From Internal Response to External Intent: Data-Efficient Machine Learning for Human State, Motion, and Embodied Intelligence

DISSERTATION

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY
(Electrical and Computer Engineering)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Sarmad Mehrdad

January 2026

From Internal Response to External Intent: Data-Efficient Machine Learning for Human State, Motion, and Embodied Intelligence

DISSERTATION

Submitted in Partial Fulfillment of

the Requirements for

the Degree of

DOCTOR OF PHILOSOPHY (Electrical and Computer Engineering)

at the

NEW YORK UNIVERSITY
TANDON SCHOOL OF ENGINEERING

by

Sarmad Mehrdad

January 2026

Approved:



Department Chair Signature

December 31, 2025

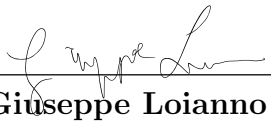
Date

Approved by the Guidance Committee:

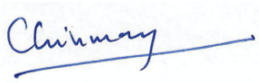
Major: PhD Electrical and Computer Engineering



Ludovic Righetti
Professor
NYU Tandon School of Engineering
12/24/2025
Date



Giuseppe Loianno
Associate Professor
University of California Berkeley
Research Associate Professor
NYU Tandon School of Engineering
12/28/2025
Date



Chinmay Hegde
Associate Professor
NYU Tandon School of Engineering
12/24/2025
Date



Vincent Bonnet
Associate Professor
University of Toulouse
LAAS-CNRS
12/28/2025
Date

Copies of this dissertation are obtainable from

UMI Dissertation Publishing
ProQuest CSA
789 E. Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Vita

Sarmad Mehrdad received the Bachelor of Science degree in Mechanical Engineering from K. N. Toosi University of Technology, Tehran, Iran, in August 2015. He subsequently moved to the United States to pursue graduate studies and earned the Master of Science degree in Mechanical Engineering in August 2019. During his master's studies, he worked on the development of affordable educational platforms for mobile robotics. He also served as a teaching assistant for both undergraduate and graduate courses and as an instructor for undergraduate mechanical engineering laboratory courses.

He began his doctoral studies in June 2020 at New York University, Tandon School of Engineering, where he pursued the Ph.D. degree in Electrical and Computer Engineering, which he completed in January 2026. During his doctoral studies, he was the recipient of multiple fellowships from New York University. He served as the lead instructor for robotics in two consecutive NYU K–12 summer programs and as a teaching assistant for both undergraduate and graduate-level courses. His research focused primarily on learning from human demonstrations and objective learning, with an emphasis on human intent inference and robotic optimal cost learning. In addition to his primary research direction, he contributed to projects in wearable devices, biosignal analysis involving human subjects, and soft robotics. He is a co-inventor on a patent resulting from his work on wearable technologies. His research has been published in several peer-reviewed conferences and journals. He has also served as a technical reviewer for multiple robotics publications, including IEEE Transactions on Robotics and IEEE Robotics and Automation Letters.

Acknowledgements

First and foremost, this thesis could not have been accomplished without the constant guidance and care of Prof. Ludovic Righetti. Instead of telling me what to do, he suggested alternatives. Instead of expecting fruits, he allowed fruition. Every meeting was positive; every conversation was engaging and full of ideas. He not only showed me how to be a researcher, but also how to be an advisor. The only thing you are bad at, Ludovic, is accepting a compliment on how great you are. I can never find the words to thank you as deeply as I intend to.

I would also like to thank my committee members, Profs. Chinmay Hegde, Giuseppe Loianno, and Vincent Bonnet, for their support and valuable feedback, which made this thesis what it is today. My sincerest gratitude to you all for agreeing to guide me along this path, and for the many constructive conversations we shared throughout this journey.

I would be remiss not to mention several key figures throughout my PhD. Prof. Ivan Selesnick (former Chair of Electrical and Computer Engineering), Prof. Katsuo Kurabayashi (Chair of Mechanical and Aerospace Engineering), Prof. Vikram Kapila (Director of the PhD Hub), Dean Jelena Kovačević (former Dean of the Tandon School of Engineering), and Jamie Lloyd (Assistant Director of the PhD Hub). Your support is one of the main reasons I did not give up on my PhD during some of the hardest times of my academic journey. Thank you for not giving up on me, and for helping me when I needed it the most.

To all my labmates at the Machines in Motion Lab, thank you for making the lab a safe place for ideas. I learned so much from you through our group meetings and research discussions. In particular, I learned to read papers without knowing who the authors are, so as to relinquish any unconscious bias. Thank you all. And before I forget, thank you, Avadesh, for all the fun and intense conversations about IRL research. I could never keep up with your machine-gun mouth, your super-brain, and your unconditional candidness.

To my great and amazing friend and labmate, Roohollah Khorrambakht: you are an absolute joy to be around. Any conversation with you is worth remembering. Your witty analyses, your hidden knowledge across countless topics, your eloquence, and your stoic calm in every situation make you the best companion one could ask for. I cannot think of a single instance in which you were not smiling. Thank you

for all the times I felt better after talking to you.

To all the students with whom I share another common denominator—Haoran Zhou, Jerry Cheng, Suzanne Oliver, Eleni Miliotou, Charudatta Mhasde, Han Wu, Mahsa Eslami, Ali Rasteh, Roohollah Khorrambakht, and Ahmadreza Ahmadjou—I am proud to have known you, and I deeply admire your courage.

To my friends who helped me keep my smile through these years—Ahmadreza, Alireza, Omid, Roohollah, Ali's (plural), Farnoosh, Iliya, Nika, and so many others—thank you for all the tea times we shared. We not only shared laughter, but also lived through a horrible war in our home country together. I could not have navigated that time without you.

To my dear Alireza Sarmadi: We did not just share a good portion of our names; we shared countless memories. From the very first day you helped me move in, you have been the most reliable and positive person I could call in times of need. Thank you for being such a beautiful soul.

To all my favorite bands—Blind Guardian, Iron Maiden, Fleshgod Apocalypse, The Black Dahlia Murder, Be'lakor, Ne Obliviscaris, and many others—the cathartic experience of watching you perform live, helped me release so much of the stress I endured throughout these years. Thank you.

To my best friend at NYU, Ahmadreza Ahmadjou: I never thought you would become my best friend the first time I saw you, but you proved me wrong. Thank you for being in my life. Despite the times you drive me absolutely insane, I love you, brother.

To my wonderful family—my dad, Gholamreza Mehrdad; my mom, Maryam Latifimoghaddam; and my sister, Sanam—you were always there for me, unconditionally supporting me every step of the way, from two continents away. I am incredibly lucky to have you as my family, and whoever I am, and however I succeed, is entirely indebted to you. Thank you, Dad, for instilling in me your solid logic. Thank you, Mom, for gifting me your beautiful emotions. Thank you, Sanam, for always forgiving me for missing parts of your golden years of success and adulthood. Shine, always. I am so proud of you, *khahari*.

And lastly, to my best friend in life, Shafagh Keyvanian: I could never have done this without you. Thank you for saying *yes* to me. Thank you for always cheering me up, for constantly looking after me, and for relieving my stress through your unmatched reasoning. Thank you for being the answer to my *whys*. And, of course, thank you for always—always—having my back. I love you.

Sarmad Mehrdad
January 2026

To *Shafagh*

ABSTRACT

From Internal Response to External Intent: Data-Efficient Machine Learning for Human State, Motion, and Embodied Intelligence

by

Sarmad Mehrdad

Advisor: Prof. Ludovic Righetti, Ph.D.

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy (Electrical and Computer Engineering)

January 2026

This thesis investigates how intelligent systems can understand human state, behavior, and intent under conditions of data scarcity, variability, and suboptimality. Robots operating alongside humans must infer internal state, interpret behavioral patterns, and anticipate future actions despite limited sensing and incomplete demonstrations. To address these challenges, this work develops data-efficient machine learning methods across three complementary axes of human-centered modeling: **learning human states**, **learning human behavior**, and **learning human intent**.

First, we study the problem of physiological state inference using minimal temporal data. Based on dilated LSTMs and Temporal Convolutional Networks, we design deep sequence models that predict imminent health deterioration of

COVID-19 patients from only three vital signs and basic demographic information. Trained on 37,006 patients, these models achieve high prognostic performance across 3- to 24-hour horizons, demonstrating that clinically meaningful patterns can be extracted from extremely low-dimensional inputs.

