value $f(\mathbf{x}_i)$ and the actual label $\mathbf{y}_i$, and N is the number of samples in the training dataset.

The goal of ERM is to select a function f^* from the hypothesis space $\mathcal{H}$ that minimizes

this empirical risk:

$$f^* = \arg\min_{f \in \mathcal{H}} \hat{R}(f). \tag{2.6.1}$$

To minimize empirical risk, gradient-based optimization methods such as gradient descent (GD), stochastic gradient descent (SGD), and adaptive moment estimation (Adam) are commonly used. For convex objective functions, gradient-based algorithms can converge to the unique global minimum with appropriate hyperparameter choices. However, when optimizing neural networks, the loss landscape is typically non-convex. As a result, there is generally no theoretical guarantee that gradient-based methods will reach a global optimum f^* for feedforward neural networks.

Recent research on convergence properties in gradient-based algorithms often focuses on the dynamics of the optimization process. One approach analyzes convergence rates by relaxing assumptions on the objective function. For example, it has been shown that for objective functions with a unique global minimum, gradient descent converges to the global minimum at a geometric rate if the function satisfies conditions such as the Polyak–Lojasiewicz inequality [Karimi et al., 2016, Li and Orabona, 2019, Khaled and Richtárik, 2020, Mei et al., 2021], weak quasi-convexity [Hardt et al., 2018], or the restricted Secant inequality [Hardt et al., 2018, Yi et al., 2020, Guille-Escuret et al., 2022]. Another line of work explores algorithmic regularization. Empirical evidence suggests that large-scale neural networks used in practice are highly over-parameterized, often containing far more trainable parameters than training examples. Consequently, optimization objectives for such high-capacity models tend to have many global minima that interpolate the data perfectly. The choice of optimization algorithm thus introduces an implicit inductive bias, guiding the solution toward specific types of global minima. One of the first results in this direction shows that for almost all linearly separable datasets, gradient descent with any initialization and any bounded stepsize converge in direction to maximum margin separator with unit l_2 norm [Soudry et al., 2018]. Both approaches build on the assumption of the existence and

separation of global minima, while the global picture of the distribution of critical points is not well-understood. Through algebro-geometric approach, we study the geometry of the optimization landscape and provide the first explicit upper bound to the number of critical points when optimizing over a special family of polynomial neural networks.

Note that in the neural network setting, the hypothesis class $\mathcal{H}$ is just the corresponding neuromanifold, and the empirical minimization problem becomes

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \ell_{F_{\boldsymbol{\theta}}} \quad \text{where } \ell_{F_{\boldsymbol{\theta}}} \coloneqq \frac{1}{N}\sum_{i=1}^{N} \ell(\mathbf{y}_i, F_{\boldsymbol{\theta}}(\mathbf{x}_i)) \tag{2.6.2}$$

Considering the ℓ_2 loss, we minimize the average distance between $F_{\boldsymbol{\theta}}(\mathbf{x})$ and $\mathbf{y}$. It turns out that, analogously to maximum likelihood estimation or Euclidean distance minimization, there exists a degree of this optimization problem constituting a measure for the algebraic complexity of solving this problem. In §2.6.1, we study the static optimization properties of a PNN through the generic ED degree of the corresponding neurovariety. In §2.6.2 and §2.6.4, we explain the dynamic optimization process of gradient descent, followed by backpropagation. The algorithm biases induced by the optimization methods, known as *implicit bias*, on general neural network architectures are still largely unknown. While limited theoretical results are known for the dynamic optimization process of neural networks in general, recent work has shown that gradient descent finds a global minimum of linear neural networks (i.e., polynomial neural networks with $r = 1$) under mild assumptions [Arora et al., 2018]. In §2.6.5, we review the trajectory-based convergence results on linear neural networks and phrase open questions for polynomial neural networks with activation degree $r \geq 2$. Sections §2.6.2, §2.6.4 and §2.6.5 are expository and do not contain new results.

2.6.1 The Learning Degree of a PNN

Let us consider the case of PNNs again. Often one tries to solve the optimization problem (2.6.1) using gradient based methods, see §2.6.2. Therefore, one is interested in computing

the critical points on the neurovariety of the loss function $\ell_{p_{\mathbf{w}}}$ as defined in (2.6.2) for a PNN $p_{\mathbf{w}}$. In particular, the number of such points provides an upper bound for the number of functions the optimization process can converge to when using gradient descent.

Definition 2.6.1. If there is a finite number of critical points of $\ell_{p_{\mathbf{w}}}$ on the regular locus of $\mathcal{V}_{\mathbf{d},r}$ which is constant for a general choice of training data $(\mathbf{x}_i, \mathbf{y}_i)_i$, we call this number the *learning degree* of the network $p_{\mathbf{w}}$ with respect to loss ℓ, and denote it by $\mathrm{Ldeg}_\ell(p_{\mathbf{w}})$.

In the following we are studying the case of the Euclidean loss function. We show that in this case the learning degree is independent of the choice of training data $(\mathbf{x}_i, \mathbf{y}_i)_i$. An upper bound of the learning degree is provided by the generic Euclidean distance (ED) degree. We would like to thank Matthew Trager for pointing out the connection to the generic ED degree.

Definition 2.6.2 ([Breiding et al., 2023, Definition 2.8])**.** Let X be a variety, let $\Lambda = (\lambda_1, \ldots, \lambda_n) \in \mathbb{R}^n_+$ and let $\mathbf{u} \in \mathbb{R}^n$ be a general point. The Λ-weighted Euclidean distance degree of X is the number of (complex) critical points on X of the function

$$\|\mathbf{x} - \mathbf{u}\|^2_\Lambda \coloneqq \sum_{i=1}^{n} \lambda_i (x_i - u_i)^2.$$

For general weights Λ, this number is independent of Λ, and we call it the *generic Euclidean distance (ED) degree* of X, which we denoted as $\mathrm{EDdeg}_{\mathrm{gen}}(X)$.

Let $(p^{(i)}_{\mathbf{w}}(\mathbf{x}))_i \in (\mathrm{Sym}_d(\mathbb{R}^n))^m$ be a tuple of polynomials in n variables $\mathbf{x} = (x_1, \ldots, x_n)$ occurring as the output of a fixed PNN with weight vector $\mathbf{w}$. Suppose we want the PNN to learn the tuple of polynomials $(f_i(\mathbf{x}))_i \in (\mathrm{Sym}_d(\mathbb{R}^n))^m$. To this end, we sample N input vectors $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N \in \mathbb{R}^n$ according to some probability distribution. We evaluate $(f_i(\mathbf{x}))_i$ at these points to obtain $\hat{\mathbf{y}}_j \coloneqq (f_i(\hat{\mathbf{x}}_j))_i \in \mathbb{R}^m$ for $j = 1, \ldots, N$; the pairs $(\hat{\mathbf{x}}_j, \hat{\mathbf{y}}_j)_j$ constitute the training data.

The learning process is defined by minimizing the average Euclidean distance between the $\hat{\mathbf{y}}_j$ and the points obtained from evaluating the network at the samples $\hat{\mathbf{x}}_j$. This process

corresponds to the empirical risk minimization with ℓ_2 loss. Concretely, we have the following problem:

$$\text{minimize } \frac{1}{N}\sum_{j=1}^{N} ||(p_{\mathbf{w}}^{(i)}(\hat{\mathbf{x}}_j))_i - \hat{\mathbf{y}}_j||^2 \text{ over weights } \mathbf{w}. \tag{2.6.3}$$

Note that $(p_{\mathbf{w}}^{(i)}(\hat{\mathbf{x}}_j))_i, \hat{\mathbf{y}}_j \in \mathbb{R}^m$ for all $j = 1, \ldots, N$, and we have

$$\frac{1}{N}\sum_{j=1}^{N} ||(p_{\mathbf{w}}^{(i)}(\hat{\mathbf{x}}_j))_i - \hat{\mathbf{y}}_j||^2 = \frac{1}{N}\sum_{i=1}^{m}\sum_{j=1}^{N} ||p_{\mathbf{w}}^{(i)}(\hat{\mathbf{x}}_j) - (\hat{\mathbf{y}}_j)_i||^2.$$

Our goal is to show that for each $i = 1, 2, \ldots, m$, this is a quadratic form in the coefficients of $p_{\mathbf{w}}^{(i)}$ and f_i over training data.

Write

$$p_{\mathbf{w}}^{(i)}(\mathbf{x}) = \sum_{I \in \binom{[n]}{d}} \rho_I^i \mathbf{x}^I \quad \text{and} \quad f_i(\mathbf{x}) = \sum_{J \in \binom{[n]}{d}} \phi_J^i \mathbf{x}^J.$$

We define a matrix E as follows: E is a block-diagonal matrix with blocks E^i for $i = 1, 2, \ldots, m$, and each block E^i has as row and column labels the multiindices $\alpha, \beta \in \binom{[n]}{d}$. Then the entry $E^i_{\alpha,\beta}$ is defined as

$$E^i_{\alpha,\beta} \coloneqq \frac{1}{N}\sum_{j=1}^{N} \hat{\mathbf{x}}_j^{\alpha+\beta}.$$

Then we can write

$$\frac{1}{N}\sum_{j=1}^{N} ||p_{\mathbf{w}}^{(i)}(\hat{\mathbf{x}}_j) - f_i(\hat{\mathbf{x}}_j)||^2 = \sum_{\alpha,\beta}(\rho_\alpha^i - \phi_\alpha^i)E^i_{\alpha,\beta}(\rho_\beta^i - \phi_\beta^i).$$

Therefore, the loss function is an $E^i_{\alpha,\beta}$-weighted Euclidean distance in the space $(\mathrm{Sym}_d(\mathbb{R}^n))^m$, the ambient space of the neurovariety $\mathcal{V}_{\mathbf{d},r}$. For a general choice of training data $(\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i)_i$, each block E^i of the linear transformation is generic. Note, however, that all blocks E^i, for $i = 1, \ldots, m$, will be equal, hence, the transformation E might be non-generic. We summarize the discussion in the theorem below.

Theorem 2.6.3. *The learning degree* $\mathrm{Ldeg}_{\ell_2}(p_{\mathbf{w}})$ *of a PNN* $p_{\mathbf{w}}$ *with architecture* $(\mathbf{d}, r)$ *with*

respect to the Euclidean loss function exists. It is at most the generic Euclidean distance degree of its neurovariety, i.e.,

$$\mathrm{Ldeg}_{\ell_2}(p_{\mathbf{w}}) \leq \mathrm{EDdeg}_{\mathrm{gen}}(\mathcal{V}_{\mathbf{d},r}). \tag{2.6.4}$$

If $p_{\mathbf{w}}$ has only a single output neuron, one has equality in (2.6.4)*.*

Note that for large number of samples, the quadratic form defined by E is in fact independent of the samples.

Remark 2.6.4. Suppose $\mathbf{x}_1, \ldots, \mathbf{x}_N$ are drawn independently from a fixed distribution $\mathcal{D}$ with known moments μ_k, $k = 1, 2, \ldots$ By the law of large numbers, for any $\epsilon > 0$,

$$\mathbb{P}(|E^i_{\alpha,\beta} - \mu_{\alpha+\beta}| > \epsilon) \to 0 \text{ as } N \to \infty.$$

Then for any $\epsilon > 0$,

$$\mathbb{P}\left(\left|\frac{1}{N}\sum_{j=1}^{N} ||p^{(i)}_{\mathbf{w}}(\hat{\mathbf{x}}_j) - f_i(\hat{\mathbf{x}}_j)||^2 - (\boldsymbol{\rho}^i - \boldsymbol{\phi}^i)^T A(\boldsymbol{\rho}^i - \boldsymbol{\phi}^i)\right| > \epsilon\right) \to 0 \text{ as } N \to \infty.$$

where A is a fixed matrix of dimension $N \times N$ whose entries are determined by the distribution $\mathcal{D}$.

For the architecture $\mathbf{d} = (2, 2, k),\ r = 2$, we compute the bound in (2.6.4) explicitly.

Theorem 2.6.5. *For $k \geq 2$, the generic ED degree of $\mathcal{V}_{(2,2,k),2}$ is $8k^2 - 12k + 3$. Hence, the learning degree of the PNN with architecture $(\mathbf{d} = (2, 2, k), r = 2)$ is at most $8k^2 - 12k + 3$.*

The generic ED degree of a variety can be computed as the sum of its polar degrees [Breiding et al., 2023, Corollary 2.14]. For a smooth variety X, this can be reformulated in terms of degrees of Chern classes of X, see [Breiding et al., 2023, Theorem 4.20]. However,

$V \coloneqq \mathcal{V}_{(2,2,k),2}$ is not smooth: recall that V consists of $k \times 3$ matrices with rank ≤ 2; this determinantal variety has a nonempty singular locus consisting of all matrices with rank ≤ 1. Therefore, we have to make use of the more involved machinery of *Chern–Mather classes*. The reader wishing to learn more about Chern classes in the context of optimization is referred to [Breiding et al., 2023, §4.3]. We give a brief definition of Chern–Mather classes below, for more details the reader is referred to [Yokura, 1986].

Let $X \hookrightarrow \mathbb{P}^N$ be a projective variety of pure dimension d. Let X_{sm} denote the open subvariety of X of nonsingular points and consider the embedding

$$g\colon X_{\mathrm{sm}} \hookrightarrow \mathrm{Gr}(d, T\mathbb{P}^N), \quad x \mapsto T_x X_{\mathrm{sm}}.$$

Then $\hat{X} \coloneqq \overline{\mathrm{im}(g(X_{\mathrm{sm}}))}$ is called the *Nash blow-up* of X. The restriction of the projection $\mathrm{Gr}(d, T\mathbb{P}^N) \to \mathbb{P}^N$ to $\hat{X}$ gives the *Nash blow-up map* $\nu\colon \hat{X} \to X$. Restricting the tautological subbundle of $\mathrm{Gr}(d, T\mathbb{P}^N)$ to $\hat{X}$ gives the *Nash tangent bundle* $\widehat{TX}$ of $\hat{X}$.

Definition 2.6.6. The i^{th} *Chern–Mather class* $c_i^M(X)$ of X is the pushforward

$$c_i^M(X) \coloneqq \nu_* \left(c_i(\widehat{TX} \cap [\hat{X}]) \right).$$

The total Chern–Mather class $c^M(X)$ is the sum of all $c_i^M(X)$.

The following result expresses the generic ED degree of a determinantal variety in terms of degrees of Chern–Mather classes.

Lemma 2.6.7 ([Zhang, 2018, Proposition 5.5])**.** *Let $V_{m,n,r} \hookrightarrow \mathbb{P}^{mn-1}$ be the determinantal variety of $m \times n$ matrices with rank $\leq r$. Then*

$$\mathrm{EDdeg}_{\mathrm{gen}}(V_{m,n,r}) = \sum_{l=0}^{(m+k)(n-k)-1} \sum_{i=0}^{l} (-1)^i \binom{(m+k)(n-k)-i}{(m+k)(n-k)-l} \beta_{(m+k)(n-k)-1-i},$$

where $k = m - r$ and $c^M(V_{m,n,r}) = \sum_{l=0}^{mn-1} \beta_l H^l \in \mathbb{Z}[H]/\langle H^{mn} \rangle = A(\mathbb{P}^{mn-1})$ with $H =$

$c_1(\mathcal{O}_{\mathbb{P}^{mn-1}}(1))$.

The Chern–Mather class of a determinantal variety has a quite involved, yet very explicit description due to Zhang.

Theorem 2.6.8 ([Zhang, 2018, Theorem 4.3]). *With the notation as in Lemma 2.6.7,*

$$c^M(V_{m,n,r}) = \operatorname{trace}(A(m,n,r) \cdot \mathcal{H}(m,n,r) \cdot B(m,n,r)).$$

Here, A, B and $\mathcal{H}$ are the following $(m(n-k)+1) \times (m(n-k)+1)$ matrices (remember $k = m - r$):

$$\begin{aligned} A(m,n,r)_{i,j} &= \int_{\mathrm{Gr}(k,n)} c(\mathcal{T}_{\mathrm{Gr}(k,n)}) c_i(\mathcal{Q}^{\vee m}) c_{j-i}(\mathcal{S}^{\vee m}) \cap [\mathrm{Gr}(k,n)], \\ B(m,n,r)_{i,j} &= \binom{m(n-k)-j}{i-j}, \\ \mathcal{H}(m,n,r)_{i,j} &= H^{mk+j-i}, \end{aligned}$$

where $\mathcal{S}$ and $\mathcal{Q}$ are the universal sub- and quotient bundle of the Grassmannian, respectively, and $i, j = 0, 1, \ldots, m(n-k)$.

Proof of Theorem 2.6.5. We first need to compute the Chern–Mather class $c^M(V) = c^M(V_{k,3,2})$. Note that in this case $k = 1$, so $\mathrm{Gr}(k,n) = \mathrm{Gr}(1,3) \cong \mathbb{P}^2$. Let $A(\mathbb{P}^2) = \mathbb{Z}[h]/\langle h^3 \rangle$ be the Chow ring of $\mathbb{P}^2$ where $h = c_1(\mathcal{O}_{\mathbb{P}^2}(1))$. The universal subbundle $\mathcal{S}$ is just $\mathcal{O}_{\mathbb{P}^2}(-1)$ and we have

$$c(\mathcal{T}_{\mathbb{P}^2}) = 1 + 3h + 3h^2, \quad c(\mathcal{S}^{\vee k}) = 1 + kh + \frac{1}{2}k(k-1)h^2, \quad c(\mathcal{Q}^{\vee k}) = 1 - kh + \frac{1}{2}k(k+1)h^2.$$

Using these, we compute, for example,

$$A(k,3,2)_{1,2} = \int_{\mathbb{P}^2} (1 + 3h + 3h^2)(-kh)kh \cap [\mathbb{P}^2] = -k^2.$$

Similarly, we obtain that $A(k,3,2)$ is the $(2k+1)\times(2k+1)$ matrix

$$A(k,3,2)=\begin{pmatrix} 3 & 3k & \frac{1}{2}k(k-1) & 0 & \dots & 0 \\ 0 & -3k & -k^2 & 0 & \dots & 0 \\ 0 & 0 & \frac{1}{2}k(k+1) & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 \end{pmatrix}.$$

The matrix $B(k,3,2)$ is given by

$$B(k,3,2)=\begin{pmatrix} \binom{2k}{0} & 0 & 0 & \dots & 0 \\ \binom{2k}{1} & \binom{2k-1}{0} & 0 & \dots & 0 \\ \binom{2k}{2} & \binom{2k}{1} & \binom{2k}{0} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \binom{2k}{2k} & \binom{2k-1}{2k-1} & \binom{2k-2}{2k-2} & \dots & \binom{0}{0} \end{pmatrix}$$

and the matrix $\mathcal{H}(k,3,2)$ is

$$\mathcal{H}(k,3,2)=\begin{pmatrix} H^k & H^{k+1} & H^{k+2} & \dots & 0 \\ H^{k-1} & H^k & H^{k+1} & \dots & H^{3k-1} \\ H^{k-1} & H^{k-1} & H^k & \dots & H^{3k-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & H^k \end{pmatrix},$$

where $H = c_1(O_{\mathbb{P}^{3k-1}}(1))$ is the generator of $A(\mathbb{P}^{3k-1})$. Carrying out the multiplication

$B(k,3,2)A(k,3,2)\mathcal{H}(k,3,2)$, one finds that the diagonal has entries

$$\begin{pmatrix} \binom{2k}{0}\left(3H^k+3kH^{k-1}+\frac{1}{2}k(k-1)H^{k-2}\right) \\ \binom{2k}{1}\left(3H^{k+1}+3kH^k+\frac{1}{2}k(k-1)H^{k-1}\right)+\binom{2k-1}{0}\left(-3kH^k-k^2H^{k-1}\right) \\ \binom{2k}{2}\left(3H^{k+2}+3kH^{k+1}+\frac{1}{2}k(k-1)H^k\right)+\binom{2k-1}{1}\left(-3kH^{k+1}-k^2H^k\right)+\binom{2k}{0}\frac{1}{2}k(k+1)H^{k-1} \\ \vdots \\ \binom{2k}{2k}\cdot 0+\binom{2k-1}{2k-1}\left(-3kH^{3k-1}-k^2H^{3k-2}\right)+\binom{2k-2}{2k-2}\frac{1}{2}k(k+1)H^{3k-3} \end{pmatrix}.$$

Computing the trace we find that

$$\begin{aligned} c^M(V_{k,3,2}) = \sum_{j=-2}^{2k-1}\Bigg[3\binom{2k}{j}+3k\left(\binom{2k}{j+1}-\binom{2k-1}{j}\right)+\frac{1}{2}k(k-1)\binom{2k}{j+2} \\ +\frac{1}{2}k(k+1)\binom{2k-2}{j}-k^2\binom{2k-1}{j+1}\Bigg] \end{aligned}$$

and hence, by Lemma 2.6.7,

$$\begin{aligned} \mathrm{EDdeg}_{\mathrm{gen}}(\mathcal{V}_{(2,2,k),2}) = \sum_{l=0}^{2(k+1)-1}\sum_{i=0}^{l}(-1)^i\binom{2(k+1)-i}{2(k+1)-l}\Bigg[3\binom{2k}{i-2}+3k\left(\binom{2k}{i-1}\right. \\ \left.-\binom{2k-1}{i-2}\right)+\frac{1}{2}k(k-1)\binom{2k}{i}+\frac{1}{2}k(k+1)\binom{2k-2}{i-2}-k^2\binom{2k-1}{i-1}\Bigg]. \end{aligned}$$

Rearranging the double sum to

$$\begin{aligned} \sum_{i=0}^{2(k+1)-1}\left(\sum_{l=i}^{2(k+1)-1}\binom{2(k+1)-i}{2(k+1)-l}\right)(-1)^i\Bigg[3\binom{2k}{i-2}+3k\left(\binom{2k}{i-1}\right. \\ \left.-\binom{2k-1}{i-2}\right)+\frac{1}{2}k(k-1)\binom{2k}{i}+\frac{1}{2}k(k+1)\binom{2k-2}{i-2}-k^2\binom{2k-1}{i-1}\Bigg] \end{aligned}$$

and noting that

$$\sum_{l=i}^{2(k+1)-1}\binom{2(k+1)-i}{2(k+1)-l}=2^{2(k+1)-i}-1,$$

one verifies that the above expression indeed equals $8k^2-12k+3$. □

We would like to thank Mark Kong for providing the rearrangement argument of the double sum.

Remark 2.6.9. For the architecture $(\mathbf{d} = (2, 2, 3), r = 2)$, Theorem 2.6.8 yields $\mathrm{Ldeg}_{\ell_2}(p_{\mathbf{w}}) \leq 39$. A numerical computation for a random choice of block E^i in the linear transformation E suggests that the actual learning degree is $\mathrm{Ldeg}_{\ell_2}(p_{\mathbf{w}}) \leq 3$. Moreover, we observe that all critical points of the loss function are actually real. This phenomenon should be further studied in future work.

2.6.2 Gradient-based optimization

In order to solve for (2.6.1), one uses iterative optimization algorithms, including gradient descent and its variants, to find a local minimum of the objective function. In each step, the parameter vector $\boldsymbol{\theta}$ is adjusted in the direction opposite to the gradient of the function at the current point, as the gradient points in the direction of the steepest ascent. The update rule can be expressed as

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \nabla_{\boldsymbol{\theta}} \ell_{F_{\boldsymbol{\theta}^{(t)}}}. \tag{2.6.5}$$

The tuning parameter η is referred to as the *learning rate* of the optimization process. In practice, the value of η is crucial as it determines how big a step is taken on each iteration. If η is too small, the algorithm may take too long to converge; if it is too large, the algorithm may overshoot the minimum or fail to converge.

For a polynomial neural network $p_{\mathbf{w}}$ with weights $\mathbf{w} = (W_1, W_2, \ldots, W_L)$, at time step t, each W_i is updated by

$$W_i^{(t+1)} = W_i^{(t)} - \eta \nabla_{W_i} \ell_{p_{\mathbf{w}^{(t)}}} \tag{2.6.6}$$

where $\nabla_{W_i} \ell_{p_{\mathbf{w}^{(t)}}}$ is computed efficiently by backpropagation. In subsection 2.6.4, we explain the backpropagation algorithm, which can potentially find its applications in algebraic

statistics beyond the neural network settings.

2.6.3 Experiments

We generated synthetic data consisting of $m = 5000$ datasets, each containing $N = 50$ data points. For each dataset, input pairs $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2)^\top$ were sampled uniformly from the range $[-1, 1]$. The corresponding outputs $\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2, \hat{y}_3)^\top$ were generated using quadratic functions with randomly sampled coefficients:

$$\begin{aligned} \hat{y}_1 &= c_{1,1}\hat{x}_1^2 + c_{1,2}\hat{x}_1\hat{x}_2 + c_{1,3}\hat{x}_2^2, \\ \hat{y}_2 &= c_{2,1}\hat{x}_1^2 + c_{2,2}\hat{x}_1\hat{x}_2 + c_{2,3}\hat{x}_2^2, \\ \hat{y}_3 &= c_{3,1}\hat{x}_1^2 + c_{3,2}\hat{x}_1\hat{x}_2 + c_{3,3}\hat{x}_2^2, \end{aligned} \tag{2.6.7}$$

where the coefficients $c_{i,j}$ were sampled from a normal distribution $\mathcal{N}(0, 1)$.

We employ a polynomial neural network $p_{\mathbf{w}}$ with architecture $\mathbf{d} = (2, 2, 3)$ and quadratic activation function ($r = 2$). The parameters $\mathbf{w}$ consist of two weight matrices $\mathbf{w} = (W_1, W_2)$ with $W_1 \in \mathbb{R}^{2\times 2}$ and $W_2 \in \mathbb{R}^{3\times 2}$.

Each network was trained on its corresponding dataset using stochastic gradient descent (SGD) with an initial learning rate of $\eta = 0.1$. The mean squared error (MSE) loss function was minimized:

$$L(W_1, W_2) = \frac{1}{N}\sum_{i=1}^{N} \|p_{\mathbf{w}}(\hat{\mathbf{x}}^{(i)}) - \hat{\mathbf{y}}^{(i)}\|_2^2.$$

A learning rate scheduler reduced the learning rate by half every 1000 epochs. Training proceeded for a maximum of 15000 epochs or until the maximum gradient magnitude fell below a threshold of 1×10^{-4}, indicating convergence. Later, we refer to this hyperparameter as the gradient norm threshold.

After training, we extracted the quadratic coefficients learned by each network to represent the functions it had approximated. The extraction was performed by analytically expressing

the network's output as a quadratic function of the inputs. For each output unit j, the coefficients (a_{1j}, a_{2j}, a_{3j}) were computed as

$$\begin{aligned} a_{1j} &= v_{j1}w_{11}^2 + v_{j2}w_{21}^2, \\ a_{2j} &= 2(v_{j1}w_{11}w_{12} + v_{j2}w_{21}w_{22}), \\ a_{3j} &= v_{j1}w_{12}^2 + v_{j2}w_{22}^2, \end{aligned} \tag{2.6.8}$$

where v_{jk} are entries of W_2 and w_{kl} are entries of W_1.

To identify distinct functions learned by the network, we compared the extracted coefficients using a coefficient comparison method with a specified tolerance ϵ. Two functions given by coefficients $a_{ij}^{(1)}$ and $a_{ij}^{(2)}$ as in (2.6.8) were considered the same if

$$\left| a_{ij}^{(1)} - a_{ij}^{(2)} \right| < \epsilon \quad \forall i, j.$$

We systematically adjusted the tolerance ϵ to explore its impact on the number of distinct functions identified. By varying ϵ, we could control the granularity of function distinction, allowing minor variations in coefficients to be considered functionally equivalent.

In addition, we performed perturbations on the function's coefficients to assess whether each learned function corresponded to a local minimum in the neuromanifold. Small perturbations δ were added to the coefficients,

$$\tilde{c}_{ij} = c_{ij} + \delta_{ij}, \quad \delta_{ij} \in [-\varepsilon, \varepsilon],$$

where ε is a small value (e.g., 1×10^{-4}). The perturbed coefficients also give rise to a perturbed neural network which we denote by $\tilde{p}_{\mathbf{w}}$. The perturbed loss function then becomes

$$L_\delta(W_1, W_2) = \frac{1}{N} \sum_{i=1}^{N} \left\| \tilde{p}_{\mathbf{w}}(\hat{\mathbf{x}}^{(i)}) - \tilde{\mathbf{y}}^{(i)} \right\|_2^2,$$

where $\tilde{\mathbf{y}}^{(i)}$ are defined as in (2.6.7) but using the perturbed coefficients $\tilde{c}_{ij}$. If the value of the original loss function was less than or equal to the losses of all perturbed functions, we considered it to be a local minimum in the neuromanifold.

Through our experimental setup, we were able to

- train 5000 datasets with gradient norm with 15000 epochs and gradient norm threshold 10^{-4};

- identify seventeen distinct functions whose coefficients differ by tolerance threshold $\epsilon = 1/10$ and that were learned with frequency no less than 10 out of $5,000$ datasets;

- check that for sixteen out of seventeen functions, the coefficient matrices of the functions have rank one;

- verify the local minimality of the one learned function whose coefficient matrix has rank two in the neuromanifold by evaluating the loss landscape around the function.

The functions with rank-one coefficient matrices correspond to the singular points of the neuromanifold. Since the coefficients of quadratic functions in (2.6.7) were generated randomly from the normal distribution, we expect the resulting function to be a nonsingular point of the neuromanifold. Therefore we consider only the learned function that is the nonsingular point of the neuromanifold. It is also the most frequently learned function by the network and it is depicted in Figure 2.3. Having one nonsingular local minimum is consistent with Remark 2.6.9 that the learning degree is at most three.

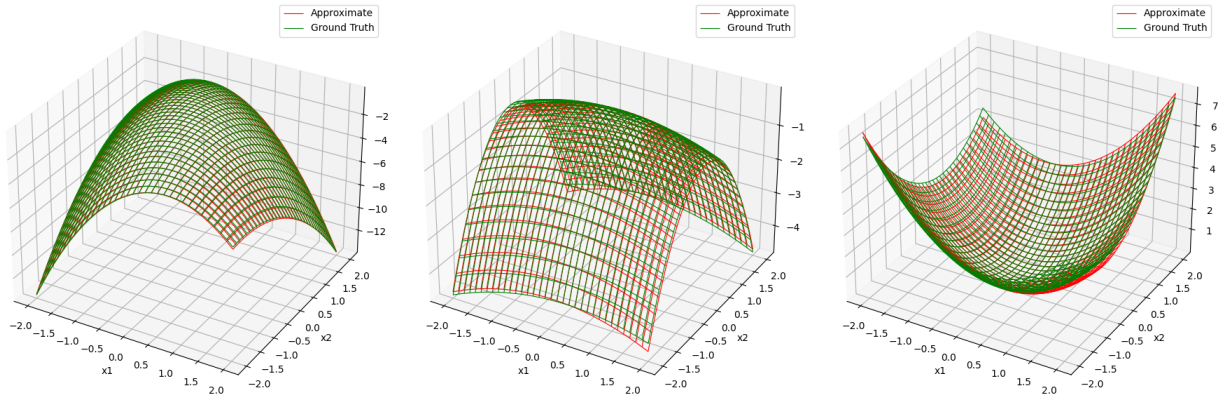


Figure 2.3: Comparison of the most frequent quadratic polynomial learned by a PNN with $\mathbf{d} = (2, 2, 3)$ and $r = 2$ in each coordinate with the ground truth. The green manifold is generated based on the ground truth, while the red manifold is the most frequent function learned by the network.

The experiment implementation is available at [Kubjas et al., 2024].

Remark 2.6.10. The initialization of the network parameters poses challenges due to their tendency to suffer from vanishing and exploding gradients. As the depth of a polynomial neural network scales up, the degree of the polynomial increases and the network becomes sensitive to weight variations, causing gradients to either diminish or escalate exponentially during backpropagation. Poor initialization can lead to prolonged training times, convergence to suboptimal solutions, or even complete failure of the training process. While the general optimal initialization scheme remains an open research problem, in these experiments, we carefully initialize the weight and apply gradient clipping to regularize the training process.

2.6.4 Backpropagation

To compute the gradient of the loss function efficiently one uses the *backpropagation algorithm*. Since Werbos introduced this algorithm in his 1974 PhD thesis [Werbos, 1974], backpropagation lies at the heart of modern successes in deep learning. For the computational algebraic geometer backpropagation is of interest as it provides a way to compute Jacobians of a parameter map $\Psi_{\mathbf{d},r}$ exponentially faster than by naïve differentiation. We give a brief

overview of the backpropagation algorithm following [Nielsen, 2015, Chapter 2] and explain how it can be used to compute Jacobian ranks.

Let ℓ be a loss function (e.g., as in (2.6.3)) satisfying the following assumptions:

1. ℓ is smooth;

2. ℓ can be written as a sum of loss functions $\ell = \sum_{j=1}^{N} \ell_{\hat{\mathbf{x}}_j}$, where each $\ell_{\hat{\mathbf{x}}_j}$ depends only on the sample $\hat{\mathbf{x}}_j$;

3. ℓ depends only on the output of the network $p_{\mathbf{w}}(\mathbf{x})$, not on the state of intermediate layers; the training data $f(\hat{\mathbf{x}}_j)$ are considered as parameters.

Backpropagation provides a way to compute the gradient $\nabla_{\mathbf{w}} \ell(\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_N)$ efficiently. Note that in particular, if $\ell_{\hat{\mathbf{x}}}$ is just the network $F_{\boldsymbol{\theta}}(\hat{\mathbf{x}})$ itself, this gives an algorithm for computing the gradient $\nabla_{\mathbf{w}} F_{\boldsymbol{\theta}}(\hat{\mathbf{x}})$.

Let z_j^l denote the input into the j^{th} neuron of the l^{th} layer. The *error* δ_j^l of this neuron is defined as

$$\delta_j^l := \frac{\partial \ell}{\partial z_j^l}.$$

Theorem 2.6.11 ([Nielsen, 2015]). *Let a_j^l be the output of the j^{th} neuron in the l^{th} layer, and let L denote the last layer of the network. Then the following equations hold:*

$$\delta_j^L = \frac{\partial \ell}{\partial a_j^L} \sigma'_{L,j}(z_j^L), \quad \delta_j^l = \sigma'_{l,j}(z_j^l) \sum_{k=1}^{d_l} w_{l+1,j,k} \delta_k^{l+1}, \quad \frac{\partial \ell}{\partial w_{l,j,k}} = a_k^{l-1} \delta_j^l.$$

Proof. These are simple consequences of applying the chain rule. □

These equations give rise to Algorithm 1 below. The idea is to first compute the outputs of each neuron a_j^l via a "forward" run through the network and then compute the errors δ_j^l using the first two equations of Theorem 2.6.11 running "backwards" through the network. Finally, the gradient can be computed using the third equation from Theorem 2.6.11.

Algorithm 1 Backpropagation

Require: A neural network $F_{\boldsymbol{\theta}}$, a cost function C, and a single input sample $\hat{\mathbf{x}}$ satisfying the assumptions above

Ensure: The gradient $\nabla_{\mathbf{w}}\ell(\hat{\mathbf{x}})$

$a^0 \leftarrow \hat{\mathbf{x}}$

for $l \leftarrow 1$ to L **do** ▷ forward loop

  $z^l \leftarrow W_l\, a^{l-1}$

  $a^l \leftarrow \sigma_l(z^l)$

end for

$\delta^L \leftarrow \nabla_{\mathbf{a}}\ell \;\odot\; \sigma_L'(z^L)$ ▷ $\odot$ = elementwise product

for $l \leftarrow L-1$ downto 1 **do** ▷ backward loop

  $\delta^l \leftarrow \left(W_l^\top\, \delta^{l+1}\right) \;\odot\; \sigma_l'(z^l)$

end for

return $\nabla_{\mathbf{w}}\ell(\hat{\mathbf{x}}) \;=\; \left(a_k^{l-1}\;\delta_j^l\right)_{j,k,l}$

If we choose the loss function ℓ to be the network $F_{\boldsymbol{\theta}}$ itself, backpropagation returns the gradient $\nabla_{\mathbf{w}}F_{\boldsymbol{\theta}}(\hat{\mathbf{x}})$. However, when we are interested in computing the Jacobian of $\Psi_{\mathbf{d},r}$, we need to compute the partial derivatives $\partial_{w_{j,k,l}}c_I^{(i)}$, whereas

$$\frac{\partial F_{\boldsymbol{\theta}}^{(i)}}{\partial w_{j,k,l}}(\hat{\mathbf{x}}) = \sum_I \frac{\partial c_I^{(i)}}{\partial w_{j,k,l}}\hat{\mathbf{x}}^I. \tag{2.6.9}$$

By choosing $N \coloneqq \binom{r^{L-1}+d_0-1}{d_0-1}$ many samples $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_N$ and applying backpropagation to each of them, we obtain a linear system from (2.6.9) in the unknowns $\frac{\partial c_I^{(i)}}{\partial w_{j,k,l}}$. For a generic choice of samples, this system has a unique solution and we obtain the Jacobian $J_{\Psi_{\mathbf{d},r}}$.

This routine allows us to compute Jacobians and in particular the dimension of the neurovariety exponentially faster than via the naïve method of computing all derivatives of the parametrization $\Psi_{\mathbf{d},r}$. It can be easily adapted for computations over finite fields which makes dimension computations even faster. In [Kileel et al., 2019b] this routine has already been implemented, however (as of Oct 9 2024), their implementation at [Kileel et al., 2019a] contains a mistake in the set-up of the linear system described above. We provide a corrected implementation at [Kubjas et al., 2024].

2.6.5 Case study: Linear Neural Networks

In addition to static properties of neural network optimization, we are also interested in the trajectory of practical optimization algorithms. This problem presents a more complex challenge, particularly in its general form. Limited results are known for the special subfamily of polynomial neural networks with activation degree $r = 1$. In this subsection, we survey prior results on the trajectory of the gradient descent algorithms applied to linear neural networks from literature in the algebraic statistics and machine learning communities separately. An interesting open question is to extend the analysis to polynomial neural networks with general activation degrees.

Linear neural networks are a special class of polynomial neural networks with the activation function $\rho_1(\mathbf{x}) = \mathbf{x}$. The associated map from the set of parameters to vectors of homogeneous polynomials of degree one in d_0 many variables is given by

$$\Psi_{\mathbf{d},1} : \mathbb{R}^{d_0 \times d_1} \times \cdots \times \mathbb{R}^{d_{L-1} \times d_L} \to (\mathrm{Sym}_1(\mathbb{R}^{d_0}))^{d_L}, \quad \mathbf{w} \mapsto p_{\mathbf{w}} = \begin{bmatrix} p_{\mathbf{w}}^{(1)} \\ \vdots \\ p_{\mathbf{w}}^{(d_L)} \end{bmatrix}.$$

The neuromanifold for the architecture $\mathbf{d}$ is

$$\mathcal{M}_{\mathbf{d},1} = \{M \in \mathbb{R}^{d_0 \times d_L} \mid \mathrm{rk}(M) \leq \min\{d_0, d_1, \ldots, d_L\}\}.$$

Note that in the linear case the neuromanifold is always equal to the neurovariety.

We denote the map from $\mathbb{R}^{d_0 \times \cdots \times d_L}$ to $\mathbb{R}^{d_0 \times d_L}$ as

$$\mu_{\mathbf{d}}(W_1, W_2, \ldots, W_L) = W_L \cdots W_2 W_1.$$

Proposition 2.6.12 ([Trager et al., 2019, Theorem 4]). *Let* $\tilde{d} = \min(\mathbf{d})$, $\mathbf{w} = (W_1, \ldots, W_L)$ *and* $\overline{W} = \mu_{\mathbf{d}}(\mathbf{w})$.

- *(Filling case) If* $\tilde{d} = \min\{d_0, d_L\}$*, the differential* $d\mu_d(\mathbf{w})$ *has maximal rank equal to* $\dim(\mathcal{M}_{\mathbf{d},1}) = d_0 d_L$ *if and only if for any* $i \in [L-1]$*, either* $\mathrm{rk}(W_j) = d_L$ *for all* $j > i$ *or* $\mathrm{rk}(W_j) = d_0$ *for all* $j < i+1$*.*

- *(Non-filling case) If* $\tilde{d} < \min\{d_0, d_L\}$*, the differential* $d\mu_d(\mathbf{w})$ *has maximal rank equal to* $\dim(\mathcal{M}_{\mathbf{d},1}) = \tilde{d}(d_0 + d_L - \tilde{d})$ *if and only if* $\mathrm{rk}(\overline{W}) = \tilde{d}$*.*

Besides static properties of the loss landscape, in the realm of neural networks, optimization often focuses on devising and refining algorithms to efficiently find the best model parameters that minimize a loss function. Key aspects of optimization include developing and analyzing optimization algorithms like gradient descent and its variants, understanding the impact of different loss functions, and ensuring model generalization through regularization. Additionally, researchers delve into convergence analysis to assess how and when algorithms reach optimal solutions. Since neural networks are, in general, complex models, the loss landscape is often non-convex, resulting in a significant risk of converging to a suboptimal local minimum. Analyzing the convergence trajectory is one of the most challenging problems in machine learning theory. We recall results on the convergence trajectory of deep linear neural networks with respect to ℓ_2 loss in Theorem 2.6.13. This analysis provides crucial insights into the efficiency and effectiveness of how learning algorithms perform on these architectures.

Theorem 2.6.13 ([Arora et al., 2018, Theorem 1], informal)**.** *Consider training deep linear neural networks with* ℓ_2 *loss using the gradient descent algorithm. Assume that at initialization, the weight matrices are well-conditioned. In addition, there exists* $\delta > 0$ *such that at any time step* T*,*

$$\|W_{j+1}^T W_{j+1} - W_j W_j^T\|_F \leq \delta, \tag{2.6.10}$$

for all $j = 1, 2, \ldots, L-1$*. For any* $\epsilon > 0$*, with a sufficiently small learning rate, there exists* T_0 *such that the loss is no greater than* ϵ *for any* $T \geq T_0$*.*

Theorem 2.6.13 shows that, when minimizing the ℓ_2 loss for a deep linear network over a

dataset with zero mean and unit variance, gradient descent converges to the global minimum at a linear rate, given mild assumptions. Similar results have been shown for deep linear neural networks with losses beyond the ℓ_2 loss. In [Bréchet et al., 2023], the authors consider deep linear neural networks trained with the Bures–Wasserstein distance, a loss function commonly used in generative models. They analyze not only the critical points of the loss landscape but also the convergence of both gradient flow and gradient descent for the Bures–Wasserstein loss.

The convergence trajectory of polynomial neural networks with activation degree $r \geq 2$ remains an open problem. Heuristically, having a higher degree of activation reduces the symmetry in the parameter space, which may lead to interesting geometric properties in the trajectory of various optimization algorithms. We aim to establish theoretical guarantees for the convergence of general polynomial neural networks in our future work.