Second, we develop statistically expressive probabilistic models to learn human behavior from small sets of demonstrations. We introduce FiMReSt, a regulated finite mixture of multivariate skew-t kernels, enabling stable, flexible modeling of skewed, heavy-tailed motion data. We also propose Skew-Normal Mixture Regression (SNMR) for learning compact, generalizable trajectory representations, and extend classical Learning-from-Demonstration frameworks by incorporating context-aware optimality scores that mitigate the impact of suboptimal or inconsistent demonstrations. These methods allow robots to learn behavior distributions more accurately, especially in few-shot conditions.

Finally, we address the challenge of inferring human intent through Inverse Reinforcement Learning (IRL). We propose Minimum Observation IRL (MO-IRL), an iterative algorithm capable of recovering latent cost functions using sparse, partially observed human trajectories while exploiting natural non-optimality. Applied to both synthetic systems and biomechanical human-arm models, MO-IRL uncovers meaningful cost structures and enables accurate prediction of human demonstration and motion. We further demonstrate subject-independent cost inference, revealing shared behavioral principles across individuals.

Together, these contributions establish a unified framework for data-efficient learning in embodied intelligence to support robust inference of human states, behaviors, and intents under real-world constraints. The methods developed here help move robotics closer to truly human-centered, adaptive, and interpretable interaction capabilities.

Table of Contents

Vita v

Acknowledgements vi

Abstract x

Table of Contents xii

List of Publications xvi

List of Figures xvii

List of Tables xxiv

List of Appendices xxvi

1 Introduction 1

1.1 Deep Sequence Models 4
1.2 Probabilistic models 6
1.2.1 Model Rigidity 6
1.2.2 Sub-optimality in Human Data 8
1.3 Inverse Reinforcement Learning 9
1.3.1 Ill-posed Optimality - Underdefined Non-Optimality 10
1.3.2 Human Intent Ambiguity 10
1.4 Statement of Purpose 11
1.5 Outline 12

I Learning Human *Signals* **14**

2 Deterioration Prediction of COVID-19 Patients using Deep Learning Methods **16**
2.1 Methodology . . . 20
2.1.1 Data Pre-processing . . . 21
2.1.2 DL-based Deterioration Prediction Model . . . 23
2.2 Results and Discussion . . . 27
2.2.1 Patient Cohort . . . 27
2.2.2 Model Performance . . . 28
2.2.3 Ablation Studies . . . 28
2.2.4 Occlusion Analysis . . . 29
2.2.5 Experimenting with Shorter Observation Windows . . . 31
2.2.6 Comparison with COVID-19 Negative Patient Data . . . 33
2.3 Conclusion . . . 34

II Learning Human *Behavior* **37**

3 FiMReS*t*: Finite Mixture of Multivariate Regulated Skew-*t* Kernels **39**
3.1 Preliminaries . . . 41
3.2 ERM Algorithm . . . 43
3.2.1 E-Step . . . 45
3.2.2 R-Step . . . 47
3.2.3 M-Step . . . 48
3.3 Results . . . 49
3.3.1 Datasets . . . 49
3.3.2 Comparative Study . . . 51
3.3.3 Initial Conditions . . . 58
3.4 Discussions . . . 58

4 SNMR: Skew-Normal Mixture Regression – A flexible probabilistic trajectory generator for Learning from Demonstration **60**
4.1 Preliminaries . . . 61

4.1.1 Probability Density Function . . . 61
4.1.2 EM Algorithm . . . 63
4.1.3 Initialization . . . 65
4.2 Methods . . . 67
4.2.1 SNMR-Ex . . . 69
4.2.2 SNMR-Max . . . 71
4.3 Results . . . 72
4.4 Discussions . . . 79

5 Context-Aware Learning from Suboptimal Kinesthetic Demonstrations 81
5.1 Preliminaries . . . 82
5.2 Methods . . . 84
5.2.1 Context-Aware EM Algorithm . . . 84
5.2.2 Context-based α-functions . . . 87
5.3 Evaluation . . . 87
5.4 Results . . . 90
5.5 Discussions . . . 94

III Learning Human *Intent* 96

6 Cost Function Estimation Using Inverse Reinforcement Learning with Minimal Observations 98
6.1 Preliminaries . . . 100
6.2 Methods . . . 101
6.2.1 Problem Statement . . . 101
6.2.2 Computing a step direction . . . 102
6.2.3 Step Acceptance Method . . . 103
6.2.4 Sub-sampling Trajectories . . . 104
6.2.5 Regularization . . . 104
6.2.6 Moving Window . . . 105
6.3 Results . . . 105
6.3.1 Point Mass Environment . . . 106

6.3.2 Robot Simulation 109
6.4 Discussions 115

7 Human Motion Prediction using Inverse Reinforcement Learning with Minimal Observations 117
7.1 Methods 118
7.1.1 Experimental protocol and mechanical model 118
7.1.2 Optimal control problem 119
7.1.3 Minimum Observation IRL (MO-IRL) 121
7.1.4 Learning and Validation 123
7.2 Results 124
7.3 Discussions 129
7.4 Conclusion 131

8 Conclusions and Future Work 133

A Ablation Study Results 136

B Hierarchical Representation of the Multivariate Skew-t Distribution 139

C Conditional Expected Values of Truncated t-Distribution 141

List of Publications

This dissertation is based on the following publications:

Mehrdad, Sarmad, Avadesh Meduri, and Ludovic Righetti. "Cost Function Estimation Using Inverse Reinforcement Learning with Minimal Observations." IEEE/RSJ IROS (2025).

Mehrdad, Sarmad, Vincent Bonnet, and Ludovic Righetti. "Learning Human Reaching Optimality Principles from Minimal Observation Inverse Reinforcement Learning." Submitted to IEEE RA-L (2026).

Sabbah, Maxime, Filip Bečanović, **Sarmad Mehrdad**, Ludovic Righetti, Bruno Watier, and Vincent Bonnet. "Minimal Observations Inverse Reinforcement Learning for Predicting Human Box-Lifting Motions." In IEEE Humanoids (2025).

Mehrdad, Sarmad, and S. Farokh Atashzar. "FiMReSt: Finite Mixture of Multivariate Regulated Skew-t Kernels–A Flexible Probabilistic Model for Multi-Clustered Data with Asymmetrically-Scattered Non-Gaussian Kernels." arXiv preprint arXiv:2305.09071 (2023).

Mehrdad, Sarmad, Farah E. Shamout, Yao Wang, and S. Farokh Atashzar. "Deep learning for deterioration prediction of COVID-19 patients based on time-series of three vital signs." Scientific reports (2023)

Mehrdad, Sarmad, Yao Wang, and S. Farokh Atashzar. "Perspective: wearable internet of medical things for remote tracking of symptoms, prediction of health anomalies, implementation of preventative measures, and control of virus spread during the era of COVID-19." Frontiers in Robotics and AI (2021).

List of Figures

1.1 Schematic thesis outline 4

1.2 (a) Comparison of t-distribusions with zero mean, unit variance, and various DoF values. When $\nu \to \infty$, the probability density will become a Gaussian distribution. (b) Venn diagram visualizes the relationship between multivariate skew-t distribution over MT, MSN, and Gaussian distributions. $\boldsymbol{\Delta}$ indicates the skewness in distribution. 8

2.1 **Data pre-processing pipeline.** We encode the non-SEQ data and pre-process the SEQ data: (i) normalize via Z-score normalization, (ii) model the time-series using cubic spline interpolation, (iii) and resample at every 15 minutes. 22

2.2 **Architecture of the LSTM-based deep neural network.** The SEQ vital sign network processes the SEQ data through an LSTM-based network, whereas the non-SEQ module encompasses the non-SEQ data. 22

2.3 **Architecture of the TCN-based deep neural network.** The SEQ vital sign network processes the SEQ data through a series of residual TCN blocks followed by a FC layer, while the non-SEQ module processes the non-SEQ data. 23

2.4 **Application of data inclusion and exclusion criteria and distribution of vital signs. (A)** In this flowchart, we illustrate the application of the inclusion and exclusion criteria, where **n** represents the number of patients after each step. **(B)** The boxplot of the vital signs recorded from the patients at the end of the 24-hour input window (t=0) corresponds to the prediction time. **(C)** The boxplot of the vital signs recorded from the patients at the beginning of the 24-hour input window (t=-24). We observe differences between the two groups (evaluated using the T-test), which motivates the design of the proposed temporal model. 25

2.5 **Distribution of samples over time.** We show the number of patients who deteriorated vs. those who did not deteriorate in our final filtered dataset (n=37,006). 26

2.6 Performance results after each phase in the training strategy across all prediction horizons. 30

2.7 **Ablation study results.** Performance results for *RSVS-Net* and *CSVS-Net* (Input: non-SEQ data and SEQ vital sign data), *MLVS-Net* (Input: non-SEQ data and a single set of vital signs), and *nSHS-Net* (Input: non-SEQ only). 31

2.8 **Occlusion analysis: (A)**: Occlusion of the clinical and comorbidity characteristics. Observation: Occlusion of Age decreases performance more significantly than others. **(B)**: Occlusion of SEQ vital sign data. Observation: the HR contributes more significantly to the model performance than the SpO2 and temperature. 32

2.9 **Various observation windows (6 to 24 hours).** The legend of the figures indicates the observation duration. Performance results of the *RSVS-Net* and *CSVS-Net* given different observation duration. 33

2.10 **Performance comparison of *CSVS-Net* trained on COVID-19 Positive and COVID-19 Negative cohorts:** Observation: The accuracy and AUROC metrics of the model trained on the COVID-19 negative cohort are comparable to those of the model trained on the COVID-19 positive cohort, while the AUPRC is significantly lower in comparison. 35

2.11 **Occlusion analysis for COVID-19 Negative Patient Cohort: (A)**: Occlusion of the clinical and comorbidity characteristics. Observation: Occlusion of Age decreases performance more significantly than others. **(B)**: Occlusion of SEQ vital sign data. Observation: SpO2 vital sign has more impact on the model performance than HR and temperature. 35

3.1 Example of DoF instability when fitting a FM-CFUST model over AIS 2D dataset (discussed in Section 3.3.1) 44
3.2 3D plot of DoF and Log-likelihood vs. iteration for fitting the FiMReS*t* and FM-CFUST over the AIS 2D dataset. It can be seen that, unlike FM-CFUST, FiMReS*t* will preserve the converging behavior of the DoFs. 52
3.3 Probability density function visualization of the FiMReS*t* model over AIS 2D dataset. 52
3.4 Comparison of DoF behavior during fitting iterations of FiMReS*t* and FM-CFUST over AIS 3D dataset 53
3.5 Probability density function visualization of the FiMReS*t* kernels over AIS 3D dataset. Higher probabilities are shown by darker red color in the visualization. The outermost 3D layer indicates 1% of the maximum probability produced by the FiMReS*t* model in the 3D space. 54
3.6 Comparison of GMM, SNMM, STMM, FM-CFUST and FiMReS*t* on the synthetic dataset 56
3.7 The comparison between GMM, SNMM, STMM, FM-CFUST, and FiMReS*t* over coherence datasets 57
3.8 DoF convergence of the FiMReS*t* over the AIS 3D model given different initial values. The red lines show the stabilized DoF. . . . 58

4.1 Maximum probability trajectory of a 2-dimensional MSN kernel . . 71
4.2 (A): Synthetic 2-dimensional Dataset, (B): GMM fit over the dataset, Log-Likelihood = 246.487 (C): SNMM fit over the same dataset, Log-Likelihood = 274.893, (D): GMR, SNMR_{EX}, and SNMR_{Max} comparison. 72

4.3 (A): Synthetic 2-dimensional Dataset, (B): GMM fit over the dataset, Log-Likelihood = 577.475 (C): SNMM fit over the same dataset, Log-Likelihood = 700.356, (D): GMR, $SNMR_{EX}$, and $SNMR_{Max}$ comparison. 74
4.4 The comparison of a 4-cluster $SNMR_{Max}$ trajectory with various number of clusters in GMR; (A): 4-cluster GMR, (B): 5-cluster GMR, (C): 6-cluster GMR, (D): 7-cluster GMR, (E): 8-cluster GMR, (F): 9-cluster GMR 75
4.5 (A): Robot Data for 35 demonstrations on the target trajectory. (B): GMR and SNMR comparison (3 clusters) (C): GMR and SNMR comparison (4 clusters) (D): GMR and SNMR comparison (5 clusters) (E): GMR and SNMR comparison (6 clusters) 77
4.6 (A): GMR (3 clusters, 35 demos) vs. SNMR (3 clusters, 35 demos) (B): GMR (3 clusters, 35 demos) vs. SNMR (3 clusters, 15 demos) (C): GMR (3 clusters, 35 demos) vs. SNMR (3 clusters, 5 demos) (D): GMR (5 clusters, 35 demos) vs. SNMR (3 clusters, 5 demos) . 78
4.7 $SNMR_{Max}$ and GMR performance comparison using different number of clusters on robot datasets with different number of demonstrations. GMR and $SNMR_{Max}$ trajectories are depicted by blue and red dashed lines respectively. 79

5.1 The α maps resulting from: (A): the distance from the first object on the left. (B): the distance from the second object in the middle. (C): the distance from the third object on the right. (D): the starting point of the task.(E): the goal of the task. (F): the normalized element-wise product of all α-maps. 90
5.2 (A): The plot of per-demonstration α function for each demonstration in the dataset. (B): Visualization of the per-point α function for each point within each demonstration. 91

5.3 **(A):** Example set of 20 demonstrations. The green circles indicate the obstacles in the task space. **(B):** Per-demonstration approach. The dataset consists of four with collision cases and four without. **(C):** Per-demonstration approach (blue), Per-datapoint approach (black), and baseline GMR (red). The dataset consists of four trajectories with collision cases and one without. **(D):** Few-Shot Learning. The dataset consists of four colliding trajectories. **(E):** One-Shot Learning. The dataset contains only one demonstration, which collides with all three obstacles. **(F):** Randomized Obstacles. The obstacle locations are moved to new locations. The per-datapoint approach generates trajectories that avoid new obstacles in the new task setting. 92

6.1 The performance of the IRL algorithms on three point mass environments (one per column). Results are shown in pairs of rows for MO-IRL, PI^2-IRL, and IS-IRL. Top row of each pair shows the sampled trajectory set for each algorithm. Bottom rows of pairs show the set of OC rollouts based on the learned cost function. The initial optimal trajectory is shown in green in the bottom row. In the bottom rows, rollouts with obstacle collision are shown in red. . 110

6.2 Performance of MO-IRL and IS-IRL throughout their iterations. First column indicates the deviation of the resulting trajectory from optimality, second column shows the cost differences of the optimal and the solved trajectory given the estimated weight set, and the third column shows the cost difference of the optimal and solved trajectory given the optimal weights at each iteration. Vertical dashed lines show the termination of associated algorithms. PI^2-IRL's performance is shown by a horizontal red dashed line. 111

6.3 Comparison of each improving step for modifying the IRL performance. **(a)** shows the algorithm without any step acceptance, regularization, and sub-sampling. **(b)** is the same as (a) with sub-sampling. **(c)** shows (b) with added step acceptance method. **(d)** is (b) but with regularization. **(e)** shows the full version with sub-sampling, regularization, and step acceptance method. 112

6.4 Sampled trajectory sets for the IRL approaches. MO-IRL carries out a different way of sampling that is not necessarily local to the optimal trajectory. PI^2 IRL uses only the initial noisy local rollouts. IS-IRL concludes the iterations with a trajectory set containing both initial local rollouts, and OC's solutions. The green path seen on the left figure is the optimal trajectory. 113
6.5 Iterative performance comparison between MO-IRL and IS-IRL. PI^2-IRL's performance is indicated by the horizontal dashed red line. 114
6.6 Comparison of robot behavior in the same obstacle environment with different goal locations. MO-IRL, IS-IRL, and optimal solution given the desired weights, are shown in blue, yellow, and green respectively. 115
6.7 Comparison of robot behavior in different challenging obstacle environment and goal location. The images are for the same test from different angles for better visibility. The color-coding is the same as in Fig. 6.6. 116

7.1 (a) Biomechanical model definition, showing the beginning and the end of the pointing task. (b) Five different initial postures for the pointing task [24]. 119
7.2 Averaged estimated weights, and cost contribution, across all subjects for initial postures 1-5, produced through SDPD analysis. Plots show a prominent importance of the Joint Acceleration feature in the cost function (Φ_4). 124
7.3 RMSE values boxplot comparison of all investigated cases against the baseline, averaged across all subjects. Red line in each boxplot indicates the mean value. 125
7.4 RMSE values (deg) boxplot comparison of all methods against the baseline, averaged across all initial postures. Red line in each boxplot indicates the mean value. 125
7.5 Weights and cost contributions of general cost weights proposed for the reaching task in SIPI case. The RMSE comparison boxplots between all existing demonstrations and predictions are presented in the figure at the bottom. 127

7.6 Comparison of SDPD, SDPI, SIPI, and baseline predictions of the subject's joint values, against the human demonstration. 128

List of Tables

1.1 **Summary of related work.** Overview of related work on the diagnosis of patients with COVID-19, patient severity assessment, and patient prognosis. 5