2.7 Conclusion

We have studied the functional space of neural networks with polynomial activation functions from an algebro-geometric perspective. These networks, characterized by their ability to model complex, high-degree polynomial relationships in data, provide a unique lens through which the intricate dynamics of deep learning models can be better understood. Moreover, since a large class of functions can be efficiently approximated by polynomials, delving into the polynomial framework can be important for the development of more efficient and accurate models.

In this chapter, we have focused on the *geometry* of polynomial neural networks. We have characterized neuromanifolds and neurovarieties for some specific architectures, and studied their dimensions. The geometry and dimension of the neuromanifold relates directly to the expressive power of the neuromanifold. Beyond expressivity, we also studied the optimization process of polynomial neural networks, and bound the number of critical points

of a loss function using the *learning degree*. Besides our conjectures from §2.5 concerning the dimension of neurovarieties, there are further questions to be studied, for example: What are the learning degrees of more general architectures? How many of the critical points of the loss functions are real?

Bibliography

[Alexander and Hirschowitz, 1995] Alexander, J. and Hirschowitz, A. (1995). Polynomial interpolation in several variables. *Journal of Algebraic Geometry*, 4(4):201–222.

[Arora et al., 2018] Arora, S., Cohen, N., Golowich, N., and Hu, W. (2018). A convergence analysis of gradient descent for deep linear neural networks. *arXiv preprint arXiv:1810.02281*.

[Bernardi et al., 2018] Bernardi, A., Blekherman, G., and Ottaviani, G. (2018). On real typical ranks. *Bollettino dell'Unione Matematica Italiana*, 11:293–307.

[Boullé et al., 2020] Boullé, N., Nakatsukasa, Y., and Townsend, A. (2020). Rational neural networks. *Advances in neural information processing systems*, 33:14243–14253.

[Brambilla and Ottaviani, 2008] Brambilla, M. C. and Ottaviani, G. (2008). On the Alexander–Hirschowitz theorem. *Journal of Pure and Applied Algebra*, 212(5):1229–1251.

[Bréchet et al., 2023] Bréchet, P., Papagiannouli, K., An, J., and Montúfar, G. (2023). Critical points and convergence analysis of generative deep linear networks trained with bures-wasserstein loss. *arXiv preprint arXiv:2303.03027*.

[Breiding et al., 2023] Breiding, P., Kohn, K., and Sturmfels, B. (2023+). Metric algebraic geometry. to appear in Springer Nature.

[Brodsky and Sturmfels, 2011] Brodsky, S. B. and Sturmfels, B. (2011). Tropical quadrics through three points. *Linear algebra and its applications*, 435(7):1778–1785.

[Bruns and Vetter, 2006] Bruns, W. and Vetter, U. (2006). *Determinantal rings*, volume 1327. Springer.

[Chrysos et al., 2020] Chrysos, G. G., Moschoglou, S., Bouritsas, G., Panagakis, Y., Deng, J., and Zafeiriou, S. (2020). P-nets: Deep polynomial neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7325–7335.

[Comon et al., 2008] Comon, P., Golub, G., Lim, L.-H., and Mourrain, B. (2008). Symmetric tensors and symmetric tensor rank. *SIAM Journal on Matrix Analysis and Applications*, 30(3):1254–1279.

[Comon and Mourrain, 1996] Comon, P. and Mourrain, B. (1996). Decomposition of quantics in sums of powers of linear forms. *Signal Processing*, 53(2-3):93–107.

[Comon and Ottaviani, 2012] Comon, P. and Ottaviani, G. (2012). On the typical rank of real binary forms. *Linear and multilinear algebra*, 60(6):657–667.

[Dey et al., 2016] Dey, S., Naskar, S., Mukhopadhyay, T., Gohs, U., Spickenheuer, A., Bittrich, L., Sriramula, S., Adhikari, S., and Heinrich, G. (2016). Uncertain natural frequency analysis of composite plates including effect of noise–a polynomial neural network approach. *Composite Structures*, 143:130–142.

[Eisenbud, 2013] Eisenbud, D. (2013). *Commutative algebra: with a view toward algebraic geometry*, volume 150. Springer Science & Business Media.

[Finkel et al., 2024] Finkel, B., Rodriguez, J. I., Wu, C., and Yahl, T. (2024). Activation thresholds and expressiveness of polynomial neural networks. *arXiv preprint arXiv:2408.04569*.

[Ghazali et al., 2011] Ghazali, R., Hussain, A. J., and Liatsis, P. (2011). Dynamic ridge polynomial neural network: Forecasting the univariate non-stationary and stationary trading signals. *Expert Systems with Applications*, 38(4):3765–3776.

[Grigsby et al., 2022] Grigsby, J. E., Lindsey, K., Meyerhoff, R., and Wu, C. (2022). Functional dimension of feedforward ReLU neural networks. *arXiv preprint arXiv:2209.04036*.

[Guille-Escuret et al., 2022] Guille-Escuret, C., Ibrahim, A., Goujaud, B., and Mitliagkas, I. (2022). Gradient descent is optimal under lower restricted secant inequality and upper error bound. *Advances in Neural Information Processing Systems*, 35:24893–24904.

[Hardt et al., 2018] Hardt, M., Ma, T., and Recht, B. (2018). Gradient descent learns linear dynamical systems. *Journal of Machine Learning Research*, 19(29):1–44.

[Haykin, 1998] Haykin, S. (1998). *Neural networks: A comprehensive foundation*. Prentice Hall PTR.

[Hornik et al., 1989] Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

[Huang et al., 2003] Huang, L.-L., Shimizu, A., Hagihara, Y., and Kobatake, H. (2003). Face detection from cluttered images using a polynomial neural network. *Neurocomputing*, 51:197–211.

[Karimi et al., 2016] Karimi, H., Nutini, J., and Schmidt, M. (2016). Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, pages 795–811. Springer.

[Khaled and Richtárik, 2020] Khaled, A. and Richtárik, P. (2020). Better theory for sgd in the nonconvex world. *arXiv preprint arXiv:2002.03329*.

[Kileel et al., 2019a] Kileel, J., Trager, M., and Bruna, J. (2019a). Github repository https://github.com/mtrager/polynomial_networks.

[Kileel et al., 2019b] Kileel, J., Trager, M., and Bruna, J. (2019b). On the expressive power of deep polynomial neural networks. *Advances in neural information processing systems*, 32.

[Kubjas et al., 2024] Kubjas, K., Li, J., and Wiesmann, M. (2024). MathRepo page PolynomialNeuralNetworks. `https://mathrepo.mis.mpg.de/PolynomialNeuralNetworks`.

[Landsberg, 2011] Landsberg, J. M. (2011). *Tensors: geometry and applications*, volume 128. American Mathematical Soc.

[Li and Orabona, 2019] Li, X. and Orabona, F. (2019). On the convergence of stochastic gradient descent with adaptive stepsizes. In *The 22nd international conference on artificial intelligence and statistics*, pages 983–992. PMLR.

[Mei et al., 2021] Mei, J., Gao, Y., Dai, B., Szepesvari, C., and Schuurmans, D. (2021). Leveraging non-uniformity in first-order non-convex optimization. In *International Conference on Machine Learning*, pages 7555–7564. PMLR.

[Michałek et al., 2016] Michałek, M., Moon, H., Sturmfels, B., and Ventura, E. (2016). Real rank geometry of ternary forms. *Annali di Matematica Pura ed Applicata*, 196(3):1025–1054.

[Nayak and Misra, 2018] Nayak, S. C. and Misra, B. B. (2018). Estimating stock closing indices using a ga-weighted condensed polynomial neural network. *Financial Innovation*, 4(1):21.

[Nielsen, 2015] Nielsen, M. A. (2015). *Neural Networks and Deep Learning*. Determination Press.

[Oh et al., 2003] Oh, S.-K., Pedrycz, W., and Park, B.-J. (2003). Polynomial neural networks architecture: analysis and design. *Computers & Electrical Engineering*, 29(6):703–725.

[Soudry et al., 2018] Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. (2018). The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57.

[Trager et al., 2019] Trager, M., Kohn, K., and Bruna, J. (2019). Pure and spurious critical points: a geometric study of linear networks. *arXiv preprint arXiv:1910.01671*.

[Werbos, 1974] Werbos, P. (1974). Beyond regression: New tools for prediction and analysis in the behavioral sciences. *PhD thesis, Committee on Applied Mathematics, Harvard University, Cambridge, MA*.

[Yavartanoo et al., 2021] Yavartanoo, M., Hung, S.-H., Neshatavar, R., Zhang, Y., and Lee, K. M. (2021). Polynet: Polynomial neural network for 3d shape recognition with polyshape representation. In *2021 International Conference on 3D Vision*, pages 1014–1023. IEEE.

[Yi et al., 2020] Yi, X., Zhang, S., Yang, T., Chai, T., and Johansson, K. H. (2020). Exponential convergence for distributed optimization under the restricted secant inequality condition. *IFAC-PapersOnLine*, 53(2):2672–2677.

[Yokura, 1986] Yokura, S. (1986). Polar classes and Segre classes on singular projective varieties. *Transactions of the American Mathematical Society*, 298(1):169–191.

[Zhang, 2018] Zhang, X. (2018). Chern classes and characteristic cycles of determinantal varieties. *Journal of Algebra*, 497:55–91.

Chapter 3

Rational Neural Networks

Rational activation functions of the form

$$\rho(z) = \frac{P(z)}{Q(z)}, \qquad P, Q \text{ polynomials},$$

have recently emerged as compact and expressive alternatives to classical nonlinearities. Their ability to capture curvature, saturation, and asymptotic behavior with low-degree components has enabled practical improvements across a range of architectures. Recent works demonstrate that rational approximations can effectively substitute standard activations [Boulle et al., 2020, Delfosse et al., 2024], that end-to-end training of rational forms can improve robustness and accuracy [Trimmel et al., 2022], and that learning univariate transformations along edges—as in Kolmogorov–Arnold networks (KANs) [Liu et al., 2025, Sidharth et al., 2024, Aghaei et al., 2024]—provides an expressive alternative to fixed nonlinearities. Taken together, these developments highlight the promise of rational functions as flexible and increasingly relevant building blocks in neural network design.

At the same time, rational activations exhibit analytic behaviors that do not arise for polynomial, ReLU-type, or fixed smooth nonlinearities. A distinctive feature is the presence

of poles: if the denominator Q has a real zero at r, then locally

$$\rho(z) = \frac{A}{z-r} + h(z), \qquad \rho'(z) = -\frac{A}{(z-r)^2} + h'(z),$$

for some residue $A \neq 0$ and a bounded smooth correction h. Small changes in a pre-activation near r can therefore produce disproportionately large variations in both forward values and gradients. Since pre-activations depend affinely on the parameters, gradient-based updates may inadvertently move some coordinates toward near-pole regions, leading to rapid growth in second-order behavior and, in practice, to brittle optimization dynamics. Such phenomena are intrinsic to rational activations and suggest that the denominator requires a more systematic treatment than is provided by standard norm or spectral constraints on the weights.

Several of the approaches mentioned above implicitly regularize pre-activation behavior—either by fitting rational functions on a prescribed interval where the denominator is well behaved, or by choosing parameterizations that tend to promote positivity and smoothness empirically. However, the denominator itself evolves under training, and even when weight norms are well controlled, nothing prevents the real roots of Q from approaching values actually attained by the network during optimization. This raises natural questions about how to ensure that pre-activations remain uniformly separated from pole regions throughout training, and how such guarantees translate into analytic control over activations, derivatives, and second-order behavior. Providing explicit, training-time certificates for denominator positivity offers one principled way to address these questions and complements the empirical insights suggested by existing rational activation methods.

A separate line of work investigates the geometry of the loss landscape. Prior results show that even fixed analytic activations—including smooth rational functions with no real poles—can give rise to spurious valleys when the network width is insufficient [Venturi et al., 2019]. These results illuminate intrinsic limitations of certain architectures and demonstrate that challenging landscape features may arise independently of singularities.

Rational functions with real poles, and the additional flexibility that comes from learning their coefficients, give rise to geometric effects of a different nature and motivate the analysis developed below. Understanding these effects is increasingly relevant as trainable rational activations become more widely used and as their behavior depends in subtle ways on how the activation varies both in its input and in its own parameters. The developments in this chapter build on and extend these perspectives. At a high level, we:

- introduce smooth denominator parameterizations that maintain a strict positivity margin on a certified pre-activation interval throughout training, providing explicit control over poles and near-pole behavior;

- derive computable bounds on ρ and its derivatives, yielding quantitative control of smoothness, second-order behavior, and stability of the empirical loss; and

- analyze the loss landscape induced by rational activations, isolating a pole-induced barrier mechanism that appears when the activation is fixed and has a real pole, and showing that these barriers vanish generically once the coefficients of P and Q become trainable.

Together, these results complement prior empirical and theoretical work by offering a unified, certificate-based viewpoint on the stability and optimization of rational activations. In Section 3.1 we develop the denominator parameterizations and positivity certificates used throughout the chapter; later sections apply this framework to control derivatives and to study the geometry of the loss landscape.

3.1 Rational Networks and Positivity-Safe Parameterizations

Rational activations provide expressive nonlinearities but introduce a central difficulty that does not arise for standard choices such as ReLU, polynomials, or fixed smooth functions: the behavior of the denominator. During training, pre-activations evolve with the network parameters, and without additional structure there is no guarantee that $Q(z)$ remains uniformly separated from its real roots on the region visited by the network. Controlling this behavior is essential for stability, for obtaining bounds on the derivatives of the activation, and for understanding the resulting optimization landscape.

This section develops the methodological framework used in the remainder of the chapter. We introduce positivity-safe parameterizations of the denominator that enforce a uniform margin $Q(z) \geq \tau > 0$ on a certified pre-activation interval that is tracked during training. These parameterizations are smooth in their coefficients, compatible with gradient-based optimization, and admit explicit envelopes for Q, Q', and Q''.

We begin by formalizing rational networks and the distinction between fixed- and trainable-coefficient settings. We then present two complementary denominator constructions: an interval-local sum-of-squares form that guarantees positivity on a prescribed compact domain, and a global stable-factor form that guarantees positivity on all of $\mathbb{R}$. Finally, we introduce a simple surrogate that discourages near-common factors between numerator and denominator on the certified interval, improving identifiability and conditioning when (P, Q) are trained jointly. Together, these tools provide the structural foundation for the stability and landscape results developed later in the chapter.

3.1.1 Rational networks and notation

A rational activation is a univariate function of the form

$$\rho(x) \;=\; \frac{P(x)}{Q(x)},$$

where $P, Q \in \mathbb{R}[x]$ are real polynomials of degrees d_P and d_Q with coefficients

$$\eta \;=\; (a_0, \ldots, a_{d_P},\, c_0, \ldots, c_{d_Q}).$$

We distinguish two regimes. In a *fixed-coefficient network* (FCN), the coefficient vector η is held constant during training, whereas in a *trainable-coefficient network* (TCN), η is optimized jointly with the weights and biases.

A multilayer rational network applies ρ elementwise at each hidden layer. Given an input vector $\mathbf{x}^{(0)} \in \mathbb{R}^{d_0}$, define for $\ell = 1, \ldots, L$,

$$\mathbf{z}^{(\ell)} = W^{(\ell)}\mathbf{x}^{(\ell-1)} + \mathbf{b}^{(\ell)}, \qquad \mathbf{x}^{(\ell)} = \rho\big(\mathbf{z}^{(\ell)}\big),$$

where ρ acts coordinatewise. The final layer is taken to be affine,

$$\Phi(\mathbf{x}^{(0)}) = W^{(L+1)}\mathbf{x}^{(L)} + \mathbf{b}^{(L+1)}.$$

Each hidden layer thus implements a rational map of its input, and the overall network realizes a multivariate rational function $\mathbf{x}^{(0)} \mapsto \Phi(\mathbf{x}^{(0)})$.

The main analytic difficulty stems from possible poles of Q. If $Q(z_0) = 0$ for some real z_0, then ρ and its derivatives can become unbounded near z_0, and pre-activations that approach such a root can give rise to large gradients and unstable second-order behavior. This motivates denominator parameterizations that provably avoid poles on the pre-activation domain encountered during training.

Fix $R > 0$, and let

$$I \;:=\; [-R, R]$$

denote a compact interval that contains all pre-activations reached during training. Our goal is to construct families Q_η with the following properties:

- **Uniform positivity.** There exists a margin $\tau > 0$ such that

$$Q_\eta(z) \;\geq\; \tau \qquad \text{for all } z \in I,$$

  ensuring the denominator remains strictly positive on I and preventing near-pole behavior.

- **Smooth dependence on parameters.** The map $\eta \mapsto Q_\eta$ is smooth, so that the positivity margin and the shape of Q_η can be tuned by gradient-based methods in TCNs.

- **Explicit envelopes.** We derive closed-form bounds for Q_η, Q'_η, and Q''_η on I; these yield uniform bounds on ρ, ρ', and ρ'', and consequently control layerwise Lipschitz and smoothness constants of the empirical loss.

In the remainder of this section we develop two constructions of Q_η that guarantee positivity on a compact interval (Section 3.1.2) or globally (Section 3.1.3), while preserving the expressivity of the activation unit.

3.1.2 Interval–Sum-of-squares parameterization

To enforce denominator positivity on a prescribed interval, we use a classical representation of univariate polynomials that are nonnegative on $[a, b]$. The Markov–Lukács theorem gives a sum-of-squares decomposition with factors that vanish at the endpoints, which in turn leads to a natural denominator construction with a strict positivity margin on a compact interval.

Theorem 3.1.1 (Markov–Lukács representation)**.** *If $q \in \mathbb{R}[z]$ is nonnegative on $[a, b]$, then there exist $s, t \in \mathbb{R}[z]$ such that*

$$q(z) = s(z)^2 + (z - a)(b - z)\, t(z)^2, \qquad z \in [a, b].$$

If $\deg q = n$, one may take $\deg s \le \lfloor n/2 \rfloor$ and $\deg t \le \lfloor (n-1)/2 \rfloor$. Conversely, every polynomial of this form is nonnegative on $[a, b]$.

Specializing to the symmetric interval $I = [-R, R]$, we obtain the following denominator parameterization.

Definition 3.1.2 (Interval–Sum-of-squares (Interval–SOS) parameterization)**.** For fixed $R > 0$ and $\tau > 0$, define

$$Q_\eta(z) \;=\; \tau \;+\; s_\eta(z)^2 \;+\; (R^2 - z^2)\, t_\eta(z)^2, \qquad |z| \le R,$$

where $s_\eta, t_\eta \in \mathbb{R}[z]$ are smoothly parameterized polynomials.

By construction, $Q_\eta(z) \ge \tau$ for all $z \in I$, and the map $\eta \mapsto Q_\eta$ is polynomial in the parameters. The degree of Q_η is determined by the degrees of s_η and t_η, with

$$\deg Q_\eta = \max\{2d_s,\, 2d_t + 2\},$$

where $d_s := \deg s_\eta$ and $d_t := \deg t_\eta$.

Taken together, these properties make Interval–SOS a convenient interval-local mechanism for enforcing a denominator margin on a compact pre-activation domain. In settings where the interval I is fixed or changes only infrequently, it provides a flexible and analytically transparent choice. When global positivity is desired, or when the certified interval is allowed to expand during training, a complementary construction based on stable factors offers an alternative.

3.1.3 Global positivity via Stable–Factor parameterization

In situations where the range of pre-activations may expand significantly during training, it can be advantageous to enforce positivity of the denominator on all of $\mathbb{R}$. For this purpose we introduce a global denominator parameterization based on products of quadratic stable factors with an explicit margin.

Definition 3.1.3 (Stable–Factor parameterization)**.** Let $\tau > 0$. For an integer $N_Q \geq 1$, define

$$Q_\eta(z) = \tau \prod_{r=1}^{N_Q} \big(1 + \alpha_r (z - \mu_r)^2\big),$$

where $\mu_r \in \mathbb{R}$ are centers and $\alpha_r \geq 0$ are scale parameters.

Since each factor $1+\alpha_r(z-\mu_r)^2$ is nonnegative and equals 1 when $\alpha_r = 0$, this construction guarantees

$$Q_\eta(z) \;\geq\; \tau \qquad \text{for all } z \in \mathbb{R}.$$

If exactly m of the α_r are nonzero, then $\deg Q_\eta = 2m$. The parameterization is smooth in $(\tau, \{\alpha_r\}, \{\mu_r\})$, and partial derivatives with respect to these quantities admit simple closed forms.

Stable–Factor trades some interval-specific flexibility for global control: it enforces positivity everywhere on $\mathbb{R}$ and requires no reparameterization when the certified pre-activation interval is enlarged. ERA [Trimmel et al., 2022] employs a related factorization to encourage positivity; here the margin τ is made explicit and coupled with an interval certificate.

3.1.4 Certifying and maintaining the pre-activation interval

Let $\rho(z) = P(z)/Q_\eta(z)$ be a rational activation with denominator constructed by Interval–SOS or Stable–Factor, and for $R > 0$ define

$$M_0(R) \;:=\; \sup_{|z| \leq R} |\rho(z)|.$$

Under either parameterization restricted to $I = [-R, R]$, we have $Q_\eta(z) \geq \tau > 0$ for all $z \in I$ and thus $M_0(R) < \infty$. We now state a simple sufficient condition that ensures all pre-activations remain in I throughout the network.

Lemma 3.1.4 (Interval invariance)**.** *Consider a feedforward network*

$$z^{(\ell)} = W^{(\ell)} x^{(\ell-1)} + b^{(\ell)}, \qquad x^{(\ell)} = \rho\big(z^{(\ell)}\big) \quad \textit{applied elementwise},$$

for layers $\ell = 1, 2, \ldots$. Let $\|\cdot\|_\infty$ denote the sup-norm on vectors and

$$\|A\|_{\infty\to\infty} := \sup_{x \neq 0} \frac{\|Ax\|_\infty}{\|x\|_\infty} = \max_i \sum_j |A_{ij}|$$

be the induced $\ell_\infty \to \ell_\infty$ operator norm (maximum absolute row-sum). Define

$$S_\ell := \|W^{(\ell)}\|_{\infty\to\infty}, \qquad \beta_\ell := \|b^{(\ell)}\|_\infty,$$

and, for $R > 0$,

$$M_0(R) := \sup_{|z| \leq R} |\rho(z)|.$$

Assume the input satisfies $\|x^{(0)}\|_\infty \leq B_{\text{in}}$, and suppose $R > 0$ is such that

$$S_1 B_{\text{in}} + \beta_1 \leq R, \qquad S_\ell M_0(R) + \beta_\ell \leq R \quad \forall \ell \geq 2.$$

Then, for every layer $\ell \geq 1$, one has $\|z^{(\ell)}\|_\infty \leq R$, and hence every pre-activation entry $z_i^{(\ell)}$ lies in $[-R, R]$.

Proof. We argue by induction on the layer index ℓ.

Base case ($\ell = 1$). Using the definition of the induced norm,

$$\|z^{(1)}\|_\infty = \|W^{(1)} x^{(0)} + b^{(1)}\|_\infty \leq \|W^{(1)} x^{(0)}\|_\infty + \|b^{(1)}\|_\infty \leq \|W^{(1)}\|_{\infty\to\infty} \|x^{(0)}\|_\infty + \beta_1.$$

By the assumptions $\|x^{(0)}\|_\infty \le B_{\text{in}}$ and $S_1 B_{\text{in}} + \beta_1 \le R$, we obtain $\|z^{(1)}\|_\infty \le S_1 B_{\text{in}} + \beta_1 \le R$.

Induction step. Assume for some $\ell \ge 2$ that $\|z^{(\ell-1)}\|_\infty \le R$. Since $x^{(\ell-1)} = \rho(z^{(\ell-1)})$ is applied elementwise and every coordinate of $z^{(\ell-1)}$ lies in $[-R, R]$, the definition of $M_0(R)$ gives

$$\|x^{(\ell-1)}\|_\infty = \max_i \left|\rho(z_i^{(\ell-1)})\right| \le \sup_{|z|\le R} |\rho(z)| = M_0(R).$$

Therefore,

$$\|z^{(\ell)}\|_\infty = \|W^{(\ell)} x^{(\ell-1)} + b^{(\ell)}\|_\infty \le \|W^{(\ell)} x^{(\ell-1)}\|_\infty + \|b^{(\ell)}\|_\infty$$

$$\le S_\ell \|x^{(\ell-1)}\|_\infty + \beta_\ell \le S_\ell M_0(R) + \beta_\ell \le R,$$

where the last inequality is the assumed condition for layer ℓ. Equivalently, each pre-activation entry satisfies $|z_i^{(\ell)}| \le R$, i.e., $z_i^{(\ell)} \in [-R, R]$. □

In practical implementations, the quantities S_ℓ and β_ℓ are controlled via spectral normalization and weight decay. The bound $M_0(R)$ can be computed from the envelopes of ρ on $[-R, R]$, and the radius R is increased if batches approach the boundary. For Interval–SOS, enlarging R requires rebuilding the representation, whereas Stable–Factor requires no reparameterization. In both cases, strict positivity of Q_η (and the corresponding bound on $M_0(R)$) is re-established after each update. Prior work [Trimmel et al., 2022] provides a partial-fractions implementation for fixed-coefficient rational MLPs.

3.1.5 Avoiding near-common factors

If $P, Q \in \mathbb{R}[z]$ share a nonconstant factor, say $P = f\,\tilde{P}$ and $Q = f\,\tilde{Q}$ with $\deg f \ge 1$, then the activation $\rho = P/Q = \tilde{P}/\tilde{Q}$ is unchanged, while gradients with respect to the raw coefficients become ill-conditioned near the common zeros. Coefficients corresponding to f are non-identifiable, and gradients can blow up or vanish. Algebraically, the resultant $\mathrm{Res}(P, Q)$ vanishes if and only if P and Q have a common complex root [Cox et al., 1997, Sturmfels,

2002, Ch. 5]. Computing $\mathrm{Res}(P, Q)$ at every iteration (for instance, via the determinant of the Sylvester matrix) costs $O((d_P+d_Q)^3)$ and is numerically delicate at higher degrees.

We therefore introduce a differentiable, scale-invariant surrogate on the certified interval $I = [-R, R]$ that is inexpensive to evaluate, smooth in the coefficients, and provably detects *real* common roots in I as the evaluation grid is refined.

Evaluation grid and reciprocal-energy surrogate. We evaluate (P, Q) on a finite set of nodes in I using the Chebyshev nodes of the first kind, scaled to $[-R, R]$, with $M \geq 1$ points

$$z_m \;=\; R\cos\Big(\frac{(2m-1)\pi}{2M}\Big), \qquad m = 1, \ldots, M.$$

These nodes (an affine image of the standard Chebyshev points on $[-1, 1]$) have near-minimal Lebesgue constant and hence support stable sup-norm approximation; see, for example, [Trefethen, 2013, Ch. 7]. By refining the grid we simply mean increasing the number of nodes M. The nearest-node distance then decays as $O(1/M)$ (Lemma 3.1.5).

Given the nodes $\{z_m\}_{m=1}^M$, define gridwise peak magnitudes

$$M_P := \max_{1\leq m\leq M} |P(z_m)|, \qquad M_Q := \max_{1\leq m\leq M} |Q(z_m)|$$

and normalized evaluations

$$p_m := \frac{|P(z_m)|}{M_P + \varepsilon}, \qquad q_m := \frac{|Q(z_m)|}{M_Q + \varepsilon}, \qquad \varepsilon > 0.$$

This max-normalization removes trivial scale effects whenever $M_P, M_Q \gg \varepsilon$.

We quantify near-coincidence of zeros via the *reciprocal energy*

$$E_M^\star(P, Q) \;:=\; \frac{1}{M}\sum_{m=1}^{M} \frac{1}{p_m^2 + q_m^2}, \tag{3.1.1}$$

and introduce the regularization term

$$\mathcal{R}_{\mathrm{sur}}(\eta) \;=\; \lambda_{\mathrm{gcd}} \cdot \log\bigl(\varepsilon_0 + E_M^\star(P,Q)\bigr), \tag{3.1.2}$$

with a small $\varepsilon_0 > 0$ to tame outliers. Intuitively, if at some z_m both P and Q are simultaneously small relative to their gridwise peaks, the corresponding summand in $E_M^\star$ is large; as M grows (grid refinement), a genuine real common root in I forces $E_M^\star(P,Q) \to \infty$ (Proposition 3.1.7). For numerical robustness one may also use the mollified variant

$$E_{M,\kappa}(P,Q) \;:=\; \frac{1}{M}\sum_{m=1}^{M} \frac{1}{p_m^2 + q_m^2 + \kappa^2},$$

with fixed or scheduled $\kappa > 0$ (Remark 3.1.8); the analysis below is stated for $E_M^\star$.

We now record the relevant geometric and local analytic facts.

Lemma 3.1.5 (Nearest Chebyshev node)**.** *For any $z^\star \in [-R,R]$ and any $M \geq 1$, there exists a Chebyshev node $z_{m^\star}$ with*

$$|z_{m^\star} - z^\star| \leq \frac{\pi R}{2M}.$$

Proof. Write $z = R\cos\theta$, so the nodes correspond to

$$\theta_m = \frac{(2m-1)\pi}{2M}, \qquad m = 1,\dots,M.$$

For any $\theta^\star \in [0,\pi]$, choose m such that $|\theta_m - \theta^\star| \leq \frac{\pi}{2M}$. Since $|dz/d\theta| = R|\sin\theta| \leq R$, we obtain

$$|z_m - z^\star| \leq R\,|\theta_m - \theta^\star| \leq \frac{\pi R}{2M}.$$

□

Lemma 3.1.6 (Local vanishing orders)**.** *If $P,Q \in \mathbb{R}[z]$ vanish at $z^\star$ with multiplicities*

$r, s \geq 1$, *then there exist constants* $C_P, C_Q > 0$ *and a neighborhood of* $z^\star$ *on which*

$$|P(z)| \;\leq\; C_P\,|z - z^\star|^r, \qquad |Q(z)| \;\leq\; C_Q\,|z - z^\star|^s.$$

Proposition 3.1.7 (Blow-up of the surrogate at real common roots)**.** *Assume* P *and* Q *share a real root* $z^\star \in [-R, R]$ *with multiplicities* $r, s \geq 1$*. Then there exist constants* $C > 0$ *and* M_0 *such that for all* $M \geq M_0$*,*

$$E^\star_M(P, Q) \;\geq\; C\,M^{\,2\min\{r,s\}-1}.$$

In particular, $\mathcal{R}_{\mathrm{sur}}(\eta) \xrightarrow[M\to\infty]{} \infty$*.*

Proof. By Lemma 3.1.5, pick $m^\star$ with $|z_{m^\star} - z^\star| \leq \pi R/(2M)$. By Lemma 3.1.6,

$$|P(z_{m^\star})| \leq C_P\Big(\frac{\pi R}{2M}\Big)^r, \qquad |Q(z_{m^\star})| \leq C_Q\Big(\frac{\pi R}{2M}\Big)^s.$$

Using $p_m = |P(z_m)|/(M_P + \varepsilon)$ and $q_m = |Q(z_m)|/(M_Q + \varepsilon)$, we obtain

$$p^2_{m^\star} + q^2_{m^\star} \;\leq\; \frac{(C'_P)^2}{(M_P+\varepsilon)^2}\,M^{-2r} \;+\; \frac{(C'_Q)^2}{(M_Q+\varepsilon)^2}\,M^{-2s} \;\leq\; \frac{C''}{M^{\,2\min\{r,s\}}}$$

for constants $C'_P, C'_Q, C'' > 0$ independent of M. Hence the $m^\star$ term in $E^\star_M$ is at least $(C'')^{-1}M^{2\min\{r,s\}}$, and averaging over M nodes yields the claimed lower bound. □

Remark 3.1.8. For the mollified quantity $E_{M,\kappa}$ (with denominator $p^2_m + q^2_m + \kappa^2$), the divergence in Proposition 3.1.7 is lost for fixed $\kappa > 0$ (values saturate at $1/\kappa^2$). To retain divergence one may schedule $\kappa = \kappa(M)$ with $\kappa(M) = o(M^{-\min\{r,s\}})$; then the same order $M^{2\min\{r,s\}-1}$ holds. In numerical experiments we employ a small fixed κ to avoid extreme gradients; this keeps the surrogate sharply sensitive to near-common factors without formal divergence.

If $Q(z) \geq \tau > 0$ on I, then $q_m \geq \tau/(M_Q + \varepsilon)$ for all m, and hence

$$E_M^\star(P, Q) \;\leq\; \frac{1}{M} \sum_{m=1}^{M} \frac{1}{q_m^2} \;\leq\; \Big(\frac{M_Q + \varepsilon}{\tau}\Big)^2,$$

a finite constant independent of M, even if P has many real zeros. Thus the regularization term $\mathcal{R}_{\text{sur}}$ selectively penalizes only configurations where P and Q become simultaneously small at some $z \in I$, precisely the near-common-factor regime that harms identifiability of the raw coefficients.

In adaptive-activation architectures, one may either impose a single set of rational coefficients across all layers or allow each layer to learn its own. *Shared* rational coefficients use one global parameter vector η for all hidden layers, enforcing common derivative envelopes (M_0, M_1, M_2) and, through the chain rule, typically yielding tighter global Lipschitz and smoothness bounds. By contrast, *per-layer* coefficients $\eta^{(\ell)}$ offer greater expressive power but can amplify these bounds multiplicatively along depth.

A practical compromise is to adopt a shared baseline with controlled deviations,

$$\eta^{(\ell)} = \eta + \Delta^{(\ell)}, \qquad \Omega_{\text{share}} = \lambda_{\text{share}} \sum_{\ell=1}^{L} \|\Delta^{(\ell)}\|_2^2,$$

which maintains a strong shared prior while permitting layer-specific adjustments. For identifiability, we fix the denominator scale via the positivity margin (the τ-term) and apply a mild penalty to the numerator coefficients (e.g., $\lambda\|a\|_2^2$). Combined with $\mathcal{R}_{\text{sur}}$, these choices eliminate trivial rescalings and discourage near-common factors on the predefined compact interval I.

3.2 Derivative Control and Optimization Smoothness

3.2.1 Single-unit envelopes on a compact interval

We now quantify how positivity of the denominator on a compact interval translates into uniform control of the activation and its derivatives. Throughout this subsection we assume that the denominator satisfies

$$Q(z) \;\geq\; \tau > 0 \qquad \text{for all } z \in I = [-R, R],$$

for some fixed $R, \tau > 0$. In particular, $\rho = P/Q$ is real-analytic on I, and all quantities below are finite. For a function f and an interval J we write

$$\|f\|_{\infty,J} := \sup_{z\in J} |f(z)|.$$

We abbreviate

$$M_P := \|P\|_{\infty,I}, \quad M_Q := \|Q\|_{\infty,I}, \qquad M_{P^{(k)}} := \|P^{(k)}\|_{\infty,I}, \quad M_{Q^{(k)}} := \|Q^{(k)}\|_{\infty,I}.$$

Our first step is to control polynomial derivatives on $[-R, R]$ in terms of their sup-norm on that interval.

Lemma 3.2.1 (Markov inequalities on a compact interval)**.** *Let p be a real polynomial of degree $d \geq 1$. Then*

$$\|p'\|_{\infty,[-R,R]} \;\leq\; \frac{d^2}{R}\,\|p\|_{\infty,[-R,R]}, \qquad \|p''\|_{\infty,[-R,R]} \;\leq\; \frac{d^2(d^2-1)}{3R^2}\,\|p\|_{\infty,[-R,R]}.$$

Moreover, the constants are sharp on $[-1, 1]$.

Proof. Let $q(t) := p(Rt)$ for $t \in [-1, 1]$. The classical Markov inequalities on $[-1, 1]$ state

that

$$\|q'\|_{\infty,[-1,1]} \le d^2 \|q\|_{\infty,[-1,1]}, \qquad \|q''\|_{\infty,[-1,1]} \le \frac{d^2(d^2-1)}{3} \|q\|_{\infty,[-1,1]},$$

see, e.g., [Rivlin, 1990, Thm. 1.5] or [Trefethen, 2013, Ch. 7]. By the chain rule,

$$q'(t) = R\, p'(Rt), \qquad q''(t) = R^2\, p''(Rt).$$

Taking suprema over $t \in [-1,1]$ and noting $\|q\|_{\infty,[-1,1]} = \|p\|_{\infty,[-R,R]}$ yields the stated bounds. □

We now derive uniform envelopes for ρ and its first two derivatives on I in terms of (R,τ), the degrees (d_P, d_Q), and the sup-norms M_P, M_Q.

Lemma 3.2.2 (Activation and derivative envelopes on I)**.** *Let $\rho = P/Q$ with (P,Q) coprime and $\deg P = d_P$, $\deg Q = d_Q$. Then*

$$\|\rho\|_{\infty,I} \le \frac{M_P}{\tau}, \tag{3.2.1}$$

$$\|\rho'\|_{\infty,I} \le \frac{M_{P'}}{\tau} \;+\; \frac{M_P\, M_{Q'}}{\tau^2} \;\le\; \frac{M_P}{\tau^2}\Big(\frac{d_P^2}{R}\,\tau \;+\; \frac{d_Q^2}{R}\,M_Q\Big), \tag{3.2.2}$$

$$\begin{aligned} \|\rho''\|_{\infty,I} &\le \frac{M_{P''}\, M_Q^2 \;+\; M_P\, M_Q\, M_{Q''} \;+\; 2\, M_{P'}\, M_Q\, M_{Q'} \;+\; 2\, M_P\, M_{Q'}^2}{\tau^3} \\ &\le \frac{M_P\, M_Q^2}{\tau^3}\left(\frac{d_P^2(d_P^2-1)}{3R^2} + \frac{d_Q^2(d_Q^2-1)}{3R^2} + 2\,\frac{d_P^2 d_Q^2}{R^2} + 2\,\frac{d_Q^4}{R^2}\right). \end{aligned} \tag{3.2.3}$$

Proof. Since $Q \ge \tau$ on I, we immediately have

$$|\rho(z)| = \frac{|P(z)|}{|Q(z)|} \le \frac{M_P}{\tau} \quad \text{for all } z \in I,$$

which proves (3.2.1).

For the first derivative, use the quotient rule:

$$\rho'(z) = \frac{P'(z)Q(z) - P(z)Q'(z)}{Q(z)^2}.$$

Taking absolute values and using $|Q| \geq \tau$ on I gives

$$|\rho'(z)| \leq \frac{|P'(z)|}{|Q(z)|} + \frac{|P(z)|\,|Q'(z)|}{|Q(z)|^2} \leq \frac{M_{P'}}{\tau} + \frac{M_P\, M_{Q'}}{\tau^2},$$

and taking suprema over I yields the first inequality in (3.2.2). The second inequality in (3.2.2) follows by applying Lemma 3.2.1 with $p = P$ and $p = Q$:

$$M_{P'} \leq \frac{d_P^2}{R} M_P, \qquad M_{Q'} \leq \frac{d_Q^2}{R} M_Q.$$

For the second derivative, differentiate once more, or use the standard expression obtained by differentiating $\rho' = (P'Q - PQ')/Q^2$:

$$\rho''(z) = \frac{P''(z)Q(z)^2 \;-\; P(z)Q(z)Q''(z) \;-\; 2P'(z)Q(z)Q'(z) \;+\; 2P(z)\big(Q'(z)\big)^2}{Q(z)^3}.$$

Taking absolute values and using $|Q| \geq \tau$ on I yields

$$|\rho''(z)| \;\leq\; \frac{M_{P''}\, M_Q^2 \;+\; M_P\, M_Q\, M_{Q''} \;+\; 2\, M_{P'}\, M_Q\, M_{Q'} \;+\; 2\, M_P\, M_{Q'}^2}{\tau^3},$$

and taking suprema over I proves the first inequality in (3.2.3). For the fully explicit bound, apply Lemma 3.2.1 to P and Q:

$$M_{P''} \leq \frac{d_P^2(d_P^2 - 1)}{3R^2} M_P, \quad M_{Q''} \leq \frac{d_Q^2(d_Q^2 - 1)}{3R^2} M_Q, \quad M_{P'} \leq \frac{d_P^2}{R} M_P, \quad M_{Q'} \leq \frac{d_Q^2}{R} M_Q,$$

which, substituted into the previous inequality and grouped by the common factor $M_P M_Q^2/\tau^3$, give the second inequality in (3.2.3). □

Remark 3.2.3 (Scaling and sharpness)**.** The Markov constants in Lemma 3.2.1 are sharp on $[-1, 1]$, with equality attained by suitable scalar multiples of the Chebyshev polynomial T_d. Consequently, the envelopes in Lemma 3.2.2 have the correct order of growth in (d_P, d_Q) and R: up to multiplicative constants depending on the sup-norms M_P, M_Q and the margin τ, one has

$$\|\rho'\|_{\infty,I} = \Theta\Big(\frac{d^2}{R}\Big), \qquad \|\rho''\|_{\infty,I} = \Theta\Big(\frac{d^4}{R^2}\Big)$$

with $d := \max\{d_P, d_Q\}$. Thus shrinking the interval I (smaller R) and increasing the positivity margin τ both improve the derivative bounds, whereas increasing degree worsens them polynomially. This scaling behavior is intrinsic and reflects the extremal role of Chebyshev polynomials in Markov-type inequalities.

Remark 3.2.4 (Coefficients and Chebyshev bases)**.** In the constructions of Section 3.1, the denominator Q_η is built from auxiliary polynomials s_η, t_η that are parameterized in a Chebyshev basis on $[-R, R]$. For instance, after rescaling $z = R\cos\theta$, one may write

$$P(z) = \sum_{k=0}^{d_P} \alpha_k \, T_k\Big(\frac{z}{R}\Big),$$

where T_k is the kth Chebyshev polynomial of the first kind and α_k are trainable coefficients. The extremal property $|T_k(u)| \le 1$ for $u \in [-1, 1]$ implies the simple coefficient bound

$$M_P = \|P\|_{\infty,I} \;\le\; \sum_{k=0}^{d_P} |\alpha_k|.$$

Moreover, the identity $T_k'(u) = k\, U_{k-1}(u)$, where U_{k-1} is the Chebyshev polynomial of the second kind, together with $|U_{k-1}(u)| \le k$ on $[-1, 1]$, yields

$$\|P'\|_{\infty,I} \;=\; \sup_{|z|\le R} \Big|\frac{d}{dz}P(z)\Big| \;=\; \frac{1}{R} \sup_{|u|\le 1} \Big|\sum_{k=1}^{d_P} \alpha_k T_k'(u)\Big| \;\le\; \frac{1}{R}\sum_{k=1}^{d_P} |\alpha_k|\, k^2.$$

Analogous estimates hold for Q and its derivatives. Inserting such bounds into Lemma 3.2.2

provides envelopes for ρ, ρ', ρ'' that are explicit not only in $(R, \tau, d_P, d_Q, M_P, M_Q)$ but directly in terms of the underlying trainable coefficients. We keep the presentation in Lemma 3.2.2 at the level of M_P, M_Q for clarity, but these coefficient-level bounds are useful when connecting regularization on coefficients to control of the derivative envelopes.