2.1 **Overview of patient cohort.** We summarize in this table the patient characteristics, including demographics, and distribution of vital signs, for patients who deteriorated and patients who did not deteriorate. 18
2.2 **Model performance.** Performance summary of the proposed networks for three phases of training at all horizons. 29

3.1 Log-Likelihood of various mixture models over the AIS 2D dataset . 51
3.2 Log-Likelihood of the various mixture modeling techniques over AIS 3D dataset 53
3.3 Defining parameters of the FiMReS*t* model given the AIS 3D dataset 54
3.4 Log-Likelihood of the synthetic dataset given different models . . . 55
3.5 Log-likelihood of GMM, SNMM, STMM, FM-CFUST, and FiMReS*t* over coherence parts 1, 2, and 3 57

6.1 Point Mass costs given optimal cost function 109
6.2 Point Mass IRL computation time (sec) 112
6.3 Robot Simulation costs given optimal cost function 115

7.1 (Discretized) Biomechanical cost functions 121
7.2 RMSE results of SDPD weights for reaching task generated by MO-IRL, in comparison with the baseline. 126
7.3 RMSE results of SDPI weights for reaching task generated by MO-IRL, in comparison with the baseline. 126

7.4 RMSE results of SIPI weights for reaching task generated by MO-IRL, in comparison with the baseline. 129

A.1 Numerical comparison of *CSVS-Net*, *RSVS-Net*, *MLVS-Net*, and *nSHS-Net* for 3-24 hours of prediction horizons 136

A.2 Numerical comparison of all the occlusions on clinical and comorbidity characteristics and SEQ vital sign data for 3-24 hours of prediction horizons for *RSVS-Net* 137

A.3 Numerical comparison of all the occlusions on clinical and comorbidity characteristics and SEQ vital sign data for 3-24 hours of prediction horizons for *CSVS-Net* 138

C.1 Description of the notations 142

List of Appendices

Ablation Study Results 136
Hierarchical Representation of the Multivariate Skew-t Distribution 139
Conditional Expected Values of Truncated t-Distribution 141

Chapter 1

Introduction

Robots are increasingly expected to operate in spaces shared with humans, collaborate with them, and adapt to the complexities of real-world environments. As these expectations grow, so does the need for learning frameworks that allow robots to understand the people they interact with. This goes beyond the physical motion, towards the deeper levels of physiology, behavior, and intent. Human-centered robotics therefore requires more than control policies or perceptual pipelines; it requires computational models capable of capturing how humans function, how they act, and why they choose the actions they do. Designing such models is profoundly challenging because humans exhibit immense variability across individuals, contexts, and tasks, and because the available data about human behavior or internal state is often sparse, noisy, or incomplete.

Robotic systems, however, do not have the luxury of perfect sensing or unlimited data. They must interpret human actions and states (sometimes in real time), often in the presence of uncertainty, partial observability, and strict safety or performance constraints. This reality motivates a shift toward structured, data-efficient learning methods that exploit both physical insight and statistical modeling, enabling robust inference even when demonstrations or physiological measurements are limited. Instead of relying solely on high-dimensional datasets or end-to-end neural models, which may fail to generalize across individuals or degrade under distribution shift, robotics benefits from designing learning algorithms that incorporate domain knowledge, biomechanical structure, and interpretable cost formulations.

A central challenge is that humans must be understood at multiple levels. At

the level of **internal sginal**, physiological variations, neuromuscular activation patterns, and biomarker dynamics shape how people feel, behave, and perform tasks. Understanding these internal processes can enable robots and automated systems to monitor health, assess risk, or provide personalized feedback. At the level of **external behavior**, humans express their goals, preferences, and strategies through motion. Replicating this motion, and inferring the human expectation of the motion is essential for seamless human–robot collaboration. Ultimately, at the **human intent** level, humans evaluate the best scenario of motion given optimal objectives given their preference and dynamics. Understanding the intent, would not only predict human motion itself, but also explains *why* the motion takes place in the manner it does. These three levels are deeply connected: understanding human physiology provides insights about capability and state, while understanding behavior and intent allows robots to anticipate what a person may do next. A robotics system capable of leveraging these perspectives is better equipped to operate safely, effectively, and intelligently.

This thesis adopts this perspective by organizing its contributions into three complementary directions: **learning human *signals***, **learning human *behavior***, and **learning human *intent***. Although these domains appear different, they share a unifying focus on *understanding humans from limited data*, and on extracting meaningful structure that can support prediction, interpretation, and ultimately robotic decision-making.

The first direction, *learning human signals*, addresses how **artificial intelligence (AI)** methods can be leveraged to interpret human physiological signals and biomarkers in settings where data scarcity and uncertainty are prevalent. As demonstrated during the COVID-19 pandemic and in broader biomedical contexts, physiological datasets often contain noise, missing values, and severe imbalance between healthy and pathological samples. Yet automated systems must still extract meaningful patterns to support prognosis, early detection, or personalized intervention. By developing models capable of learning under such constraints, we gain tools that deepen our computational understanding of human internal state — a critical foundation for future embodied systems that must recognize fatigue, stress, impairment, or health deterioration before interacting physically.

The second direction, *learning human behavior*, focuses on how robots can learn

behavior and infer intent directly from human demonstrations. This dimension is central to embodied intelligence: robots should not merely mimic human trajectories, but understand the underlying decision-making principles that produce them. To this end, probabilistic modeling frameworks allow us to capture the distributional structure of human movement, especially when only a small number of demonstrations are available. Extensions of classical **Learning from Demonstration (LfD)** techniques — including more expressive, skewed, or higher-order variants of Gaussian mixture models — enable richer representations of human motion, better regression performance under limited data, and improved generalization across individuals.

The third direction, *learning human intent*, goes beyond behavior imitation, where **Inverse Reinforcement Learning (IRL)** offers a principled framework for learning *why* humans move the way they do by identifying latent cost functions that best explain observed actions. Such formulations, when paired with biomechanical models and optimal control theory, allow robots to predict human intent, anticipate motion evolution, and adapt their assistance or collaboration strategies accordingly.

These three learning axes form a coherent progression toward human-centered robotics and learning (Figure 1.1). They reflect a shift from interpreting human signals to interpreting human purpose, and together provide a multilayered modeling foundation for intelligent robots that interact closely with people. Moreover, across both axes runs a common methodological thread: the pursuit of learning frameworks that remain robust and expressive despite limited data, variable human characteristics, and real-world constraints. Whether analyzing physiological biomarkers or inferring intent from minimal motion observations, the goal is to design models that use structure to compensate for scarcity. Although the thesis spans two different settings, it follows a coherent methodological progression under the umbrella of machine learning.

- **Deep Sequence Models**: Learning internal state from physiological signals
- **Probabilistic models**: Learning behavior from human demonstrations
- **Inverse Reinforcement Learning**: Learning human intent and cost functions

Crucially, all three are used in **data-efficient** regimes.

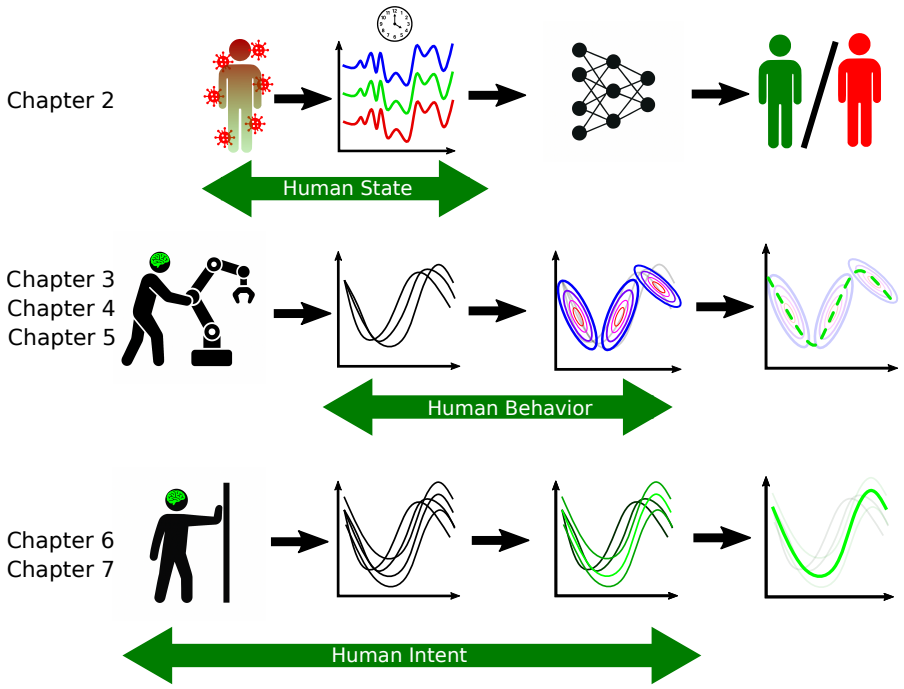


Figure 1.1: Schematic thesis outline

1.1 Deep Sequence Models

The significant shock imposed by the novel coronavirus (COVID-19) pandemic fundamentally challenged the delivery and management of health care services globally [187]. According to the World Health Organization, more than 620 million patients have been diagnosed with COVID-19 as of October 2022, and there are around 6.52 million deaths [3]. Since March 2020, 96.2 million patients have been admitted to emergency departments across the United States [154].

Patients with COVID-19 can experience rapid deterioration entailing the need for invasive measures associated with high morbidity or mortality [173]. During the pandemic, patient prognosis was challenging, especially in the early days when the knowledge about the disease was limited, and any modifications in admission protocols could significantly alter the patient outcomes [188]. This highlighted the importance of routine patient monitoring to ensure that patients with the highest risk of deterioration receive early attention [175].

Due to the saturation of healthcare systems and concerns over unnecessary exposure, many outpatients or those in nursing centers were advised to monitor

Table 1.1: **Summary of related work.** Overview of related work on the diagnosis of patients with COVID-19, patient severity assessment, and patient prognosis.

Reference	**Task**	**Data**	**Machine learning model**
[7, 43, 48, 70, 76, 85, 89, 99, 113, 124, 134, 138, 143, 162, 165, 166, 181, 186, 189, 190, 199, 201]	Diagnosis	CXR, CT Scan, EHR, CCC	CNN, GBM, VGG19, APACHE, ResNet50, VAE, LSTM
[14, 44, 68, 160, 197, 204]	Severity assessment	CXR, CCC	SVM, DenseNet, CNN, MLP
[12, 15, 29, 55, 131, 167]	Prognosis	CXR, CT Scan	CNN, DenseNet121-FPN, GLM, GBM, XGBoost, DT, AlexNet, Inception-V4

symptoms remotely and report through telemedicine [28]. Hence, patients would avoid visiting emergency care facilities unless symptoms were considered significantly severe and would require immediate and specialized attention [179]. Although this practice could reduce exposure and unnecessary loads on emergency services [137], it could also result in poor patient prognosis. In fact, for some patients, especially those with comorbidities, the development of symptoms was followed by sudden, drastic, and unexpected deterioration resulting in morbidity, even after discharge from a clinic [62].

Considering the scale of data gathered from a plethora of patients with COVID-19 admitted to emergency departments worldwide, many Deep Learning (DL) and Machine Learning (ML) methods were developed for early diagnosis [11], patient severity assessment [104], or prognosis prediction [161, 196]. These methods include the use of a Convolutional Neural Network (CNN), Variational Auto-Encoder (VAE), Gradient Boosting Machine (GBM), Multi-Layer Perceptron (MLP), Decision Tree (DT), Generalized Linear Model (GLM), Support Vector Machine (SVM), and Long-Short Term Memory (LSTM), to name a few. The datasets used in the aforementioned studies can range from Electronic Health Records (EHR) and Clinical and Comorbidity characteristics (CCC) to imaging data such as Chest X-Ray (CXR) and Computed Tomography (CT) Scan. In Table 1.1, we summarize relevant work based on the choice of datasets and models for the different prediction

tasks.

1.2 Probabilistic models

Learning from demonstration based on probabilistic models has attracted a great deal of interest recently in robotics as a technique to train autonomous systems the motion skills of a human in performing a task after capturing a set of observations by trainer [13, 27, 42, 150, 163]. This approach relaxes the need for expert knowledge to be always present during operation since it incorporates human intuition into the task execution. Based on this approach, data (often recorded as motions of the robot) prescribed by the human operator during the demonstration phase is statistically modeled and generalized to capture the human's expectation of a robot's behavior conducting a task [53, 126]. This modeling is done through fitting a mixture of probabilistic clusters to the dataset, using an iterative algorithm called Expectation-Maximization (EM) [50], which maximizes the expectation of log-likelihood as the measure of model's accuracy.

1.2.1 Model Rigidity

Clustering algorithms have been utilized broadly in the literature for unsupervised learning, decoding the inherent clusters of a given data or signal space to better understand underlying hidden nature and patterns and to also detect anomalies, outliers, and specific abnormal trends [33, 57, 112, 200]. The applications can range from fault detection in industries [73, 83, 195], up to cancer detection in health sectors [47, 86], and even recently in robot learning where clustering methods are used to model the most probable behavior of a human trainer, in the context of learning from demonstration [54, 127, 156, 157]. Classic clustering methods range from simple algorithms such as K-means [67] up to more comprehensive methods using different probabilistic kernels such as Gaussian Mixture Models (GMM). Probabilistic clustering using GMM was first suggested for speaker identification [152, 153, 155] and has gained significant attention from numerous researchers for data segregation in applications such as biological data evaluations [10], robotics [8, 145], and pattern recognition [41, 103]. Recently, GMM-based learning from

demonstration has been adopted in several domains of robotics [84, 91, 114, 118]. Although GMM has shown an acceptable modeling capacity throughout the literature, it is suffering from fundamental constraints. In general, GMM fails to model complex demonstrations. More specifically, the performance (accuracy and generalizability) of GMM-based techniques can be seriously challenged depending on the symmetricity and smoothness of the data clusters. The limitations of GMM are recognized in statistical modeling literature and can challenge complex human-robot interactions. Thus, new fundamental mathematical derivations of an iterative EM algorithm are required to optimize and design a descriptive mixture model.

For this reason, there has been an accelerated surge in proposing non-normal mixture modeling approaches such as mixture models based on generalized hyperbolic distribution [32], general split Gaussian distribution [171], and generalized hyperbolic factor analyzers [192]. Details can be found in [120] and references therein. Lin et al. proposed a solution by designing the skew-normal mixture model (SNMM) [106] based on multivariate skew normal (MSN) distribution and the corresponding explicit EM algorithm, which mathematically relaxes the dependency of mixture modeling on the elliptical nature of Gaussian kernels.

In order to further improve the SNMM technique and to accommodate for the datasets with outliers and heavy tails, noted in the literature [159], a series of efforts was initiated [94] to fundamentally replace the multivariate skew normal kernel with the multivariate skew-t (MST) distribution kernel, which is a skewed version of the multivariate t (MT) distribution. This attempt is due to the higher "flexibility in reach" and generalizability that the skew-t distribution enables by incorporating the statistical degree of freedom (DoF) denoted in the literature by ν. Fig. 1.2 shows the schematic view of the generalizability of the multivariate skew-t over multivariate skew normal, multivariate t, and Gaussian distributions. Thus, ideally, the mixture modeling techniques derived from skew-t distribution would be more reliable for modeling a mixture of clusters subject to skewness, outliers, heavy tails, and data dropout [122].

The EM algorithm for training a mixture of multivariate skew-t distributions cannot be implemented in closed-form since some conditional expected values involved in this approach are mathematically intractable. Thus, there were several attempts (such as [159] and [17]) to train a mixture model based on a simplified

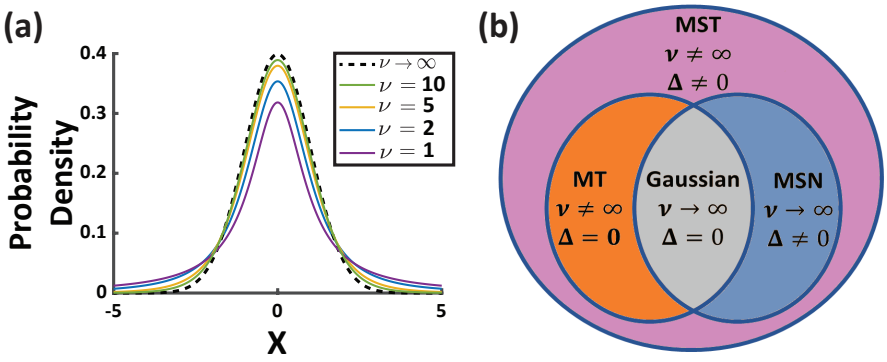


Figure 1.2: (a) Comparison of t-distribusions with zero mean, unit variance, and various DoF values. When $\nu \to \infty$, the probability density will become a Gaussian distribution. (b) Venn diagram visualizes the relationship between multivariate skew-t distribution over MT, MSN, and Gaussian distributions. $\boldsymbol{\Delta}$ indicates the skewness in distribution.

formulation of skew-t distribution, which reduces the dimensionality of the skewness matrix down to unity. Such simplification allowed researchers to propose an explicit solution [35, 147, 184] and thus realize an estimate of skew-t mixture modeling. However, as expected, the aforementioned simplification limited the ability and flexibility of the resulting EM algorithm in modeling the skewness of the kernels.