3.2.2 Smoothness and Lipschitz constants for one-hidden-layer networks

We now propagate the single-unit envelopes of Section 3.2.1 through a one-hidden-layer network and obtain explicit smoothness and Lipschitz constants for the predictor and for its empirical square loss. Throughout this subsection we assume that the pre-activation interval has been certified as in Section 3.1, so that all hidden pre-activations lie in

$$I = [-R, R], \qquad Q(z) \geq \tau > 0 \quad \forall z \in I,$$

and therefore the derivative envelopes of Lemma 3.2.2 apply.

For convenience, recall

$$M_0 := \sup_{z \in I} |\rho(z)|, \qquad M_1 := \sup_{z \in I} |\rho'(z)|, \qquad M_2 := \sup_{z \in I} |\rho''(z)|,$$

which are explicitly bounded in terms of $(R, \tau, d_P, d_Q, M_P, M_Q)$ by Lemma 3.2.2.

We consider a one-hidden-layer, width-p network

$$f_\Theta(x) \;=\; \mathbf{v}^\top \rho(\mathbf{W}x + \mathbf{b}), \qquad \Theta = (\mathbf{W}, \mathbf{b}, \mathbf{v}),$$

and the empirical square loss

$$L(\Theta) = \frac{1}{2N} \sum_{i=1}^{N} \big(f_\Theta(x_i) - y_i\big)^2,$$

with inputs bounded by $\|x_i\|_2 \le B$ and labels bounded by $|y_i| \le Y_\star$. We equip the parameter space with the Euclidean product norm

$$\|\Theta\|^2 = \|\mathbf{W}\|_F^2 + \|\mathbf{b}\|_2^2 + \|\mathbf{v}\|_2^2.$$

Proposition 3.2.5. *Under the assumptions above, the gradient of the empirical loss L is Lipschitz with respect to the Euclidean parameter norm. In particular, one may take*

$$L_* \le \underbrace{p\, M_0^2 + \|\mathbf{v}\|_2^2 M_1^2\,(B^2+1)}_{\sup_i \|\nabla f_\Theta(x_i)\|^2} + \underbrace{\big(\|\mathbf{v}\|_2 M_0 + Y_\star\big)\Big(\|\mathbf{v}\|_2 M_2\,(B{+}1)^2 + M_1\,(B{+}1)\Big)}_{\sup_i |f_\Theta(x_i)-y_i|\,\|\nabla^2 f_\Theta(x_i)\|}. \quad (3.2.4)$$

In particular, L is L_*-smooth on any parameter set where M_0, M_1, M_2 and $\|\mathbf{v}\|_2$ are finite.

Proof. For any (x, y), write

$$g(\Theta) := \frac{1}{2}\big(f_\Theta(x) - y\big)^2.$$

Differentiating,

$$\nabla g = (f-y)\,\nabla f, \qquad \nabla^2 g = \nabla f\,(\nabla f)^\top + (f-y)\,\nabla^2 f.$$

Hence

$$\|\nabla^2 g(\Theta)\| \;\le\; \|\nabla f(\Theta)\|^2 + |f_\Theta(x) - y|\,\|\nabla^2 f(\Theta)\|. \quad (3.2.5)$$

Any upper bound on the right-hand side is a Lipschitz constant for ∇g at (x, y).

Since the empirical loss is the average

$$L(\Theta) = \frac{1}{N}\sum_{i=1}^N g_i(\Theta), \qquad g_i(\Theta) := \frac{1}{2}\big(f_\Theta(x_i) - y_i\big)^2,$$

and the operator norm is convex, we obtain

$$\|\nabla^2 L(\Theta)\| \le \frac{1}{N}\sum_{i=1}^{N} \|\nabla^2 g_i(\Theta)\| \le \sup_{1\le i\le N} \|\nabla^2 g_i(\Theta)\|.$$

Thus any bound on the right-hand side of (3.2.5), uniform over i, is an admissible Lipschitz constant for ∇L.

We now bound $\|\nabla f\|$ and $\|\nabla^2 f\|$ for fixed x and Θ.

Bounding $\|\nabla f\|$. Let $z = \mathbf{W}x + \mathbf{b} \in \mathbb{R}^p$, and denote

$$\varphi = \rho(z), \qquad \varphi' = \rho'(z), \qquad \varphi'' = \rho''(z),$$

where ρ and its derivatives act elementwise. A direct computation yields

$$\nabla_{\mathbf{v}} f = \varphi, \qquad \nabla_{\mathbf{W}} f = (\varphi' \odot \mathbf{v})\, x^\top, \qquad \nabla_{\mathbf{b}} f = \varphi' \odot \mathbf{v}.$$

Therefore,

$$\begin{aligned}\|\nabla f\|^2 &= \|\nabla_{\mathbf{v}} f\|_2^2 + \|\nabla_{\mathbf{W}} f\|_F^2 + \|\nabla_{\mathbf{b}} f\|_2^2 \\ &\le \|\varphi\|_2^2 + \|\varphi' \odot \mathbf{v}\|_2^2 \|x\|_2^2 + \|\varphi' \odot \mathbf{v}\|_2^2.\end{aligned}$$

On the certified interval I, $|\rho(z_j)| \le M_0$ and $|\rho'(z_j)| \le M_1$ for all coordinates j, so

$$\|\varphi\|_2^2 = \sum_{j=1}^{p} \rho(z_j)^2 \le p\, M_0^2, \qquad \|\varphi' \odot \mathbf{v}\|_2^2 = \sum_{j=1}^{p} (\rho'(z_j) v_j)^2 \le M_1^2\, \|\mathbf{v}\|_2^2.$$

Using $\|x\| \le B$, we conclude

$$\|\nabla f\|^2 \le p\, M_0^2 + \|\mathbf{v}\|_2^2 M_1^2 (\|x\|^2 + 1) \le p\, M_0^2 + \|\mathbf{v}\|_2^2 M_1^2 (B^2 + 1),$$

which gives the first term in (3.2.4).

Bounding $\|\nabla^2 f\|$. To bound the Hessian, consider a unit direction $\Delta\Theta = (\Delta\mathbf{W}, \Delta\mathbf{b}, \Delta\mathbf{v})$ with $\|\Delta\Theta\| = 1$ and estimate the second directional derivative

$$D^2 f(\Theta)[\Delta\Theta, \Delta\Theta] := \frac{d^2}{dt^2} f_{\Theta + t\Delta\Theta}(x)\Big|_{t=0} = \Delta\Theta^\top \nabla^2 f(\Theta)\, \Delta\Theta.$$

Let $u := \Delta\mathbf{W}\, x + \Delta\mathbf{b} \in \mathbb{R}^p$. The chain rule gives

$$\frac{d}{dt} f_{\Theta + t\Delta\Theta}(x)\Big|_{t=0} = (\varphi' \odot \mathbf{v})^\top u + \varphi^\top \Delta\mathbf{v},$$

and differentiating once more yields

$$\frac{d^2}{dt^2} f_{\Theta + t\Delta\Theta}(x)\Big|_{t=0} = (\mathbf{v} \odot \varphi'')^\top (u \odot u) + 2\,\varphi'^\top (\Delta\mathbf{v} \odot u).$$

Hence

$$\left|\Delta\Theta^\top \nabla^2 f(\Theta)\, \Delta\Theta\right| \le \|\mathbf{v}\|_2\, M_2\, \|u\|_2^2 + 2M_1\, \|\Delta\mathbf{v}\|_2\, \|u\|_2. \tag{3.2.6}$$

We bound $\|u\|_2$ in terms of $\|\Delta\Theta\|$:

$$\|u\|_2 = \|\Delta\mathbf{W}\, x + \Delta\mathbf{b}\|_2 \le \|\Delta\mathbf{W}\|_F\, \|x\|_2 + \|\Delta\mathbf{b}\|_2 \le (B+1)\sqrt{\|\Delta\mathbf{W}\|_F^2 + \|\Delta\mathbf{b}\|_2^2} \le B + 1,$$

using $\|\Delta\Theta\|^2 = \|\Delta\mathbf{W}\|_F^2 + \|\Delta\mathbf{b}\|_2^2 + \|\Delta\mathbf{v}\|_2^2 = 1$. Write $t := \sqrt{\|\Delta\mathbf{W}\|_F^2 + \|\Delta\mathbf{b}\|_2^2}$ and $s := \|\Delta\mathbf{v}\|_2$, so $s^2 + t^2 = 1$ and $2st \le s^2 + t^2 = 1$. From (3.2.6), we obtain

$$\left|\Delta\Theta^\top \nabla^2 f(\Theta)\, \Delta\Theta\right| \le \|\mathbf{v}\|_2 M_2 (B+1)^2 t^2 + 2M_1 (B+1) st \le \|\mathbf{v}\|_2 M_2 (B+1)^2 + M_1 (B+1).$$

Taking the supremum over all $\|\Delta\Theta\| = 1$ shows

$$\|\nabla^2 f(\Theta)\| \le \|\mathbf{v}\|_2 M_2 (B+1)^2 + M_1 (B+1).$$

Combine with $|f - y|$. Finally, for each data point,

$$|f_\Theta(x) - y| \le |f_\Theta(x)| + |y| \le \|\mathbf{v}\|_2 M_0 + Y_\star.$$

Substituting the bounds for $\|\nabla f\|^2$ and $\|\nabla^2 f\|$ into (3.2.5) and taking the supremum over the dataset yields (3.2.4). □

Proposition 3.2.6. *On the region*

$$\mathcal{X}_R := \{x \in \mathbb{R}^d : \ \|\mathbf{W}x + \mathbf{b}\|_\infty \le R\},$$

we have

$$\mathrm{Lip}\left(f_\Theta|_{\mathcal{X}_R}\right) \ \le \ \|\mathbf{W}\|_2 \, \|\mathbf{v}\|_2 \, M_1.$$

Proof. By differentiating f_Θ with respect to x,

$$\nabla_x f_\Theta(x) = \mathbf{W}^\top\big(\rho'(\mathbf{W}x + \mathbf{b}) \odot \mathbf{v}\big).$$

Hence

$$\|\nabla_x f_\Theta(x)\|_2 \le \|\mathbf{W}\|_2 \, \|\mathbf{v}\|_2 \, \|\rho'(\mathbf{W}x + \mathbf{b})\|_\infty \le \|\mathbf{W}\|_2 \, \|\mathbf{v}\|_2 \, M_1$$

for all $x \in \mathcal{X}_R$, and the Lipschitz bound follows from the mean value inequality. □

Corollary 3.2.7. *Let L_* be given by* (3.2.4)*. For full-batch gradient descent on $L(\Theta)$ with any constant step size $\eta \in (0, 2/L_*)$, the loss is nonincreasing along the iterates. In particular, choosing $\eta \le c/L_*$ with $c \in (0, 1]$ yields a conservative step size. The constant L_* is explicit in $(R, \tau, d_P, d_Q, M_P, M_Q)$ via Lemma 3.2.2 and in the observed norms $\|\mathbf{W}\|_F, \|\mathbf{b}\|_2, \|\mathbf{v}\|_2$.*

This follows directly from the descent lemma for L_*-smooth functions.

Remark 3.2.8. The bounds in Propositions 3.2.5 and 3.2.6 depend on the activation only through (M_0, M_1, M_2). Under the positivity-safe parameterizations of Section 3.1, the margin condition $Q_\eta(z) \geq \tau > 0$ on $I = [-R, R]$ and the polynomial structure of (P, Q) guarantee that these envelopes remain finite and can be expressed in closed form in terms of $(R, \tau, d_P, d_Q, M_P, M_Q)$. Thus, enforcing denominator positivity not only prevents poles but also yields explicit quantitative control over the smoothness of the empirical loss and the Lipschitz dependence of the network on its inputs.

3.3 Landscape Analysis with Fixed Rational Activations

The previous sections developed positivity-safe parameterizations for rational activations and established uniform derivative control for the resulting networks. In this part of the chapter, we shift focus from analytic regularity to the geometric structure of the loss landscape when the rational activation is held fixed.

When the activation $\rho = P/Q$ is fixed, two mechanisms can give rise to adverse landscape phenomena:

- **Intrinsic-dimension spurious valleys.** If ρ is smooth, bounded, and non-polynomial (for example, a rational function with no real poles), then the associated family of ridge functions spans an infinite-dimensional subspace of L^2. The resulting loss landscape contains arbitrarily bad spurious valleys on suitable data distributions, as shown by Venturi, Balduzzi, and Bruna [Venturi et al., 2019]. These valleys arise from an expressivity mismatch and are independent of analyticity or poles.

- **Pole-induced barriers.** If ρ has real poles, the activation and its derivatives exhibit unbounded behavior near these singularities. Even on compact input domains, attempting to move a neuron across a pole forces its readout weight to shrink and produces

steep gradients and large Gauss–Newton curvature. Any continuous path that shifts a neuron across a pole must incur a fixed, non-negligible increase in population loss.

These two obstructions arise for entirely different reasons. We begin in Section 3.3.1 with the intrinsic-dimension mechanism, focusing on bounded rational activations that satisfy the hypotheses of Venturi, Balduzzi, and Bruna.

3.3.1 Spurious valleys for fixed rational activations

We study the landscape of the population square loss for a one-hidden-layer model when the activation rational function $\rho = P/Q$ is fixed.

Definition 3.3.1 (Sublevel sets and spurious valleys)**.** For $c \in \mathbb{R}$, the sublevel set is $\Omega_L(c) := \{\theta \in \Theta : L(\theta) \leq c\}$. A *spurious valley at level* c is a path-connected component of $\Omega_L(c)$ that does not contain any global minimizer of L.

Definition 3.3.2 (Arbitrarily bad spurious valleys)**.** We say that L has *arbitrarily bad spurious valleys* if for every $M > 0$ there exists a nonempty open set $\Omega \subset \Theta$ such that

$$\min_{\theta\in\Omega} L(\theta) \;\geq\; \inf_{\Theta} L + M, \qquad \sup_{\theta\in\Omega} L(\theta) \;\leq\; \min_{\theta\in\Omega} L(\theta) + \tfrac{M}{2},$$

and for every continuous path $\gamma : [0,1] \to \Theta$ with $\gamma(0) \in \Omega$ and $\gamma(1) \in \arg\min_\Theta L$, $\max_{t\in[0,1]} L(\gamma(t)) \;\geq\; \min_{\theta\in\Omega} L(\theta) + M$.

Setting. We consider a single-output, one-hidden-layer model of width $p \geq 1$,

$$\Phi_\theta(x) = \sum_{j=1}^{p} v_j\, \rho(w_j^\top x + b_j), \qquad \theta = (\{w_j, b_j, v_j\}_{j=1}^{p}),$$

with population squared loss $L(\theta) = \frac{1}{2}\,\mathbb{E}\big[(\Phi_\theta(X) - Y)^2\big]$, where the expectation is with respect to a data distribution (X, Y) on $\mathbb{R}^d \times \mathbb{R}$.

Theorem 3.3.3 (Existence of arbitrarily bad valleys with fixed rational activations)**.** *Let* $\rho : \mathbb{R} \to \mathbb{R}$ *be a* fixed*, continuous, non-polynomial rational function such that* $\rho \geq 0$ *on* $\mathbb{R}$ *and* $\rho \in L^2(\mathbb{R}, e^{-z^2}\, dz)$*. Then for every width* $p \geq 1$ *there exists a data distribution* (X, Y) *for which the one-hidden-layer population loss admits* arbitrarily bad spurious valleys *in the sense of Def. 3.3.2.*

Proof. This is a direct corollary of [Venturi et al., 2019, Thm. 13 and Cor. 15]. Consider the family of ridge functions $\{x \mapsto \rho(w^\top x + b) : (w, b) \in \mathbb{R}^d \times \mathbb{R}\}$, viewed as elements of L^2 of the input distribution. The *lower intrinsic dimension* of ρ is the dimension of the smallest closed linear subspace of L^2 that contains all single-neuron ridge functions; equivalently, it is the number (possibly ∞) of independent directions spanned by these features as (w, b) vary.[1] When this dimension is infinite, any fixed finite width p yields a model whose realizable function class is too thin relative to the geometry of the span, and the results of [Venturi et al., 2019] show that this mismatch produces arbitrarily bad spurious valleys.

Since one can choose a non-polynomial rational ρ satisfying the stated hypotheses (e.g., Example 3.3.4), the claim follows. □

Example 3.3.4 (A concrete fixed rational activation)**.**

$$\rho(z) = \frac{z^2 + 1}{z^2 + 2}$$

is continuous, non-polynomial, nonnegative, and has no real poles; it is bounded, hence $\rho \in L^2(\mathbb{R}, e^{-z^2}\, dz)$. Theorem 3.3.3 therefore applies.

The parameterizations in Sec. 3.1 guarantee $Q(z) > 0$ (globally for Stable–Factor; on $[-R, R]$ for Interval–SOS). To meet the nonnegativity hypothesis in Theorem 3.3.3, one may choose a nonnegative numerator (e.g., $P(z) = s(z)^2 + \lambda$) so $\rho = P/Q \geq 0$. Boundedness and nonpolynomiality are generic, so the theorem applies to such fixed rationals. Conversely, if Q

[1] In the Gaussian setting, this can be characterized via the Hermite expansion of ρ: if ρ has infinitely many nonzero Hermite coefficients, then the lower intrinsic dimension is infinite; see [Venturi et al., 2019, Sec. 4].

admits a real zero, then typically $\rho \notin L^2(\mathbb{R}, e^{-z^2}\,dz)$ and [Venturi et al., 2019] does not apply; we next isolate a near-pole mechanism on compact supports that still induces barriers.

3.3.2 Pole-induced barriers on compact supports

We now study the landscape when the fixed activation $\rho = P/Q$ has a real simple pole. This regime is distinct from the intrinsic-dimension phenomenon of Section 3.3.1. We show that spurious valley already appears on a discrete compact domain with two inputs.

Assume throughout that

$$Q(r) = 0, \qquad Q'(r) \neq 0, \qquad A := \frac{P(r)}{Q'(r)} \neq 0, \tag{3.3.1}$$

so r is a simple real pole with residue A. Consider two inputs $x_-, x_+ \in \mathbb{R}^d$, drawn with probability $1/2$ and labeled

$$Y(x_-) = 0, \qquad Y(x_+) = 1,$$

and a width-p one-hidden-layer model

$$\Phi_\theta(x) = \sum_{j=1}^{p} v_j\, \rho(w_j^\top x + b_j), \qquad \theta = \big(\{w_j, b_j, v_j\}_{j=1}^p\big).$$

The population loss is

$$L(\theta) = \frac{1}{2}\Big((\Phi_\theta(x_-))^2 + (\Phi_\theta(x_+) - 1)^2\Big).$$

We begin by quantifying the local behavior of ρ near the pole r and its effect on a single neuron.

Lemma 3.3.5. *Under* (3.3.1)*, there exist constants* $C, \varepsilon_0 > 0$ *such that for all* $0 < |\varepsilon| \le \varepsilon_0$*,*

$$\rho(r+\varepsilon) = \frac{A}{\varepsilon} + R(\varepsilon), \qquad \rho'(r+\varepsilon) = -\frac{A}{\varepsilon^2} + R_1(\varepsilon), \tag{3.3.2}$$

with $|R(\varepsilon)| \le C$ *and* $|R_1(\varepsilon)| \le C$.

Fix $\delta > 0$ *and a compact interval* $I \subset \{z : |z - r| \ge \delta\}$. *Suppose a neuron satisfies*

$$z_+ = r + \varepsilon \quad \text{at } x_+, \qquad z_- \in I \quad \text{at } x_-,$$

and its contribution at x_+ *obeys*

$$|v\,\rho(z_+)| \le T$$

for some $T > 0$. *Then, for all sufficiently small* $|\varepsilon|$,

$$|v| \le \frac{2T}{|A|}|\varepsilon|, \qquad |v\,\rho(z_-)| \le \frac{2TM_I}{|A|}|\varepsilon|, \qquad M_I := \sup_{z \in I} |\rho(z)| < \infty.$$

Proof. Since ρ has a simple pole at r, there exists a holomorphic function h on a neighborhood of r with

$$\rho(z) = \frac{A}{z - r} + h(z).$$

Differentiating gives $\rho'(z) = -A(z-r)^{-2} + h'(z)$. Choose $\varepsilon_0 > 0$ such that $|h(z)|, |h'(z)| \le C$ whenever $|z - r| \le \varepsilon_0$. Writing

$$R(\varepsilon) := h(r + \varepsilon), \qquad R_1(\varepsilon) := h'(r + \varepsilon),$$

we obtain (3.3.2) with $|R(\varepsilon)|, |R_1(\varepsilon)| \le C$.

The bound $|v\,\rho(z_+)| \le T$ reads

$$|v|\Big|\frac{A}{\varepsilon} + R(\varepsilon)\Big| \le T.$$

For $|\varepsilon| \le \min\{\varepsilon_0, |A|/(2C)\}$, we have

$$\Big|\frac{A}{\varepsilon} + R(\varepsilon)\Big| \ge \frac{|A|}{|\varepsilon|} - C \ge \frac{|A|}{2|\varepsilon|},$$

and hence $|v| \le 2T|\varepsilon|/|A|$. The second inequality follows from $|\rho(z_-)| \le M_I$ and $|v\rho(z_-)| \le |v|M_I$. □

In words, if a neuron stays numerically well behaved at a near-pole input, its readout weight must shrink linearly in its distance to the pole, so its contribution at safe pre-activations $z_- \in I$ becomes negligible. We now isolate a region of parameter space where all neurons remain a fixed distance from the pole and the total readout mass is controlled.

Fix $0 < \delta < R$ and define "safe" intervals

$$I^- = [r - R,\, r - \delta], \qquad I^+ = [r + \delta,\, r + R].$$

Assume P has no zeros in $I^- \cup I^+$, and set

$$m := \inf_{z \in I^- \cup I^+} |\rho(z)| > 0, \qquad M := \sup_{z \in I^- \cup I^+} |\rho(z)| < \infty.$$

Definition 3.3.6 (Feasible set away from the pole)**.** For $\Lambda > 0$, define

$$\mathcal{K} := \Big\{\theta :\ z_{i,j} \in I^- \cup I^+ \text{ for } i \in \{-,+\},\ 1 \le j \le p,\ \sum_{j=1}^{p} |v_j| \le \Lambda\Big\}.$$

Inside $\mathcal{K}$, every neuron output is bounded by M at each input, and

$$|\Phi_\theta(x_\pm)| \le \sum_{j=1}^{p} |v_j|\, |\rho(z_{\pm,j})| \le \Lambda M.$$

We also single out the same-side region where neurons are kept on one side of the pole:

$$\mathcal{K}_{\mathrm{s}} := \Big\{\theta \in \mathcal{K} :\ \forall j,\ (z_{-,j}, z_{+,j}) \subset I^- \text{ or } (z_{-,j}, z_{+,j}) \subset I^+\Big\}.$$

Theorem 3.3.7 (Compact-support barrier)**.** *Assume* $\Lambda < 1/(\sqrt{2}\,M)$ *and set*

$$B_{\mathrm{s}} := \frac{1}{2}\big(1 - \Lambda\sqrt{2}\,M\big)^2 > 0.$$

Then there exists $B_{\mathrm{bar}} \in (0, B_{\mathrm{s}})$ *such that:*

1. *Every* $\theta \in \mathcal{K}_{\mathrm{s}}$ *satisfies* $L(\theta) \geq B_{\mathrm{s}}$.
2. *Let* $\gamma : [0,1] \to \Theta$ *be continuous with* $\gamma(0) \in \mathcal{K}_{\mathrm{s}}$ *and* $\inf_t L(\gamma(t)) < B_{\mathrm{s}}$. *Then there exists* $t_\star \in (0,1)$ *such that* $\gamma(t_\star) \notin \mathcal{K}$ *and* $L(\gamma(t_\star)) \geq B_{\mathrm{bar}}$.

Proof. (1) For any $\theta \in \mathcal{K}$ we have $|\Phi_\theta(x_\pm)| \leq \Lambda M$, hence

$$\|(\Phi_\theta(x_-), \Phi_\theta(x_+)) - (0,1)\|_2 \geq 1 - \Lambda\sqrt{2}\,M,$$

and so $L(\theta) \geq B_{\mathrm{s}}$. In particular, this holds for $\theta \in \mathcal{K}_{\mathrm{s}}$.

(2) Let γ be as in the statement. Define

$$t_0 := \inf\{t \in [0,1] : L(\gamma(t)) < B_{\mathrm{s}}\}.$$

By (1), $t_0 > 0$ and $\gamma(t_0) \notin \mathcal{K}_{\mathrm{s}}$. Next let

$$t_\star := \inf\{t \in [t_0, 1] : \gamma(t) \notin \mathcal{K}\}.$$

Since $\mathcal{K}$ is closed and γ is continuous, we have $\gamma([t_0, t_\star)) \subset \mathcal{K}$ and $\gamma(t_\star) \notin \mathcal{K}$.

At $t_\star$, either (i) $\sum_{j=1}^{p} |v_j| > \Lambda$, or (ii) some pre-activation $z_{i,j}$ leaves $I^- \cup I^+$.

In case (i), by continuity there exists $t \leq t_\star$ with $\sum_j |v_j| = \Lambda$, and at that point the argument in (1) shows $L(\gamma(t)) \geq B_{\mathrm{s}}$. We may then take $B_{\mathrm{bar}} \leq B_{\mathrm{s}}$.

In case (ii), there exists a unit j and input x_i with $|z_{i,j} - r| < \delta$. Assume without loss of generality $i = +$, so $z_{+,j} = r + \varepsilon$. By continuity of $L(\gamma(t))$ and the definition of t_0, we may

restrict attention to t close to $t_\star$ with $L(\gamma(t)) \le B_{\mathrm{s}} + 1$. Thus

$$|R_\pm| := |\Phi_{\gamma(t)}(x_\pm) - Y(x_\pm)| \le \sqrt{2(B_{\mathrm{s}} + 1)} =: C_0.$$

Consequently

$$|\Phi_{\gamma(t)}(x_+)| \le |R_+| + |Y(x_+)| \le C_0 + 1 =: T_0.$$

Subtracting the contributions of all units $k \neq j$ and using $|\rho(z_{+,k})| \le M$ yields

$$|v_j \rho(z_{+,j})| \le T_0 + \Lambda M.$$

Applying Lemma 3.3.5 with $T := T_0 + \Lambda M$ and choosing δ small enough so that $|\varepsilon| \le \varepsilon_0$, we obtain $|v_j| \le C|\varepsilon|$ and $|v_j \rho(z_{-,j})| \le C|\varepsilon|$ for some constant $C > 0$.

Combining these bounds with the contributions of the remaining units gives

$$\|(\Phi_{\gamma(t_\star)}(x_-), \Phi_{\gamma(t_\star)}(x_+)) - (0,1)\|_2 \ge 1 - \Lambda\sqrt{2}\, M - C'|\varepsilon|$$

for some $C' > 0$ independent of ε. Choosing δ (hence $|\varepsilon|$) sufficiently small ensures $C'|\varepsilon| \le \frac{1}{2}(1 - \Lambda\sqrt{2}M)$ whenever $|z_{+,j} - r| < \delta$, and hence

$$L(\gamma(t_\star)) = \frac{1}{2}\|(\Phi_{\gamma(t_\star)}(x_-), \Phi_{\gamma(t_\star)}(x_+)) - (0,1)\|_2^2 \ge \frac{1}{2}\Big(\tfrac{1}{2}(1 - \Lambda\sqrt{2}M)\Big)^2 =: B_{\mathrm{bar}} > 0.$$

This proves (2). □

3.3.3 Trainable rational activations and generic escape from fixed minima

The landscape obstructions identified in Sections 3.3.1 and 3.3.2 arise when the rational activation $\rho = P/Q$ is held fixed. We now show that these fixed-activation pathologies are

nongeneric once the coefficients of P and Q are allowed to vary. Even in the simple two-point setting used above, a strict suboptimal local minimizer of the fixed-activation model becomes a nonstationary point for almost every choice of activation parameters.

Throughout this subsection we impose the following two conditions:

(C1) There exists a fixed activation parameter $\hat{\alpha}$ and a set of network parameters $\hat{\Theta}_{\text{net}}$ such that $\hat{\Theta}_{\text{net}}$ is a *strict local minimizer* of $\Theta_{\text{net}} \mapsto L(\Theta_{\text{net}}, \hat{\alpha})$ and

$$L(\hat{\Theta}_{\text{net}}, \hat{\alpha}) - \inf_{\Theta} L(\Theta) \;\geq\; M > 0.$$

In particular $L(\hat{\Theta}_{\text{net}}, \hat{\alpha}) > 0$.

(C2) At this minimizer, at least one hidden unit separates the two inputs and has nonzero readout weight:

$$u_{-,j_\star} \neq u_{+,j_\star}, \qquad \hat{v}_{j_\star} \neq 0,$$

and $Q_{\hat{\alpha}}(u_{i,j_\star}) \neq 0$ for $i \in \{-,+\}$.

These conditions hold whenever a fixed-activation network gets "stuck" at a suboptimal configuration and at least one unit is active and nondegenerate.

Set $u_{i,j} = \hat{w}_j^\top x_i + \hat{b}_j$ and

$$\widehat{y}_i(\alpha) = \sum_{j=1}^{p} \hat{v}_j \rho(u_{i,j}; \alpha), \qquad R_i(\alpha) = \widehat{y}_i(\alpha) - Y(x_i), \qquad i \in \{-,+\}.$$

Then

$$L(\hat{\Theta}_{\text{net}}, \alpha) = \tfrac{1}{2}\big(R_-(\alpha)^2 + R_+(\alpha)^2\big), \qquad \nabla_\alpha L(\hat{\Theta}_{\text{net}}, \alpha) = R_-(\alpha)C_-(\alpha) + R_+(\alpha)C_+(\alpha),$$

where each activation-feature vector $C_i(\alpha) \in \mathbb{R}^{d_P+d_Q+1}$ is

$$C_i(\alpha) = \sum_{j=1}^{p} \hat{v}_j \, \nabla_\alpha \rho(u_{i,j}; \alpha).$$

Differentiating $\rho(u;\alpha) = P_\alpha(u)/Q_\alpha(u)$ gives the analytic formulas

$$\frac{\partial \rho}{\partial a_k}(u;\alpha) = \frac{u^k}{Q_\alpha(u)}, \qquad \frac{\partial \rho}{\partial c_k}(u;\alpha) = -\frac{P_\alpha(u)u^k}{Q_\alpha(u)^2}, \tag{$*$}$$

so each entry of $C_i(\alpha)$ is a real-analytic function of α on the domain where $Q_\alpha(u_{i,j}) \neq 0$.

Define

$$\mathcal{Z} = \{\alpha : \ C_-(\alpha) \text{ and } C_+(\alpha) \text{ are linearly dependent}\}.$$

Since each $C_i(\alpha)$ is analytic, $\mathcal{Z}$ is the common zero set of finitely many analytic 2×2 minors of the matrix $[C_-(\alpha)\ C_+(\alpha)]$.

Theorem 3.3.8 (Generic escape from fixed strict minima)**.** *Under* (C1)–(C2)*, there exists a neighborhood U of $\hat{\alpha}$ such that for Lebesgue-almost-every $\alpha \in U$,*

$$\nabla_\alpha L(\hat{\Theta}_{\text{net}}, \alpha) \neq 0.$$

Thus a strict suboptimal local minimizer for the fixed activation becomes nonstationary for generic choices of the activation coefficients.

Proof. By (C2), for the distinguished unit $j_\star$ one has $u_{-,j_\star} \neq u_{+,j_\star}$ and $\hat{v}_{j_\star} \neq 0$. From $(*)$, the evaluations

$$\frac{u_{i,j_\star}^k}{Q_\alpha(u_{i,j_\star})}, \qquad \frac{P_\alpha(u_{i,j_\star})u_{i,j_\star}^k}{Q_\alpha(u_{i,j_\star})^2}$$

depend real-analytically on α. For generic α, these families take different values at $i = -$ and $i = +$ because $u_{-,j_\star} \neq u_{+,j_\star}$, so the resulting vectors $C_-(\alpha)$ and $C_+(\alpha)$ are not colinear. Thus at least one 2×2 minor of $[C_-(\alpha)\ C_+(\alpha)]$ is not identically zero as a function of α, and

hence $\mathcal{Z}$ is a proper real-analytic subset of some neighborhood U of $\hat{\alpha}$. In particular, $\mathcal{Z}$ has Lebesgue measure zero in U.

Now write the stationarity condition explicitly:

$$\nabla_\alpha L(\hat{\Theta}_{\text{net}}, \alpha) = 0 \quad \Longleftrightarrow \quad R_-(\alpha)C_-(\alpha) + R_+(\alpha)C_+(\alpha) = 0.$$

If $\alpha \notin \mathcal{Z}$, then $C_-(\alpha)$ and $C_+(\alpha)$ are linearly independent, so the above identity forces

$$R_-(\alpha) = R_+(\alpha) = 0.$$

The set

$$\mathcal{R} := \{\alpha : \ R_-(\alpha) = 0, \ R_+(\alpha) = 0\}$$

is the common zero set of the analytic functions R_- and R_+, and is therefore either all of U or a proper analytic subset. By (C1), $L(\hat{\Theta}_{\text{net}}, \hat{\alpha}) > 0$, so $(R_-(\hat{\alpha}), R_+(\hat{\alpha})) \neq (0, 0)$ and $\mathcal{R} \neq U$. Hence $\mathcal{R}$ is a proper analytic subset of U and thus has Lebesgue measure zero.

We have shown that

$$\{\alpha \in U : \ \nabla_\alpha L(\hat{\Theta}_{\text{net}}, \alpha) = 0\} \ \subset \ \mathcal{Z} \cup \mathcal{R},$$

a union of two proper analytic subsets of U. Therefore this set has Lebesgue measure zero, and the claim follows. □

3.4 Experiments

We evaluate on Split CIFAR-10 and Split CIFAR-100 with T=10 sequential tasks (disjoint class partitions) in the *class-incremental* single-head setting. We report: (i) average accuracy

$$\mathrm{ACC} \;=\; \tfrac{1}{T}\sum_{t=1}^{T} a_{T,t},$$

(ii) forgetting

$$\mathrm{F} \;=\; \tfrac{1}{T-1}\sum_{t=1}^{T-1}\Big(\max_{s\le T} a_{s,t} - a_{T,t}\Big),$$

(iii) the minimum certified denominator margin

$$\min Q \;=\; \min_{b,\ell,j} Q_\eta\big(z_j^{(\ell)}(x_b)\big),$$

and (iv) normalized wall-clock per epoch relative to ReLU.

Here $a_{s,t}$ denotes test accuracy on task t after completing task s, and $z_j^{(\ell)}$ are pre-activations. All methods use identical data augmentation and optimizer; hyperparameters are tuned on the first task only and then reused for all tasks. Unless noted, we report mean $\pm$ standard deviation across identical random seeds.

For Split CIFAR-100, the 100 classes are partitioned into T=10 tasks of 10 classes each (classes $\{0{:}9\}, \{10{:}19\}, \dots$). For Split CIFAR-10, the 10 classes are partitioned into T=10 tasks of 1 class each. "Disjoint class partitions" means each training task contains a *distinct* set of labels with no overlap across tasks; at test time we evaluate under the single shared classifier on the union of all seen classes (single head). Unless otherwise indicated, results use the canonical class order above. To assess order robustness we also evaluate K random class-order permutations with identical budgets and seeds across methods.

We certify positivity of the denominator on $[-R, R]$ (Sec. 3.1). A one-time calibration on task 1 collects pre-activations z from 2 k held-out examples and sets R to the smallest

Table 3.1: **Split CIFAR-100 (10 tasks). ACC**: average accuracy (%); **F**: forgetting (%); $\min Q$: minimum certified denominator margin; **Time**: per-epoch runtime normalized to ReLU (lower is better). *Fixed Rational (no cert)* uses the same (d_P, d_Q) as the certified variant but *no* positivity/margin constraints; coefficients are obtained by least-squares fitting to SiLU on $[-R, R]$ and then *frozen*. *Trainable Rational (no cert)* trains coefficients end-to-end without certification.

Method	ACC ↑	F ↓	$\min Q$ ↑	Time ↓
ReLU	34.2±1.7	6.9±0.5	–	1.00
SiLU	37.1±1.7	6.2±0.8	–	1.01
Fixed Rational (no cert)	33.2±0.9	6.5±0.6	0.000 (*violations*)	1.04
Trainable Rational (no cert)	32.6±1.2	7.1±0.9	0.000 (*spikes*)	2.06
Pos-Safe (Interval–SOS)	38.8±0.8	6.8±0.4	0.653±0.020	4.08
Pos-Safe (Stable–Factor)	**43.6**±1.2	**5.3**±0.8	0.663±0.016	4.05

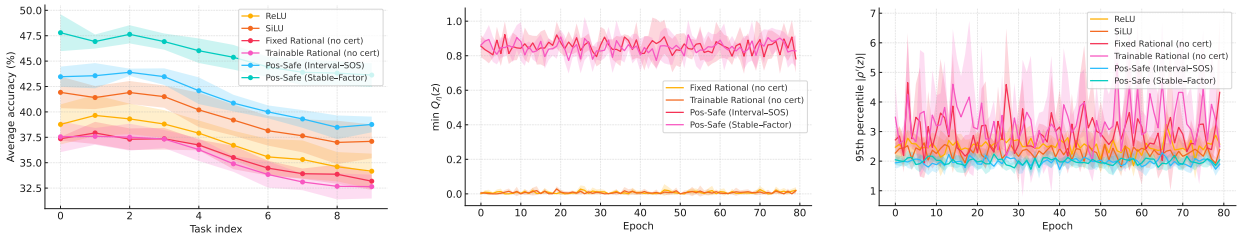


Figure 3.1: **Continual learning results** (mean ± std over seeds). **Left:** continual accuracy across task switches. **Middle:** minimum denominator margin $\min Q_\eta(z)$ per epoch. **Right:** 95$^{\text{th}}$ percentile of $|\rho'(z)|$ per epoch. Positivity-safe rationals maintain higher ACC, strictly positive margins, and bounded derivatives; non-certified rationals show violations/spikes.

value covering the 99.9th percentile of $|z|$ with a 10% safety margin; R is then *frozen* for all tasks/methods. The margin floor ε (denoted τ_0 in ablations) is chosen on task 1 by grid search to satisfy two targets: (i) derivative quantile $\mathrm{q}_{95}(|\rho'(z)|) \le \kappa$ and (ii) minimum certified margin $\min Q \ge \tau_0$, without ACC degradation exceeding a preset Δ; the selected τ_0 is reused across the task sequence.

Per epoch, we log (i) the minimum denominator margin $\min Q_\eta(z)$ over realized pre-activations and (ii) the 95$^{\text{th}}$ percentile of $|\rho'(z)|$. Regularized models are expected to maintain strictly positive margins and bounded derivative quantiles; unregularized rationals may exhibit spikes or violations.

Sharing rational coefficients across layers reduces variance and forgetting. Per-layer

Table 3.2: **Shared vs. per-layer coefficients (Split CIFAR-100).** Gradient volatility via $q_{95}(|\rho'(z)|)$ and the 99th percentile of $\|\nabla_\theta L\|_2$ across mini-batches.

Param. tying	ACC ↑	F ↓	$\min Q$ ↑	$q_{95}(\lvert\rho'\rvert)$ ↓	99p $\|\nabla L\|$ ↓
Shared	$\mathbf{43.6 \pm 1.2}$	$\mathbf{5.3 \pm 0.8}$	0.663 ± 0.016	2.6 ± 0.2	860 ± 45
Per-layer	42.8 ± 1.3	6.2 ± 0.9	0.607 ± 0.028	4.2 ± 0.3	1120 ± 60
Per-layer + Ω_{share}	43.4 ± 1.1	5.6 ± 0.8	0.648 ± 0.021	3.0 ± 0.2	930 ± 50

Table 3.3: **Schedules.** Step schedule with $\eta = c/L_*$ (c=0.1–0.3) versus cosine annealing with warm restarts (SGDR). Robustness to task switches is quantified by the 99th percentile of $\|\nabla_\theta L\|_2$ at task boundaries. **Time** is per-epoch runtime normalized to ReLU.

Schedule	ACC ↑	F ↓	$\min Q$ ↑	99p $\|\nabla L\|$ ↓	Time ↓
Step ($\eta = c/L_*$)	43.2 ± 1.0	$\mathbf{5.5 \pm 0.7}$	$\mathbf{0.660 \pm 0.017}$	$\mathbf{880 \pm 70}$	4.05 ± 0.06
Cosine (tuned SGDR)	$\mathbf{43.5 \pm 1.1}$	5.9 ± 0.8	0.651 ± 0.018	1110 ± 80	4.08 ± 0.05

parameters can match ACC but amplify bound constants and gradient volatility unless regularized (e.g., with Ω_{share}). Evidence: Tab. 3.2 reports $q_{95}(|\rho'(z)|)$ and the 99th percentile of $\|\nabla_\theta L\|_2$ across mini-batches.

A conservative step size $\eta = c/L_*$ with $c \in [0.1, 0.3]$ trains reliably without per-task retuning. A *tuned cosine* schedule refers to cosine annealing with warm restarts with restart period and base LR selected on task 1. Cosine can match or slightly exceed ACC but is less robust at task switches, with larger gradient spikes; certification narrows this gap (Tab. 3.3).

3.5 Conclusion

This chapter used rational activations as a case study in how analytic structure influences the geometry of learning. For fixed rational activations, we observed two distinct sources of difficulty: spurious valleys arising from representational mismatch, and barriers in the loss surface created by poles. Allowing the rational coefficients to be trainable transforms this picture. Once P and Q are free to move—subject to explicit positivity constraints—the landscape becomes less fragile: directions of descent that were previously unavailable reappear, and fixed local minima are no longer stationary for almost all activation parameters.

The positivity-safe parameterizations developed here show that analytic guarantees and expressive learning need not be in tension. By constraining the denominator and quantifying derivative behavior, one can control singularities while still obtaining practical benefits such as improved accuracy and stability in continual learning. This highlights a promising direction in which analytic certification informs model design and optimization.

Much remains to be explored. Extending certification to adaptive domains, analyzing richer input distributions, and integrating activation-parameter geometry with stochastic training remain open challenges. Nevertheless, the perspective developed here suggests that activation design and parameterization are powerful levers for influencing optimization geometry, and mathematical structure can be used intentionally to shape the behavior of modern learning systems.

Bibliography

[Aghaei et al., 2024] Aghaei, A. A., Ebadi, N. M., Taghirad, H. D., Yadegar, J., Aghamohammadi, M., and Nieto, M. N. (2024). rKAN: Rational kolmogorov-arnold networks.

[Boulle et al., 2020] Boulle, N., Nakatsukasa, Y., and Townsend, A. (2020). Rational neural networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 14243–14253. Curran Associates, Inc.

[Cox et al., 1997] Cox, D., Little, J., and O'Shea, D. (1997). *Ideals, Varieties, and Algorithms*. Springer.

[Delfosse et al., 2024] Delfosse, Q., Schramowski, P., Mundt, M., Molina, A., and Kersting, K. (2024). Adaptive rational activations to boost deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*.

[Liu et al., 2025] Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T. Y., and Tegmark, M. (2025). KAN: Kolmogorov–arnold networks. In *International Conference on Learning Representations (ICLR)*.

[Rivlin, 1990] Rivlin, T. J. (1990). *The Chebyshev Polynomials*. Pure and Applied Mathematics. Wiley, New York, 2nd edition.

[Sidharth et al., 2024] Sidharth, S. S., Gokul, R., and Ponnambalam, N. (2024). Chebyshev polynomial-based kolmogorov-arnold networks: An efficient architecture for nonlinear function approximation.

[Sturmfels, 2002] Sturmfels, B. (2002). *Solving Systems of Polynomial Equations*. American Mathematical Society.

[Trefethen, 2013] Trefethen, L. N. (2013). *Approximation Theory and Approximation Practice*. Other Titles in Applied Mathematics. SIAM, Philadelphia, PA.

[Trimmel et al., 2022] Trimmel, M., Zanfir, M., Hartley, R., and Sminchisescu, C. (2022). ERA: Enhanced rational activations. In Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Computer Vision – ECCV 2022*, pages 722–738, Cham. Springer Nature Switzerland.

[Venturi et al., 2019] Venturi, L., Bandeira, A. S., and Bruna, J. (2019). Spurious valleys in one-hidden-layer neural network optimization landscapes. *Journal of Machine Learning Research*, 20(133):1–34.

Chapter 4

Estimating Means of Bounded Random Variables by Betting

This is chapter, we follow up on Waudby-Smith and Ramdas' work [Waudby-Smith and Ramdas, 2020b] in generating confidence intervals and time-uniform confidence sequences for mean estimation with bounded observations. Their methodology utilizes composite nonnegative martingales and establishes a connection to game-theoretic probability. Our comments will focus on numerical comparisons with alternative methods. The corresponding code is available at `https://github.com/Likelyt/Estimate-mean-with-betting`.

4.1 Methods

Consider a sequence of random variables $(X_t)_{t=1}^{\infty}$, drawn from a distribution $P \in \mathcal{P}^{\mu}$, where $\mathcal{P}^{\mu}$ represents the set of all distributions on $[0,1]^{\infty}$. We assume $\mathbb{E}_P[X_t|X_1,\ldots,X_{t-1}] = \mu$ for some unknown $\mu \in [0,1]$. The objective is to construct a time-uniform confidence sequence $(C_t)_{t=1}^{\infty}$ that satisfies the condition

$$\sup_{P \in \mathcal{P}^{\mu}} P(\exists t \geq 1 : \mu \notin C_t) \leq \alpha. \tag{4.1.1}$$

The confidence sequence guarantees that, for any fixed n, C_n is a $(1-\alpha)$-confidence interval for μ. We review three methods for generating such confidence sequences.

(a) The *predictable plug-in empirical Bernstein* (Pr-EB) method, discussed in Section 3 of [Waudby-Smith and Ramdas, 2020b], combines the Robbins' method of mixture [Robbins, 1970] with exponential supermartingales.

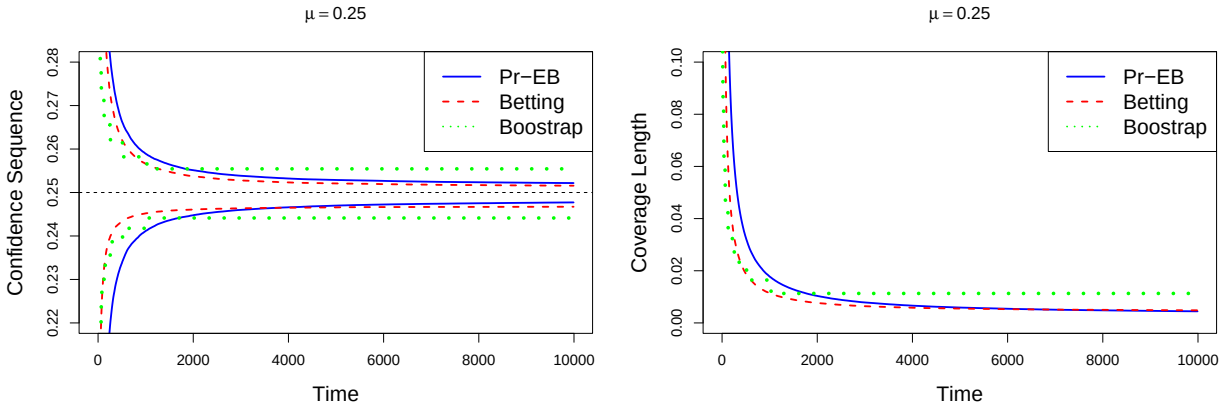
(b) The *hedged capital process* (Betting) method, introduced in Section 4 of [Waudby-Smith and Ramdas, 2020b], is a *novel* approach that enjoys the interpretation of wealth accumulation in a game and has connections to the game-theoretic probability [Shafer and Vovk, 2005].

(c) The *Bootstrap* resampling method is implemented using the R package BOOT [Canty and Ripley, 2022]. With $B = 200$ bootstrap replicates and $L = 10$ batches, we calculate a separate confidence interval for data sequence $2^l \leq t < 2^{l+1}$ within each batch $l = 1, \ldots, L$. The confidence intervals are constructed using the $\frac{\alpha}{2L}$- and $(1 - \frac{\alpha}{2L})$-quantiles of the bootstrap means.

4.2 Numerical Studies

Synthetic Data Example We sequentially generate data from Beta$(10, 30)$ distribution for $1 \leq t \leq 10^4$, and construct a 95% confidence sequence C_t in Eq. (4.1.1) using methods (a)-(c) in Section 4.1.

Figure 4.1 shows that the Betting method outperforms the Bootstrap method with a higher lower bound in the confidence sequence and consistently tighter intervals for $t \geq 1500$. This aligns with the intuition that the Bootstrap method results in wider intervals due to dividing the confidence budget among data subsets when dealing with a large number of sequential data points. Moreover, the Betting method outperforms the Pr-EB method with consistently tighter confidence sequences for $t \leq 3000$. For $t \geq 3000$, both methods yield



(a) Confidence sequence

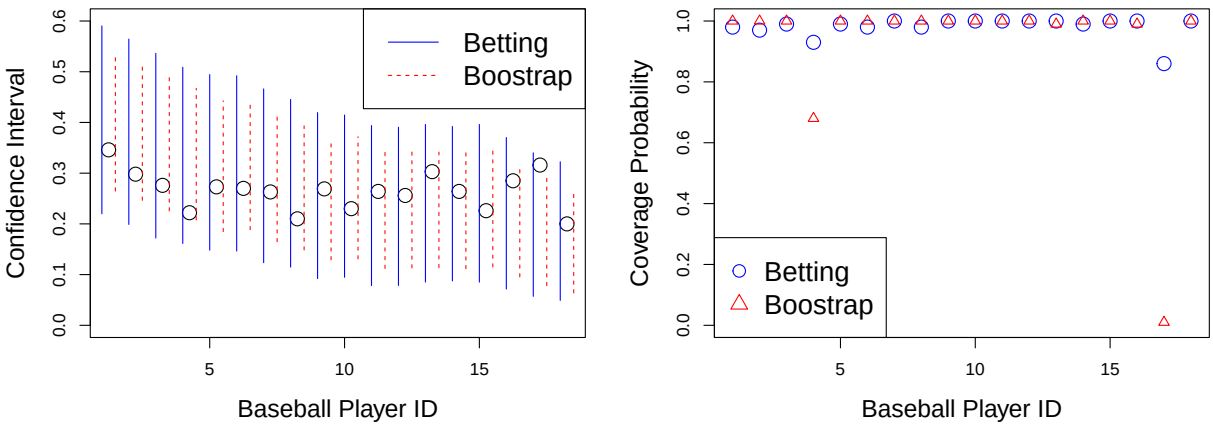
(b) Coverage length

Figure 4.1: Comparisons based on the synthetic data of Beta distribution.

comparable coverage lengths. The Betting method's sequence shows a slight shift towards smaller values, reflecting the left-skewed ground truth distribution, while the Pr-EB method produces a symmetric interval around the estimated mean.

Real Data Example We analyze a batting dataset of 18 Major League players from the 1970 season, available in [Efron and Hastie, 2021] or the R package EfronMorris. The goal is to construct a 95% confidence interval for the true batting level of each player based on their first 45 at-bats. The following results are obtained from 100 data replications.

Figure 4.2(a) shows the average confidence intervals, where the black circle represents the true batting level of each player. Note that while the Bootstrap method fails to cover the true batting level of player 17, the Betting method successfully includes it. Figure 4.2(b) presents the coverage probability of both methods. The Betting method achieves higher coverage probability for the true batting levels of players 4 and 17 compared to the Bootstrap method. These results indicate that the Betting method outperforms the Bootstrap method in constructing accurate confidence intervals for this baseball data.



(a) Confidence interval

(b) Coverage Probability

Figure 4.2: Comparisons based on the baseball batting data from [Efron and Hastie, 2021].

4.3 Extensions

The work in [Waudby-Smith and Ramdas, 2020b] may inspire several directions for future research. One such direction is the generalization of the Betting method to handle multi-dimensional observations. Currently, the Betting method relies on the assumption that the underlying capital process is a martingale. However, when estimating the mean of multidimensional data, applying Ville's inequality [Waudby-Smith and Ramdas, 2020a] for confident rejection of the null hypothesis becomes more challenging. Defining the hedged capital process in a vector space introduces complexities when simultaneously estimating multiple attributes. Another direction is the extension of the Betting method to online decision-making scenarios, such as dynamic treatment, online recommendation, and dynamic pricing. These tasks frequently involve constructing confidence sequences to determine optimal actions. The application of the Betting method in these contexts seems to be interesting.

Bibliography

[Canty and Ripley, 2022] Canty, A. and Ripley, B. D. (2022). *boot: Bootstrap R (S-Plus) Functions*. R package version 1.3-28.1.

[Efron and Hastie, 2021] Efron, B. and Hastie, T. (2021). *Computer Age Statistical Inference: Algorithms, Evidence, and Data Science*, volume 6. Cambridge University Press.

[Robbins, 1970] Robbins, H. (1970). Statistical methods related to the law of the iterated logarithm. *The Annals of Mathematical Statistics*, 41(5):1397–1409.

[Shafer and Vovk, 2005] Shafer, G. and Vovk, V. (2005). *Probability and Finance: It's Only a Game!*, volume 491. John Wiley & Sons.

[Waudby-Smith and Ramdas, 2020a] Waudby-Smith, I. and Ramdas, A. (2020a). Confidence sequences for sampling without replacement. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20204–20214. Curran Associates, Inc.

[Waudby-Smith and Ramdas, 2020b] Waudby-Smith, I. and Ramdas, A. (2020b). Estimating means of bounded random variables by betting. *arXiv preprint arXiv:2010.09686*.

Chapter 5

Geometric algorithms for predicting resilience and recovering damage in neural networks

Biological neural networks have evolved to maintain performance despite significant circuit damage. To survive damage, biological network architectures have both intrinsic resilience to component loss and also activate recovery programs that adjust network weights through plasticity to stabilize performance. Despite the importance of resilience in technology applications, the resilience of artificial neural networks is poorly understood, and autonomous recovery algorithms have yet to be developed. In this paper, we establish a mathematical framework to analyze the resilience of artificial neural networks through the lens of differential geometry. Our geometric language provides natural algorithms that identify local vulnerabilities in trained networks as well as recovery algorithms that dynamically adjust networks to compensate for damage. We reveal striking weight perturbation vulnerabilities in common image analysis architectures, including MLP's and CNN's trained on MNIST and CIFAR-10 respectively. We also uncover high-performance recovery paths that enable the same networks to dynamically re-adjust their parameters to compensate for damage. Broadly,

our work provides procedures that endow artificial systems with resilience and rapid-recovery routines to enable their deployment for critical applications.

5.1 Introduction

Brains are remarkable machines whose computational capabilities have inspired many breakthroughs in machine learning [Iliadis et al., 2020, Fukushima, 1980, Wołczyk et al., 2019, Zador, 2019]. However, the resilience of the brain, its ability to maintain computational capabilities in harsh conditions and following circuit damage, remains poorly developed in current artificial intelligence paradigms [Hendrycks and Dietterich, 2019] . Biological neural networks are known to implement redundancy and other architectural features that allow circuits to maintain performance following loss of neurons or lesion to sub-circuits [Gonzalez et al., 2019, Castor and El Massioui, 2018, Marder and Goaillard, 2006, Srinivasan and Stevens, 2011, Richter et al., 2019]. In addition to architectural resilience, biological neural networks execute recovery programs that allow circuits to repair themselves through the activation of network plasticity following damage [Arlotta and Berninger, 2014, Gates et al., 2000, Murphy and Corbett, 2009]. For example, recovery algorithms reestablish olfactory and visual behaviors in mammals following sensory specific cortical circuit lesions [Keck et al., 2013, Hong et al., 2018]. Through resilience and recovery mechanisms, biological neural networks can maintain steady performance in the face of dynamic challenges like changing external environments, cell damage, partial circuit loss as well as catastrophic injuries like the loss of large sections of the cortex. [Lewin, 1980, Scoville and Milner, 1957, Hammerschmidt et al., 2015, Gilbert and Li, 2012].

Like brains, artificial neural networks must increasingly execute critical applications that require robustness to both hardware component damage and memory errors that could corrupt network weights. Recent studies have highlighted the importance of network robustness to soft errors that can lead to weight corruption and network failure [Li et al., 2017] in

applications including (i) decision-making in the healthcare industry, (ii) image and sensor analysis in self-driving cars and (iii) robotic control systems. Errors in dynamic access memory can occur due to malicious attacks (the RowHammer), but a particular focus has been on errors induced by high energy particles [Arechiga and Michaels, 2018] that occur at surprising rates [Schroeder et al., 2009]. Further, the rising implementation of neural networks on physical hardware (like neuromorphic, edge devices) [Monroe, 2014, Mach and Becvar, 2017], where networks can be disconnected from the internet and are under control of an end user, necessitates the need for damage-resilient and dynamically recovering artificial neural networks.

The resilience of living neural networks motivates theoretical and practical efforts to understand the resilience of artificial neural networks and to design new algorithms that reverse engineer resilience and recovery into artificial systems [Firesmith, 2019]. Recent studies [Morcos et al., 2018, Cheney et al., 2017] have demonstrated empirically that MLP and CNN architectures can be surprisingly robust to large scale node deletion. However, there is currently little understanding of the empirically observed resilience or what ultimately causes networks to fail. Mathematical frameworks will be important for understanding the resilience neural networks and for developing recovery procedures that can maintain network performance during damage.

We propose a mathematical framework grounded in differential geometry for studying the resilience and the recovery of artificial neural nets. We formalize damage/response behavior as dynamic movement on a curved pseudo-Riemannian manifold. Our geometric language provides new procedures for identifying network vulnerabilities by predicting local perturbations that adversely impact the functional performance of the network. Further, we demonstrate that geodesics, minimum length paths, on the weight manifold provide high performance recovery paths that the network can traverse to maintain performance while damaged. Our algorithms allow networks to maintain high-performance during rounds of damage and repair through computationally efficient weight-update algorithms that do not

require conventional retraining. Broadly, our work provides procedures that will help endow artificial systems with resilience and autonomous recovery policies to emulate the properties of biological neural networks.

5.2 Analyzing network resilience with differential geometry

We develop a geometric framework for understanding how artificial neural networks respond to damage using differential geometry to analyze changes in functional performance given changes in network weights. Two recent papers have highlighted intrinsic robustness properties of layered neural networks [Morcos et al., 2018, Cheney et al., 2017]. We provide a geometric approach for understanding robustness as arising from underlying geometric properties of the weight manifold that are quantified by the metric tensor. The geometric approach allows us to identify vulnerabilities in common neural network architectures as well as define new strategies for repairing damaged networks.

We represent a feed-forward neural network as a smooth, $\mathbb{C}^\infty$function $f(\mathbf{x}, \mathbf{w})$, that maps an input vector, $\mathbf{x} \in \mathbb{R}^{\mathrm{k}}$, to an output vector, $f(\mathbf{x}, \mathbf{w}) = \mathbf{y} \in \mathbb{R}^{\mathrm{m}}$. The function, $f(\mathbf{x}, \mathbf{w})$, is parameterized by a vector of weights, $\mathbf{w} \in \mathbb{R}^{\mathrm{n}}$, that are typically set in training to solve a specific task. We refer to $W = \mathbb{R}^n$ as the *weight space* (W) of the network, and we refer to $\mathcal{F} = \mathbb{R}^m$ as the *functional manifold* [Mache et al., 2006]. In addition to f, we will sometimes be interested in considering a loss function, $L : \mathbb{R}^{\mathrm{m}} \times \mathbb{R} \to \mathbb{R}$, that provides a scalar measure of network performance for a given task (Figure 1).

We ask how the performance of a trained neural network, $\mathbf{w_t}$, will change when subjected to weight perturbation, shifting $\mathbf{w_{trained}} \to \mathbf{w_{damaged}}$. We use differential geometry to develop a mathematical theory, rooted in a functional notion of distance, to analyze how arbitrary weight perturbations $\mathbf{w_t} \to \mathbf{w_d}$ impact functional performance of a network. Specifically, we

construct a local distance metric, $\mathbf{g}$, that can be applied at any point in W to measure the functional impact of an arbitrary network perturbation.

To construct a metric mathematically, we fix the input, $\mathbf{x}$, into a network and ask how the output of the network, $f(\mathbf{w}, \mathbf{x})$, moves on the functional manifold, $\mathcal{F}$, given an infinitesimal weight perturbation, $\mathbf{du}$, in W where $\mathbf{w_d} = \mathbf{w_t} + \mathbf{du}$. For an infinitesimal perturbation $\mathbf{du}$,

$$f(\mathbf{x}, \mathbf{w_t} + \mathbf{du}) \approx f(\mathbf{x}, \mathbf{w_t}) + \mathbf{J_{w_t}}\ \mathbf{du}, \tag{5.2.1}$$

where $\mathbf{J_{w_t}}$ is the Jacobian of $f(\mathbf{x}, \mathbf{w})$ for a fixed $\mathbf{x}$, $J_{i,j} = \frac{\partial f_i}{\partial w^j}$, evaluated at $\mathbf{w_t}$. We measure the change in functional performance given $\mathbf{du}$ as the mean squared error

$$d(\mathbf{w_t}, \mathbf{w_d}) = |f(\mathbf{x}, \mathbf{w_t}) - f(\mathbf{x}, \mathbf{w_d})|^2 = \mathbf{du}^T\ (\mathbf{J_{w_t}}^T\ \mathbf{J_{w_t}})\ \mathbf{du} \tag{5.2.2}$$

$$= \mathbf{du}^T\ \mathbf{g_{w_t}}\ \mathbf{du}, \tag{5.2.3}$$

$$\tag{5.2.4}$$

where $\mathbf{g_{w_t}} = \mathbf{J_{w_t}}^T \mathbf{J_{w_t}}$ is the metric tensor evaluated at the point $\mathbf{w_t} \in W$. The metric tensor $\mathbf{g}$ is an $n \times n$ symmetric matrix that defines an inner product and local distance metric, $\langle \mathbf{du}, \mathbf{du} \rangle_\mathbf{w} = \mathbf{du^T}\ \mathbf{g_w}\ \mathbf{du}$, on the tangent space of the manifold, $T_w(W)$ at each $\mathbf{w} \in W$.

Explicitly,

$$g_{ij} = \sum_{k=1}^{m} \frac{\partial f_k(\mathbf{x}, \mathbf{w})}{\partial \mathbf{w^i}} \frac{\partial f_k(\mathbf{x}, \mathbf{w})}{\partial \mathbf{w^j}}, \tag{5.2.5}$$

where the partial derivatives $\frac{\partial f_k(\mathbf{x}, \mathbf{w})}{\partial \mathbf{w^i}}$ measure change in functional output of a network given a change in weight. In the appendix, we extend the metric formulation to cases where we consider a set, $\mathbf{X}$, of training data and view $\mathbf{g}$ as the average of metrics derived from individual training examples. The metric, $\mathbf{g}$, provides a local measure of *functional distance* on the pseudo-Riemmanian manifold $(W, \mathbf{g})$. At each point in weight space, the metric defines the length, $\langle \mathbf{du}, \mathbf{du} \rangle_\mathbf{w}$, of a local perturbation by its impact on the functional output of the

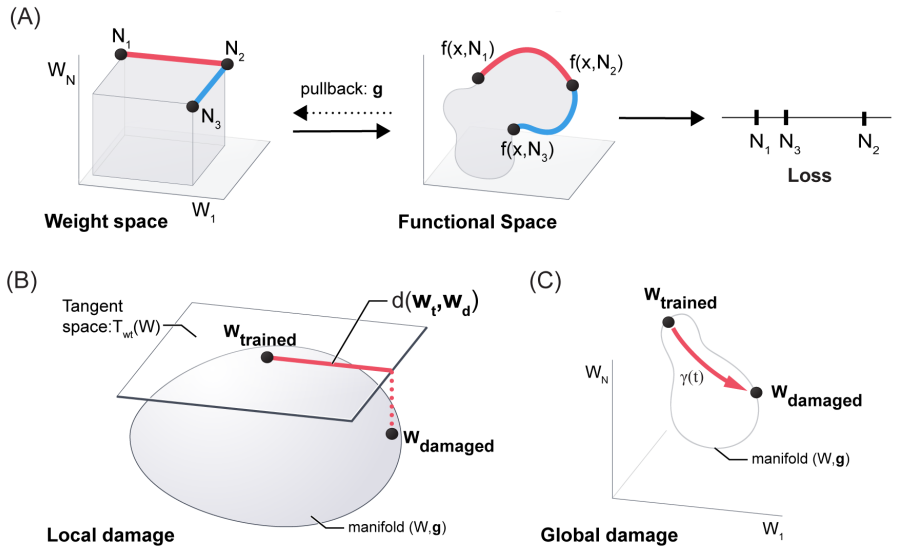


Figure 5.1: **Geometric framework for analyzing neural network resilience** (A) Three networks (N_1, N_2, N_3) in weights space W and their relative distance in functional space and loss space. Damage is analyzed by asking how movement in weight space changes functional performance and loss through introduction of a pullback metric $\mathbf{g}$. (B) We consider local damage to a network as an infinitesimal perturbation that can be analyzed in the tangent space of a trained network. (C) Global damage is modeled as long range movement of network weights along a path, $\gamma(t)$, in weight space.

network (Figure 1b).

Globally, we can use the metric to determine the functional performance change across a path connected set of networks. Mathematically, the metric changes as we move in W due to the curvature of the ambient space that reflects changes in the vulnerability of a network to weight perturbation (Figure 1c). As a network moves along a path, $\gamma(t) \in W$ from a given trained network $\gamma(0) = \mathbf{w_t}$ to a damaged network $\gamma(1) = \mathbf{w_d}$, we can analyze the integrated impact of damage on network performance along $\gamma(t)$ by using the metric to calculate the length of the path $\gamma(t)$ as:

$$L(\gamma) = \int_0^1 \langle \frac{d\gamma(t)}{dt}, \frac{d\gamma(t)}{dt} \rangle_{\gamma(t)} \, dt, \tag{5.2.6}$$

where $\langle \frac{d\gamma(t)}{dt}, \frac{d\gamma(t)}{dt} \rangle_{\gamma(t)} = \frac{d\gamma(t)}{dt}^T \mathbf{g}_{\gamma(t)} \frac{d\gamma(t)}{dt}$ is the infinitesimal functional change accrued while traversing path $\gamma(t) \in W$.

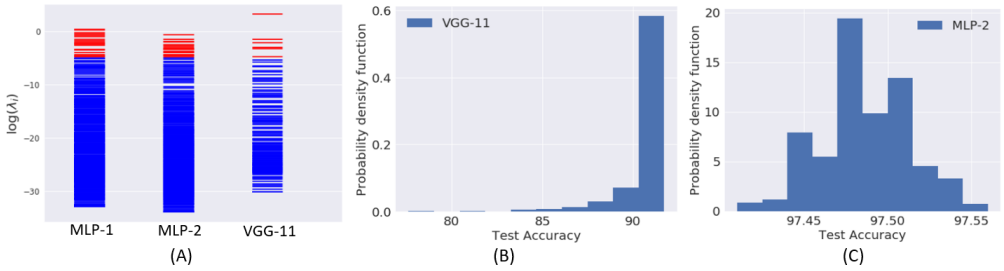
In what follows, we study the resilience of neural networks by analyzing the structure of the metric tensor along paths in weight space. We show that the metric tensor can be used to develop recovery procedures by finding 'geodesic paths', minimum length paths, in the pseudo-Riemannian manifold that allow networks to respond to damage while suffering minimal performance degradation.

5.3 The geometry of local damage and network vulnerability

We, first, apply our mathematical framework to analyze the response of trained neural networks to small, local weight perturbations. Empirical studies have demonstrated that trained networks are often robust to small, local weight perturbation [Morcos et al., 2018, Cheney et al., 2017]. We connect local resilience to the spectral properties of the metric tensor, $\mathbf{g}$, at a given position, $\mathbf{w_t}$, in weight space. We find that networks are typically robust to random local weight perturbations but also have catastrophic vulnerabilities to specific low magnitude weight perturbations that dramatically alter network performance.

To understand local damage, we consider a trained network, $\mathbf{w_t}$, and we subject the network to an infinitesimal weight perturbations in a direction $\mathbf{du} = c_i\ \mathbf{dw^i}$ yielding the perturbed weights $\mathbf{w'} = \mathbf{w_t} + \mathbf{du}$. We use $\mathbf{dw^i}$ to indicate an infinitesimal displacement vector in the direction w_i. Formally, we view $\mathbf{du}$ as a vector in the tangent space of W at $\mathbf{w_t}$, $T_{w_t}(W)$ (Figure 1B). The metric tensor, evaluated at the point $\mathbf{w_t}$ provides a local measure of functional performance change induced by the perturbation along $\mathbf{du}$ through Equation 5.3.2.

As a positive semi-definite, symmetric matrix, $\mathbf{g}$ (evaluated at $\mathbf{w_t}$) has an orthonormal eigenbasis $\{\mathbf{v_i}\}$ with eigenvalues λ_i, $\lambda_i \geq 0$. The eigenvalue λ_i locally determines how a perturbation along the eigenvector $\mathbf{v_i}$ will alter functional performance. Expanding an



Trained network	Dog	Horse	frog	airplane	91%
Random weight perturbation	Dog	horse	frog	airplane	91%
Adversarial weight perturbation	automobile	automobile	frog	automobile	13%

Test accuracy

(D)

Trained network	6	3	6	6	98%
Random weight perturbation	6	3	6	6	97%
Adversarial weight perturbation	5	3	6	5	70%

Test accuracy

(E)

Figure 5.2: **Metric tensor explains local resilience and predicts catastrophic vulnerabilities.** (A) Spectra of the metric tensor for MLP-1, MLP-2 and VGG-11 (B,C) Test performance of networks perturbed within a unit-ball in W(B) perturbed VGG-11 trained on CIFAR-10, (C) perturbed MLP-2 trained on MNIST (D,E) Designing adversarial perturbations to destroy trained networks' performance. (D) adversarial perturbation within unit-ball in W lowers accuracy to 13% in VGG-11, (E) adversarial weight perturbation within unit-ball in W lowers accuracy to 70% in MLP-2.

arbitrary perturbation, $\mathbf{du}$ in the basis $\{\mathbf{v_i}\}$, as $\mathbf{du} = \sum_i c_i\ \mathbf{v_i}$, the functional performance change of the network is

$$d(\mathbf{w_t}, \mathbf{w_t} + \mathbf{du}) = \mathbf{du^T}\ \mathbf{g_{w_t}}\ \mathbf{du} \tag{5.3.1}$$

$$= \sum_i c_i^2 \lambda_i \tag{5.3.2}$$

where $c_i = \langle \mathbf{du}, \mathbf{v_i} \rangle$ quantifies the contribution of vector $\mathbf{v_i}$ to $\mathbf{du}$. Thus, the performance change, $d(\mathbf{w_t}, \mathbf{w_t} + \mathbf{du})$, incurred by a network, following perturbation $\mathbf{du}$ is determined by the magnitude of each λ_i and the projection of $\mathbf{du}$ onto $\mathbf{v_i}$. The eigenvalues λ convert weight changes into change in functional performance and so have units of $\frac{\text{performance change}}{\text{weight change}}$. A network will be resilient to weight perturbations directed along eigenvectors, $\mathbf{v_i}$, with *small* eigenvalues ($\lambda_i < 10^{-3}$). Alternately, networks are vulnerable to perturbations along directions with larger eigenvalues ($\lambda_i > 10^{-3}$). Our definition of resilient directions, $\lambda_i < 10^{-3}$,

is an operational direction that selects directions where a unit of weight change will produce a performance change of less than 10^{-3} or .1%.

Mathematically, we can understand the resilience of networks to randomly distributed weight perturbations by calculating the average response of a network to Gaussian weight perturbations, $\mathbf{du} \sim P(\mathbf{du})$, where $P(du_i) = \mathcal{N}(0, \frac{\sigma}{d})$ ($n = \dim(W)$ and $\mathbb{E}[||\mathbf{du}||_2] = \sigma$) . The expectation of the induced performance change for such a Gaussian perturbation is

$$\mathbb{E}_{du_i \sim \mathcal{N}(0, \frac{\sigma}{d})}[d(\mathbf{w}, \mathbf{w} + \mathbf{du})] = \frac{\sigma}{d} \sum_i \lambda_i \tag{5.3.3}$$

$$< \sigma \rho \lambda_1, \tag{5.3.4}$$

where ρ indicates the fraction of vulnerable directions, and λ_1 is the largest eigenvalue of $\mathbf{g}$.

Empirically, we find that trained networks are, perhaps as expected, robust to 'random' local perturbation (Figure 2) due to a large fraction of resilient eigendirections ($\rho < 10^{-3}$). Such local network robustness holds for a series of trained network architectures including (i) Multi-layer perceptrons (MLP-1, MLP-2) trained on MNIST and (ii) Convolutional neural networks (VGG-11) trained on CIFAR-10. MLP-1 is a single hidden layer network, with variable number of hidden nodes, while MLP-2 is the LeNet architecture borrowed from [LeCun et al., 1998] (2 hidden layers, with 300 and 100 hidden nodes respectively). VGG-11 for CIFAR-10 is adapted from [Simonyan and Zisserman, 2014][1].

Consistent with their eigenspectra (VGG-11: $\rho < 10^{-4}$, MLP-1, MLP-2: $\rho < 10^{-3}$) , both MLP and CNN architectures exhibit minimal performance degradation for unit-ball perturbations [2] ($\sigma = 1$, Figure 2A) . When perturbed along 1000 directions of unit-norm, the trained MLP-2 (initial test accuracy of 98%) maintains accuracy of 97.2-97.6% (Figure 2C). Perturbation of VGG-11 trained on CIFAR-10 (initial test accuracy of 91%) yields networks with test accuracy between 88-91% (Figure 2B).

[1]The network architecture, pre-trained models and optimization algo's are specified in the appendix

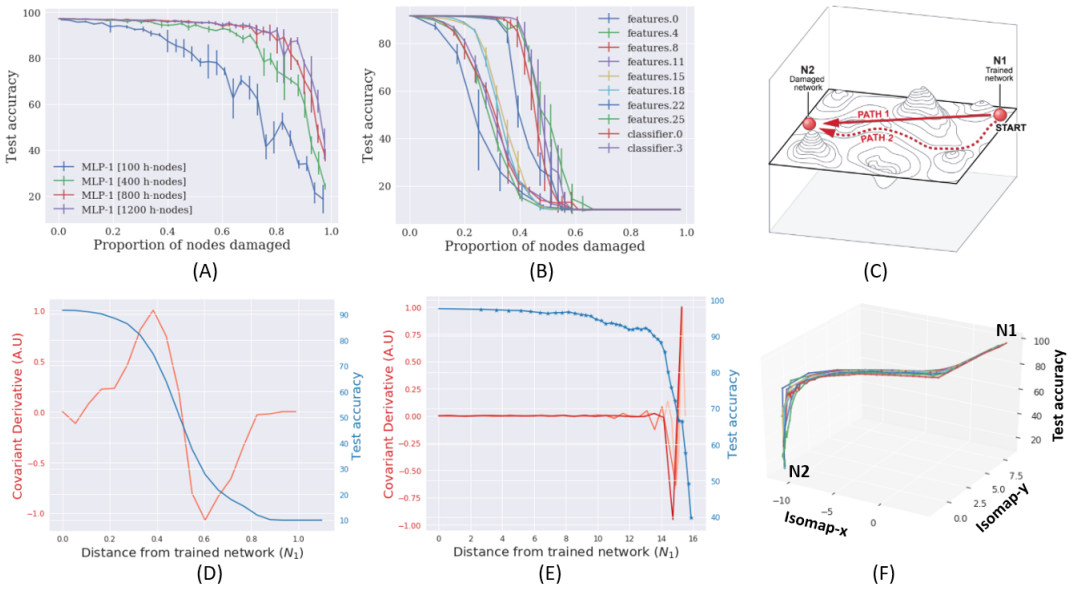
[2]Unit-ball perturbations, due to the high dimensionality of the space, induce an average weight change of $< 10^{-6}$ for individual weights

Resilience to such small local perturbations might be expected, but our framework also exposes hidden catastrophic vulnerabilities to perturbations of the same order in both networks. By designing adversarial weight perturbations to lie along the 'vulnerable' eigenvectors of $\mathbf{g}$ ($\mathbf{v_i}$ with large λ_i), we can induce sharp performance declines across architectures (Figure 2D,E). For the VGG-11 network trained on CIFAR-10, an adversarial weight perturbation decreases accuracy from 91% to 13% (Figure 2D). Similarly, adversarial perturbation reduces the performance of MLP-2 network trained on MNIST from 98% to 70% (Figure 2E). For the CIFAR-10 network, a relatively small perturbation causes the network to make critical classification errors making the erroneous inference of most CIFAR-10 images to being in the class of 'automobiles'. In this way, the local geometry of the weight manifold allows us to discover subtle weight perturbations that cause catastrophic changes in network performance for small change in network weights.

5.4 Acceleration identifies global break-down points in a network

From earlier studies [Morcos et al., 2018, Cheney et al., 2017], we know that trained MLP's and CNN's can be surprisingly robust to much more profound global damage including large scale node deletion. In this section, we develop a concept of break-down acceleration using the *covariant derivative* of a network along paths connecting the trained network and damaged network in W. Break-down acceleration *predicts* failure points that emerge in weight space through rapid changes in the curvature of the weight space, and ultimately allows us to develop procedures to thwart break-down by avoiding acceleration.

Mathematically, we represent global damage as a path in weight space, $\gamma(t) \in W$ with $t \in [0, 1]$, that connects a trained network, $\gamma(0) = \mathbf{w_t}$, to its damaged counterpart $\gamma(1) = \mathbf{w_d}$ (Figure 3C). Practically, global damage might emerge as a discrete event (node deletion),



(a)

Figure 5.3: **Break-down acceleration characterizes network break-down points following damage** Performance of an (A) MLP-1 network (1 hidden-layer, variable hidden nodes) and (B) VGG-11 during simulated damage to distinct layers. Both networks experience sharp performance break down when network damage exceeds (A) ∼90% of hidden-nodes for MLP-1 and (B) ∼60% of nodes in any layer for VGG-11. (C,D,E) **Damage paths in manifold (W, g).** (C) A cartoon of the loss landscape showing multiple break-down paths from the trained network to the damaged network (D,E) The covariant derivative of the accuracy along multiple damage paths for (D) MLP-2 and (E) VGG-11 are shown. A steep increase in the covariant derivative (acceleration) along damage paths corresponds to the networks' sharp break-down to global damage.(F) Multiple damage paths (colored lines) shown from trained MLP-2 (N1) to its damaged counter-part (N2). The z-axes is the test-accuracy of the networks, while x,y axes are the isomap embedding of networks in a 2D space.

our analysis provides a continuous approximation to discrete network damage. As a network moves along a path from $\mathbf{w_t}$ to $\mathbf{w_d}$, the metric tensor itself changes, changing its spectra and its vulnerability.

Along a path, $\gamma(t) \in W$ the velocity vector, $v(t) = \frac{d\gamma}{dt}$, quantifies the change in the functional performance of a network per unit time. Mathematically, we define the break-down speed (s) of a network along a path in weight space as the norm of the network's velocity vector computed using the metric tensor $s(t) = \langle \frac{d\gamma}{dt}, \frac{d\gamma}{dt} \rangle_{\gamma(t)} = \sum_{ij} g_{ij} w_{ti}\ w_{tj}$. Non-linear break-down points emerge along paths in W when break-down speed undergoes a rapid acceleration, so that $\frac{ds}{dt} >> 0$. We can calculate the break-down speed and acceleration

explicitly for a network following simple straight or Euclidean path from a trained to damaged configuration. Taking $w_d = 0$, $\gamma(t) = \mathbf{w_t}(1-t)$ and $\frac{d\gamma(t)}{dt} = -\mathbf{w_t}$, we have

$$\frac{ds}{dt} = \sum_{i,j} \sum_{k} \frac{dg_{ij}}{dw_k} w_{tk}\ w_{ti}\ w_{tj} \tag{5.4.1}$$

where g_{ij} is evaluated along $\gamma(t)$. The change in the metric tensor $\frac{dg_{ij}}{dw_k}$ along a path $\gamma(t)$, thus, determines whether performance decays at a constant $\frac{ds}{dt} = 0$, $\frac{dg_{ij}}{dk} = 0$ or at an accelerating $\frac{ds}{dt} > 0$, $\frac{dg_{ij}}{dw_k} > 0$ rate. For curved paths break-down acceleration can be analyzed using an object known as the covariant derivative, $\nabla_{\gamma(t)} v(t)$ (Appendix).

In practice, calculation of the break-down acceleration identifies, damage failure points in real neural networks. For example, both MLP-1 and VGG-11 architectures tolerate considerable node deletion (Figure 3A). MLP-1, (one hidden layer, 400 hidden units) trained on MNIST, tolerates damage to 80% of the network nodes reducing functional performance of the network by merely ∼10%. Similarly, VGG-11 trained on CIFAR-10 tolerates ∼ 60% node damage in any layer without performance degradation. However, both networks exhibit drastic break-down in functional performance beyond these node damage thresholds (Figure 3A,B). Mathematically, break-down points occur where the acceleration of the network, as measured by the covariant derivative along the damage path, rapidly increases (Figure 3D-F) Steep increases in the covariant derivative identify points of loss acceleration corresponding to the functional breakdown of both the networks analyzed.

5.5 Geodesic paths enable network recovery

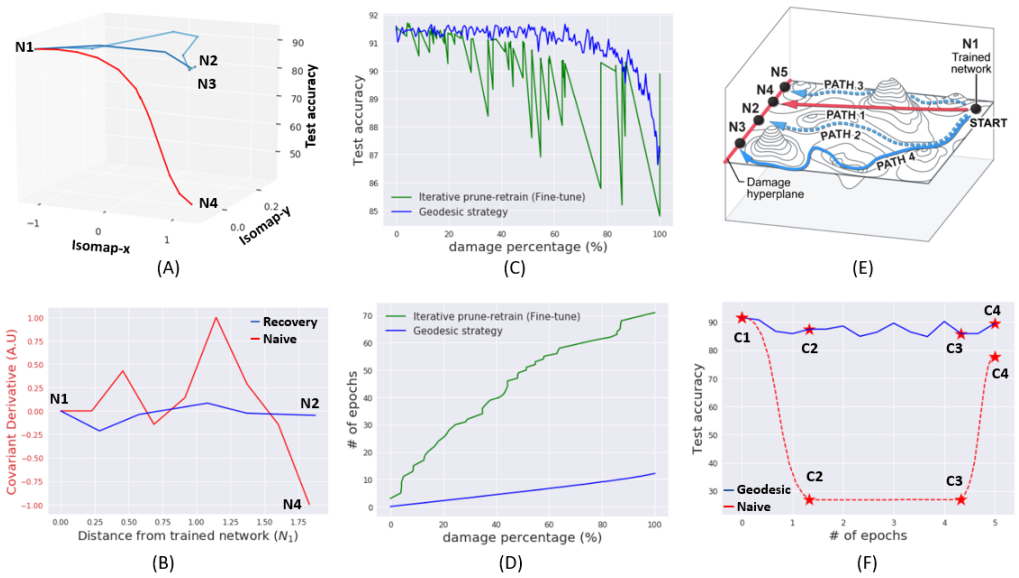
Thus, globally, network break-down occurs along a damage path in Wdue to abrupt changes in the curvature of the underlying functional landscape that result in abrupt change in the metric. The mathematical connection between break-down and curvature suggests a strategy for designing recovery protocols that can adapt a neural network's weights to compensate for

damage. Inspired by recovery mechanisms in neuroscience that compensate for damage by altering the weights of undamaged nodes. We apply the concept of break-down acceleration to develop recovery procedures for artificial neural networks that compensate for damage through continuous adjustment of the undamaged weights by minimizing the acceleration along the path.

Mathematically, minimum acceleration paths in weight space are known as geodesic paths. Geodesic paths, by definition, provide both minimum length and minimum acceleration paths in weight space. Specifically, we consider a trained network, $\mathbf{w}$, subjected to weight damage that zeros a subset of weights, $w_i = 0$, for $i \in n_{\text{damaged}}$. Our strategy responds to damage by adjusting undamaged weights, w_i for $i \notin n_{\text{damaged}}$ to maximize network performance by moving the network along a geodesic in W. Geodesic paths can be computed directly using our metric $\mathbf{g}$ and also represent the minimum distance paths (with distance defined in equation-5.2.6) between two points on W.

In general, geodesic paths are typically calculated using the geodesic equation (Appendix) an ordinary differential equation that uses derivatives of the metric tensor to identify minimum acceleration paths in a space given an initial velocity. However, solutions to the geodesic equation are computationally prohibitive for large neural networks as they require evaluation of the Christoffel symbols which scale as a third order polynomial of the number of parameters in the neural network ($\mathcal{O}(n^3)$).