1.2.2 Sub-optimality in Human Data

Kinesthetic demonstration is a common approach that allows the human to intuitively provide examples of task conduction [13, 150]. In traditional probabilistic LfD, after several demonstrations conducted by a trainer, the robot learning algorithm generates a probabilistic model that encapsulates the most probable representation of the trainer's intuition for task conduction. The model will generate policies needed to control the robot so that the task conduction follows the trainer's intuition. Probabilistic LfD methods have been recognized as a powerful tool in teaching robots while encoding the variability in complex tasks. Some methods include Gaussian Mixture Model-Gaussian Mixture Regression (GMM-GMR)[87, 144], Probabilistic Movement Primitives (ProMP)[115, 142], Kernelized Movement Primitives (KMP)[74, 170], and Gaussian Processes (GP)[49, 164].

Since existing probabilistic LfD methods equally consider all demonstrations and uniformly use the dataset to model the dynamics of the task, their performance

relies heavily on the *quality* of the demonstrated set. Thus, demonstrations should be conducted correctly to satisfy the task's criteria so that the robot correctly grasps the underlying human intuition of the task. It should be noted that the demonstration quality depends on task setting and context [116]. For example, if the task is to move a cup filled with hot water from one location to another, the quality may depend on the smoothness, ability to avoid obstacles in the workspace, and whether or not the goal location was reached. However, other tasks, such as navigating a maze as quickly as possible, may value speed over smoothness. When providing demonstrations, a human cannot always provide the 'highest' quality of demonstrations that meet all requirements. As a result, the trajectories can be suboptimal, there is a possibility of contact with obstacles or deviating from the boundaries of the task. Such incidences may happen only for a portion of a demonstration, or they may affect the majority of one or more demonstration(s). Hence, demonstrations can be subject to outliers and variances in how the task was completed, which adversely affect the overall quality of the dataset [116]. It should be noted that the structure of the environment (for example, the locations of the obstacles) may vary post-demonstration, causing a significant contextual difference between the demonstration condition and the one during the task conduction. Considering the above-mentioned challenges, the question addressed here is how to conduct LfD while relaxing the conventional assumption of perfect task execution by the trainer. In GMM-based LfD methods, the EM algorithm does not consider the quality of the demonstrations nor the context/setting of the task. Therefore, sub-optimal demonstrations need to be manually filtered out from the dataset, if possible.

1.3 Inverse Reinforcement Learning

Regardless of how expressive probabilistic models may be, they are descriptive rather than principled, since they model *what* was done, not *why*. They do not infer the objective function that explains behavior, nor do they generalize well outside observed conditions. This limitation motivates the transition to Inverse Reinforcement Learning (IRL).

1.3.1 Ill-posed Optimality - Underdefined Non-Optimality

IRL and Inverse Optimal Control (IOC) both aim to extract the cost function of a task given the observation(s) of an "expert" or "optimal" task execution [130]. The problem of extracting a cost function from expert observations has several benefits. Primarily, it goes beyond explaining the expert behavior to one environment in a particular task, and tries to explains *why* the task was executed in the observed manner. Moreover, this allows for a generalized behavior of the system facing various environments and inevitable external inputs where simply *mimicking* the expert cannot be fruitful. Because of this, IRL has become attractive for researchers in various fields such as optimal control [34], human intent inference [111], human movement analysis [141] and humanoids [125] (cf. [4, 5] for extended reviews of the topic).

However, extracting an optimized cost function given an expert demonstration is challenging at best. The most important challenge is that there exists multiple cost functions that can explain the observed expert movement, making IRL an ill-posed problem full of local optima. This is in contrast to reinforcement learning (RL) or optimal control (OC), where the cost and task space information is readily available and therefore understood easily. It is also fairly complicated for IRL to sample non-optimal demonstrations for training, and providing expert examples for IRL is not always feasible.

1.3.2 Human Intent Ambiguity

Understanding the optimal principles that govern simple human motions such as arm reaching is important for both neuroscience and robotics. In neuroscience, these principles offer insight into how the central nervous system plans goal-directed movements with a redundant musculoskeletal system and help identify deviations from optimality in clinical populations. In robotics and human–robot interaction, modeling human reaching as an optimal control problem enables bio-inspired motion prediction and supports the generation of large-scale synthetic trajectories for imitation learning.

Despite redundancy, human movements display robust invariants, suggesting that the nervous system resolves redundancy through specific organizational prin-

ciples often framed as optimal cost weights [23]. Although the existence of an optimality principle is widely accepted [20, 90, 158, 176], it remains unclear how cost weightings adapt across tasks [38] or within a single movement. For example, individuals may slow down near the target to improve accuracy, indicating a trade-off between task-related and body-related costs that many existing models fail to capture [202].

Optimal control theory provides a common framework for exploring these principles. Many models have been proposed [24, 177], yet most rely on a single cost function per task and often yield high prediction errors. Sylla et al. [177], using a computationally expensive bi-level method, reported average errors of about 7 degrees and values above 30 degrees for some joints, even in simple reaching tasks. Moreover, quantitative evaluations are often missing or rely on tailored metrics [24], raising concerns about predictive validity.

A major limitation is the difficulty of applying current IOC and IRL methods to estimate cost weights directly from human data. Most approaches identify a single weight set per task [105] because introducing time-varying weights leads to a large number of parameters that standard algorithms cannot handle efficiently. The standard bi-level formulation, in which weights are optimized through repeated solutions of a nested optimal control problem, is computationally expensive and can require several days of computation. This formulation is also prone to convergence to local minima, particularly in high-dimensional settings. To address these issues, alternative formulations based on the residuals of the Karush-Kuhn-Tucker conditions have been proposed. These methods aim to eliminate the need for repeated trajectory optimization. However, they remain highly sensitive to measurement noise and modeling errors, both of which are common in human motion data [21, 45] making them unsuitable for human motion analysis.

1.4 Statement of Purpose

This thesis aims to address various shortcomings in human-centered learning, through providing data-efficient algorithms to overcome data scarcity, sub-optimality, and variability. The thesis addresses the following overarching questions:

1. How can machine learning methods extract clinically and behaviorally mean-

ingful patterns from limited human data?

2. What modeling frameworks best represent human motion variability and allow generalization from a small number of demonstrations?

3. How can IRL be formulated to recover intent and cost functions under sparse observations and natural human suboptimality?

4. How can unified principles of data-efficient learning support the broader goal of human-centered and embodied intelligence?

1.5 Outline

The outline of this thesis is as the following:

1. Part I: Learning Human *Signals*

   - Chapter 2: LSTM and TCN models using only three physiological signals for COVID-19 health deterioration prediction. Experimental validation demonstrating robustness despite extremely low input dimensionality.

2. Part II: Learning Human *Behavior*

   - Chapter 3: Development and application of a regulated skew-t based mixture model for compact, expressive representation of human motion and teaching data.
   - Chapter 4: Representation of skew-Normal mixture regression for human behavior extraction from intricate demonstrations.
   - Chapter 5: Incorporating optimality scores in conventional GMM-based LfD paradigm to enhance the learned behavior from human demonstrations.

3. Part III: Learning Human *Intent*

   - Chapter 6: A novel iterative IRL formulation capable of inferring cost functions from limited and partially observed human demonstrations.

- Chapter 7: Human objective learning through an extended IRL method to devise subject independent cost function to define general human intent

Part I

Learning Human *Signals*

Chapter 2

Deterioration Prediction of COVID-19 Patients using Deep Learning Methods

In this chapter, we aim to learn human states and the consequent temporal changes. To solely focus on learning the human states, it is preferred to study the human when these states are not affected by human decisions. Hence, a study on human states when influenced by an external motivator such as a disease can present this setting. To that aim, this chapter presents our approach for prediction of human states for COVID-19 positive patients to determine whether or not the patient state will deteriorate.