Therefore, we developed an approximation to the geodesic equation using a first order expansion of the loss function. Given a trained network, our procedure updates the weights of the network to optimize performance given a direction of damage. To discover a geodesic path $\gamma(t)$, we begin at a trained network and iteratively solve for the tangent vector , $\theta(w)$, at every point, $\mathbf{w} = \gamma(t)$, along the path, starting from $\mathbf{w_t}$ and terminating at the damage hyperplane, W_d. The damage hyperplane is the set of all networks, $\mathbf{w} \in W$, such that $w_i = 0$,



(a)

Figure 5.4: **Geodesic paths allow damage compensation through weight adjustment**: **(A)** Test accuracy of geodesic recovery paths (blue) versus naive damage paths (red) for VGG-1 network while 30 convolution filters and 1000 nodes from fully-connected layers are damaged. While naive path exhibits sharp break-down, geodesic procedure adjusts undamaged weights to maintain test accuracy. **(B)** Magnitude of the covariant derivative (break-down acceleration) for geodesic (blue) and naive damage paths (red). **C** Test accuracy (C) and **(D)** number of network update epochs (D) for geodesic recovery (blue) vs fine-tuning (green) while 50 (out of 60) conv-filters are deleted from layer1 in VGG11. Geodesic recovery requires $\leq$10 total update epochs .**(E)** A depiction of multiple recovery paths on the loss landscape from trained network (N1) to networks on the damage hyper-plane (N2, N3, N4, N5). The z-axes is network-loss, while the x,y axes are neural net weights. **(F)** Geodesic strategy (blue) allows networks to dynamically transition between configurations: C1, trained VGG11 network ; C2, 50 conv-filters removed from C1; C3, 1000 additional nodes removed from classifier-layers in C2; C4, 30 conv-filters in conv-layer1 restored to C3. Dynamic transitioning enabled within 5 epochs. Naive strategy is red).

for $i \in n_{\text{damaged}}$. We specifically solve

$$\text{argmin}_{\theta(\mathbf{w})} \ \langle \theta(\mathbf{w}), \theta(\mathbf{w}) \rangle_{\mathbf{w}} - \beta \ \theta(\mathbf{w})^T v_w \text{ subject to: } \theta(\mathbf{w})^T \theta(\mathbf{w}) \leq 0.01. \tag{5.5.1}$$

The tangent vector $\theta(\mathbf{w})$ is obtained by simultaneously optimizing two objective functions: (1) minimizing the increase in functional distance along the path measured by the metric tensor ($\mathbf{g_w}$) [min: $(\langle \theta(\mathbf{w}), \theta(\mathbf{w}) \rangle_{\mathbf{w}}) = (\theta(\mathbf{w})^T \mathbf{g_w} \theta(\mathbf{w}))$] and (2) maximizing the dot-product between the tangent vector and $v_{\mathbf{w}}$, vector pointing in the direction of the hyperplane [max:

$(\theta(\mathbf{w})^T v_{\mathbf{w}})]$ to enable movement towards the damage hyperplane. By finding geodesic paths to the damage hyperplane, we find weight adjustments that can be made within a network during damage to maintain performance (Figure 4E).

Our optimization procedure is a quadratic program that trades off, through the hyper-parameter β, motion towards the damage hyper-plane and the maximization of the functional performance of the intermediate networks along the path (optimization procedure elaborated in the appendix). The strategy discovers multiple paths from the trained network $\mathbf{w_t}$ to W_d, damage hyper-plane, (depicted as path-1 to path-5 in Figure 4a) where networks maintain high functional performance during damage. Of the many paths obtained, we can select the path with the shortest total length (with respect to the metric $\mathbf{g}$) as the best approximation to the geodesic in the manifold.

The geodesic strategy enables damage compensation through continuously updating weights in the network. We apply the geodesic strategy to discover recovery paths from a trained network (VGG11) to a pre-defined damage hyperplane[3]. The recovery path is a high-performance path with all networks performing above 87% test accuracy, and the recovery path maintains low break-down acceleration when compared to the naive (linear) path (Figure 4B). A similar analysis for MLP's is presented in the appendix.

While high-performance paths can also be discovered through heuristic fine-tuning, the geodesic procedure is both rationale and computationally efficient. Specifically, an iterative prune-train cycle achieved through structured pruning of a single node at a time, coupled with SGD re-training [Han et al., 2015, Frankle and Carbin, 2018] (Figure 4C) requires 70 training epochs to identify a recovery path. In comparison, the geodesic strategy finds paths that quantitatively out-perform the iterative prune-train procedure and obtains these paths with only 10 training epochs (figure 4C,D).

Additionally, the same geodesic strategy enables us to dynamically shift networks between

[3]Fig-4A, 4B: Damage hyperplane for VGG11 is defined by := deletion of 30 conv-filters from layer1,2 and 1000 nodes from fully-connected layer1,2.

different weight configurations (eg from a dense to sparse or vice-versa) while maintaining performance (Figure 4F). The rapid shifting of networks is relevant for networks on neuromorphic hardware to ensure that the real-time functionality of the hardware isn't compromised while transitioning between different power configurations.

5.6 Discussion

We have established a mathematical framework to analyze resilience of neural networks through the lens of differential geometry. We introduce a functional distance metric on a Riemmanian weight manifold and apply the metric tensor, covariant derivative, and the geodesic to predict the response of networks to local and global damage. Mathematically, our work forms new connections between machine learning and differential geometry. Practically, we develop new procedures for (i) identifying vulnerabilities in neural networks and (ii) compensating for network damage in real-time through computationally efficient weight updates, enabling their rapid recovery. As neural networks are increasingly deployed on edge devices with increased susceptibility to damage, we believe these methods could be useful in a variety of practical applications.

5.7 Broader Impact

The field of AI has grown by leaps and bounds in the last few years. As a result, AI is increasingly being built into many critical applications across the society. Additionally, to cater to the rising need of AI systems for real-time applications, AI systems have been transitioning from cloud-implementation to edge devices and neuromorphic hardware. Some of the real-time critical applications that have actively adopted AI systems include (1) decision making in the health-care industry, (2) real-time image and sensor analysis in self-driving cars, (3) incorporation into IoT sensors and devices installed in most households and (4)

robotic control systems.

The failure of AI in any of these applications could be catastrophic. For instance, errors committed by AI systems while classifying radiology reports in the health-care industry, or the faulty real-time analysis of stream of images being processed by AI systems in self-driving cars could lead to human casualties. Hence, it has become extremely important for us to understand how neural network architectures (performing critical applications) react to perturbations, that could arise from many sources. AI implemented on the cloud are a victim of DRAM (dynamic random access memory) errors that can occur at surprising rates, either due to malicious attack or induced by high energy particles. Additionally, the growing implementation of AI networks on physical hardware (for instance, neuromorphic, edge devices) has made the need for discovering damage-resilient networks and rapidly recovery damaged networks a necessity.

Our paper lays down the mathematical framework to study resilience and robustness of neural networks to damage and proposes algorithms to rapidly recover networks experiencing damage. Our research will be extremely important for AI systems implemented across many applications, as damage of systems is inevitable and needs to be protected against. Although resilience and robustness of AI systems is very important, there aren't very many principled studies on the same. To reduce the gap in our knowledge on the resilience of AI, we propose a principled framework to understand the vulnerabilites of AI networks. One of the immediate applications of our contribution would be the design of damage-resilient networks and rapid recovery algorithms implemented on neuromorphic hardware. We believe this research will be foundational as neural networks are becoming ubiquitous across many applications, ranging from rovers sent to mars to radiology applications.

Bibliography

[Arechiga and Michaels, 2018] Arechiga, A. P. and Michaels, A. J. (2018). The robustness of modern deep learning architectures against single event upset errors. In *2018 IEEE High Performance extreme Computing Conference (HPEC)*, pages 1–6. IEEE.

[Arlotta and Berninger, 2014] Arlotta, P. and Berninger, B. (2014). Brains in metamorphosis: reprogramming cell identity within the central nervous system. *Current opinion in neurobiology*, 27:208–214.

[Castor and El Massioui, 2018] Castor, N. and El Massioui, F. (2018). Resilience after a neurological pathology: What impact on the cognitive abilities of patients with brain damage? *Neuropsychological rehabilitation*, pages 1–19.

[Cheney et al., 2017] Cheney, N., Schrimpf, M., and Kreiman, G. (2017). On the robustness of convolutional neural networks to internal architecture and weight perturbations. *arXiv preprint arXiv:1703.08245*.

[Firesmith, 2019] Firesmith, D. (2019). System resilience: What exactly is it?

[Frankle and Carbin, 2018] Frankle, J. and Carbin, M. (2018). The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

[Fukushima, 1980] Fukushima, K. (1980). Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36(4):193–202.

[Gates et al., 2000] Gates, M. A., Fricker-Gates, R. A., and Macklis, J. D. (2000). Reconstruction of cortical circuitry. In *Progress in brain research*, volume 127, pages 115–156. Elsevier.

[Gilbert and Li, 2012] Gilbert, C. D. and Li, W. (2012). Adult visual cortical plasticity. *Neuron*, 75(2):250–264.

[Gonzalez et al., 2019] Gonzalez, W. G., Zhang, H., Harutyunyan, A., and Lois, C. (2019). Persistence of neuronal representations through time and damage in the hippocampus. *Science*, 365(6455):821–825.

[Hammerschmidt et al., 2015] Hammerschmidt, K., Whelan, G., Eichele, G., and Fischer, J. (2015). Mice lacking the cerebral cortex develop normal song: insights into the foundations of vocal learning. *Scientific reports*, 5:8808.

[Han et al., 2015] Han, S., Pool, J., Tran, J., and Dally, W. (2015). Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143.

[Hendrycks and Dietterich, 2019] Hendrycks, D. and Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*.

[Hong et al., 2018] Hong, Y. K., Lacefield, C. O., Rodgers, C. C., and Bruno, R. M. (2018). Sensation, movement and learning in the absence of barrel cortex. *Nature*, 561(7724):542–546.

[Iliadis et al., 2020] Iliadis, L. S., Kurkova, V., and Hammer, B. (2020). Brain-inspired computing and machine learning.

[Keck et al., 2013] Keck, T., Keller, G. B., Jacobsen, R. I., Eysel, U. T., Bonhoeffer, T., and Hübener, M. (2013). Synaptic scaling and homeostatic plasticity in the mouse visual cortex in vivo. *Neuron*, 80(2):327–334.

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[Lewin, 1980] Lewin, R. (1980). Is your brain really necessary? *Science*, 210(4475):1232–1234.

[Li et al., 2017] Li, G., Hari, S. K. S., Sullivan, M., Tsai, T., Pattabiraman, K., Emer, J., and Keckler, S. W. (2017). Understanding error propagation in deep learning neural network (dnn) accelerators and applications. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–12.

[Mach and Becvar, 2017] Mach, P. and Becvar, Z. (2017). Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys & Tutorials*, 19(3):1628–1656.

[Mache et al., 2006] Mache, D. H., Szabados, J., and de Bruin, M. G. (2006). *Trends and Applications in Constructive Approximation*, volume 151. Springer Science & Business Media.

[Marder and Goaillard, 2006] Marder, E. and Goaillard, J.-M. (2006). Variability, compensation and homeostasis in neuron and network function. *Nature Reviews Neuroscience*, 7(7):563–574.

[Monroe, 2014] Monroe, D. (2014). Neuromorphic computing gets ready for the (really) big time.

[Morcos et al., 2018] Morcos, A. S., Barrett, D. G., Rabinowitz, N. C., and Botvinick, M. (2018). On the importance of single directions for generalization. *arXiv preprint arXiv:1803.06959*.

[Murphy and Corbett, 2009] Murphy, T. H. and Corbett, D. (2009). Plasticity during stroke recovery: from synapse to behaviour. *Nature Reviews Neuroscience*, 10(12):861–872.

[Richter et al., 2019] Richter, A., Krämer, B., Diekhof, E. K., and Gruber, O. (2019). Resilience to adversity is associated with increased activity and connectivity in the vta and hippocampus. *NeuroImage: Clinical*, 23:101920.

[Schroeder et al., 2009] Schroeder, B., Pinheiro, E., and Weber, W.-D. (2009). Dram errors in the wild: a large-scale field study. *ACM SIGMETRICS Performance Evaluation Review*, 37(1):193–204.

[Scoville and Milner, 1957] Scoville, W. B. and Milner, B. (1957). Loss of recent memory after bilateral hippocampal lesions. *Journal of neurology, neurosurgery, and psychiatry*, 20(1):11.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[Srinivasan and Stevens, 2011] Srinivasan, S. and Stevens, C. F. (2011). Robustness and fault tolerance make brains harder to study. *BMC biology*, 9(1):46.

[Wołczyk et al., 2019] Wołczyk, M., Tabor, J., Śmieja, M., and Maszke, S. (2019). Biologically-inspired spatial neural networks. *arXiv preprint arXiv:1910.02776*.

[Zador, 2019] Zador, A. M. (2019). A critique of pure learning and what artificial neural networks can learn from animal brains. *Nature communications*, 10(1):1–7.

Chapter 6

Machine Learning in Digital Twins

Digital twin (DT) and artificial intelligence (AI) technologies have grown rapidly in recent years and are considered by both academia and industry to be key enablers for Industry 4.0. As a digital replica of a physical entity, the basis of DT is the infrastructure and data, the core is the algorithm and model, and the application is the software and service. The grounding of DT and AI in industrial sectors is even more dependent on the systematic and in-depth integration of domain-specific expertise. This survey comprehensively reviews over 300 manuscripts on AI-driven DT technologies of Industry 4.0 used over the past five years and summarizes their general developments and the current state of AI-integration in the fields of smart manufacturing and advanced robotics. These cover conventional sophisticated metal machining and industrial automation as well as emerging techniques, such as 3D printing and human–robot interaction/cooperation. Furthermore, advantages of AI-driven DTs in the context of sustainable development are elaborated. Practical challenges and development prospects of AI-driven DTs are discussed with a respective focus on different levels. A route for AI-integration in multiscale/fidelity DTs with multiscale/fidelity data sources in Industry 4.0 is outlined.

6.1 Introduction

Industry 4.0 and smart manufacturing are crucial fundamentals of modern industry and the national economy. Industry 4.0 aims to construct a universal networked architecture that addresses the interoperability and compatibility issues within and across all levels of the automation systems and factories, thus improving the flexibility and agility of conventional manufacturing. Equally indispensable to smart manufacturing is advanced robotics, which serves as an intelligent agent appearing in every corner of production lines. With the profound research and development of Industry 4.0 and artificial intelligence (AI), digital twin (DT) has drawn growing research attention [Tao et al., 2018a, Scharl and Praktiknjo, 2019, Wan et al., 2019, Shirowzhan et al., 2020]. As a digital replica of a physical entity, the *basis* of DT is the infrastructure and data, the *core* is the algorithm and model, and the *application* is the software and service. In recent years, the progressive aggravation of environmental problems, such as carbon emission and nuclear pollution, has required national industries to shift from conventional extensive economic growth to sustainable development. Achieving holistic sustainability commonly requires a balance within the *financial*, *environmental*, *social* and *governance* dimensions, i.e., *FESG* factors [Lokuwaduge and Heenetigala, 2017]. This increases the costs of manufacturing enterprises and simultaneously raises severe challenges for their organizations and processes. Against this backdrop, AI-powered DT technology is expected to adapt the traditional model-based approaches to the evolving boundary conditions and provide a demand-oriented, real-time capable evaluation basis to efficiently support decision making in multi-objective problems. There is already much research that discusses and characterizes DT from the view of general concepts and technologies [Tao et al., 2018a, Tao et al., 2019b, Fuller et al., 2020, Jones et al., 2020, Lim et al., 2020, Liu et al., 2020c, Sepasgozar, 2021] as well as certain fields, without targeted focus on AI, this novel enabler, i.e., product design [Schleich et al., 2017, Tao et al., 2019a], modeling and simulation [Rasheed et al., 2020], fault diagnostics and prognostics [Tao et al., 2018b].

Practically, different engineering application scenarios have separate challenges and concerns. The grounding of DT and AI is even more dependent on the systematic and in-depth integration of domain-specific expertise. Currently, there is still a lack of comprehensive industry-oriented review of "AI + DT" technologies in the context of sustainability and circular economy. To further contribute to developing and landing of these general-purpose technologies (GPT) in smart manufacturing and advanced robotics, the following research questions (RQ) are proposed in conducting this survey:

RQ1: What are the current research and concrete case solutions on DTs?

RQ2: What is the current state of AI integration in DTs in the above two areas?

RQ3: What are the benefits of AI-enabled DTs, considering sustainability?

RQ4: What are the challenges in practice and future work with AI-enabled DTs?

This paper endeavors to address this research gap by revisiting current developments on DTs from a domain-specific perspective, analyzing implemented AI methods in each subarea, sorting out the role they play in sustainable development, and summarizing practical challenges in various application fields. The following contributions are delivered in this study:

1. The general development and application cases with common AI methods in AI-driven DTs of Industry 4.0 are concluded.

2. The advantages of AI-driven DTs in sustainable development are elaborated regarding the *FESG* factors, which enable a quantitative assessment of sustainability.

3. Challenges and development prospects of AI-driven DTs in smart manufacturing and advanced robotics are discussed with a respective focus on different levels.

4. A route for AI-integration in multiscale/fidelity DTs with multiscale/fidelity data sources along the product lifecycle is outlined.

6.1.1 Topic Definitions

Digital Twin

The concept of DT was described by the National Aeronautics and Space Administration (NASA) as a multiphysics, multiscale, probabilistic simulation that uses physical models, sensor updates, fleet history, etc., to mirror the life of its twin [Glaessgen and Stargel, 2012]. Later, DT was indicated by Grieves and Vickers as a dynamic model based on massive amounts of information and computing capability that change over the lifecycle, including creation, production, operations and disposal [Grieves and Vickers, 2017]. Based on the architecture in [Grieves, 2014], Tao et al. proposed an extended five-dimension DT model, comprising a physical entity, a virtual entity, service, data and connection [Tao et al., 2018b].

Digital Shadow

The concept "digital shadow" was mainly advocated by the German Academic Society for Production Engineering (WGP), whereby it is understood to be the sufficiently accurate representation of the processes in production, development and adjacent areas with the aim of generating a real-time capable evaluation basis of all relevant data [Bauernhansl, 2016]. Compared with the DT, a digital shadow does not require a high-resolution database, but a complete one [Wohlfeld et al., 2017]. Together with all of its subsystems, the digital shadow is designed as an information system or a multi-perspective information model to allow a more efficient operation of value creation systems, and is thus considered an enabler for data analytics in product lifecycle management (PLM) [Bauernhansl et al., 2018, Riesener et al., 2019].

Digital Thread

The concept of "digital thread" was initially proposed by the U.S. Air Force as a framework to merge detailed design models with model-based systems engineering (MBSE) conceptual

and top-level architectural models [West and Pyster, 2015]. This idea was further driven by the National Institute of Standards and Technology (NIST) with the purpose of exchanging information, including product design and quality and equipment performance and health, across the product lifecycle [Helu and Hedberg, 2015]. A single digital thread is, thus, created with the model-based ensemble of data in design, manufacturing, and inspection, which enables full-process traceability in a seamless, real-time, collaborative development among the project participants [Hedberg et al., 2016].

6.1.2 Topic Delimitation and Coverage

This survey provides in-depth insight into the current progress of AI-driven DT technologies, including the three aforementioned concepts, in Industry 4.0. Although there are various interpretations of the connotations and extensions of the DT, they share the same philosophy, namely, how to utilize digital replicas with near real-time capabilities to effectively enhance traditional organizations and processes across the product lifecycle, thereby improving industry competitiveness and optimizing resource allocation. Correspondingly, 5G communication and Internet of Things (IoT) technologies as well as standalone machine learning (ML) technologies without digital replicas are not the focus here. Over 300 manuscripts are covered on this basis.

6.1.3 Paper Organization

As illustrated in Figure 6.1, the rest of the review paper is organized as follows. In Section 6.2, we analyze the digital production twins toward sustainable resilient manufacturing at three different levels; in Section 6.3, we discuss the applications of DT in robots and human–robot interaction/human–robot collaboration. After dissecting AI-enabled case studies and branch-specific challenges of the development and deployment of DTs in industry verticals, Section 6.4 compares AI methods horizontally; Section 6.5 concludes the contributions of this paper

and addresses the future work.

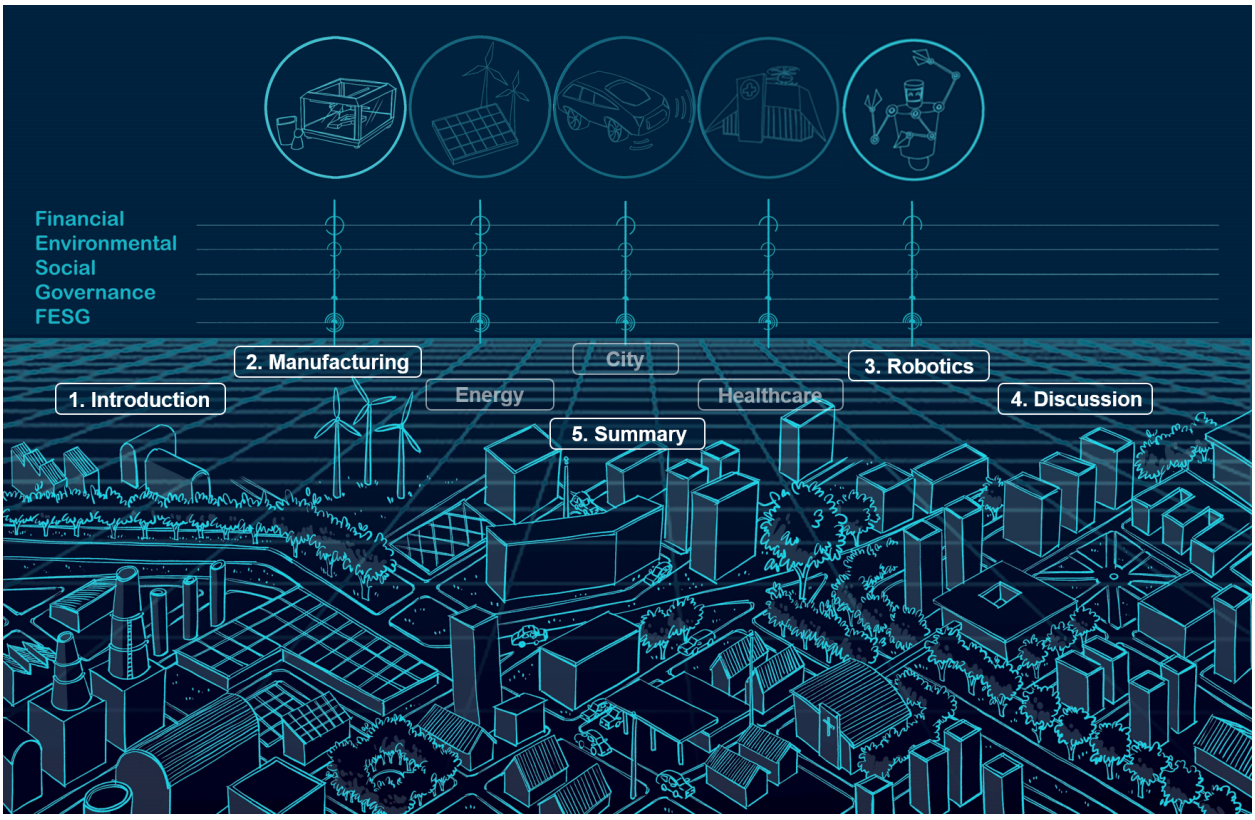


Figure 6.1: Overview of the review.

6.2 Sustainable Resilient Manufacturing

6.2.1 Overview

The core task of manufacturing industry lies in producing qualitative products in a productive and available manner. Balancing these partially competing objectives under time-varying boundary conditions is becoming a challenge in the course of ever-shortened decision-making horizons [Brecher et al., 2017]. In recent years, intensive research has been conducted in the areas of Industry 4.0 [Wagner et al., 2017], cyber–physical production systems [Monostori et al., 2016, Jeschke et al., 2017], and integrative production techniques [Brecher et al., 2012] to address the VUCA (volatile, uncertain, complex, ambiguous) market environment. The

widespread application of simulation models and ubiquitous connectivity provide a solid foundation for building digital production twin throughout the product lifecycle, which is considered a key enabler for future manufacturing transformation and upgrading in the era of big data [Rosen et al., 2015, Tao et al., 2018a, Qi and Tao, 2018]. In the context of the circular economy and sustainability pledges (e.g., European Green Deal), traditional resource-intensive productivity thinking is being replaced by a future vision of a more ecologically minded society, namely "sustainable productivity" [Boos, 2021]. This understanding of productivity incorporates *FESG* factors as a novel indicator for quantitatively assessing sustainable production as well as the performance (with tangible and intangible services as well as business models) and value creation systems (consisting of resource, process, and organization) of manufacturing companies, thus pushing them to embrace the required sustainability transformation, as illustrated in Figure 6.2. Various research of DTs, including general developments and AI-integrated cases, in terms of enhancing the trilemma of productivity, availability, and quality (*financial*) toward sustainable resilient manufacturing (*environmental*, *social*, *governance*) are discussed in the following three levels: factory and shop floor (Section 6.2.2), machinery and equipment (Section 6.2.3), as well as process and material (Section 6.2.4).

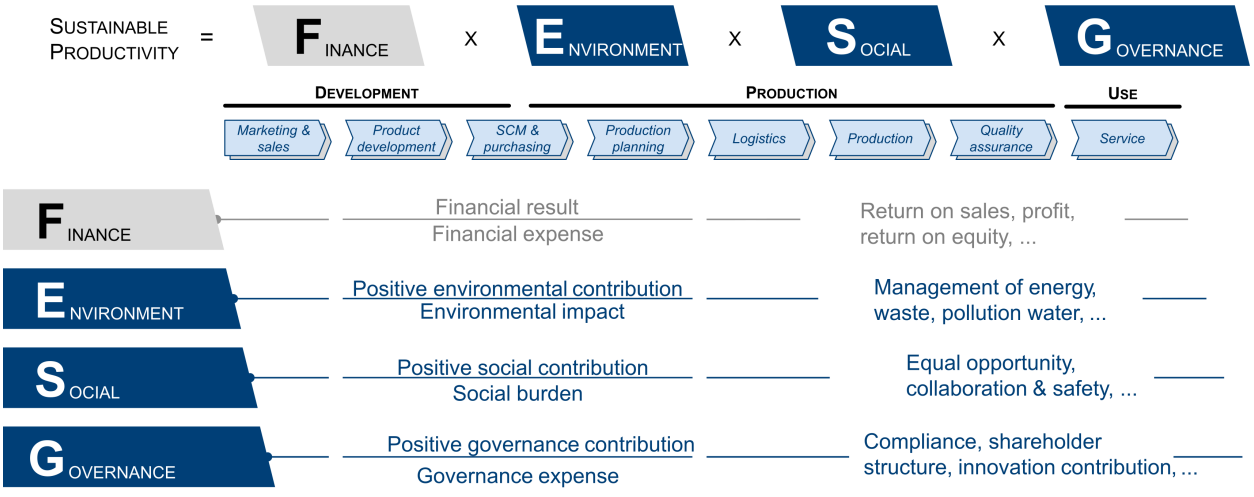


Figure 6.2: An illustration of holistic assessment of sustainable production [Boos, 2021]. © WZL—RWTH Aachen.

6.2.2 Factory and Shop Floor

General Developments

The megatrend toward the volatile market environment and individual customer demand poses new challenges, primarily for production systems and management in industrial enterprises, which can significantly influence the profitability and productivity of manufacturing. Within the various presented concepts and frameworks [Helu et al., 2017, Zheng et al., 2019, Ding et al., 2019, Tao and Zhang, 2017, Pennekamp et al., 2019, May et al., 2020], automated production systems, including mixed reality assistance systems [Roh, 2018, Rebmann et al., 2020], could be rapidly modularized [Um et al., 2017] and reconfigured [Leng et al., 2020, Feng et al., 2020, Talkhestani et al., 2018], enhanced with AI [Jazdi et al., 2021, Ma et al., 2021] and sensors [Cai et al., 2017, Chhetri et al., 2019] and, in combination with cloud and edge computing [Brecher et al., 2019a], transformed into distributed control systems, while detailed production environments can be generated and updated in the form of 3D point clouds [Biesinger et al., 2018, Delisle et al., 2018, Minos-Stensrud et al., 2018, Hellmuth et al., 2020, Sommer et al., 2019, Pan and Zhang, 2021]. Based on these infrastructures, DT demonstrates the capability of handling increasingly complex operational problems, such as production planning and scheduling [Schuh et al., 2016, Fang et al., 2019, Guo et al., 2020b, Zhang et al., 2020d], production monitoring and control [Borangiu et al., 2020, Zhang et al., 2020b, Karanjkar et al., 2019, Leng et al., 2019, Guo et al., 2020a, Liu et al., 2021], quality control and management [Wagner et al., 2018, Grégorio et al., 2020, Wunderlich et al., 2018, Polini and Corrado, 2020, Schmitt and Voigtmann, 2018, Wagner et al., 2020, Schmitt et al., 2019, vZidek et al., 2020, Psarommatis, 2021], as well as logistics [Haße, 2019, Pan et al., 2020, Brenner and Hummel, 2017], supply chain management (SCM) [Ivanov and Dolgui, 2020, Marmolejo-Saucedo, 2020, Cirullies and Schwede, 2021], disassembly and remanufacturing [Tozanli et al., 2020, Wang and Wang, 2019, Wang et al., 2020e].

AI-Integration

The availability of industrial production data in a networked system landscape acts in this background as a technical enabler to increase the relevance of topics, such as AI and data-driven approaches. This further opens up new potentials for optimizing (novel) manufacturing systems, e.g., line-less mobile assembly systems in Figure 6.3, which enable agile assembly of large components by leveraging modeling and scheduling systems [Hüttemann et al., 2019]. The primary objective of utilizing AI at this level is to improve the adaptability of DTs to dynamically changing boundary conditions on the factory and shop floor scale. Typical application subfields with AI-integrated DTs are *production planning* (Section 6.2.2 A), *production control* (Section 6.2.2 B), and *quality control* (Section 6.2.2 C), as shown in Table 6.1.

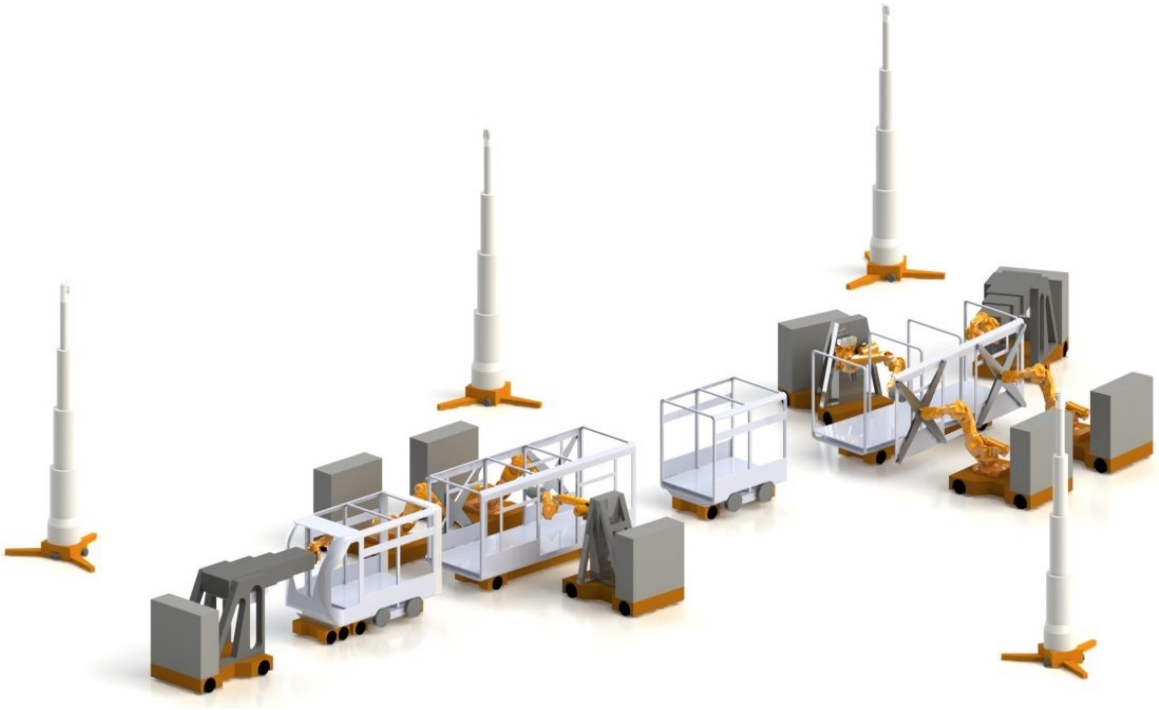


Figure 6.3: An illustration of DT in manufacturing: line-less mobile assembly system [Hüttemann et al., 2019]. © WZL—RWTH Aachen.

A. Production Planning According to the maturity model of production planning and control (PPC) proposed by Busch et al. [Busch et al., 2019], toward digitally connected,

Table 6.1: Summary of AI-enabled DTs in smart manufacturing: **factory and shop floor level** (Section 6.2.2).

Subfield	**AI-Category**	**Key Methods**	**Application-Case**	**Ref.**
Production Planning (Section 6.2.2 A)	Supervised Learning	Decision tree	Material selection; tool holder selection; tool wear level prediction	[Rojek et al., 2021]
	Reinforcement Learning	DQN	Dynamic scheduling of flexible manufacturing systems	[Hu et al., 2020]
	Computational Intelligence	GA, DES	Optimization of production scheduling	[Negri et al., 2020]
		GA	Optimization of smart assembly lines	[Rezaei Aderiani et al., 2021]
		Supernetwork	Scheduling optimization of a gear production workshop	[Liu et al., 2020d]
		Multi-objective optimization	Design of automatic flow-shop manufacturing system	[Liu et al., 2019]
Production Control (Section 6.2.2 B)	Supervised Learning	PGM	Resource allocation for sequential manufacturing operations	[Ding et al., 2019]
		DNN	Assembly commissioning process optimization	[Sun et al., 2020]
		AdaBoost, CART, Gradient boosting	Identification of real-time factors that influence throughput in a semiconductor fab	[Singgih, 2021]
		AdaBoost, XGBoost, decision tree	Optimum yield of the light oil	[Min et al., 2019]
	Reinforcement Learning	Deep reinforcement learning	Geometrical quality improvement in assembly	[Cronrath et al., 2019]
		Q-Learning	Box sorting in a material flow control system	[Jaensch et al., 2019a]
		DQN	Optimization of the workflow in a semiconductor wafer processing plant	[Waschneck et al., 2018]
		DQN	Optimization of conveyor systems	[Wang et al., 2020b]
		TRPO	Optimizing order dispatching	[Kuhnle et al., 2020]
		TRPO	Human behavior forecasting	[May et al., 2021]

DQN: Deep Q-Network; GA: Genetic Algorithms; DES: Discrete Event Simulation; PGM: Probabilistic Graphical Model; DNN: Deep Neural Network; AdaBoost: Adaptive Boosting; CART: Classification and Regression Tree; XGBoost: eXtreme Gradient Boosting; TRPO: Trust Region Policy Optimization; ANN: Artificial Neural Network; CNN: Convolutional Neural Network; k-NN: k-Nearest Neighbors algorithm; SVM: Support Vector Machine; ResNet: Residual Neural Network.

intelligent and adaptive PPC systems, AI is envisioned to support production planners in determining plans with improved key performance indicators (KPI), derive optimization measures and autonomously implement the identified measures prospectively to achieve better sequencing and reallocation of resources (*E*-factor). At the *green design* and *production planning* stage, decision tree could be applied in DT to create classic rules used in smart systems, thus facilitating decision making in multidimensional processes and strategic planning [Rojek et al., 2021]. Hu et al. introduced a Petri-net-based dynamic scheduling approach via a deep Q-network (DQN) with graph convolution network (GCN) to solve the dynamic scheduling problems involving shared resources, route flexibility, and stochastic arrivals of raw products [Hu et al., 2020]. Metaheuristic methods, such as the genetic algorithm (GA) and other optimization methods [Rezaei Aderiani et al., 2021, Liu et al., 2019, Liu et al., 2020d, Negri et al., 2020] were similarly widely employed to deal with the scheduling problems in production lines.

B. Production Control At the *production control* stage, DNN [Sun et al., 2020], decision tree [Min et al., 2019, Singgih, 2021], and tree-based ensemble models, such as AdaBoost [Min et al., 2019, Singgih, 2021] and XGBoost [Min et al., 2019], were implemented in assorted digital production twins to optimize resource allocation and manufacturing performance indicators in a timely manner (*E*-factor). However, the multi-objective problems at the factory level are usually interpreted as non-deterministic, polynomial-time hard, due to the complexity and dynamics in production environments. To address this challenge, reinforcement learning (RL), such as DQN and deep RL, were employed as a substitute for heuristic optimization and supervised approaches in various investigations, where the major task is normally mathematically formalized as a Markov decision process (MDP), with the objective of autonomously achieving the global optimal economic and logistic KPIs in a factory or logistic simulation environment [Waschneck et al., 2018, Wang et al., 2020b, Jaensch et al., 2019a, Kuhnle et al., 2020] (*EG*-factor). In order to incorporate humans as a critical element

in smart manufacturing, May et al. presented a concept for the situational selection of production control agents by forecasting human behavior modeled through a reinforcement learner [May et al., 2021] (*ES*-factor).

C. Quality Control At the *quality control* stage, classical supervised ML models, such as ANN, decision tree, and SVM, were expected to detect or predict potential deformations and surface deviations in production [Li et al., 2020b, Yacob et al., 2019]. Deep learning (DL) computer vision models, including residual and convolutional neural networks were deployed to recognize eventual quality issues during the automatic production and machining features of parts [Alexopoulos et al., 2020, Zhou et al., 2020], which could be further utilized to enhance the quality and efficiency of assembly processes [Guo et al., 2018] (*E*-factor), or retraced to the production planning stage in order to support decision making on the basis of historical production knowledge [Zhang et al., 2020a], as a "smart expert" in a collaborative environment (*SG*-factor). Following the general concept of integrating ML methods into the digital production twins [Jaensch et al., 2019b], DTs of production systems in combination with MBSE can be modeled and adapted modularly as a virtual testbed, which in turn could provide a runtime environment for simulation-based optimization [Delbrügger and Rossmann, 2019, Delbrügger et al., 2019].

Interim Summary

Business profitability remains the prerequisite for sustainable operation. DTs at this level elevate the productivity, resilience, and transparency of production processes, which enable end-to-end availability of data along the entire value chain as well as a holistic sustainability assessment on this basis. From a business development perspective, AI-enabled DT can additionally be considered a service agent [Meierhofer et al., 2020], providing innovative smart services via DT network platforms [Li et al., 2020c] and subscription business models [Kampker et al., 2019], thus contributing sustainably to long-term innovation for manufacturers, and

helping them in accomplishing a paradigm shift from the one-time provision of production hardware to the ongoing delivery of manufacturing solutions (*SG*-factor). The importance of a diversified product and service portfolio is particularly evident in times of global crisis. For small- and medium-sized enterprises (SMEs), innovative services from research institutes and major manufacturing equipment suppliers can facilitate the conduction of the comprehensive balance sheet assessment covering FESG factors in order to achieve a smooth transition to sustainability.

6.2.3 Machinery and Equipment

General Developments

The availability of production machinery and equipment has a direct impact on the efficiency of manufacturing processes and overall equipment effectiveness, and is, therefore, equally significant for sustainable production, especially regarding energy consumption and resource utilization (*E*-factor). Since their degradation and damage level can be affected by working operations as well as other disturbances in harsh manufacturing environments, such as lubricants, soiling, and temperature, the results from traditional wear and fatigue models are connected with considerable uncertainties. DT-driven condition monitoring (CM) and predictive maintenance (PdM) highlight the possibility for the process-parallel monitoring and diagnosis of the health status of the critical components (i.e., tools [Xi et al., 2021, Botkina et al., 2018], bearings [Mi et al., 2020], ball screws [Luo et al., 2019], gears [Tao et al., 2018b, Aivaliotis et al., 2019, He et al., 2021c], pumps [Bergs et al., 2019]) and the energy efficiency of the equipment [Zhang et al., 2018, Blume et al., 2020] in order to handle the conflict between the unplanned maintenance operations and the resulting costs and productivity, particularly in SMEs, due to their limited capacity for the full deployment of a PdM strategy [Surico et al., 2020]. Additionally, DT-based optimal control [Lynn et al., 2018, Xu and Guo, 2021] as well as machine dynamics issues [Wagg et al., 2020, Huynh and Altintas, 2021] from a rotating

Table 6.2: Summary of AI-enabled DTs in smart manufacturing: **machinery and equipment level** (Section 6.2.3).

Subfield	AI-Category	Key Methods	Application-Case	Ref.
Condition Monitoring (Section 6.2.3 A)	Supervised Learning	ANN	Prediction of process forces	[Königs et al., 2017]
		CNN	Prediction of process forces	[Su et al., 2021]
		CNN, SVDD	Defect recognition of steel surfaces	[Yiping et al., 2021]
		CNN-DLSTM based transfer learning	Fault detection of rolling bearings	[Wang et al., 2020f]
	Unsupervised Learning	GAN	Prediction of machining vibration signals	[Zotov et al., 2020]
		Dictionary learning, transfer learning	Wave field prediction for damage detection with ultrasonic guided wave	[Alguri et al., 2021]
	Computational Intelligence	Fuzzy inference	Brake CM of an overhead crane	[Autiosalo et al., 2021]

system [Wang et al., 2019] to feed drive [Tseng et al., 2019, Wang et al., 2020a] are other important aspects.

AI-Integration

Conventional model-based approaches of CM and PdM lie in the evaluation of process indicators, which are either recorded from sensors directly or determined indirectly by them. While the installation of external sensors (e.g., force measurement platforms and rotating dynamometers) increases costs and, more seriously, could negatively affect machine properties and manufacturing stability, DT combined with ML as a promising (soft) sensing technique provides an economically reasonable and sufficiently accurate approach to identifying such indirectly measurable process parameters (*EG*-factor). Table 6.2 provides an overview of AI-enabled cases in *condition monitoring* (Section 6.2.3 A), *predictive maintenance* (Section 6.2.3 B), and *dynamics and control* (Section 6.2.3 C).

A. Condition Monitoring For *condition monitoring* of machining tools and processes, Königs et al. [Königs et al., 2017] proposed a hybrid modeling method, whereby an ANN utilized machine internal signals and the cutter-workpiece engagement map generated from a real-time virtual machining simulation as inputs to predict the cutting force as an essential indicator. Su et al. proposed another machining force prediction model based on the

Table 6.2: *Cont.*

Subfield	**AI-Category**	**Key Methods**	**Application-Case**	**Ref.**
Predictive Maintenance (Section 6.2.3 B)	Supervised Learning	PGM, MCMC	Prediction of stress-intensity factors and RUL	[Leser et al., 2020]
		RCM	Prediction of RUL of a drilling machine	[Cattaneo and MacChi, 2019]
		RF, particle filter	Prediction of tool wear	[Luo et al., 2020]
		Deep Stacked GRU	Prediction of tool wear	[Qiao et al., 2019]
		LSTM	Equipment utilization prediction	[Wang and Luo, 2021]
		LSTM	Tool condition prognostic model	[Xie et al., 2020]
		LSTM	Estimation of RUL of the machine components	[Anis et al., 2020]
	Unsupervised Learning	GMM	Tool failure prediction	[Ladj et al., 2020]
		SSAE-PHMM	Prediction of tool wear	[Zhang et al., 2021]
		SSAE, deep transfer learning	Fault prognosis in a car body-side production line	[Xu et al., 2019]
		GAN, VAE	Generation of a health indicator for PHM of rotating systems	[Booyse et al., 2020]
		CAE	Construction of a health indicator for bearings	[Kaji et al., 2020]
		Distributed k-means	Assessing MAS for collaborative PdM	[Salvador Palau et al., 2019]
	Computational Intelligence	Bayesian network	Mission planning under uncertainty with respect to fatigue cracking	[Karve et al., 2020]
Dynamics & Control (Section 6.2.3 C)	Supervised Learning	RNN	Prediction of dynamic states in metal cutting	[Kabaldin et al., 2019]
		ANN	Prediction of resonances frequencies of a thin bulk acoustic wave resonator	[Simon et al., 2020]
	Computational Intelligence	Gaussian process	Estimation of single-degree-of-freedom dynamic systems	[Chakraborty et al., 2021]
		Gaussian process	Prediction of the dynamic response	[Gardner et al., 2020]
		GWO	Optimization of motion control system in machine tools	[Haber et al., 2020]

SVDD: Support Vector Data Description; RCM: Random Coefficient Model; RF: Random Forest; MCMC: Markov Chain Monte Carlo; GRU: Gated Recurrent Units; (D)LSTM: (Deep) Long Short Term Memory; GMM: Gaussian Mixture Model; GAN: Generative Adversarial Network; SSAE-PHMM: Stack Sparse AutoEncoder Parallel Hidden Markov Model; VAE: Variational AutoEncoder; CAE: Convolutional AutoEncoder; MAS: Multi-Agent System; RNN: Recurrent Neural Network; GWO: Grey Wolf Optimization.

cutter frame image data using CNN [Su et al., 2021]. Gao et al. [Yiping et al., 2021] introduced a deep lifelong learning method based on CNN and an SVDD-based (support vector data descriptor) detector, which enables recognizing multiple tool defects (crazing, patches, scratches, etc.) with novel classes by learning relevant tool images. CNN-DLSTM based transfer learning [Wang et al., 2020f], generative models [Zotov et al., 2020], and dictionary learning [Alguri et al., 2021] were furthermore employed to assist in abnormal signals detection and fault prognosis (*E*-factor).

B. Predictive Maintenance Based on the monitoring status, health indicators, such as remaining useful life (RUL), could be estimated, particularly under non-stationary operation conditions, thus enabling effective *predictive maintenance* planning (*E*-factor). Besides statistical [Cattaneo and MacChi, 2019, Leser et al., 2020] and hybrid modeling [Luo et al., 2020] approaches, DL models for time-series forecasting, such as LSTM [Anis et al., 2020, Wang and Luo, 2021, Xie et al., 2020], were adopted in numerous studies to estimate wear status and equipment utilization. However, the acquisition of massive, structured and labeled data, especially regarding complex rotating equipment and components, is normally tied up with high costs since the fault-free operation is a frequent case in production. Considering this fact, several attempts with unsupervised and semi-supervised learning models, e.g., GMM (Gaussian mixture model) [Ladj et al., 2020], SSAE (stacked sparse autoencoder) [Xu et al., 2019, Zhang et al., 2021], and GAN (generative adversarial network) [Booyse et al., 2020], were investigated in order to reduce the reliance on historical failure data in terms of prognostics and health management (PHM). Regarding the cost factor, Palau et al. [Salvador Palau et al., 2019] provided a methodology to assess the optimal multi-agent system (MAS) architecture for collaborative PdM in large fleets of industrial assets by using a distributed k-means clustering algorithm (*ES*-factor).

C. Dynamics and Control The manufacturing stability and reliability are furthermore closely related to the dynamic behavior of production machines and process machine interactions (e.g., chatter), namely, *dynamics and control*, which are due to complex damping effects and structure modification (e.g., tool changing), which are, in practice, difficult to estimate accurately and/or efficiently (*E*-factor). In this respect, Kabaldin et al. [Kabaldin et al., 2019] selected RNN to estimate the statistical model of the dynamic state in cutting. ML models, such as ANN [Simon et al., 2020], and probabilistic modeling methods, such as the Gaussian process [Chakraborty et al., 2021, Chakraborty and Adhikari, 2021, Gardner et al., 2020], could likewise be adopted to develop a surrogate model and implemented in a control context [He et al., 2021b, Gardner et al., 2020, Haber et al., 2020]. Alternatively, model order reduction techniques can transfer highly detailed and complex simulation models to other domain and life cycle phase, e.g., building efficient finite element model for dynamic structural analysis through reducing the degree of freedom, while maintaining required accuracies and predictability [Hartmann et al., 2018, Podskarbi and Knezevic, 2020, Zambrano et al., 2020].

Interim Summary

As the backbone of the manufacturing industry, research on machinery and equipment has a protracted history, yet complex and varying working conditions with relatively sparse datasets in practice frequently make it challenging to transfer research findings, commonly in the form of elaborate analytical/empirical models, to real industrial environments. From a data perspective, the networked production landscape provides an additional unique opportunity to use each manufacturing system as a "test bench" in order to continuously enhance the database and the amount of labeled training samples regarding these indirectly measurable and non-measurable indicators. The improved availability of scarce datasets prospectively extends the previous model boundaries and transferability through ML/DL in equipment fault diagnosis and system behavior prediction, which remain as central concerns within the scope of sustainable manufacturing (*E*-factor). High data rates empowered by 5G technology [Kehl

et al., 2020] similarly open up novel prospects for research on AI-driven DTs in the field of PHM, e.g., for sensing high-frequency phenomena in high speed/performance cutting. Figure 6.4 illustrates the application of digital twin for CM and PdM in the networked, adaptive production.

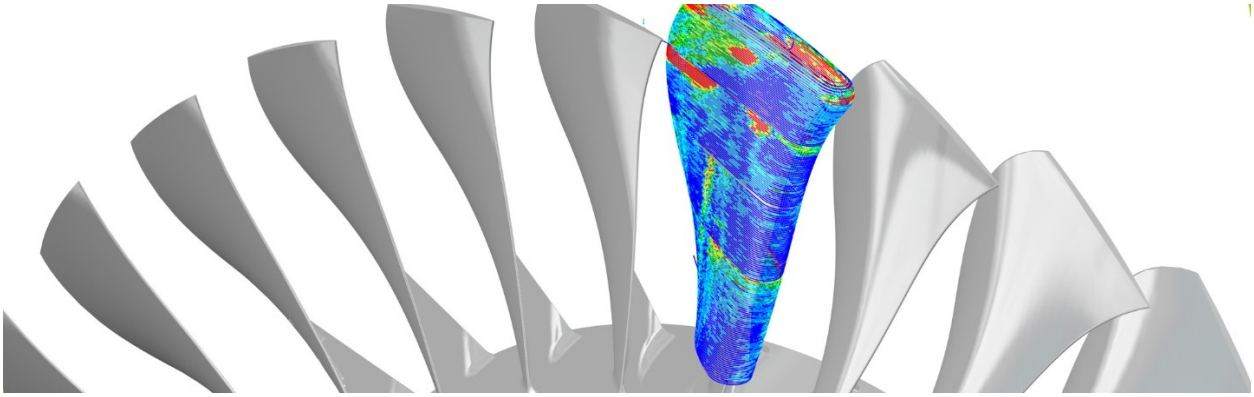


Figure 6.4: An illustration of DT in manufacturing: CM in the networked, adaptive production. © Fraunhofer IPT.

6.2.4 Process and Material

General Developments

At this level, we are more concerned about the quality issues and mechanical properties of manufactured parts. The quality of parts is normally determined in quality assurance after the entire machining process by testing quality specifications, such as form and position tolerances, as well as surface roughness. Regarding the optimization of the current quality control loop and the associated costs, the digital process twin represents a core element in modern manufacturing. For instance, the surface quality of parts can be estimated in parallel to the process within a GPU-enabled (graphical processing unit) material removal simulation with a subsequent virtual measurement, which significantly reduces the latency between machining and the detection of defective parts [Königs and Brecher, 2018, Brecher et al., 2019b]. A sufficiently accurate virtual representation of the machining process [Brecher et al., 2017, Denkena et al., 2021, Afrasiabi et al., 2020, Cao et al., 2020, Hänel et al., 2020] also

enables cause-and-effect analysis and, therefore, robust process design and control [Tong et al., 2020, Armendia et al., 2019, Brecher et al., 2019c, Ward et al., 2021, Heo and Yoo, 2021] as well as the exploitation of potential process productivity [Brecher et al., 2019d]. Further cases of production processes with knowledge-based approaches include welding [Söderberg et al., 2018, Papacharalampopoulos et al., 2020], injection molding [Loaldi et al., 2020, Bibow et al., 2020], linear winding [Weigelt et al., 2019], tape laying [Schulz et al., 2020], metal forming [Havinga et al., 2020], laser polishing [Bordatchev et al., 2019], automated fiber placement [Zambal et al., 2018], sheet molding compound [Görthofer et al., 2019], fused filament fabrication [Moretti et al., 2021], and metal additive manufacturing (AM) [Huang et al., 2020, Reisch et al., 2020, Ünal-Saewe et al., 2020, Heo et al., 2021, Ertveldt et al., 2020].

AI Integration

While the refining scale of models allows deeper insights into the mechanism of manufacturing processes, their modeling takes correspondingly more time. In addition to leveraging the parallel computing power of GPUs for near real-time process simulation, it is common practice in engineering to compensate online with the results of preprocessed offline simulations. ML and DL methods can thus be trained as surrogate models in order to efficiently update time-consuming numerical simulations at the process and material scale. These ML/DL-equipped lightweight models could be used for faster production ramp-up as well as soft sensory for inline-quality monitoring (*EG*-factor). They also deliver additional process understanding, thus optimizing the space-time yield and accelerating process development [Bayer et al., 2021]. Table 6.3 presents a summary of cases of AI-driven digital process twins for major manufacturing techniques involving *metal cutting* (Section 6.2.4 A), *metal AM and laser material processing* (Section 6.2.4 B), and *composite material processing* (Section 6.2.4 C).

Table 6.3: Summary of AI-enabled DTs in smart manufacturing: **process and material level** (Section 6.2.4).

Subfield	AI-Category	Key Methods	Application-Case	Ref.
Metal Cutting (Section 6.2.4 A)	Supervised Learning	PIO, SVM	Prediction of surface roughness	[Zhao et al., 2020]
		Ensemble methods, ANN	Modeling of the rheological behavior of drilling fluids	[Samnejad et al., 2020]
		ANN	Prediction of stress and fatigue damage (FE surrogate) of flexible risers	[Repalle et al., 2020]
		DNA-based computing, Markov chain	Prediction of surface roughness	[Ghosh et al., 2020]
	Reinforcement Learning	DDPG	Optimization of decision-making based on performance and machinability of parts	[Zhou et al., 2021]
	Computational Intelligence	PSO	Inverse determination of material model parameters from cutting simulation	[Hardt et al., 2021]
Metal AM and Laser Material Processing (Section 6.2.4 B)	Supervised Learning	SVM	Prediction of the occurrence of defects in metal AM (LPBF, LMD)	[Gaikwad et al., 2020b]
		CNN, LSTM, RNN	Quality assurance in metal AM (LPBF)	[Gaikwad et al., 2020a]
		CART	Prediction of additive manufacturability	[Ko et al., 2019]
		HMM	Model adaptivity and quality assessment of laser material removal processes	[Stavropoulos et al., 2020]
	Unsupervised Learning	k-means	Anomaly detection and process optimization of 3D laser cutting processes	[Stojanovic and Milenovic, 2019]
Composite Material Processing (Section 6.2.4 C)	Supervised Learning	CNN, transfer learning	Detection of dry points in the production of carbon fiber reinforced plastics	[Stieber et al., 2020]
		AdaBoost, XGBoost, RF	Prediction of temperature distribution of thermoplastic composites	[Hürkamp et al., 2020]
		DNN	FE surrogate for a composite textile draping process	[Pfrommer et al., 2018]
		PML	Prediction of material properties of a composite material system	[Ghanem et al., 2020]
	Computational Intelligence	ISRES	Identification of material parameters of a prepreg sheet	[Chen et al., 2021]
Joining	Supervised Learning	DNN, GA	Prediction of distortion in welding	[Asadi et al., 2020]
Forming	Supervised Learning	ANN	Prediction of the ingate velocity during sand mold filling	[Ktari and El Mansori, 2020]

PIO: Pigeon-Inspired optimization; DDPG: Deep Deterministic Policy Gradient; PSO: Particle Swarm Optimization; HMM: Hidden Markov Model; PML: Probabilistic Machine Learning; ISRES: Improved Stochastic Ranking Evolution Strategy.

A. Metal Cutting For conventional *metal cutting* techniques, e.g., milling and drilling, the part quality is primarily determined by sophisticated machine behavior and cutter-workpiece engagement, which can become particularly intricate in multi-axis machining or for parts with thin-walled structures. Zhao et al. constructed a self-learning surface roughness prediction model based on pigeon-inspired optimization and SVM in order to stabilize the part quality with a self-adaptation adjustment method [Zhao et al., 2020]. Approaches combining ANN and semantic modeling for fatigue and quality prediction were discussed in [Samnejad et al., 2020, Repalle et al., 2020, Ghosh et al., 2020]. Following the philosophy of DfX (Design for X), Zhou et al. utilized the DDPG (deep deterministic policy gradient) approach to optimize decision making, according to the performance and machinability of parts, which could shorten cycles and save costs in the product development [Zhou et al., 2021] (SG-factor). At the material scale, evolutionary algorithms, such as PSO, were investigated by Hardt et al. to inversely identify the material model parameters in finite element (FE) simulations of orthogonal cutting processes [Hardt et al., 2021].

B. Metal AM and Laser Material Processing For advanced *metal AM and laser material processing* techniques, e.g., laser powder bed fusion (LPBF) and laser melting deposition (LMD), research efforts focus on the subtle impacts of thermal effects on materials, such as the microstructure and parts' distortion, during such non-contact processes. As Gaikward et al. in [Gaikwad et al., 2020b] presented the temperature distribution of parts, predicted based on a graph–theoretical computational heat transfer approach and subsequently combined it with an SVM model in order to detect potential quality faults in printing processes. Another attempt of grey box modeling for build quality in dependency of process parameters and in situ sensor signatures was proposed in [Gaikwad et al., 2020a] by Gaikward et al., where the a priori knowledge of physical processes was incorporated into three sequentially connected shallow ANNs and consequently achieved better performance in comparison with purely data-driven methods (CNN, LSTM, RNN, among others). With respect to DfX and

knowledge engineering, Ko et al. employed CART to predict additive manufacturability, which was further fed back to a knowledge-query formulation phase in order to continuously construct and broaden an AM knowledge base [Ko et al., 2019] (*EG*-factor). In addition, HMM and k-means demonstrated their applications for quality assessing and monitoring in [Stavropoulos et al., 2020, Stojanovic and Milenovic, 2019].

C. Composite Material Processing For similarly novel but not yet matured *composite material processing* techniques, e.g., lightweight production of fiber-reinforced polymers, hybrid modeling of non-measurable process variables and process signatures refined therefrom, are anticipated to provide more process understanding and transparency (*EG*-factor). Stieber et al. proposed a CNN-based transfer learning approach to in situ monitor the polymerization progress of resin transfer molding (RTM) [Stieber et al., 2020]. Hürkamp et al. implemented AdaBoost, XGBoost, and random forest as finite element surrogate models to predict temperature distribution during the fabrication of overmolded thermoplastic composites [Hürkamp et al., 2020]. Similarly, DNN was applied by Pfrommer et al. as surrogate modeling to optimize the manufacturing process parameters of a composite textile draping process [Pfrommer et al., 2018]. A probabilistic ML approach with statistical inference was developed by Ghanem et al. to efficiently update numeric simulations of a composite material system and reveal multi-scaling relationships of mechanical properties and behaviors [Ghanem et al., 2020]. Kanyuck et al. introduced a methodology for identifying sheet material parameters using the ISRES algorithm and implemented a thin-shell simulator for predicting the material behavior, which enabled a defect-free layup of prepreg composite sheets in human–robot collaborative cells [Chen et al., 2021] (*ES*-factor).

Interim Summary

AI-driven digital process twins are envisioned to learn and interpret implicit correlations between manufacturing processes and material/process/environmental parameters from an

aggregation of (heterogeneous) data with the objective of optimizing process development, production ramp-up, and quality assurance cycle. From an engineering implementation perspective, we have noted that despite the significance of algorithms and models, novel sensor technologies [Brecher et al., 2019b, Denkena et al., 2017, Min et al., 2020, Postel et al., 2019, Mayes et al., 2019, Schäkel et al., 2018] and networked digital process chains [Kwon et al., 2020, Ramnath et al., 2020, Liu et al., 2020b, Bonnard et al., 2019] and [Ganser et al., 2021] should not be neglected, as they are essential pillars for constructing DTs and can considerably influence the effectiveness and efficiency of their development and deployment in practice. Figure 6.5 shows an example of a DT dynamically mapping the manufacturing process of an aerospace part and the data sources involved from the contextualized CAD-CAM-CNC-CAQ process chain. As yet, these immature manufacturing techniques, such as 3D printing and lightweight production of metals and composites, are still set up in a trial-and-error manner [Zhang et al., 2020c]. Their maturation will not only lead to innovative concepts (SG-factor) and resource savings (E-factor) in the design and manufacturing phase, but also provides a pivotal basis for the sustainable operation, maintenance, and reuse of the product downstream, thus reducing energy consumption and carbon emissions across the product lifecycle. The resulting sustainability upgrades are, therefore, all-encompassing.

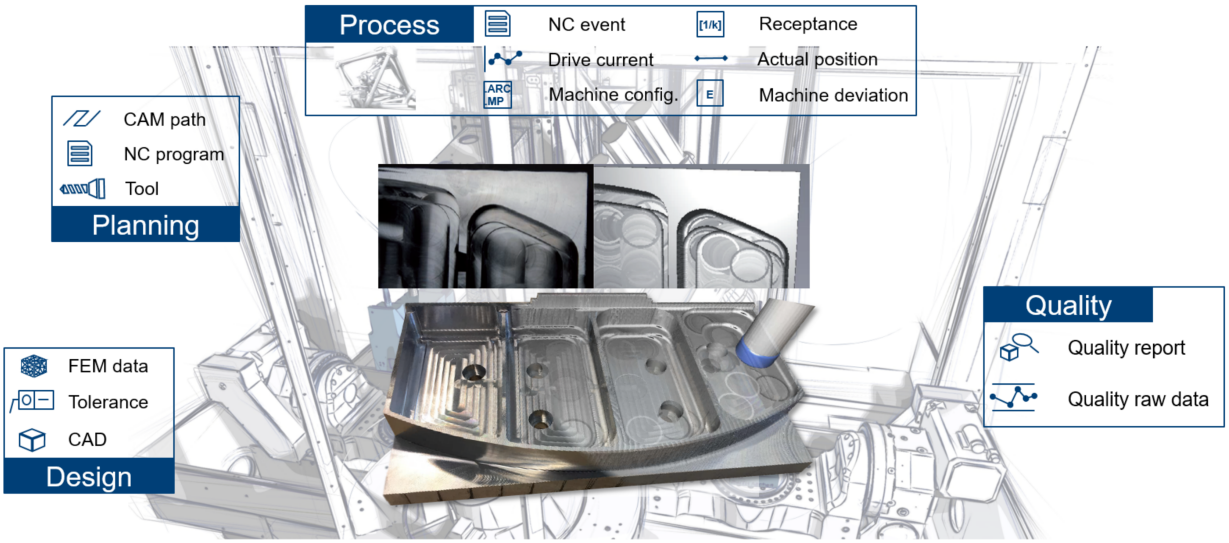


Figure 6.5: An illustration of DT in manufacturing: workpiece quality monitor [Brecher et al., 2017]. © WZL—RWTH Aachen.

6.2.5 Challenges and Outlook

Key practical challenges and future work are included as follows.

1. Vertical interoperability in the production context (*basis/infrastructure*): despite various proposed concepts and cases, a standard framework for developing and deploying multi-scale DTs combining with AI methods has not yet been established. A valid reference framework covering separate manufacturing levels should be defined in the future. For this purpose, interdisciplinary cooperation is indispensable [Dittrich et al., 2020].

2. Horizontal interoperability in the production context (*basis/data*): sufficiently high data quality is a fundamental prerequisite for building AI-driven DTs. In the production context, however, the usage of heterogeneous data sources from different software and hardware across the product life cycle is associated with high costs. Through consistently and comprehensively contextualizing and interconnecting of these data silos, the full potential of DTs can be exploited in terms of further optimizing the agility, traceability, resiliency, and transparency of current manufacturing. Interfaces for standard information exchange are, therefore, imperative [Wagner et al., 2019].

3. High-fidelity, lightweight models with uncertainty quantification (*core*): real manufacturing processes are characterized by numerous, partly stochastic variables with highly complicated cause–effect relationships. In the course of increasing product variants driven by the market, frequently changing technical boundary conditions bring more parameterization efforts and potential uncertainties. Conventional, complex and non-real-time capable models are, therefore, gradually reaching their limits in the industrial environment. In this sense, data-driven approaches with the incorporation of prior knowledge promise to extend the current model boundaries, thereby improving the industrial objectives in terms of productivity, availability, and quality (F-factor).

4. Smart services and business models (*application*): novel digital platforms contribute to long-range collaboration and innovation, ensembling isolated AI-equipped DT-solutions from diverse levels, and thus offering new opportunities for manufacturers to deliver products and solutions sustainably, e.g., through the X-as-a-Service (XaaS) model. Moreover, the transparency distilled from the entire product value chain serves as an on-demand and real-time capable analytical foundation for the holistic assessment of sustainable resilient manufacturing in order to achieve a balance in the *financial*, *environmental*, *social*, and *governance* dimensions. The concept of "sustainable productivity" based thereon is depicted in Figure 6.6 selected from the white paper by Boos [Boos, 2021]. Therein, the vision of "Internet of Production" provides the infrastructure for harnessing data along the product lifecycle [Schuh et al., 2017], while the transparency generated from the (AI-powered) DTs enables product design, manufacturing, and usage exclusively, according to actual demand-, quantity-, and user-oriented requirements [Boos, 2021].

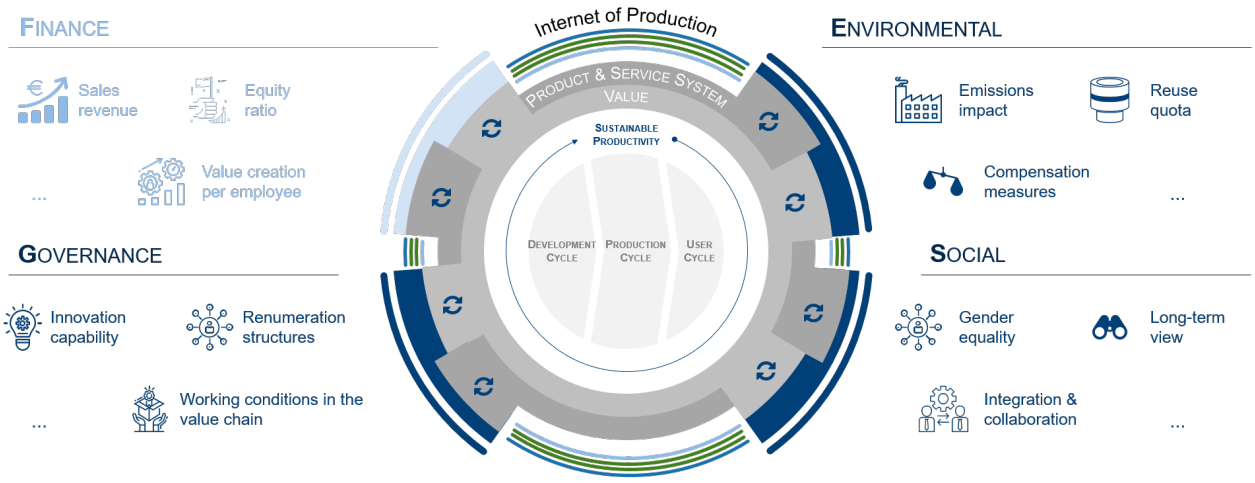


Figure 6.6: An illustration of sustainable resilient manufacturing [Boos, 2021]. © WZL—RWTH Aachen.

6.3 Advanced Robotics

6.3.1 Overview

Along with the widespread deployment of robotic systems in industry and daily life [Shen et al., 2020], having a digital twin of robots becomes more and more critical in practical scenarios, such as multi-robot coordination/collaboration as well as those that require safe human–robot interaction (HRI) and/or complex human–robot collaboration (HRC) [Cichon and Rosmann, 2017, Pairet et al., 2019, Malik and Brem, 2020] which place human safety as a high priority, thus helping to create a sustainable working environment (*SG*-factor). Many have used traditional simulation/cloud framework to attempt robot DT implementations, such as the ones shown in Figures 6.7 and 6.8. Other examples could be found in kinematics [KUTS et al., 2020], communication [Liang et al., 2020, Girletti et al., 2020], control [Kuts et al., 2019, Li et al., 2019, Xu et al., 2020], planning [Larsen, 2019], and industrial robot energy modeling [Yan et al., 2018], in use cases like welding [MPOC, 2020], cleaning [Burghardt et al., 2020], pick-and-place [Tipary and Erdős, 2021], assembly [Hoebert et al., 2019, Kousi et al., 2019, Meng et al., 2019, Pérez et al., 2020], manufacturing [He et al., 2021a], warehouse [Fedotov et al., 2020, Filocamo, 2020], maintenance [Franko et al.,

2020], and construction [Cai et al., 2020]. Some well-known robotics simulation tools are Gazebo [Agüero et al., 2015], MuJoCo [Todorov et al., 2012], and CoppeliaSim (aka V-REP) [Rohmer et al., 2013]. Recently, new concepts and cases utilizing artificial intelligence towards semi- and fully autonomous robotic systems have been reported, e.g., transfer learning [Maschler et al., 2020] and imitation learning (also known as apprenticeship learning or learning from demonstration) [Verner et al., 2017, Liang et al., 2019]. While traditional DTs have been developed for systems that we have a solid grasp of (in other words, *model-based*), data-driven and AI-equipped DTs help with complex robotic systems for which building high-fidelity dynamics models is not feasible (*model-free*). The latter has been applied in more and more cases, even for biomimetic robotic system development (e.g., robotic fish [Yu et al., 2016]). Table 6.4 summarizes AI-equipped DTs and categorizes them based on learning algorithms used and subfields, such as control (detailed in Section 6.3.2), planning (detailed in Section 6.3.3), and HRI/HRC (detailed in Section 6.3.4).

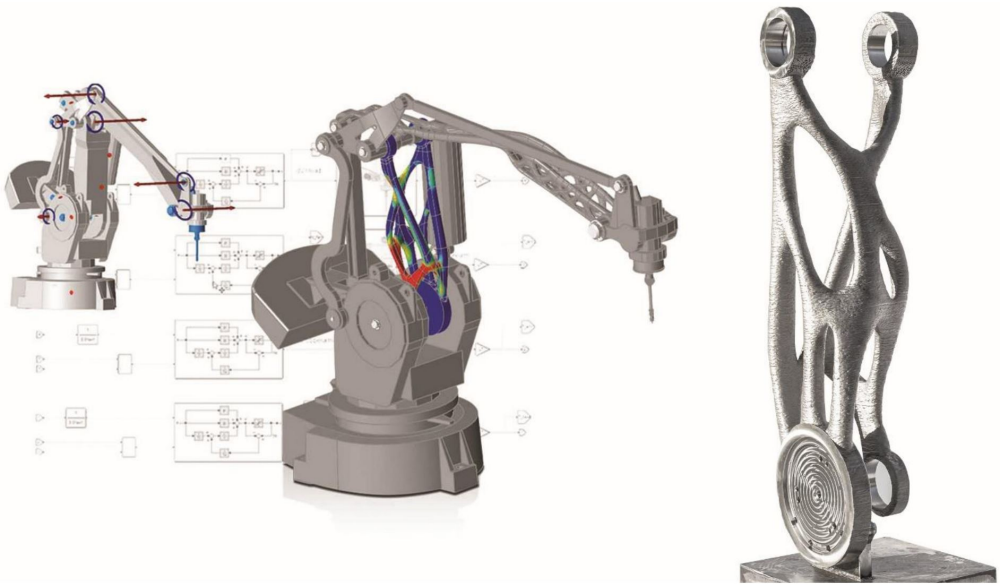


Figure 6.7: An illustration of DT in robotics: in a joint project MX3D, ABB, and Altair demonstrated how a 3D-printed robot can be improved by using a digital twin process to achieve more precise positioning. ©Altair.

Table 6.4: Summary of AI-enabled DTs in advanced robotics.

Subfield	**AI Category**	**Key Methods**	**Application-Case**	**Ref.**
Control (Section 6.3.2)	Supervised and Unsupervised Learning	SVM, PCA	Object recognition of a smart gripper	[Jin et al., 2020]
	Reinforcement learning	Trial-and-error search	Weightlifting robot control	[Verner et al., 2018]
	Supervised Learning	GD	Understanding the added value of integrated models for through-life engineering services	[Vrabič et al., 2018]
	Computational Intelligence	Vision-based Markovian chain	Automate fan-blade reconditioning for aerospace maintenance, repair and overhaul	[Oyekan et al., 2020]
		QP	Supporting rescuers on disaster-response missions	[Klamt et al., 2019]
Planning (Section 6.3.3)	Reinforcement Learning	Proximal policy optimization	Pick-and-place tasks for an industrial robotic arm	[Matulis and Harvey, 2021]
		DDPG	Control and trajectory planning of a planar 3-DOF manipulator and 3D arms of a humanoid robot	[Liu et al., 2020a]
		DQN	Automate smart manufacturing systems	[Xia et al., 2020]
		Proposed LSTM-MACG	Collision avoidance for a number of UAVs in a confined airspace	[Zhao et al., 2021]
	Computational Intelligence	Ant colony optimization	Path planning of industrial robots	[Bansal et al., 2019]

Table 6.4: *Cont.*

Subfield	AI Category	Key Methods	Application-Case	Ref.
HRI/HRC (Section 6.3.4)	Supervised Learning	CNN	Standing-posture recognition in HRC	[Li et al., 2020a]
		DL	Mechatronics system	[Cichon and Robmann, 2018]
		ANN	Enabling industrial robots to bypass obstacles	[Dröder et al., 2018]
		LSTM	Visual question answering for HMC system	[Wang et al., 2020c]
	Supervised and Unsupervised Learning	FFT-PCA-SVM	HRI welding and welder behavior analysis (identifying the professional level)	[Wang et al., 2021]
	Reinforcement Learning	DDPG	COVID-19, improve efficiency in assembling medical equipment	[Lv et al., 2021]
Predictive Maintenance	Supervised Learning	DNN	System health monitoring	[Anton et al., 2020]
			Maximizing the overall plant availability of modern manufacturing systems	[Aivaliotis et al., 2021]
Workspace Modeling	Supervised Learning	Monte Carlo method	Simulating the workspace of the mechanisms	[He et al., 2020]
Others	Supervised Learning	RF	Estimation of lawn grass lengths for robotic lawn mower	[Zushida et al., 2020]

PCA: Principal Component Analysis; GD: Gradient Descent; QP: Quadratic Programming; MACG: MultiAgent Computational Guidance; FFT: Fast Fourier Transform; UAV: Unmanned Aerial Vehicle.

6.3.2 Control

A key part of modern robotic control is feedback, which expects accurate information collected from physical sensors installed on robots and in the external environment and to provide commands for the next loop of execution. Sometimes, real time is required to enable safer controllers on these robotic systems. Many efforts utilizing AI + DT have been attempted in this subfield. Compared with traditional DTs, those equipped with AI and driven by data have advantages of gaining better generalizability and higher adaptability in a varying environment, and accomplishing nontrivial sensing/manipulation tasks. At the *sensing* stage, Jin et al. reported a smart soft-robotic gripper system based on triboelectric nanogenerator sensors to capture the continuous motion and tactile information for soft gripper control, where PCA and SVM were used to realize real-time prediction [Jin et al., 2020]. Data- or AI-driven

approaches can also be found in other touch/haptic/force sensing for obtaining better system understanding and task performance [Kapusta et al., 2016, Piacenza et al., 2020, Santina et al., 2020]. One level higher, at the *controller* stage, Verner et al. implemented online reinforcement learning via a fabricated digital twin, to enable a humanoid robot to lift a weight of unknown mass through autonomous trial-and-error search [Verner et al., 2018]. Similarly, in [Grinshpun et al., 2016], Grinshpun et al. reported the development and deployment of control algorithms for soft robots, with particular reference to industrial peg-in-hole insertion tasks. Vrabič et al. used DT and gradient descent to optimize controller parameters of a mobile robot [Vrabič et al., 2018]. More data/AI-driven examples in robot control include [Reinhart et al., 2017, Lyu and Cheah, 2020]. At the *application* level, one example is that Oyekan et al. utilized vision-based Markovian chain to automate fan-blade reconditioning for aerospace maintenance, repair and overhaul with a 6DOF robotic arm [Oyekan et al., 2020] (E-factor). Another example is that Klamt et al. built a DT for the famous CENTAURO robotic system to support rescuers on disaster response missions [Klamt et al., 2019] (S-factor).

6.3.3 Planning

Once the low-level robotic control is in good shape, the high-level robotic planning, as another critical part of realizing autonomous robotic systems, comes into play. Unlike the low-level control subfield, which emphasizes the system response and robustness, high-level planning focuses more on strategically finding a close-to-optimal solution among all feasible options, under specific constraints. Compared with traditional search-based motion planning algorithms, reinforcement learning has demonstrated huge potentials in bringing intelligence to complex systems planning, e.g., a humanoid robot with high degrees of freedom (DOFs) [Frank et al., 2014]. However, it is usually difficult to train reinforcement learning because obtaining the data from the real physical system is both finance- and time-consuming. The curse of dimensionality may also disable the system from learning something useful [Bengio et al., 2013].

As a robotic system's digital twin grows up and provides reliable data, the combination "DT + RL" seems to be a promising approach (*SG*-factor): in [Matulis and Harvey, 2021], Matulis et al. integrated digital twin and reinforcement learning for a 6DOF robotic manipulator to plan pick-and-place motions; in [Liu et al., 2020a], Liu et al. proposed a multitasking-oriented robot arm motion planning scheme based on deep reinforcement learning and twin synchro-control; in [Xia et al., 2020], Xia et al. proposed a DT to train deep reinforcement learning agent for automating smart manufacturing systems; and in [Zhao et al., 2021], Zhao et al. demonstrated collision avoidance for a number of UAVs in a confined airspace, using LSTM-MACG. While RL has become popular in recent years, it is not the only AI strategy that is integrated with DTs. For example, Bansal et al. developed an ant colony optimization algorithm for industrial robot programming in a digital twin [Bansal et al., 2019].

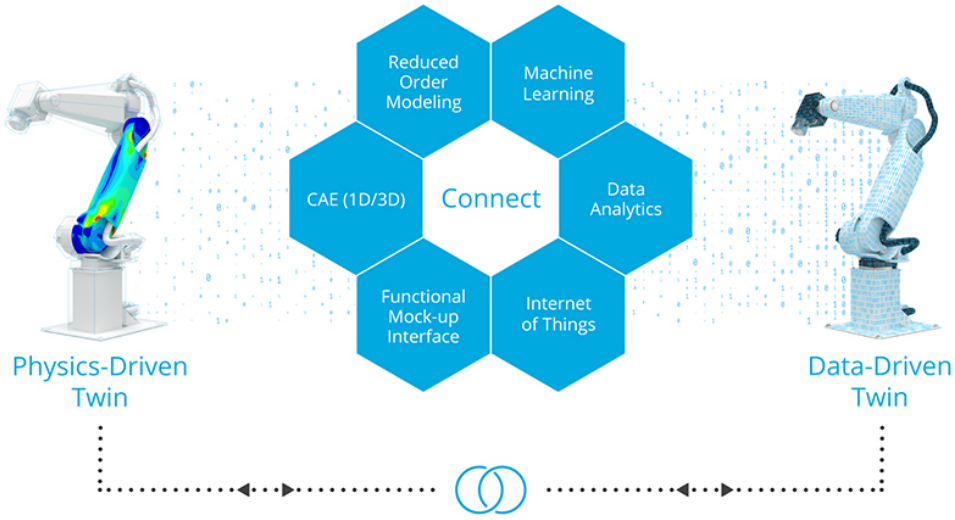


Figure 6.8: An illustration of DT in robotics: Altair digital twin platform. ©Altair.

6.3.4 HRI and HRC

Under the context of Industry 4.0, one of the very important benefits that DT could bring to robot-involved scenarios is safer human–robot interaction and human–robot collaboration [Bil-

berg and Malik, 2019, Malik and Brem, 2021] (*SG*-factor). HRI and HRC scenarios are, intuitively, more complex and challenging than robot-only applications, due to not only the uncertainties in the environment and sensors, but also the randomness and diversity of human behavior. One of the advantages of AI-equipped DTs over conventional ones is the ability to better respond to these (sometimes implicit, such as [Admoni and Scassellati, 2014]) variables that are nontrivial to exhaustively model and analyze. For example, in [Wang et al., 2020d], Wang et al. proposed a real-time process-level digital twin for collaborative human–robot construction work. The proposed DT utilized immersive virtual reality (VR) and combined the as-designed BIM model and the evolving as-built workspace geometry obtained from on-site sensors, to provide the capability for both planning and improvising. Similarly, in [Li et al., 2020a], Li et al. proposed DL-based human standing-posture recognition in HRC. Other supervised learning applications, such as visual question answering for the HMC system, are included in Table 6.4 [Cichon and Robmann, 2018, Dröder et al., 2018, Wang et al., 2020c]. A supervised/unsupervised learning example is [Wang et al., 2021], where Wang et al. used FFT–PCA–SVM–based DT for human–robot interactive welding and welder behavior analysis. In [Lv et al., 2021], Lv et al. proposed a reinforcement learning–based DT for improving medical equipment assembly efficiency during COVID-19.

6.3.5 Robot Maintenance and Other Applications

Robotic systems, like other machines with a physical entity, have downtime and need maintenance as well (*E*-factor). Without explicitly mentioning the concept of DT, Khalastchi et al. and Vallachira et al. reported applications of using data-driven methods in robot anomaly/failure detection in [Khalastchi et al., 2015] and [Vallachira et al., 2020], respectively. In [Anton et al., 2020], Anton et al. used deep learning equipped DT for global system health monitoring as well as predictive, customized maintenance. Similarly, in [Aivaliotis et al., 2021], Aivaliotis et al. integrated degradation curves in the predictive maintenance of

industrial robots. Some other applications, such as using the Monte Carlo learning method in calculating the workspace of a serial robot manipulator [He et al., 2020] and random forest based estimation of lawn grass lengths for robotic lawn mower [Zushida et al., 2020], can be found in Table 6.4.

6.3.6 Challenges and Outlook

There are several key challenges in developing and implementing digital twins in the field of robotics [Schluse and Rossmann, 2016, Stehling et al., 2017]. First, the multibody physical simulation is intrinsically difficult, due to the complex interaction properties at the interfaces of robot–robot, robot–human, and robot–environment. In addition, since robot movement can often be extremely fast (e.g., in an assembly line), real-time feedback from sensors is critical to the digital twin's effectiveness in making short-term decisions. Many researchers in academia choose to use simulators, such as Gazebo (e.g., [Verner et al., 2019, Verner et al., 2020]), as the simulation environment for developing robots, but even after years of evolution, those robotic simulators still have many unsolved limitations and may require high-performance computing (HPC) platforms. Second, inputs and disturbances from the human user add another layer of uncertainty and unpredictability to the whole collaborative system/workspace, and thus, compromise HRI/HRC safety. While standard-compliant (e.g., ISO 13482) safety measures must be facilitated on both physical and digital sides, having virtual reality or augmented reality (AR) technologies involved is another thread to make the human–robot interaction intuitive [Havard et al., 2019]. Rückert et al. also suggested the consolidation of product life cycle information within human–robot collaborative assembly tasks [Rückert et al., 2020].

6.4 Discussion

DTs have shown remarkable potentials to contribute to industrial economic growth or F-factor (i.e., the productivity, availability, and quality of manufacturing) while continuously upgrading sustainable aspects, such as the E-factor (e.g., reduced carbon emission and resource consumption through CM, PdM, 3D printing and lightweight production of metals and polymers) and SG-factor (e.g., enhanced working conditions, collaboration and innovation through HRI/HRC, XaaS model, and intelligent/soft sensing of novel production indicators) as summarized in Table 6.5. In this range of cross-domain engineering cases, AI techniques arm digital twins with tools to create models based on observed behavior and historical data, which improves the efficiency of data analysis and increases prediction accuracy by integrating data from a collection of disparate and incompatible sources.

Table 6.5: **Summary of AI-enabled DTs in sustainable development and FESG factors.**

F-Factor	**E-Factor**	**SG-Factor**
Productivity (Section 6.2.2)	Production Planning (Section 6.2.2 A)	Intelligent Sensing of Novel Indicators (Sections 6.2.2, 6.2.3, and 6.2.4)
	Production Control (Section 6.2.2 B)	
	Quality Control (Section 6.2.2 C)	
Availability (Section 6.2.3)	Condition Monitoring (Section 6.2.3 A)	Innovative Robot Planning and Control (Sections 6.3.2 and 6.3.3)
	Predictive Maintenance (Section 6.2.3 B)	
	Dynamics and Control (Section 6.2.3 C)	HRI/HRC (Section 6.3.4)
	Robot Maintenance (Section 6.3.5)	
Quality (Section 6.2.4)	Metal Cutting (Section 6.2.4 A)	DfX (Sections 6.2.4 A and 6.2.4 B)
	Metal Additive Manufacturing (Section 6.2.4 B)	XaaS and Business Model (Section 6.2.2)
	Composite Material Processing (Section 6.2.4 C)	

The AI techniques involved in digital twins can be roughly categorized into four classes: supervised learning, unsupervised learning, reinforcement learning and other intelligent computational methods. Supervised learning algorithms refer to machine learning methods in which models are trained using labels. Typical supervised learning methods used in digital twin include supper vector machine (SVM) [Gaikwad et al., 2020b, Zhao et al., 2020], decision trees [Rojek et al., 2021, Min et al., 2019, Singgih, 2021], k-nearest neighbors [Yacob et al., 2019], convolutional neural networks (CNN) [Alexopoulos et al., 2020], [Yiping et al., 2021], [Gaikwad et al., 2020a], [Stieber et al., 2020], [Li et al., 2020a] and recurrent neural networks (RNN) [Gaikwad et al., 2020a]. In practice, data labeling can be an expensive task. Most supervised learning algorithms require a large amount of labeled data at the training stage to obtain a model with high prediction accuracy. In general, the more complex the architecture is, the more data are needed to produce viable results. The results of supervised learning algorithms also depend on the selection of feature vectors and the accuracy of labeling.