Unrecognized deterioration of COVID-19 patients can lead to high morbidity and mortality. Most existing deterioration prediction models require a large number of clinical information, typically collected in hospital settings, such as medical images or comprehensive laboratory tests. This is infeasible for telehealth solutions and highlights a gap in deterioration prediction models based on minimal data, which can be recorded at a large scale in any clinic, nursing home, or even at the patient's home. In this study, we develop and compare two prognostic models that predict if a patient will experience deterioration in the forthcoming 3 to 24 hours. The models sequentially process routine triadic vital signs: (a) oxygen saturation, (b) heart rate, and (c) temperature. These models are also provided with basic patient information, including sex, age, vaccination status, vaccination date, and

status of obesity, hypertension, or diabetes. The difference between the two models is the way that the temporal dynamics of the vital signs are processed. Model #1 utilizes a temporally-dilated version of the Long-Short Term Memory model (LSTM) for temporal processes, and Model #2 utilizes a residual temporal convolutional network (TCN) for this purpose. We train and evaluate the models using data collected from 37006 COVID-19 patients at NYU Langone Health in New York, USA. The convolution-based model outperforms the LSTM based model, achieving a high Area Under the Receiver Operating Curve (AUROC) of $0.8844 - 0.9336$ for 3 to 24 hour deterioration prediction on a held-out test set. We also conduct occlusion experiments to evaluate the importance of each input feature, which reveals the significance of continuously monitoring the variation of the vital signs. Our results show the prospect for accurate deterioration forecast using a minimum feature set that can be relatively easily obtained using wearable devices and self-reported patient information.

In this study, we propose two deep neural networks to model time series of three vital signs only to predict the deterioration amongst patients with COVID-19. The ultimate goal of this work is to provide a light and scalable prediction model to support clinical decision-making for a wide range of patients in the long term, including at-home patients, outpatients, and inpatients. To minimize the size of the input feature space, we focus on three basic vital signs, namely, oxygen saturation (SpO2), heart rate (HR), and temperature. This design choice is motivated by the wide availability of wearable devices, such as smart watches, that can monitor these vital signs. We specifically exclude other vital signs, such as blood pressure, because they cannot be measured using readily available wearable systems.

To develop and evaluate the proposed models, we use real-world data collected at NYU Langone Health between January 2020 and September 2022 (detailed in Table 2.1). The model predicts deterioration at time horizons of 3 to 24 hours using the vital-signs time-series data collected in the 24 hours preceding the time of prediction (corresponding to the beginning of the prediction horizon), defined as in-hospital mortality, admission to the intensive care unit (ICU), or intubation. We refer to the sequential vital-sign data as SEQ data in the remainder of this chapter. The model is also provided with a small set of features reflecting CCC, including sex, age, vaccination status, vaccination date, and status of obesity, hypertension,

Table 2.1: **Overview of patient cohort.** We summarize in this table the patient characteristics, including demographics, and distribution of vital signs, for patients who deteriorated and patients who did not deteriorate.

Patient Data	Deterioration	No Deterioration
Characteristics		
Patients, n	6104	30902
Age (years), mean (SD)	66.0 (18.4)	63.1 (18.0)
Sex (females), n (%)	2539 (41.5)	17792 (57.5)
Diabetes w/o Complications, n (%)	1510 (24.7)	5582 (18.0)
Diabetes w/ Complications, n (%)	114 (1.8)	401 (1.2)
Hypertension, n (%)	2721 (44.5)	11748 (38.0)
Vaccination n (%)	2388 (39.1)	16776 (54.2)
Time since last vaccination (months), mean (SD)	-0.67 (6.2)	-0.94 (7.3)
Obesity, n (%)	1003 (16.4)	5370 (17.3)
Vital Signs		
SpO2 (%), mean (SD)	95.2 (4.5)	96.2 (2.7)
HR (Beats per minute), mean (SD)	93.98 (27.1)	82.9 (18.57)
Temperature (F), mean (SD)	98.37 (1.6)	98.2 (1.4)

and diabetes (referred to as non-SEQ data).

The first model includes a temporally-dilated LSTM (TD-LSTM) network [174] to process the SEQ data and an MLP that combines the last hidden state of the LSTM and the non-SEQ data. The LSTM network utilizes temporal dilation to enable access to more extended memory dynamics without exponentially increasing the size and complexity of the computational framework. The second model utilizes a residual temporal convolutional network (TCN) [18, 151] for SEQ data processing, and similar to the first model, an MLP is used to combine the output of the residual TCN blocks with the non-SEQ data. TCNs will preserve the causality of the time-series signal while using exponentially dilated convolutions to effectively extract features from the input signal's full spectrum. For each prediction horizon, a separate model is trained and optimized using the focal cross-entropy [110] loss through a three-phase training procedure.

The results show that, in the 3 to 24 hours prediction time horizons the model based on TCN achieves an AUROC of 0.8842-0.9237. While the results are not directly comparable to those in existing work due to differences in data pre-

processing, the model achieves a comparable performance. For example, the model in [168], which uses CXR images and other clinical variables, achieves 0.765 AUROC in predicting deterioration within 24 hours. In order to assess the significance of the various CCC features and the importance of the temporal history of vital signs, we also perform a sensitivity analysis through an occlusion experiment. Overall, our work highlights the feasibility of achieving high model performance for deterioration prediction amongst patients with COVID-19 using minimal feature sets, which are easy to obtain not only in the hospital setting but potentially also in nursing centers and at patient homes. The following remarks are added to further clarify the main focus of this chapter.

- ***Remark #1:*** This chapter report on an investigation that has shown for the first time that with deep temporal processing of only three vital signs and a set of patient's clinical characteristics, computational models (such as those reported in this chapter based on TD-LSTM and TCNs) have the power to predict the health deterioration of the patients with COVID-19 ahead of time. This chapter also investigates the contribution and importance of each of the vital signs and a set of comorbidity. The outcome can have a significant impact on clinical and therapeutic management in hospitals. It should be noted that this chapter does not claim to propose a new neural network architecture or a training optimization algorithm, as it is out of the scope of this chapter. Rather we focus and report on the possibility of health deterioration prediction using only three vital signs processed by the proposed deep neural network architectures.

- ***Remark #2:*** The dataset used in this study pertains to the patients with COVID-19 positive tests, but the COVID-19 infection may not necessarily be the primary reason for their hospital admission. Also, it is not possible to isolate the reason for deterioration as COVID-19 can play a compounding effect besides other comorbidities, the primary reason for admission, and underlying conditions altogether could result in deterioration. This chapter does not claim to generate a general model for any abnormal health condition; instead, specifically, we focus on patients with positive COVID-19 tests.

- ***Remark #3:*** The proposed models in this chapter are trained on COVID-

19-positive patients, and the chapter suggests that three major vital signs can be processed by the proposed models to predict the deterioration of COVID-19-positive patients. The chapter does not claim that the proposed models can be used for other respiratory illnesses, as this would require different data collection and possibly different architecture designs and training, which is out of the scope of this chapter.

2.1 Methodology

Dataset All methods were carried out in accordance with relevant guidelines and regulations. In this retrospective study, we use the "NYU Langone De-identified COVID-19 database [1]" collected from patients at the NYU Langone facilities between January 2020 and September 2022. The COVID-19 De-identified Clinical Database is a de-identified dataset of clinical activity at NYU Langone Health obtained from Epic starting January 1st, 2020. The data has been stripped of unique identifiers and dates have been shifted by an arbitrary number of days for each patient, which means that these data are not subject to HIPAA restrictions on research use, and thus IRB was exempted. More information can be found here: [1].

Data Availability The database that supports the outcomes of this paper was obtained from the NYU Langone Health's Medical Center Information Technology (MCIT). This database is available to the NYU Langone Health community to encourage exploration of the COVID-19 patient population for exploratory research, hypothesis testing, and identification of cohorts. Restrictions do apply to the public availability of the database. Data access is upon reasonable request and requires permission from NYU Langone Health (more relevant information can be found here [2]). The processed data, including figures and results supporting the conclusion made in this study, are given in the paper and can be made available upon reasonable requests from the authors.

2.1.1 Data Pre-processing

We define inclusion and exclusion criteria. First, in the case of multiple patient encounters, we use the patient's most recent encounter. Then, we include patients who either tested positive for COVID-19 at the facility or were already diagnosed with COVID-19 at the time of their admission. Next, we include in-patients with vital sign measurements. The vital signs of these patients, including SpO2, temperature, and HR, are periodically measured and recorded roughly every 4-5 hours. For each patient, age, sex, vaccination status and time, and the presence of comorbidities, including obesity, diabetes, and hypertension, are also recorded.

Similar to previous work (see [168] and reference therein), we define deterioration as the occurrence of the composite outcome of mortality, ICU admission, or intubation, i.e., any of the three events. For patient encounters with several adverse events, we only consider the occurrence of the earliest deterioration event. It should be noted that if there are multiple deteriorations of the same type (e.g., ICU admission) recorded for a patient more than a week apart, we only consider the latest as the reference time of deterioration for the patient. For patients who had deteriorated, we extracted vital sign data in the 48 hours preceding the time of deterioration.