In unsupervised learning methods, there is no labeling of data required, and the model is expected to infer patterns from the unlabeled input data. Clustering algorithms, such as principle component analysis (PCA) [Jin et al., 2020, Wang et al., 2021] and k-means methods [Stojanovic and Milenovic, 2019] and generative models using generative adversarial network (GAN) [Booyse et al., 2020, Zotov et al., 2020] and variational autoencoders (VAE) [Booyse et al., 2020] all use unlabeled data at the training stage, thus falling into the category of unsupervised learning. One of the challenges in applying unsupervised learning methods is that the number of clusters is normally not known a priori. For clustering algorithms, the clusters are determined by the metric used to measure similarity—Euclidean, cosine, Gaussian distance—of which the criteria are not clear for a given task. Reinforcement learning algorithms are concerned with how intelligent agents ought to take actions in an environment in order to maximize the notion of cumulative reward. Researchers have applied reinforcement learning algorithms, including Q-learning [Jaensch et al., 2019a], deep reinforcement learning [Cronrath et al., 2019, Waschneck et al., 2018, Xia et al., 2020]

and deep deterministic policy gradient [Liu et al., 2020a, Lv et al., 2021, Zhou et al., 2021] to optimize the decision-making process of box sorting, conveyor systems and other DT scenarios. The performance of a reinforcement learning system generally heavily depends on the correctness of data logging and the choice of reward structures. Logging to incorrect references might corrupt the information and lead the whole system to break down during training.

From the above, the fidelity of AI-driven DTs depends largely on the model granularity, the selection of which (*core* level) is, in turn, related to the environment complexity (*application* level) and dataset quality (*basis* level). Based on the review of over 300 manuscripts, we sketched a route for AI-integration in multiscale/fidelity DTs in the real-world case of multiscale/fidelity data sources, as outlined in Figure 6.9. This multiscale nature is reflected in both spatial and temporal terms. Vertically, the demand for real-time capability in digital twins alters with different levels of the automation pyramid. While the higher-scale planning and management typically require a longer response time to deal with changing production environments, time-sensitive sensing and manipulation rely on the in-depth insight of physical processes gained through refining scale modeling. Horizontally, datasets of varying difficulty and fidelity are constructed from heterogeneous sources over the product lifecycle. In the development phase, simulation tools and laboratory experiments separately provide comparatively large amounts of data with a moderate degree of fidelity and limited high-accuracy data under controlled conditions. As products are manufactured and utilized, obtaining reliable labeled data, such as quality indicators and RUL, becomes increasingly scarce and expensive. On this basis, AI methods can be developed and deployed in pipelines. In general, supervised learning remains the most stable and extensively used approach in digital twins. Reinforcement learning is often advantageous in scenarios with complex environments and long response cycles, unsupervised and semi-supervised learning are quite proactive in state detection and lifetime prediction, and traditional intelligence methods are still commonly used and can be flexibly combined with other learning methods.

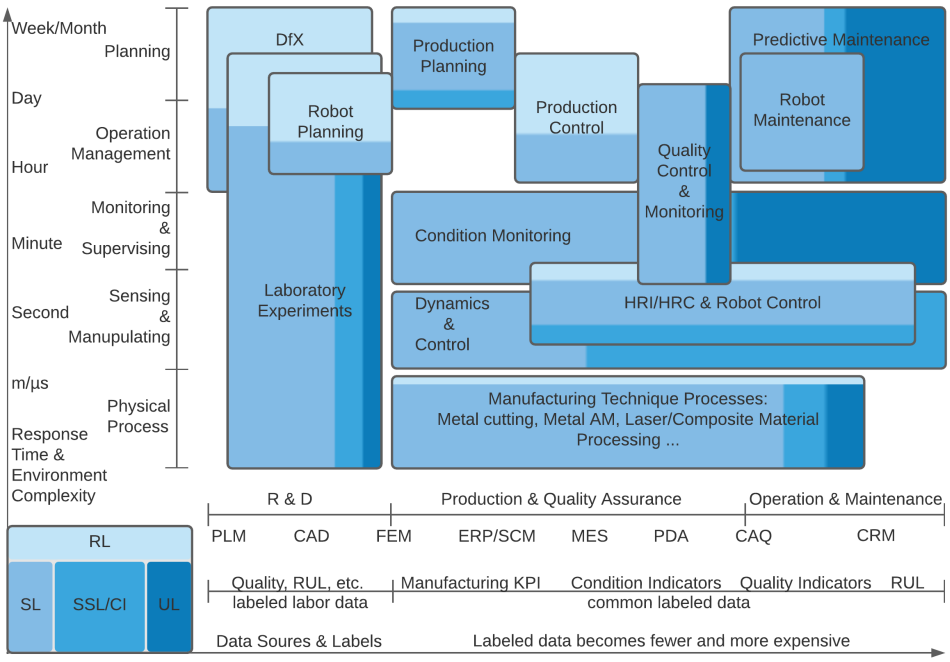


Figure 6.9: An illustration of AI-integration in multiscale/fidelity DTs along the product lifecycle.

6.5 Summary

Traditional profit-maximizing industrial technologies have transformed human society while causing significant—and mostly irreversible—negative impacts on the environment and climate. Sustainable development strategies are gaining attention, but still have a long way to go, as all aspects within the *FESG* dimensions have to be balanced in real implementation. According to a report from the WBA Tooling Academy Aachen [Boos et al., 2020], classic evaluation for manufacturing companies with a focus on value added and equity ratio could show a significantly worse outcome in the extended analysis that incorporates *ESG* criteria in the balance sheet. For sustainable thinking and practices, which have been increasingly deliberated and advocated worldwide, digitalization and AI are powerful enablers. Through consistently linking, processing, and analyzing all available data across the entire value chain, and connecting them to a digital twin or shadow, a holistic assessment of sustainability,

namely, the consideration of *ESG* factors alongside profitability (*F*-factor), can be achieved so that companies can develop and operate sustainably. For Industry 4.0, which is facing new challenges from climate-neutral products and production, AI-driven DTs are expected to provide the additional manufacturing transparency and understanding that enable a demand-oriented and real-time capable analytical foundation, considering *FESG* factors along the product lifecycle. The main contributions of this study are concluded as follows.

1. The general development and application cases with common AI methods in AI-driven DTs of Industry 4.0 are concluded.

2. The advantages of AI-driven DTs in sustainable development are elaborated regarding the *FESG* factors, which enable a quantitative assessment of sustainability.

3. Challenges and development prospects of AI-driven DTs in smart manufacturing and advanced robotics are discussed with a respective focus on different levels.

4. A route for AI integration in multiscale/fidelity DTs with multiscale/fidelity data sources along the product lifecycle is outlined.

In the past, typical production issues, such as predicting the behavior of machine tools as single systems, were already well approximated by complex analytical and empirical models; their application in industry environments, however, has often been handicapped, due to both the lack of real-time capabilities and transferability in varying frameworks. AI promises in this circumstance to extend the model boundaries of traditional model-based approaches, particularly within changing boundary conditions. As regards the relatively sparse and expensive datasets in engineering, incorporating prior knowledge can reduce the dependence of pure data-driven approaches on the amount of historical data and improve the predictivity and transferability of models. Moreover, statistically-based uncertainty quantification similarly plays an essential role in building AI-driven DTs, as it allows to assess the reliability of the modeled results and, thus, influences their acceptance and the decision-making based thereon in real, high-risk engineering.

Meanwhile, we have noted that the development and deployment of AI-enabled algorithms

and models, the *core* of the DT, are still constrained by the current infrastructure and that constructing the latter requires interdisciplinary collaboration and integration of domain-specific expertise (*basis* level). New breakthroughs in novel sensors and benefits from 5G communications and OPC UA TSN are expected in the near future. In terms of the *application* level, while smart service and new business models facilitate the paradigm shift for manufacturers, a prerequisite is the willingness to share data and knowledge with partners to a healthy extent. Standardized concepts for data ownership and data security must constitute the basis for this. A further survey on the AI-driven DT technologies in the application fields of renewable energy, smart city and mobility, and healthcare will be covered in future work. By reorganizing and aggregating several highly relevant topics both horizontally and vertically, we believe that a synergistic effect will emerge that can allow the work in this study to contribute to more AI-driven, DT-related research and help various branches build new developments in their respective sustainable and smart areas.

Bibliography

[Admoni and Scassellati, 2014] Admoni, H. and Scassellati, B. (2014). Data-driven model of nonverbal behavior for socially assistive human-robot interactions. *ICMI 2014 - Proceedings of the 2014 International Conference on Multimodal Interaction*, pages 196–199.

[Afrasiabi et al., 2020] Afrasiabi, M., Meier, L., Röthlin, M., Klippel, H., and Wegener, K. (2020). GPU-accelerated meshfree simulations for parameter identification of a friction model in metal machining. *International Journal of Mechanical Sciences*, 176(February):105571.

[Agüero et al., 2015] Agüero, C. E., Koenig, N., Chen, I., Boyer, H., Peters, S., Hsu, J., Gerkey, B., Paepcke, S., Rivero, J. L., Manzo, J., Krotkov, E., and Pratt, G. (2015). Inside

the Virtual Robotics Challenge: Simulating Real-Time Robotic Disaster Response. *IEEE Transactions on Automation Science and Engineering*, 12(2):494–506.

[Aivaliotis et al., 2021] Aivaliotis, P., Arkouli, Z., Georgoulias, K., and Makris, S. (2021). Degradation curves integration in physics-based models: Towards the predictive maintenance of industrial robots. *Robotics and Computer-Integrated Manufacturing*, 71:102177.

[Aivaliotis et al., 2019] Aivaliotis, P., Georgoulias, K., and Chryssolouris, G. (2019). The use of Digital Twin for predictive maintenance in manufacturing. *International Journal of Computer Integrated Manufacturing*, 32(11):1067–1080.

[Alexopoulos et al., 2020] Alexopoulos, K., Nikolakis, N., and Chryssolouris, G. (2020). Digital twin-driven supervised machine learning for the development of artificial intelligence applications in manufacturing. *International Journal of Computer Integrated Manufacturing*, 33(5):429–439.

[Alguri et al., 2021] Alguri, K. S., Chia, C. C., and Harley, J. B. (2021). Sim-to-Real: Employing ultrasonic guided wave digital surrogates and transfer learning for damage visualization. *Ultrasonics*, 111(November 2020):106338.

[Anis et al., 2020] Anis, M. D., Taghipour, S., and Lee, C. G. (2020). Optimal RUL estimation: A state-of-art digital twin application. *Proceedings - Annual Reliability and Maintainability Symposium*, 2020-Janua.

[Anton et al., 2020] Anton, F., Borangiu, T., Răileanu, S., and Anton, S. (2020). Cloud-Based Digital Twin for Robot Integration in Intelligent Manufacturing Systems. *Mechanisms and Machine Science*, 84:565–573.

[Armendia et al., 2019] Armendia, M., Cugnon, F., Berglind, L., Ozturk, E., Gil, G., and Selmi, J. (2019). Evaluation of machine tool digital twin for machining operations in industrial environment. *Procedia CIRP*, 82:231–236.

[Asadi et al., 2020] Asadi, M., Mohseni, M., Golkhosh, F., Kashani, M. T., Fernandez, M., and Smith, M. (2020). A Hybrid Digital-Twin Platform for Sequence Design in Welded Structures. In *Proceedings of the ASME 2020 Pressure Vessel & Piping Conference PVP2020*, pages 1–10.

[Autiosalo et al., 2021] Autiosalo, J., Ala-Laurinaho, R., Mattila, J., Valtonen, M., Peltoranta, V., and Tammi, K. (2021). Towards integrated digital twins for industrial products: Case study on an overhead crane. *Applied Sciences (Switzerland)*, 11(2):1–35.

[Bansal et al., 2019] Bansal, R., Khanesar, M. A., and Branson, D. (2019). Ant colony optimization algorithm for industrial robot programming in a digital twin. *ICAC 2019 - 2019 25th IEEE International Conference on Automation and Computing*, September:5–7.

[Bauernhansl, 2016] Bauernhansl, T. (2016). *WGP-Standpunkt Industrie 4.0*. WGP, Wissenschaftliche Gesellschaft für Produktionstechnik.

[Bauernhansl et al., 2018] Bauernhansl, T., Hartleif, S., and Felix, T. (2018). The Digital Shadow of production - A concept for the effective and efficient information supply in dynamic industrial environments. *Procedia CIRP*, 72:69–74.

[Bayer et al., 2021] Bayer, B., Diaz, R. D., Melcher, M., Striedner, G., and Duerkop, M. (2021). Digital twin application for model-based doe to rapidly identify ideal process conditions for space-time yield optimization. *Processes*, 9(7).

[Bengio et al., 2013] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

[Bergs et al., 2019] Bergs, C., Heizmann, M., Hartmann, D., and Carillo, G. L. (2019). Novel method for online wear estimation of centrifugal pumps using multi-fidelity modeling.

Proceedings - 2019 IEEE International Conference on Industrial Cyber Physical Systems, ICPS 2019, pages 185–190.

[Bibow et al., 2020] Bibow, P., Dalibor, M., Hopmann, C., Mainz, B., Rumpe, B., Schmalzing, D., Schmitz, M., and Wortmann, A. (2020). Model-Driven Development of a Digital Twin for Injection Molding. In Dustdar, S., Yu, E., Salinesi, C., Rieu, D., and Pant, V., editors, *Advanced Information Systems Engineering*, pages 85–100, Cham. Springer International Publishing.

[Biesinger et al., 2018] Biesinger, F., Meike, D., Kras, B., and Weyrich, M. (2018). A Case Study for a Digital Twin of Body-in-White Production Systems General Concept for Automated Updating of Planning Projects in the Digital Factory. *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2018-Septe:19–26.

[Bilberg and Malik, 2019] Bilberg, A. and Malik, A. A. (2019). Digital twin driven human–robot collaborative assembly. *CIRP Annals*, 68(1):499–502.

[Blume et al., 2020] Blume, C., Blume, S., Thiede, S., and Herrmann, C. (2020). Data-driven digital twins for technical building services operation in factories: A cooling tower case study. *Journal of Manufacturing and Materials Processing*, 4(4).

[Bonnard et al., 2019] Bonnard, R., Hascoët, J. Y., Mognol, P., Zancul, E., and Alvares, A. J. (2019). Hierarchical object-oriented model (HOOM) for additive manufacturing digital thread. *Journal of Manufacturing Systems*, 50(October 2018):36–52.

[Boos, 2021] Boos, W. (2021). Die Produktionswende – Turning Data into Sustainability. White Paper.

[Boos et al., 2020] Boos, W., Kelzenberg, C., Helbig, J., Busch, M., Graberg, T., and Schweins, J. (2020). Wettbewerbsfaktor Nachhaltigkeit: Ein Differenzierungsmerkmal für den Werkzeugbau.

[Booyse et al., 2020] Booyse, W., Wilke, D. N., and Heyns, S. (2020). Deep digital twins for detection, diagnostics and prognostics. *Mechanical Systems and Signal Processing*, 140:106612.

[Borangiu et al., 2020] Borangiu, T., Oltean, E., Răileanu, S., Anton, F., Anton, S., and Iacob, I. (2020). Embedded Digital Twin for ARTI-Type Control of Semi-continuous Production Processes. In *Studies in Computational Intelligence*, volume 853, pages 113–133. Springer.

[Bordatchev et al., 2019] Bordatchev, E., Cvijanovic, S., and Tutunea-Fatan, O. R. (2019). Effect of initial surface topography during laser polishing process: Statistical analysis. *Procedia Manufacturing*, 34:269–274.

[Botkina et al., 2018] Botkina, D., Hedlind, M., Olsson, B., Henser, J., and Lundholm, T. (2018). Digital Twin of a Cutting Tool. *Procedia CIRP*, 72:215–218.

[Brecher et al., 2019a] Brecher, C., Buchsbaum, M., and Storms, S. (2019a). Control from the cloud: Edge computing, services and digital shadow for automation technologies. *Proceedings - IEEE International Conference on Robotics and Automation*, 2019-May(0):9327–9333.

[Brecher et al., 2019b] Brecher, C., Eckel, H. M., Motschke, T., Fey, M., and Epple, A. (2019b). Estimation of the virtual workpiece quality by the use of a spindle-integrated process force measurement. *CIRP Annals*, 68(1):381–384.

[Brecher et al., 2017] Brecher, C., Epple, A., Fey, M., Königs, M., Neus, S., and Wellmann, F. N. (2017). Lernende Produktionssysteme. In *Internet of Production für agile Unternehmen : AWK Aachener Werkzeugmaschinen-Kolloquium 2017, 18. bis 19. Mai / Christian Brecher, Fritz Klocke, Robert Schmitt, Günther Schuh*, pages 135–161, Aachen. 29. Aachener Werkzeugmaschinen-Kolloquium, Aachen (Germany), 18 May 2017 - 19 May 2017, Apprimus Verlag.

[Brecher et al., 2012] Brecher, C., Jeschke, S., Schuh, G., Aghassi, S., Arnoscht, J., Bauhoff, F., Fuchs, S., Jooß, C., Karmann, O., Kozielski, S., Orilski, S., Richert, A., Roderburg, A., Schiffer, M., Schubert, J., Stiller, S., Tönissen, S., and Welter, F. (2012). *Integrative Production Technology for High-Wage Countries*. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Brecher et al., 2019c] Brecher, C., Wetzel, A., Berners, T., and Epple, A. (2019c). Increasing productivity of cutting processes by real-time compensation of tool deflection due to process forces. *Journal of Machine Engineering*, 19(1):16–27.

[Brecher et al., 2019d] Brecher, C., Wiesch, M., and Wellmann, F. (2019d). Productivity Increase - Model-based optimisation of NC-controlled milling processes to reduce machining time and improve process quality. *IFAC-PapersOnLine*, 52(13):1803–1807.

[Brenner and Hummel, 2017] Brenner, B. and Hummel, V. (2017). Digital Twin as Enabler for an Innovative Digital Shopfloor Management System in the ESB Logistics Learning Factory at Reutlingen - University. *Procedia Manufacturing*, 9:198–205.

[Burghardt et al., 2020] Burghardt, A., Szybicki, D., Gierlak, P., Kurc, K., Pietruś, P., and Cygan, R. (2020). Programming of industrial robots using virtual reality and digital twins. *Applied Sciences (Switzerland)*, 10(2).

[Busch et al., 2019] Busch, M., Schuh, G., Kelzenberg, C., and De Lange, J. (2019). Development of production planning and control through the empowerment of artificial intelligence. *Proceedings - 2019 2nd International Conference on Artificial Intelligence for Industries, AI4I 2019*, pages 115–118.

[Cai et al., 2020] Cai, B., Du, H., Cong, Y., Xie, F., and Zhang, J. (2020). Research on Angle Steel Tower Climbing Robot System Based on Digital Twin. In *2020 7th International Conference on Information, Cybernetics, and Computational Social Systems (ICCSS)*, 61673153, pages 749–754. IEEE.

[Cai et al., 2017] Cai, Y., Starly, B., Cohen, P., and Lee, Y. S. (2017). Sensor Data and Information Fusion to Construct Digital-twins Virtual Machine Tools for Cyber-physical Manufacturing. *Procedia Manufacturing*, 10:1031–1042.

[Cao et al., 2020] Cao, X., Zhao, G., and Xiao, W. (2020). Digital Twin–oriented real-time cutting simulation for intelligent computer numerical control machining. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 37.

[Cattaneo and MacChi, 2019] Cattaneo, L. and MacChi, M. (2019). A Digital Twin Proof of Concept to Support Machine Prognostics with Low Availability of Run-To-Failure Data. *IFAC-PapersOnLine*, 52(10):37–42.

[Chakraborty and Adhikari, 2021] Chakraborty, S. and Adhikari, S. (2021). Machine learning based digital twin for dynamical systems with multiple time-scales. *Computers and Structures*, 243:106410.

[Chakraborty et al., 2021] Chakraborty, S., Adhikari, S., and Ganguli, R. (2021). The role of surrogate models in the development of digital twins of dynamic systems. *Applied Mathematical Modelling*, 90:662–681.

[Chen et al., 2021] Chen, Y.-W., Joseph, R. J., Kanyuck, A., Khan, S., Malhan, R. K., Manyar, O. M., McNulty, Z., Wang, B., Barbič, J., and Gupta, S. K. (2021). A Digital Twin for Automated Layup of Prepreg Composite Sheets. In *Volume 1: Additive Manufacturing; Advanced Materials Manufacturing; Biomanufacturing; Life Cycle Engineering; Manufacturing Equipment and Automation*, June. American Society of Mechanical Engineers.

[Chhetri et al., 2019] Chhetri, S. R., Faezi, S., Canedo, A., and Faruque, M. A. A. (2019). QUILT: Quality inference from living digital twins in IoT-enabled manufacturing systems. *IoTDI 2019 - Proceedings of the 2019 Internet of Things Design and Implementation*, pages 237–248.

[Cichon and Robmann, 2018] Cichon, T. and Robmann, J. (2018). Digital Twins: Assisting and Supporting Cooperation in Human-Robot Teams. *2018 15th International Conference on Control, Automation, Robotics and Vision, ICARCV 2018*, 0(2):486–491.

[Cichon and Rosmann, 2017] Cichon, T. and Rosmann, J. (2017). Robotic teleoperation: Mediated and supported by virtual testbeds. *SSRR 2017 - 15th IEEE International Symposium on Safety, Security and Rescue Robotics, Conference*, Mmi:55–60.

[Cirullies and Schwede, 2021] Cirullies, J. and Schwede, C. (2021). On-demand Shared Digital Twins – An Information Architectural Model to Create Transparency in Collaborative Supply Networks. *Proceedings of the 54th Hawaii International Conference on System Sciences*, 5:1675–1684.

[Cronrath et al., 2019] Cronrath, C., Aderiani, A. R., and Lennartson, B. (2019). Enhancing digital twins through reinforcement learning. *IEEE International Conference on Automation Science and Engineering*, 2019-Augus:293–298.

[Delbrügger et al., 2019] Delbrügger, T., Meißner, M., Wirtz, A., Biermann, D., Myrzik, J., Rossmann, J., and Wiederkehr, P. (2019). Multi-level simulation concept for multidisciplinary analysis and optimization of production systems. *International Journal of Advanced Manufacturing Technology*, 103(9-12):3993–4012.

[Delbrügger and Rossmann, 2019] Delbrügger, T. and Rossmann, J. (2019). Representing adaptation options in experimentable digital twins of production systems. *International Journal of Computer Integrated Manufacturing*, 32(4-5):352–365.

[Delisle et al., 2018] Delisle, D., Schreiber, M., Krombholz, C., and Stüve, J. (2018). Production of fiber composite structures by means of cooperating robots. *Lightweight Design worldwide*, 11(2):42–47.

[Denkena et al., 2017] Denkena, B., Dahlmann, D., and Boujnah, H. (2017). Tool Deflection Control by a Sensory Spindle Slide for Milling Machine Tools. *Procedia CIRP*, 62:329–334.

[Denkena et al., 2021] Denkena, B., Pape, O., Krödel, A., Böß, V., Ellersiek, L., and Mücke, A. (2021). Process design for 5-axis ball end milling using a real-time capable dynamic material removal simulation. *Production Engineering*, 15(1):89–95.

[Ding et al., 2019] Ding, K., Chan, F. T., Zhang, X., Zhou, G., and Zhang, F. (2019). Defining a Digital Twin-based Cyber-Physical Production System for autonomous manufacturing in smart shop floors. *International Journal of Production Research*, 57(20):6315–6334.

[Dittrich et al., 2020] Dittrich, M. A., Schleich, B., Clausmeyer, T., Damgrave, R., Erkoyuncu, J. A., Haefner, B., de Lange, J., Plakhotnik, D., Scheidel, W., and Wuest, T. (2020). Shifting value stream patterns along the product lifecycle with digital twins. *Procedia CIRP*, 86:3–11.

[Dröder et al., 2018] Dröder, K., Bobka, P., Germann, T., Gabriel, F., and Dietrich, F. (2018). A machine learning-enhanced digital twin approach for human-robot-collaboration. *Procedia CIRP*, 76:187–192.

[Ertveldt et al., 2020] Ertveldt, J., Guillaume, P., and Helsen, J. (2020). MiCLAD as a platform for real-time monitoring and machine learning in laser metal deposition. *Procedia CIRP*, 94:456–461.

[Fang et al., 2019] Fang, Y., Peng, C., Lou, P., Zhou, Z., Hu, J., and Yan, J. (2019). Digital-Twin-Based Job Shop Scheduling Toward Smart Manufacturing. *IEEE Transactions on Industrial Informatics*, 15(12):6425–6435.

[Fedotov et al., 2020] Fedotov, A. A., Sergeev, S. M., Provotorova, E. N., Prozhogina, T. V., and Zaslavskaya, O. Y. (2020). The digital twin of a warehouse robot for Industry 4.0. *IOP Conference Series: Materials Science and Engineering*, 862(3):0–6.

[Feng et al., 2020] Feng, X., Zhao, Z., and Zhang, C. (2020). Simulation optimization framework for online deployment and adjustment of reconfigurable machines in job shops. *IEEE International Conference on Industrial Engineering and Engineering Management*, 2020-Decem:731–735.

[Filocamo, 2020] Filocamo, C. (2020). *Digital Twin and HMI for collaborative autonomous mobile robots in flexible logistics*. PhD thesis, Corso di laurea magistrale in Ingegneria Informatica.

[Frank et al., 2014] Frank, M., Leitner, J., Stollenga, M., Forster, A., and Schmidhuber, J. (2014). Curiosity driven reinforcement learning for motion planning on humanoids. *Frontiers in Neurorobotics*, 7(JAN):1–15.

[Franko et al., 2020] Franko, J., Du, S., Kallweit, S., Duelberg, E., and Engemann, H. (2020). Design of a multi-robot system for wind turbine maintenance. *Energies*, 13(10).

[Fuller et al., 2020] Fuller, A., Fan, Z., Day, C., and Barlow, C. (2020). Digital Twin: Enabling Technologies, Challenges and Open Research. *IEEE Access*, 8:108952–108971.

[Gaikwad et al., 2020a] Gaikwad, A., Giera, B., Guss, G. M., Forien, J. B., Matthews, M. J., and Rao, P. (2020a). Heterogeneous sensing and scientific machine learning for quality assurance in laser powder bed fusion – A single-track study. *Additive Manufacturing*, 36:101659.

[Gaikwad et al., 2020b] Gaikwad, A., Yavari, R., Montazeri, M., Cole, K., Bian, L., and Rao, P. (2020b). Toward the digital twin of additive manufacturing: Integrating thermal simulations, sensing, and analytics to detect process faults. *IISE Transactions*, 52(11):1204–1217.

[Ganser et al., 2021] Ganser, P., Landwehr, M., Schiller, S., Vahl, C., Mayer, S., and Bergs, T. (2021). Knowledge-Based Adaptation of Product and Process Design in Blisk Manufacturing. *Journal of Engineering for Gas Turbines and Power*.

[Gardner et al., 2020] Gardner, P., Dal Borgo, M., Ruffini, V., Hughes, A. J., Zhu, Y., and Wagg, D. J. (2020). Towards the Development of an Operational Digital Twin. *Vibration*, 3(3):235–265.

[Ghanem et al., 2020] Ghanem, R., Soize, C., Mehrez, L., and Aitharaju, V. (2020). Probabilistic learning and updating of a digital twin for composite material systems. *International Journal for Numerical Methods in Engineering*, December 2019:1–17.

[Ghosh et al., 2020] Ghosh, A. K., Sharif Ullah, A. M., Kubo, A., Akamatsu, T., and D'Addona, D. M. (2020). Machining phenomenon twin construction for industry 4.0: A case of surface roughness. *Journal of Manufacturing and Materials Processing*, 4(1).

[Girletti et al., 2020] Girletti, L., Groshev, M., Guimaraes, C., Bernardos, C. J., and de la Oliva, A. (2020). An Intelligent Edge-based Digital Twin for Robotics. In *2020 IEEE Globecom Workshops (GC Wkshps*, pages 1–6. IEEE.

[Glaessgen and Stargel, 2012] Glaessgen, E. H. and Stargel, D. S. (2012). The digital twin paradigm for future NASA and U.S. Air force vehicles. *Collection of Technical Papers - AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, April.

[Görthofer et al., 2019] Görthofer, J., Meyer, N., Pallicity, T. D., Schöttl, L., Trauth, A., Schemmann, M., Hohberg, M., Pinter, P., Elsner, P., Henning, F., Hrymak, A., Seelig, T., Weidenmann, K., Kärger, L., and Böhlke, T. (2019). Virtual process chain of sheet molding compound: Development, validation and perspectives. *Composites Part B: Engineering*, 169(April):133–147.

[Grégorio et al., 2020] Grégorio, J. L., Lartigue, C., Thiébaut, F., and Lebrun, R. (2020). A digital twin-based approach for the management of geometrical deviations during assembly processes. *Journal of Manufacturing Systems*, April:1–10.

[Grieves, 2014] Grieves, M. (2014). Digital twin: manufacturing excellence through virtual factory replication. *White paper*, 1:1–7.

[Grieves and Vickers, 2017] Grieves, M. and Vickers, J. (2017). Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In Kahlen, F.-J., Flumerfelt, S., and Alves, A., editors, *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*, pages 85–113. Springer International Publishing, Cham.

[Grinshpun et al., 2016] Grinshpun, G., Cichon, T., Dipika, D., and Rossmann, J. (2016). From virtual testbeds to real lightweight robots: Development and deployment of control algorithms for soft robots, with particular reference to industrial peg-in-hole insertion tasks. *47th International Symposium on Robotics, ISR 2016*, 2016:208–214.

[Guo et al., 2020a] Guo, D., Zhong, R. Y., Lin, P., Lyu, Z., Rong, Y., and Huang, G. Q. (2020a). Digital twin-enabled Graduation Intelligent Manufacturing System for fixed-position assembly islands. *Robotics and Computer-Integrated Manufacturing*, 63(November 2019):101917.

[Guo et al., 2018] Guo, F., Zou, F., Liu, J., and Wang, Z. (2018). Working mode in aircraft manufacturing based on digital coordination model. *International Journal of Advanced Manufacturing Technology*, 98(5-8):1547–1571.

[Guo et al., 2020b] Guo, H., Chen, M., Mohamed, K., Qu, T., Wang, S., and Li, J. (2020b). A digital twin-based flexible cellular manufacturing for optimization of air conditioner line. *Journal of Manufacturing Systems*, June:1–14.

[Haber et al., 2020] Haber, R., Strzelczak, S., Miljkovic, Z., Castano, F., Fumagalli, L., and Petrovic, M. (2020). Digital twin-based Optimization on the basis of Grey Wolf Method. A Case Study on Motion Control Systems. *Proceedings - 2020 IEEE Conference on Industrial Cyberphysical Systems, ICPS 2020*, pages 469–474.

[Hänel et al., 2020] Hänel, A., Schnellhardt, T., Wenkler, E., Nestler, A., Brosius, A., Corinth, C., Fay, A., and Ihlenfeldt, S. (2020). The development of a digital twin for machining processes for the application in aerospace industry. *Procedia CIRP*, 93:1399–1404.

[Hardt et al., 2021] Hardt, M., Schraknepper, D., and Bergs, T. (2021). Investigations on the Application of the Downhill-Simplex-Algorithm to the Inverse Determination of Material Model Parameters for FE-Machining Simulations. *Simulation Modelling Practice and Theory*, 107:129–148.

[Hartmann et al., 2018] Hartmann, D., Herz, M., and Wever, U. (2018). Model Order Reduction a Key Technology for Digital Twins. In Keiper, W., Milde, A., and Volkwein, S., editors, *Reduced-Order Modeling (ROM) for Simulation and Optimization: Powerful Algorithms as Key Enablers for Scientific Computing*, pages 167–179. Springer International Publishing, Cham.

[Haße, 2019] Haße, H. (2019). Digital Twin for Real-Time Data Processing in Logistics. *Artificial Intelligence and Digital Transformation in Supply Chain Management (Proceedings of the Hamburg International Conference of Logistics (HICL), 27)*, February 2020:0–1.

[Havard et al., 2019] Havard, V., Jeanne, B., Lacomblez, M., and Baudry, D. (2019). Digital twin and virtual reality: a co-simulation environment for design and assessment of industrial workstations. *Production and Manufacturing Research*, 7(1):472–489.

[Havinga et al., 2020] Havinga, J., Mandal, P. K., and van den Boogaard, T. (2020). Exploiting data in smart factories: real-time state estimation and model improvement in metal forming mass production. *International Journal of Material Forming*, 13(5):663–673.

[He et al., 2021a] He, B., Cao, X., and Hua, Y. (2021a). Data fusion-based sustainable digital twin system of intelligent detection robotics. *Journal of Cleaner Production*, 280:124181.

[He et al., 2021b] He, B., Li, T., and Xiao, J. (2021b). Digital twin-driven controller tuning method for dynamics. *Journal of Computing and Information Science in Engineering*, 21(June):1–27.

[He et al., 2021c] He, B., Liu, L., and Zhang, D. (2021c). Digital Twin-driven Remaining Useful Life Prediction for Gear Performance Degradation: A Review. *Journal of Computing and Information Science in Engineering*, 21(June):1–70.

[He et al., 2020] He, B., Zhu, X., and Zhang, D. (2020). Boundary encryption-based Monte Carlo learning method for workspace modeling. *Journal of Computing and Information Science in Engineering*, 20(3):1–6.

[Hedberg et al., 2016] Hedberg, T., Lubell, J., Fischer, L., Maggiano, L., and Feeney, A. B. (2016). Testing the digital thread in support of model-based manufacturing and inspection. *Journal of Computing and Information Science in Engineering*, 16(2):1–10.

[Hellmuth et al., 2020] Hellmuth, R., Wehner, F., and Giannakidis, A. (2020). Approach for an update method for digital factory models. *Procedia CIRP*, 93:280–285.

[Helu and Hedberg, 2015] Helu, M. and Hedberg, T. (2015). Enabling Smart Manufacturing Research and Development using a Product Lifecycle Test Bed. *Procedia Manufacturing*, 1(Wolf 2009):86–97.

[Helu et al., 2017] Helu, M., Hedberg, T., and Barnard Feeney, A. (2017). Reference architecture to integrate heterogeneous manufacturing systems for the digital thread. *CIRP Journal of Manufacturing Science and Technology*, 19:191–195.

[Heo and Yoo, 2021] Heo, E. and Yoo, N. (2021). Numerical Control Machine Optimization Technologies through Analysis of Machining History Data Using Digital Twin. *Applied Sciences*, 11(7):3259.

[Heo et al., 2021] Heo, T. W., Khairallah, S. A., Shi, R., Berry, J., Perron, A., Calta, N. P., Martin, A. A., Barton, N. R., Roehling, J. D., Roehling, T., Fattebert, J.-L., Anderson, A., Nichols, A. L., Wopschall, S., King, W. E., McKeown, J. T., and Matthews, M. (2021). A mesoscopic digital twin that bridges length and time scales for control of additively manufactured metal microstructures. *Journal of Physics: Materials*, December 2016:11–14.

[Hoebert et al., 2019] Hoebert, T., Lepuschitz, W., List, E., and Merdan, M. (2019). *Cloud-Based Digital Twin for Industrial Robotics*, volume 11710 LNAI. Springer International Publishing.

[Hu et al., 2020] Hu, L., Liu, Z., Hu, W., Wang, Y., Tan, J., and Wu, F. (2020). Petri-net-based dynamic scheduling of flexible manufacturing system via deep reinforcement learning with graph convolutional network. *Journal of Manufacturing Systems*, 55(February):1–14.

[Huang et al., 2020] Huang, H., Ma, N., Chen, J., Feng, Z., and Murakawa, H. (2020). Toward large-scale simulation of residual stress and distortion in wire and arc additive manufacturing. *Additive Manufacturing*, 34(April):101248.

[Hürkamp et al., 2020] Hürkamp, A., Gellrich, S., Ossowski, T., Beuscher, J., Thiede, S., Herrmann, C., and Dröder, K. (2020). Combining simulation and machine learning as digital twin for the manufacturing of overmolded thermoplastic composites. *Journal of Manufacturing and Materials Processing*, 4(3).

[Hüttemann et al., 2019] Hüttemann, G., Buckhorst, A. F., and Schmitt, R. H. (2019). Modelling and assessing line-less mobile assembly systems. *Procedia CIRP*, 81:724–729.

[Huynh and Altintas, 2021] Huynh, H. N. and Altintas, Y. (2021). Modeling the Dynamics of Five-Axis Machine Tool Using the Multibody Approach. *Journal of Manufacturing Science and Engineering*, 143(2).

[Ivanov and Dolgui, 2020] Ivanov, D. and Dolgui, A. (2020). A digital supply chain twin for managing the disruption risks and resilience in the era of Industry 4.0. *Production Planning and Control*, 0(0):1–14.

[Jaensch et al., 2019a] Jaensch, F., Csiszar, A., Kienzlen, A., and Verl, A. (2019a). Reinforcement learning of material flow control logic using hardware-in-the-loop simulation. *Proceedings - 2018 1st IEEE International Conference on Artificial Intelligence for Industries, AI4I 2018*, pages 77–80.

[Jaensch et al., 2019b] Jaensch, F., Csiszar, A., Scheifele, C., and Verl, A. (2019b). Digital Twins of Manufacturing Systems as a Base for Machine Learning. *Proceedings of the 2018 25th International Conference on Mechatronics and Machine Vision in Practice, M2VIP 2018*.

[Jazdi et al., 2021] Jazdi, N., Ashtari Talkhestani, B., Maschler, B., and Weyrich, M. (2021). Realization of AI-enhanced industrial automation systems using intelligent Digital Twins. *Procedia CIRP*, 97:396–400.

[Jeschke et al., 2017] Jeschke, S., Brecher, C., Meisen, T., Özdemir, D., and Eschert, T. (2017). Industrial Internet of Things and Cyber Manufacturing Systems. In Jeschke, S., Brecher, C., Song, H., and Rawat, D. B., editors, *Industrial Internet of Things. Springer Series in Wireless Technology. Springer, Cham*, number October in Springer Series in Wireless Technology, pages 3–19. Springer International Publishing, Cham.

[Jin et al., 2020] Jin, T., Sun, Z., Li, L., Zhang, Q., Zhu, M., Zhang, Z., Yuan, G., Chen, T., Tian, Y., Hou, X., and Lee, C. (2020). Triboelectric nanogenerator sensors for soft robotics aiming at digital twin applications. *Nature Communications*, 11(1):1–12.

[Jones et al., 2020] Jones, D., Snider, C., Nassehi, A., Yon, J., and Hicks, B. (2020). Characterising the Digital Twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology*, 29:36–52.

[Kabaldin et al., 2019] Kabaldin, Y. G., Shatagin, D. A., Anosov, M. S., Kolchin, P. V., and Kuz'mishina, A. M. (2019). CNC Machine Tools and Digital Twins. *Russian Engineering Research*, 39(8):637–644.

[Kaji et al., 2020] Kaji, M., Parvizian, J., and van de Venn, H. W. (2020). Constructing a reliable health indicator for bearings using convolutional autoencoder and continuous wavelet transform. *Applied Sciences (Switzerland)*, 10(24):1–21.

[Kampker et al., 2019] Kampker, A., Stich, V., Jussen, P., Moser, B., and Kuntz, J. (2019). Business models for industrial smart services - the example of a digital twin for a product-service-system for potato harvesting. *Procedia CIRP*, 83:534–540.

[Kapusta et al., 2016] Kapusta, A., Yu, W., Bhattacharjee, T., Liu, C. K., Turk, G., and Kemp, C. C. (2016). Data-driven haptic perception for robot-assisted dressing. *25th IEEE International Symposium on Robot and Human Interactive Communication, RO-MAN 2016*, pages 451–458.

[Karanjkar et al., 2019] Karanjkar, N., Joglekar, A., Mohanty, S., Prabhu, V., Raghunath, D., and Sundaresan, R. (2019). Digital twin for energy optimization in an SMT-PCB assembly line. *Proceedings - 2018 IEEE International Conference on Internet of Things and Intelligence System, IOTAIS 2018*, pages 85–89.

[Karve et al., 2020] Karve, P. M., Guo, Y., Kapusuzoglu, B., Mahadevan, S., and Haile, M. A. (2020). Digital twin approach for damage-tolerant mission planning under uncertainty. *Engineering Fracture Mechanics*, 225:106766.

[Kehl et al., 2020] Kehl, P., Lange, D., Konstantin Maurer, F., Nemeth, G., Overbeck, D., Jung, S., Konig, N., and Schmitt, R. H. (2020). Comparison of 5G Enabled Control Loops for Production. In *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*, volume 2020-Augus, pages 1–6. IEEE.

[Khalastchi et al., 2015] Khalastchi, E., Kalech, M., Kaminka, G. A., and Lin, R. (2015). Online data-driven anomaly detection in autonomous robots. *Knowledge and Information Systems*, 43(3):657–688.

[Klamt et al., 2019] Klamt, T., Kamedula, M., Karaoguz, H., Kashiri, N., Laurenzi, A., Lenz, C., Leonardis, D., Mingo Hoffman, E., Muratore, L., Pavlichenko, D., Porcini, F., Rodriguez, D., Ren, Z., Schilling, F., Schwarz, M., Solazzi, M., Felsberg, M., Frisoli, A., Gustmann, M., Jensfelt, P., Nordberg, K., Rossmann, J., Baccelliere, L., Suess, U., Tsagarakis, N. G., Behnke, S., Chen, X., Chiaradia, D., Cichon, T., Gabardi, M., Guria, P., and Holmquist, K. (2019). Flexible Disaster Response of Tomorrow: Final Presentation and Evaluation of the CENTAURO System. *IEEE Robotics and Automation Magazine*, 26(4):59–72.

[Ko et al., 2019] Ko, H., Witherell, P., Ndiaye, N. Y., and Lu, Y. (2019). Machine learning based continuous knowledge engineering for additive manufacturing. *IEEE International Conference on Automation Science and Engineering*, 2019-Augus:648–654.

[Königs and Brecher, 2018] Königs, M. and Brecher, C. (2018). Process-parallel virtual quality evaluation for metal cutting in series production. *Procedia Manufacturing*, 26:1087–1093.

[Königs et al., 2017] Königs, M., Wellmann, F., Wiesch, M., Epple, A., Brecher, C., Schmitt, R., and Schuh, G. (2017). A scalable, hybrid learning approach to process-parallel estimation of cutting forces in milling applications. *Robert Schmitt Günther Schuh (Publ.)*, 7:425–432.

[Kousi et al., 2019] Kousi, N., Gkournelos, C., Aivaliotis, S., Giannoulis, C., Michalos, G., and Makris, S. (2019). Digital twin for adaptation of robots' behavior in flexible robotic assembly lines. *Procedia Manufacturing*, 28:121–126.

[Ktari and El Mansori, 2020] Ktari, A. and El Mansori, M. (2020). Digital twin of functional gating system in 3D printed molds for sand casting using a neural network. *Journal of Intelligent Manufacturing*.

[Kuhnle et al., 2020] Kuhnle, A., Kaiser, J. P., Theiß, F., Stricker, N., and Lanza, G. (2020). Designing an adaptive production control system using reinforcement learning. *Journal of Intelligent Manufacturing*, 32(3):855–876.

[KUTS et al., 2020] KUTS, V., CHEREZOVA, N., SARKANS, M., and OTTO, T. (2020). Digital Twin: industrial robot kinematic model integration to the virtual reality environment. *Journal of Machine Engineering*, 20(2):53–64.

[Kuts et al., 2019] Kuts, V., Sarkans, M., Otto, T., Tähemaa, T., and Bondarenko, Y. (2019). Digital Twin: Concept of Hybrid Programming for Industrial Robots — Use Case. In *Volume 2B: Advanced Manufacturing*. American Society of Mechanical Engineers.

[Kwon et al., 2020] Kwon, S., Monnier, L. V., Barbau, R., and Bernstein, W. Z. (2020). Enriching standards-based digital thread by fusing as-designed and as-inspected data using knowledge graphs. *Advanced Engineering Informatics*, 46(January):101102.

[Ladj et al., 2020] Ladj, A., Wang, Z., Meski, O., Belkadi, F., Ritou, M., and Da Cunha, C. (2020). A knowledge-based Digital Shadow for machining industry in a Digital Twin perspective. *Journal of Manufacturing Systems*, July:1–12.

[Larsen, 2019] Larsen, C. (2019). Including a Collaborative Robot in Digital Twin Manufacturing Systems. *Master's thesis 2019 Digital*.

[Leng et al., 2020] Leng, J., Liu, Q., Ye, S., Jing, J., Wang, Y., Zhang, C., Zhang, D., and Chen, X. (2020). Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model. *Robotics and Computer-Integrated Manufacturing*, 63(March 2019).

[Leng et al., 2019] Leng, J., Zhang, H., Yan, D., Liu, Q., Chen, X., and Zhang, D. (2019). Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1155–1166.

[Leser et al., 2020] Leser, P. E., Warner, J. E., Leser, W. P., Bomarito, G. F., Newman, J. A., and Hochhalter, J. D. (2020). A digital twin feasibility study (Part II): Non-deterministic predictions of fatigue life using in-situ diagnostics and prognostics. *Engineering Fracture Mechanics*, 229(June 2019):106903.

[Li et al., 2020a] Li, G., Liu, Z., Cai, L., and Yan, J. (2020a). Standing-Posture Recognition in Human–Robot Collaboration Based on Deep Learning and the Dempster–Shafer Evidence Theory. *Sensors*, 20(4):1158.

[Li et al., 2020b] Li, L., Liu, D., Liu, J., Zhou, H. G., and Zhou, J. (2020b). Quality prediction and control of assembly and welding process for ship group product based on digital twin. *Scanning*, 2020.

[Li et al., 2019] Li, L., Xu, W., Liu, Z., Yao, B., Zhou, Z., and Pham, D. T. (2019). Digital twin-based control approach for industrial cloud robotics. *ASME 2019 14th International Manufacturing Science and Engineering Conference, MSEC 2019*, 1:1–7.

[Li et al., 2020c] Li, X., Cao, J., Liu, Z., and Luo, X. (2020c). Sustainable Business Model Based on Digital Twin Platform Network: The Inspiration from Haier's Case Study in China. *Sustainability*, 12(3):936.

[Liang et al., 2019] Liang, C. J., Kamat, V. R., and Menassa, C. C. (2019). Teaching robots to perform construction tasks via learning from demonstration. *Proceedings of the 36th International Symposium on Automation and Robotics in Construction, ISARC 2019*, pages 1305–1311.

[Liang et al., 2020] Liang, C.-J., McGee, W., Menassa, C., and Kamat, V. (2020). Bi-Directional Communication Bridge for State Synchronization between Digital Twin Simulations and Physical Construction Robots. *Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC)*, October.

[Lim et al., 2020] Lim, K. Y. H., Zheng, P., and Chen, C. H. (2020). A state-of-the-art survey of Digital Twin: techniques, engineering product lifecycle management and business innovation perspectives. *Journal of Intelligent Manufacturing*, 31(6):1313–1337.

[Liu et al., 2020a] Liu, C., Gao, J., Bi, Y., Shi, X., and Tian, D. (2020a). A Multitasking-Oriented Robot Arm Motion Planning Scheme Based on Deep Reinforcement Learning and Twin Synchro-Control. *Sensors*, 20(12):3515.

[Liu et al., 2020b] Liu, C., Le Roux, L., Körner, C., Tabaste, O., Lacan, F., and Bigot, S. (2020b). Digital Twin-enabled Collaborative Data Management for Metal Additive Manufacturing Systems. *Journal of Manufacturing Systems*, April:0–1.

[Liu et al., 2020c] Liu, M., Fang, S., Dong, H., and Xu, C. (2020c). Review of digital twin about concepts, technologies, and industrial applications. *Journal of Manufacturing Systems*, October 2019:1–16.

[Liu et al., 2021] Liu, Q., Leng, J., Yan, D., Zhang, D., Wei, L., Yu, A., Zhao, R., Zhang, H., and Chen, X. (2021). Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system. *Journal of Manufacturing Systems*, 58(October 2019):52–64.

[Liu et al., 2019] Liu, Q., Zhang, H., Leng, J., and Chen, X. (2019). Digital twin-driven rapid individualised designing of automated flow-shop manufacturing system. *International Journal of Production Research*, 57(12):3903–3919.

[Liu et al., 2020d] Liu, Z., Chen, W., Zhang, C., Yang, C., and Cheng, Q. (2020d). Intelligent scheduling of a feature-process-machine tool supernetwork based on digital twin workshop. *Journal of Manufacturing Systems*, July.

[Loaldi et al., 2020] Loaldi, D., Regi, F., Baruffi, F., Calaon, M., Quagliotti, D., Zhang, Y., and Tosello, G. (2020). Experimental validation of injection molding simulations of 3D microparts and microstructured components using virtual design of experiments and multi-scale modeling. *Micromachines*, 11(6):1–17.

[Lokuwaduge and Heenetigala, 2017] Lokuwaduge, C. S. D. S. and Heenetigala, K. (2017). Integrating Environmental, Social and Governance (ESG) Disclosure for a Sustainable Development: An Australian Study. *Business Strategy and the Environment*, 26(4):438–450.

[Luo et al., 2020] Luo, W., Hu, T., Ye, Y., Zhang, C., and Wei, Y. (2020). A hybrid predictive maintenance approach for CNC machine tool driven by Digital Twin. *Robotics and Computer-Integrated Manufacturing*, 65(March):101974.

[Luo et al., 2019] Luo, W., Hu, T., Zhang, C., and Wei, Y. (2019). Digital twin for CNC machine tool: modeling and using strategy. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1129–1140.

[Lv et al., 2021] Lv, Q., Zhang, R., Sun, X., Lu, Y., and Bao, J. (2021). A digital twin-driven human-robot collaborative assembly approach in the wake of COVID-19. *Journal of manufacturing systems*, November 2020.

[Lynn et al., 2018] Lynn, R., Sati, M., Tucker, T., Rossignac, J., Saldana, C., and Kurfess, T. (2018). Realization of the 5-Axis Machine Tool Digital Twin Using Direct Servo Control

from CAM. *National Institute of Standards and Technology (NIST) Model-Based Enterprise Summit*, pages 1–22.

[Lyu and Cheah, 2020] Lyu, S. and Cheah, C. C. (2020). Data-driven learning for robot control with unknown Jacobian. *Automatica*, 120:109120.

[Ma et al., 2021] Ma, X., Cheng, J., Qi, Q., and Tao, F. (2021). Artificial intelligence enhanced interaction in digital twin shop-floor. *Procedia CIRP*, 100:858–863.

[Malik and Brem, 2020] Malik, A. A. and Brem, A. (2020). Man, machine and work in a digital twin setup: a case study. *arXiv*.

[Malik and Brem, 2021] Malik, A. A. and Brem, A. (2021). Digital twins for collaborative robots: A case study in human-robot interaction. *Robotics and Computer-Integrated Manufacturing*, 68(September 2019):102092.

[Marmolejo-Saucedo, 2020] Marmolejo-Saucedo, J. A. (2020). Design and Development of Digital Twins: a Case Study in Supply Chains. *Mobile Networks and Applications*, 25(6):2141–2160.

[Maschler et al., 2020] Maschler, B., Braun, D., Jazdi, N., and Weyrich, M. (2020). Transfer Learning as an Enabler of the Intelligent Digital Twin. *Procedia CIRP*, 00:2–7.

[Matulis and Harvey, 2021] Matulis, M. and Harvey, C. (2021). A robot arm digital twin utilising reinforcement learning. *Computers & Graphics*, 95:106–114.

[May et al., 2021] May, M. C., Overbeck, L., Wurster, M., Kuhnle, A., and Lanza, G. (2021). Foresighted digital twin for situational agent selection in production control. *Procedia CIRP*, 99(i):27–32.

[May et al., 2020] May, M. C., Schmidt, S., Kuhnle, A., Stricker, N., and Lanza, G. (2020). Product Generation Module: Automated Production Planning for optimized workload and increased efficiency in Matrix Production Systems. *Procedia CIRP*, 96:45–50.

[Mayes et al., 2019] Mayes, A., Heffernan, J., Jauriqui, L., Livings, R., Biedermann, E., Aldrin, J. C., Goodlet, B. R., and Mazdiyasni, S. (2019). Process compensated resonance testing (PCRT) inversion for material characterization and digital twin calibration. *AIP Conference Proceedings*, 2102(May).

[Meierhofer et al., 2020] Meierhofer, J., West, S., Rapaccini, M., and Barbieri, C. (2020). The Digital Twin as a Service Enabler: From the Service Ecosystem to the Simulation Model. In *Lecture Notes in Business Information Processing*, volume 377 LNBIP, pages 347–359. Springer.

[Meng et al., 2019] Meng, S., Tang, S., Zhu, Y., and Chen, C. (2019). Digital Twin-Driven Control Method for Robotic Automatic Assembly System. *IOP Conference Series: Materials Science and Engineering*, 493(1):012128.

[Mi et al., 2020] Mi, S., Feng, Y., Zheng, H., Wang, Y., Gao, Y., and Tan, J. (2020). Prediction maintenance integrated decision-making approach supported by digital twin-driven cooperative awareness and interconnection framework. *Journal of Manufacturing Systems*, November 2019:0–1.

[Min et al., 2019] Min, Q., Lu, Y., Liu, Z., Su, C., and Wang, B. (2019). Machine Learning based Digital Twin Framework for Production Optimization in Petrochemical Industry. *International Journal of Information Management*, 49(April):502–519.

[Min et al., 2020] Min, S. H., Lee, T. H., Lee, G. Y., Zontar, D., Brecher, C., and Ahn, S. H. (2020). Directly printed low-cost nanoparticle sensor for vibration measurement during milling process. *Materials*, 13(13):1–12.

[Minos-Stensrud et al., 2018] Minos-Stensrud, M., Haakstad, O. H., Sakseid, O., Westby, B., and Alcocer, A. (2018). Towards automated 3D reconstruction in SME factories and digital twin model generation. *International Conference on Control, Automation and Systems*, 2018-Octob(Iccas):1777–1781.

[Monostori et al., 2016] Monostori, L., Kádár, B., Bauernhansl, T., Kondoh, S., Kumara, S., Reinhart, G., Sauer, O., Schuh, G., Sihn, W., and Ueda, K. (2016). Cyber-physical systems in manufacturing. *CIRP Annals*, 65(2):621–641.

[Moretti et al., 2021] Moretti, M., Rossi, A., and Senin, N. (2021). In-process monitoring of part geometry in fused filament fabrication using computer vision and digital twins. *Additive Manufacturing*, 37(January):101609.

[MPOC, 2020] MPOC (2020). *Digital twin methods for feedback control in multi-robot welding cells*. PhD thesis, LUT UNIVERSITY.

[Negri et al., 2020] Negri, E., Pandhare, V., Cattaneo, L., Singh, J., Macchi, M., and Lee, J. (2020). Field-synchronized Digital Twin framework for production scheduling with uncertainty. *Journal of Intelligent Manufacturing*.

[Oyekan et al., 2020] Oyekan, J., Farnsworth, M., Hutabarat, W., Miller, D., and Tiwari, A. (2020). Applying a 6 dof robotic arm and digital twin to automate fan-blade reconditioning for aerospace maintenance, repair, and overhaul. *Sensors (Switzerland)*, 20(16):1–20.

[Pairet et al., 2019] Pairet, E., Ardón, P., Liu, X., Lopes, J., Hastie, H., and Lohan, K. S. (2019). A Digital Twin for Human-Robot Interaction. *ACM/IEEE International Conference on Human-Robot Interaction*, 2019-March:372.

[Pan and Zhang, 2021] Pan, Y. and Zhang, L. (2021). A BIM-data mining integrated digital twin framework for advanced project management. *Automation in Construction*, 124(July 2020):103564.

[Pan et al., 2020] Pan, Y. H., Wu, N. Q., Qu, T., Li, P. Z., Zhang, K., and Guo, H. F. (2020). Digital-twin-driven production logistics synchronization system for vehicle routing problems with pick-up and delivery in industrial park. *International Journal of Computer Integrated Manufacturing*.

[Papacharalampopoulos et al., 2020] Papacharalampopoulos, A., Stavropoulos, P., and Petrides, D. (2020). Towards a digital twin for manufacturing processes: Applicability on laser welding. *Procedia CIRP*, 88:110–115.

[Pennekamp et al., 2019] Pennekamp, J., Glebke, R., Henze, M., Meisen, T., Quix, C., Hai, R., Gleim, L., Niemietz, P., Rudack, M., Knape, S., Epple, A., Trauth, D., Vroomen, U., Bergs, T., Brecher, C., Buhrig-Polaczek, A., Jarke, M., and Wehrle, K. (2019). Towards an infrastructure enabling the internet of production. *Proceedings - 2019 IEEE International Conference on Industrial Cyber Physical Systems, ICPS 2019*, pages 31–37.

[Pérez et al., 2020] Pérez, L., Rodríguez-Jiménez, S., Rodríguez, N., Usamentiaga, R., and García, D. F. (2020). Digital twin and virtual reality based methodology for multi-robot manufacturing cell commissioning. *Applied Sciences (Switzerland)*, 10(10).

[Pfrommer et al., 2018] Pfrommer, J., Zimmerling, C., Liu, J., Kärger, L., Henning, F., and Beyerer, J. (2018). Optimisation of manufacturing process parameters using deep neural networks as surrogate models. *Procedia CIRP*, 72:426–431.

[Piacenza et al., 2020] Piacenza, P., Behrman, K., Schifferer, B., Kymissis, I., and Ciocarlie, M. (2020). A Sensorized Multicurved Robot Finger with Data-Driven Touch Sensing via Overlapping Light Signals. *IEEE/ASME Transactions on Mechatronics*, 25(5):2416–2427.

[Podskarbi and Knezevic, 2020] Podskarbi, M. and Knezevic, D. J. (2020). Digital twin for operations-present applications and future digital thread. *Proceedings of the Annual Offshore Technology Conference*, 2020-May(May):4–7.

[Polini and Corrado, 2020] Polini, W. and Corrado, A. (2020). Digital twin of composite assembly manufacturing process. *International Journal of Production Research*, 58(17):5238–5252.

[Postel et al., 2019] Postel, M., Aslan, D., Wegener, K., and Altintas, Y. (2019). Monitoring of vibrations and cutting forces with spindle mounted vibration sensors. *CIRP Annals*, 68(1):413–416.

[Psarommatis, 2021] Psarommatis, F. (2021). A generic methodology and a digital twin for zero defect manufacturing (ZDM) performance mapping towards design for ZDM. *Journal of Manufacturing Systems*, 59(March):507–521.

[Qi and Tao, 2018] Qi, Q. and Tao, F. (2018). Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison. *IEEE Access*, 6:3585–3593.

[Qiao et al., 2019] Qiao, Q., Wang, J., Ye, L., and Gao, R. X. (2019). Digital twin for machining tool condition prediction. *Procedia CIRP*, 81:1388–1393.

[Ramnath et al., 2020] Ramnath, S., Haghighi, P., Venkiteswaran, A., and Shah, J. J. (2020). Interoperability of CAD geometry and product manufacturing information for computer integrated manufacturing. *International Journal of Computer Integrated Manufacturing*, 33(2):116–132.

[Rasheed et al., 2020] Rasheed, A., San, O., and Kvamsdal, T. (2020). Digital twin: Values, challenges and enablers from a modeling perspective. *IEEE Access*, 8:21980–22012.

[Rebmann et al., 2020] Rebmann, A., Knoch, S., Emrich, A., Fettke, P., and Loos, P. (2020). A multi-sensor approach for digital twins of manual assembly and commissioning. *Procedia Manufacturing*, 51:549–556.

[Reinhart et al., 2017] Reinhart, R. F., Shareef, Z., and Steil, J. J. (2017). Hybrid analytical and data-driven modeling for feed-forward robot control†. *Sensors (Switzerland)*, 17(2):1–19.

[Reisch et al., 2020] Reisch, R., Hauser, T., Kamps, T., and Knoll, A. (2020). Robot based wire arc additive manufacturing system with context-sensitive multivariate monitoring framework. *Procedia Manufacturing*, 51(2019):732–739.

[Repalle et al., 2020] Repalle, N., Thethi, R., Viana, P., and Tellier, E. (2020). Application of machine learning for fatigue prediction of flexible risers - Digital twin approach. *Society of Petroleum Engineers - SPE Asia Pacific Oil and Gas Conference and Exhibition 2020, APOG 2020*.

[Rezaei Aderiani et al., 2021] Rezaei Aderiani, A., Wärmefjord, K., and Söderberg, R. (2021). Evaluating different strategies to achieve the highest geometric quality in self-adjusting smart assembly lines. *Robotics and Computer-Integrated Manufacturing*, 71(September 2020):102164.

[Riesener et al., 2019] Riesener, M., Schuh, G., Dölle, C., and Tönnes, C. (2019). The digital shadow as enabler for data analytics in product life cycle management. *Procedia CIRP*, 80:729–734.

[Roh, 2018] Roh, J. K. P. (2018). Assessing the Efficiency of Information Retrieval from the Digital Shadow at the Shop Floor using IT Assistive Systems. *16th Mechatronics Forum International Conference*, pages 202–209.

[Rohmer et al., 2013] Rohmer, E., Singh, S. P., and Freese, M. (2013). V-REP: A versatile and scalable robot simulation framework. *IEEE International Conference on Intelligent Robots and Systems*, pages 1321–1326.

[Rojek et al., 2021] Rojek, I., Mikołajewski, D., and Dostatni, E. (2021). Digital twins in product lifecycle for sustainability in manufacturing and maintenance. *Applied Sciences (Switzerland)*, 11(1):1–19.

[Rosen et al., 2015] Rosen, R., Von Wichert, G., Lo, G., and Bettenhausen, K. D. (2015). About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-PapersOnLine*, 28(3):567–572.

[Rückert et al., 2020] Rückert, P., Tracht, K., Herfs, W., Roggendorf, S., Schubert, V., and Schneider, M. (2020). Consolidation of product lifecycle information within human-robot collaboration for assembly of multi-variant products. *Procedia Manufacturing*, 49:217–221.

[Salvador Palau et al., 2019] Salvador Palau, A., Dhada, M. H., and Parlikad, A. K. (2019). Multi-agent system architectures for collaborative prognostics. *Journal of Intelligent Manufacturing*, 30(8):2999–3013.

[Samnejad et al., 2020] Samnejad, M., Shirangi, M. G., and Ettehadi, R. (2020). A digital twin of drilling fluids rheology for real-time rig operations. *Proceedings of the Annual Offshore Technology Conference*, 2020-May(May):4–7.

[Santina et al., 2020] Santina, C. D., Truby, R. L., and Rus, D. (2020). Data-Driven Disturbance Observers for Estimating External Forces on Soft Robots. *IEEE Robotics and Automation Letters*, 5(4):5717–5724.

[Schäkel et al., 2018] Schäkel, M., McNab, J., Dodds, N., Peters, T., Janssen, H., and Brecher, C. (2018). Data collection and analysis for the creation of a digital shadow during the production of thermoplastic composite layers in unbonded flexible pipes. *Proceedings of the International Conference on Offshore Mechanics and Arctic Engineering - OMAE*, 5:1–9.

[Scharl and Praktiknjo, 2019] Scharl, S. and Praktiknjo, A. (2019). The Role of a Digital Industry 4.0 in a Renewable Energy System. *International Journal of Energy Research*, 43(8):3891–3904.

[Schleich et al., 2017] Schleich, B., Anwer, N., Mathieu, L., and Wartzack, S. (2017). Shaping the digital twin for design and production engineering. *CIRP Annals*, 66(1):141–144.

[Schluse and Rossmann, 2016] Schluse, M. and Rossmann, J. (2016). From Simulation to Experimentable Digital Twins. *IEEE International Symposium on Systems Engineering*, pages 1–6.

[Schmitt et al., 2019] Schmitt, R. H., Nienheysen, P., Lehmann, N., Jahangir, H., Peterek, M., and Neuenhahn, T. (2019). Digitalized Ultrasonic Inspection by Optical Tracking. *Proceedings of the 2019 IEEE/SICE International Symposium on System Integration, SII 2019*, pages 566–571.

[Schmitt and Voigtmann, 2018] Schmitt, R. H. and Voigtmann, C. (2018). Sensor information as a service–component of networked production. *Journal of Sensors and Sensor Systems*, 7(1):389–402.

[Schuh et al., 2016] Schuh, G., Blum, M., Reschke, J., and Birkmeier, M. (2016). Der digitale schatten in der auftragsabwicklung. *ZWF Zeitschrift fuer Wirtschaftlichen Fabrikbetrieb*, 111(1-2):48–51.

[Schuh et al., 2017] Schuh, G., Stich, V., Basse, F., Franzkoch, B., Harzenetter, F., Luckert, M., Prote, J., Reschke, J., Schmitz, S., Tücks, G., and Weißkopf, J. (2017). Change request im Produktionsbetrieb. In *AWK Aachener Werkzeugmaschinen-Kolloquium*, pages 109–131. Apprimus Verlag.

[Schulz et al., 2020] Schulz, M., Janssen, H., and Brecher, C. (2020). A Digital Shadow for the Infrared-based Tape Laying Process of Tailored Blanks out of Thermoplastic Unidirectional Tape. *Procedia CIRP*, 85:221–226.

[Sepasgozar, 2021] Sepasgozar, S. M. E. (2021). Differentiating Digital Twin from Digital Shadow: Elucidating a Paradigm Shift to Expedite a Smart, Sustainable Built Environment. *Buildings*, 11(4):151.

[Shen et al., 2020] Shen, Y., Guo, D., Long, F., Mateos, L. A., Ding, H., Xiu, Z., Hellman, R. B., King, A., Chen, S., Zhang, C., and Tan, H. (2020). Robots Under COVID-19 Pandemic: A Comprehensive Survey. *IEEE Access*, 9:1590–1615.

[Shirowzhan et al., 2020] Shirowzhan, S., Tan, W., and Sepasgozar, S. M. (2020). Digital twin and CyberGIS for improving connectivity and measuring the impact of infrastructure construction planning in smart cities. *ISPRS International Journal of Geo-Information*, 9(4).

[Simon et al., 2020] Simon, G., Hantos, G. B., Patel, M. S., Tweedie, A., and Harvey, G. (2020). Machine Learning Enabled FBAR Digital Twin for Rapid Optimization. *IEEE International Ultrasonics Symposium, IUS*, 2020-Septe:33–36.

[Singgih, 2021] Singgih, I. K. (2021). Production Flow Analysis in a Semiconductor Fab Using Machine Learning Techniques. *Processes*, 9(3):407.

[Söderberg et al., 2018] Söderberg, R., Wärmefjord, K., Madrid, J., Lorin, S., Forslund, A., and Lindkvist, L. (2018). An information and simulation framework for increased quality in welded components. *CIRP Annals*, 67(1):165–168.

[Sommer et al., 2019] Sommer, M., Stjepandić, J., Stobrawa, S., and von soden, M. (2019). Automatic generation of digital twin based on scanning and object recognition. *Advances in Transdisciplinary Engineering*, 10:645–654.

[Stavropoulos et al., 2020] Stavropoulos, P., Papacharalampopoulos, A., and Athanasopoulou, L. (2020). A molecular dynamics based digital twin for ultrafast laser material removal processes. *International Journal of Advanced Manufacturing Technology*, 108(1-2):413–426.

[Stehling et al., 2017] Stehling, M., Schmiedinger, T., Affenzeller, P., and ... (2017). Why Robots do not matter! Using Digital Twin and Augmented Learning for Continuous Improvement in the context of Manufacturing. *Researchgate.Net*, November 2018.

[Stieber et al., 2020] Stieber, S., Hoffmann, A., Schiendorfer, A., Reif, W., Beyrle, M., Faber, J., Richter, M., and Sause, M. (2020). Towards Real-time Process Monitoring and Machine Learning for Manufacturing Composite Structures. *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, 2020-Septe:1455–1458.

[Stojanovic and Milenovic, 2019] Stojanovic, N. and Milenovic, D. (2019). Data-driven Digital Twin approach for process optimization: An industry use case. *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, pages 4202–4211.

[Su et al., 2021] Su, S., Zhao, G., Xiao, W., Yang, Y., and Cao, X. (2021). An image-based approach to predict instantaneous cutting forces using convolutional neural networks in end milling operation. *The International Journal of Advanced Manufacturing Technology*, 115(5-6):1657–1669.

[Sun et al., 2020] Sun, X., Bao, J., Li, J., Zhang, Y., Liu, S., and Zhou, B. (2020). A digital twin-driven approach for the assembly-commissioning of high precision products. *Robotics and Computer-Integrated Manufacturing*, 61(March 2019):1–14.

[Surico et al., 2020] Surico, M., Ricatto, R., Merlo, A., Németh, I., Sardelis, A., Villoslada, M., Montejo, E., Frenkel, N., Aivaliotis, P., de la Pera Celada, I., Sidiropoulos, J., Eytan, A., Papavasileiou, A., and Aggogeri, F. (2020). PROGRAMS project approach to maintenance management. *IFAC-PapersOnLine*, 53(3):313–318.

[Talkhestani et al., 2018] Talkhestani, B. A., Jazdi, N., Schloegl, W., and Weyrich, M. (2018). Consistency check to synchronize the Digital Twin of manufacturing automation based on anchor points. *Procedia CIRP*, 72:159–164.

[Tao et al., 2018a] Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., and Sui, F. (2018a). Digital twin-driven product design, manufacturing and service with big data. *International Journal of Advanced Manufacturing Technology*, 94(9-12):3563–3576.

[Tao et al., 2019a] Tao, F., Sui, F., Liu, A., Qi, Q., Zhang, M., Song, B., Guo, Z., Lu, S. C., and Nee, A. Y. (2019a). Digital twin-driven product design framework. *International Journal of Production Research*, 57(12):3935–3953.

[Tao et al., 2019b] Tao, F., Zhang, H., Liu, A., and Nee, A. Y. (2019b). Digital Twin in Industry: State-of-the-Art. *IEEE Transactions on Industrial Informatics*, 15(4):2405–2415.

[Tao and Zhang, 2017] Tao, F. and Zhang, M. (2017). Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing. *IEEE Access*, 5:20418–20427.

[Tao et al., 2018b] Tao, F., Zhang, M., Liu, Y., and Nee, A. Y. (2018b). Digital twin driven prognostics and health management for complex equipment. *CIRP Annals*, 67(1):169–172.

[Tipary and Erdős, 2021] Tipary, B. and Erdős, G. (2021). Generic development methodology for flexible robotic pick-and-place workcells based on Digital Twin. *Robotics and Computer-Integrated Manufacturing*, 71.

[Todorov et al., 2012] Todorov, E., Erez, T., and Tassa, Y. (2012). MuJoCo: A physics engine for model-based control. *IEEE International Conference on Intelligent Robots and Systems*, pages 5026–5033.

[Tong et al., 2020] Tong, X., Liu, Q., Pi, S., and Xiao, Y. (2020). Real - time machining data application and service based on IMT digital twin. *Journal of Intelligent Manufacturing*, 31(5):1113–1132.

[Tozanli et al., 2020] Tozanli, O., Kongar, E., and Gupta, S. M. (2020). Evaluation of waste electronic product trade-in strategies in predictive twin disassembly systems in the era of blockchain. *Sustainability (Switzerland)*, 12(13).

[Tseng et al., 2019] Tseng, G. W. G., Chen, C. Q. G., Erkorkmaz, K., and Engin, S. (2019). Digital shadow identification from feed drive structures for virtual process planning. *CIRP Journal of Manufacturing Science and Technology*, 24:55–65.

[Um et al., 2017] Um, J., Weyer, S., and Quint, F. (2017). Plug-and-Simulate within Modular Assembly Line enabled by Digital Twins and the use of AutomationML. *IFAC-PapersOnLine*, 50(1):15904–15909.

[Ünal-Saewe et al., 2020] Ünal-Saewe, T., Vedder, C., Vervoort, S., and Schleifenbaum, J. H. (2020). Digitaler Zwilling im Produktlebenszyklus additiv gefertigter Komponenten. In Frenz, W., editor, *Handbuch Industrie 4.0: Recht, Technik, Gesellschaft*, pages 591–602. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Vallachira et al., 2020] Vallachira, S., Orkisz, M., Norrlof, M., and Butail, S. (2020). Data-driven gearbox failure detection in industrial robots. *IEEE Transactions on Industrial Informatics*, 16(1):193–201.

[Verner et al., 2018] Verner, I., Cuperman, D., Fang, A., Reitman, M., Romm, T., and Balikin, G. (2018). Robot online learning through digital twin experiments: A weightlifting project. *Lecture Notes in Networks and Systems*, 22:307–314.

[Verner et al., 2019] Verner, I., Cuperman, D., Gamer, S., and Polishuk, A. (2019). Training robot manipulation skills through practice with digital twin of Baxter. *International journal of online and biomedical engineering*, 15(9):58–70.

[Verner et al., 2020] Verner, I., Cuperman, D., Gamer, S., and Polishuk, A. (2020). *Digital Twin of the Robot Baxter for Learning Practice in Spatial Manipulation Tasks*, volume 80. Springer International Publishing.

[Verner et al., 2017] Verner, I. M., Cuperman, D., and Reitman, M. (2017). Robot online learning to lift weights: A way to expose students to robotics and intelligent technologies. *International Journal of Online Engineering*, 13(8):174–182.

[Vrabič et al., 2018] Vrabič, R., Erkoyuncu, J. A., Butala, P., and Roy, R. (2018). Digital twins: Understanding the added value of integrated models for through-life engineering services. *Procedia Manufacturing*, 16:139–146.

[vZidek et al., 2020] vZidek, K., Modrák, V., Pitel, J., and Šoltysová, Z. (2020). The digitization of quality control operations with cloud platform computing technologies. In *Industry 4.0 for SMEs*, pages 305–334. Palgrave Macmillan, Cham.

[Wagg et al., 2020] Wagg, D. J., Worden, K., Barthorpe, R. J., and Gardner, P. (2020). Digital Twins: State-of-The-Art and Future Directions for Modeling and Simulation in Engineering Dynamics Applications. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering*, 6(3):1–17.

[Wagner et al., 2017] Wagner, C., Grothoff, J., Epple, U., Drath, R., Malakuti, S., Grüner, S., Hoffmeister, M., and Zimermann, P. (2017). The role of the Industry 4.0 asset administration shell and the digital twin during the life cycle of a plant. *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA*, pages 1–8.

[Wagner et al., 2020] Wagner, R., Haefner, B., Biehler, M., and Lanza, G. (2020). Digital DNA in quality control cycles of high-precision products. *CIRP Annals*, 69(1):373–376.

[Wagner et al., 2018] Wagner, R., Haefner, B., and Lanza, G. (2018). Function-oriented quality control strategies for high precision products. *Procedia CIRP*, 75:57–62.

[Wagner et al., 2019] Wagner, R., Schleich, B., Haefner, B., Kuhnle, A., Wartzack, S., and Lanza, G. (2019). Challenges and potentials of digital twins and industry 4.0 in product design and production for high performance products. *Procedia CIRP*, 84:88–93.

[Wan et al., 2019] Wan, L., Nochta, T., and Schooling, J. M. (2019). Developing a city-level digital twin - Propositions and a case study. *International Conference on Smart Infrastructure and Construction 2019, ICSIC 2019: Driving Data-Informed Decision-Making*, 2019:187–193.

[Wang et al., 2020a] Wang, C. P., Erkorkmaz, K., McPhee, J., and Engin, S. (2020a). In-process digital twin estimation for high-performance machine tools with coupled multibody dynamics. *CIRP Annals*, 69(1):321–324.

[Wang et al., 2019] Wang, J., Ye, L., Gao, R. X., Li, C., and Zhang, L. (2019). Digital Twin for rotating machinery fault diagnosis in smart manufacturing. *International Journal of Production Research*, 57(12):3920–3934.

[Wang and Luo, 2021] Wang, P. and Luo, M. (2021). A digital twin-based big data virtual and real fusion learning reference framework supported by industrial internet towards smart manufacturing. *Journal of Manufacturing Systems*, 58(PA):16–32.

[Wang et al., 2021] Wang, Q., Jiao, W., Wang, P., and Zhang, Y. (2021). Digital Twin for Human-Robot Interactive Welding and Welder Behavior Analysis. *IEEE/CAA Journal of Automatica Sinica*, 8(2):334–343.

[Wang et al., 2020b] Wang, T., Cheng, J., Yang, Y., Esposito, C., Snoussi, H., and Tao, F. (2020b). Adaptive Optimization Method in Digital Twin Conveyor Systems via Range-Inspection Control. *IEEE Transactions on Automation Science and Engineering*, pages 1–9.

[Wang et al., 2020c] Wang, T., Li, J., Kong, Z., Liu, X., Snoussi, H., and Lv, H. (2020c). Digital twin improved via visual question answering for vision-language interactive mode in human–machine collaboration. *Journal of Manufacturing Systems*, July:0–1.

[Wang et al., 2020d] Wang, X., Liang, C.-J., Menassa, C., and Kamat, V. (2020d). Real-Time Process-Level Digital Twin for Collaborative Human-Robot Construction Work. *Proceedings of the 37th International Symposium on Automation and Robotics in Construction (ISARC)*, October 2020.

[Wang and Wang, 2019] Wang, X. V. and Wang, L. (2019). Digital twin-based WEEE recycling, recovery and remanufacturing in the background of Industry 4.0. *International Journal of Production Research*, 57(12):3892–3902.

[Wang et al., 2020e] Wang, Y., Wang, S., Yang, B., Zhu, L., and Liu, F. (2020e). Big data driven Hierarchical Digital Twin Predictive Remanufacturing paradigm: Architecture, control mechanism, application scenario and benefits. *Journal of Cleaner Production*, 248:119299.

[Wang et al., 2020f] Wang, Z., Liu, Q., Chen, H., and Chu, X. (2020f). A deformable CNN-DLSTM based transfer learning method for fault diagnosis of rolling bearing under multiple working conditions. *International Journal of Production Research*, 0(0):1–15.

[Ward et al., 2021] Ward, R., Sencer, B., Jones, B., and Ozturk, E. (2021). Accurate prediction of machining feedrate and cycle times considering interpolator dynamics. *The International Journal of Advanced Manufacturing Technology*, 116(1-2):417–438.

[Waschneck et al., 2018] Waschneck, B., Reichstaller, A., Belzner, L., Altenmüller, T., Bauernhansl, T., Knapp, A., and Kyek, A. (2018). Optimization of global production scheduling with deep reinforcement learning. *Procedia CIRP*, 72:1264–1269.

[Weigelt et al., 2019] Weigelt, M., Kink, J., Mayr, A., Lindenfels, J. V., Kuhl, A., and Franke, J. (2019). Digital twin of the linear winding process based on explicit finite element method. *2019 9th International Electric Drives Production Conference, EDPC 2019 - Proceedings*.

[West and Pyster, 2015] West, T. D. and Pyster, A. (2015). Untangling the Digital Thread: The Challenge and Promise of Model-Based Engineering in Defense Acquisition. *Insight*, 18(2):45–55.

[Wohlfeld et al., 2017] Wohlfeld, D., Weiss, V., and Becker, B. (2017). Digital Shadow – From production to product. In Bargende, M., Reuss, H.-C., and Wiedemann, J., editors, *17. Internationales Stuttgarter Symposium*, pages 783–794, Wiesbaden. Springer Fachmedien Wiesbaden.

[Wunderlich et al., 2018] Wunderlich, C., Tschöpe, C., and Duckhorn, F. (2018). Advanced methods in NDE using machine learning approaches. *AIP Conference Proceedings*, 1949(April 2018).

[Xi et al., 2021] Xi, T., Benincá, I. M., Kehne, S., Fey, M., and Brecher, C. (2021). Tool wear monitoring in roughing and finishing processes based on machine internal data. *International Journal of Advanced Manufacturing Technology*.

[Xia et al., 2020] Xia, K., Sacco, C., Kirkpatrick, M., Saidy, C., Nguyen, L., Kircaliali, A., and Harik, R. (2020). A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence. *Journal of Manufacturing Systems*, June:1–21.

[Xie et al., 2020] Xie, N., Kou, R., and Yao, Y. (2020). Tool condition prognostic model based on digital twin system. *Procedia CIRP*, 93:1502–1507.

[Xu and Guo, 2021] Xu, J. and Guo, T. (2021). Application and research on digital twin in electronic cam servo motion control system. *International Journal of Advanced Manufacturing Technology*, pages 1145–1158.

[Xu et al., 2020] Xu, W., Cui, J., Li, L., Yao, B., Tian, S., and Zhou, Z. (2020). Digital twin-based industrial cloud robotics: Framework, control approach and implementation. *Journal of Manufacturing Systems*, July:0–1.

[Xu et al., 2019] Xu, Y., Sun, Y., Liu, X., and Zheng, Y. (2019). A Digital-Twin-Assisted Fault Diagnosis Using Deep Transfer Learning. *IEEE Access*, 7:19990–19999.

[Yacob et al., 2019] Yacob, F., Semere, D., and Nordgren, E. (2019). Anomaly detection in Skin Model Shapes using machine learning classifiers. *International Journal of Advanced Manufacturing Technology*, 105(9):3677–3689.

[Yan et al., 2018] Yan, K., Xu, W., Yao, B., Zhou, Z., and Pham, D. T. (2018). *Digital Twin-Based Energy Modeling of Industrial Robots*, volume 946. Springer Singapore.

[Yiping et al., 2021] Yiping, G., Xinyu, L., and Gao, L. (2021). A Deep Lifelong Learning Method for Digital-Twin Driven Defect Recognition With Novel Classes. *Journal of Computing and Information Science in Engineering*, 21(3):1–9.

[Yu et al., 2016] Yu, J., Yuan, J., Wu, Z., and Tan, M. (2016). Data-Driven Dynamic Modeling for a Swimming Robotic Fish. *IEEE Transactions on Industrial Electronics*, 63(9):5632–5640.

[Zambal et al., 2018] Zambal, S., Eitzinger, C., Clarke, M., Klintworth, J., and Mechin, P.-y. (2018). Effects of defects analysis based on manufacturing data. *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*, pages 803–808.

[Zambrano et al., 2020] Zambrano, V., Rodríguez-Barrachina, R., Calvo, S., and Izquierdo, S. (2020). TWINKLE: A digital-twin-building kernel for real-time computer-aided engineering. *SoftwareX*, 11:100419.

[Zhang et al., 2020a] Zhang, C., Zhou, G., Hu, J., and Li, J. (2020a). Deep learning-enabled intelligent process planning for digital twin manufacturing cell. *Knowledge-Based Systems*, 191.

[Zhang et al., 2020b] Zhang, K., Qu, T., Zhou, D., Jiang, H., Lin, Y., Li, P., Guo, H., Liu, Y., Li, C., and Huang, G. Q. (2020b). Digital twin-based opti-state control method for a synchronized production operation system. *Robotics and Computer-Integrated Manufacturing*, 63(November 2019):101892.

[Zhang et al., 2020c] Zhang, L., Chen, X., Zhou, W., Cheng, T., Chen, L., Guo, Z., Han, B., and Lu, L. (2020c). Digital twins for additive manufacturing: A state-of-the-art review. *Applied Sciences (Switzerland)*, 10(23):1–10.

[Zhang et al., 2020d] Zhang, M., Tao, F., and Nee, A. Y. (2020d). Digital Twin Enhanced Dynamic Job-Shop Scheduling. *Journal of Manufacturing Systems*, April:0–1.

[Zhang et al., 2018] Zhang, M., Zuo, Y., and Tao, F. (2018). Equipment Energy Consumption Management in Applications. *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, pages 1–5.

[Zhang et al., 2021] Zhang, X., Liu, L., Wan, X., and Feng, B. (2021). Tool Wear Online Monitoring Method Based on DT and SSAE-PHMM. *Journal of Computing and Information Science in Engineering*, pages 1–18.

[Zhao et al., 2021] Zhao, Y., Guo, J., Bai, C., and Zheng, H. (2021). Reinforcement Learning-Based Collision Avoidance Guidance Algorithm for Fixed-Wing UAVs. *Complexity*, 2021.

[Zhao et al., 2020] Zhao, Z., Wang, S., Wang, Z., Wang, S., Ma, C., and Yang, B. (2020). Surface roughness stabilization method based on digital twin-driven machining parameters self-adaption adjustment: a case study in five-axis machining. *Journal of Intelligent Manufacturing*.

[Zheng et al., 2019] Zheng, Y., Yang, S., and Cheng, H. (2019). An application framework of digital twin and its case study. *Journal of Ambient Intelligence and Humanized Computing*, 10(3):1141–1153.

[Zhou et al., 2020] Zhou, G., Zhang, C., Li, Z., Ding, K., and Wang, C. (2020). Knowledge-driven digital twin manufacturing cell towards intelligent manufacturing. *International Journal of Production Research*, 58(4):1034–1051.

[Zhou et al., 2021] Zhou, Y., Xing, T., Song, Y., Li, Y., Zhu, X., Li, G., and Ding, S. (2021). Digital-twin-driven geometric optimization of centrifugal impeller with free-form blades for five-axis flank milling. *Journal of Manufacturing Systems*, 58(June):22–35.

[Zotov et al., 2020] Zotov, E., Tiwari, A., and Kadirkamanathan, V. (2020). Towards a Digital Twin with Generative Adversarial Network Modelling of Machining Vibration. In *International Conference on Engineering Applications of Neural Networks*, pages 190–201. Springer.

[Zushida et al., 2020] Zushida, K., Haohao, Z., Shimamura, H., Motegi, K., and Shiraishi, Y. (2020). Estimation of Lawn Grass Lengths based on Random Forest Algorithm for Robotic Lawn Mower. *2020 59th Annual Conference of the Society of Instrument and Control Engineers of Japan, SICE 2020*, pages 1628–1633.