We use this data to define "positive" windows for each prediction horizon, where $t = 0$ represents the end of the window and $t = -24$ represents the start of the window, such that, for example, in the prediction horizon of 24 hours, deterioration would have occurred at $t = 24$.

For patients who did not experience deterioration and were discharged, we use the 48 hours window preceding the last vital-sign recording and similarly use those to formulate the "negative" windows. We exclude all samples containing less than 48 hours of vital-sign monitoring, either preceding the deterioration time or discharge time.

To pre-process the time-series data, we first normalize the data using Z-score normalization based on each vital sign's mean and standard deviation. Since the vital signs are measured at irregular intervals, we resample each time series to obtain regularly sampled input for the LSTM and TCN networks. This resampling is done by first interpolating the raw non-uniformly sampled data through cubic spline interpolation and then sampling the interpolated signal every 15 minutes.

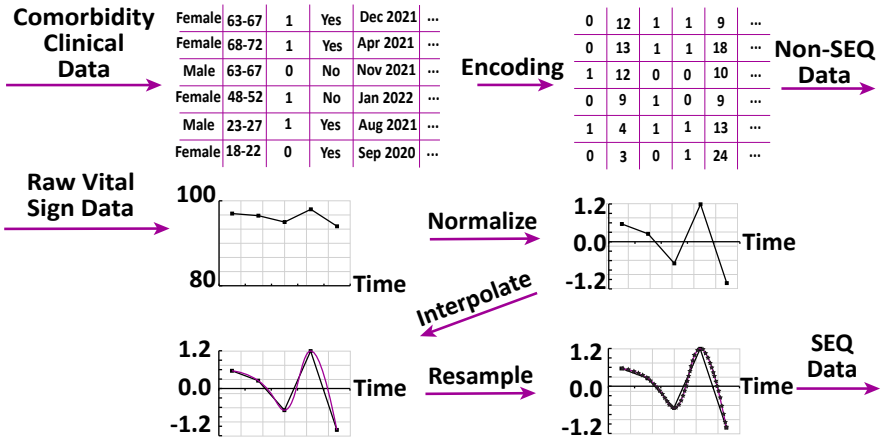


Figure 2.1: **Data pre-processing pipeline.** We encode the non-SEQ data and pre-process the SEQ data: (i) normalize via Z-score normalization, (ii) model the time-series using cubic spline interpolation, (iii) and resample at every 15 minutes.

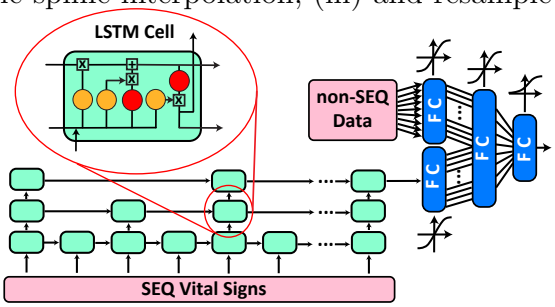


Figure 2.2: **Architecture of the LSTM-based deep neural network.** The SEQ vital sign network processes the SEQ data through an LSTM-based network, whereas the non-SEQ module encompasses the non-SEQ data.

In Figure 2.1, we show a schematic summarizing the pre-processing of the raw time-series data, which we refer to as the SEQ data.

As for the non-SEQ data, we encode patient sex, vaccination status, hypertension status, and obesity status as binary (0 or 1). For diabetes status, we use one-hot encoding to represent if a patient is non-diabetic ($[1, 0, 0]$), diabetic without complications ($[0, 1, 0]$), or diabetic with complications ($[0, 0, 1]$). We grouped the age into 18 different sub-groups and replaced each age with the corresponding age sub-group (value between 1 to 18). For vaccination time, we count the number of elapsed months between the time of the second COVID-19 vaccination shot and the day of the time of prediction ($t = 0$).

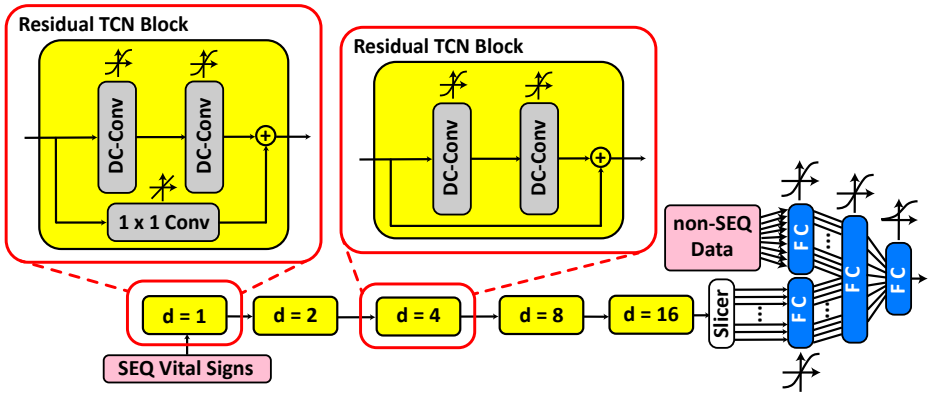


Figure 2.3: **Architecture of the TCN-based deep neural network.** The SEQ vital sign network processes the SEQ data through a series of residual TCN blocks followed by a FC layer, while the non-SEQ module processes the non-SEQ data.

2.1.2 DL-based Deterioration Prediction Model

We choose to develop and compare the performances of a LSTM-based model and a TCN-based model, because these two model architectures are commonly used for modeling time series data. Our proposed LSTM-based deep neural network architecture consists of TD-LSTM layers and fully-connected (FC) layers. The overall architecture of this network is shown in Figure 2.2, and we refer to it as the *Recurrent Sequential Vital Sign Network (RSVS-Net)*, consisting of two modules. The SEQ data is processed by a module comprised of an LSTM network and a single FC layer, while the non-SEQ data is processed by a second module consisting of an independent FC layer. The final prediction is based on both modalities.

The TCN-based network proposed in this study includes residual TCN blocks, and FC layers. We refer to this model as *Convolutional Sequential Vital Sign Network (CSVS-Net)*. The module that is processing the SEQ data in this network consists of a series of five residual TCN blocks with exponentially enlarging kernel dilations, and a single FC layer. The non-SEQ data processing module of the *CSVS-Net* is the same as the one in *RSVS-Net*. The architecture of *CSVS-Net* is shown in Figure 2.3. For brevity, we refer to the residual TCN blocks as R-blocks throughout the rest of this paper.

2.1.2.1 RSVS-Net architecture details

LSTM networks are well-known for their ability to learn from SEQ data and have been widely used in studies where the time-series data is integral to the learning of the system and predicting future events [198, 203]. The temporal module consists of a temporally-dilated LSTM, which takes three vital signs as input at each time step, and has three layers, each containing 32 hidden units. The final hidden state of the LSTM network, consisting of a dimensionality of 32, is processed by an FC layer with an output dimensionality of 16. The non-SEQ data, with a dimensionality of 9, is processed by a single FC layer, which computes an output of dimensionality of 16.

Finally, the latent representations of the two modalities are concatenated as a vector of dimensionality of 32 and then processed by an FC fusion layer with an output dimensionality of 8. This is then followed by a single FC layer with sigmoid activation and an output dimensionality of one, which represents the prediction that a sample precedes deterioration or is not within the specified time horizon. All of the FC layers use hyperbolic tangent activation except for the last layer, which uses sigmoid activation. The *RSVS-Net* model has **22,209** trainable parameters.

2.1.2.2 CSVS-Net architecture details

TCN models have gained increasing attention recently for time-series data analysis [18, 56, 69, 151, 182]. In TCNs, convolution is causal, meaning that the convolution output at time t will only rely on the signal at time t or earlier [18]. In the *CSVS-Net*, the SEQ data module comprises five serialized R-blocks. Each block has two dilated causal 1D convolutional (DC-Conv) layers. In addition, each R-block has a residual connection, so the network would become more effective in learning the extracted features and the modifications applied to the input signal by the DC-Conv layers. At the end of each R-block structure, the output of the last DC-Conv layer and the R-block's input will be summed. Therefore, these signals should have matching dimensions. To ensure this criteria is satisfied throughout the model, in the first residual R-block, a 1×1 convolutional layer with linear activation function is implemented in the residual connection to match the input data dimension to the output of the DC-Conv layers.

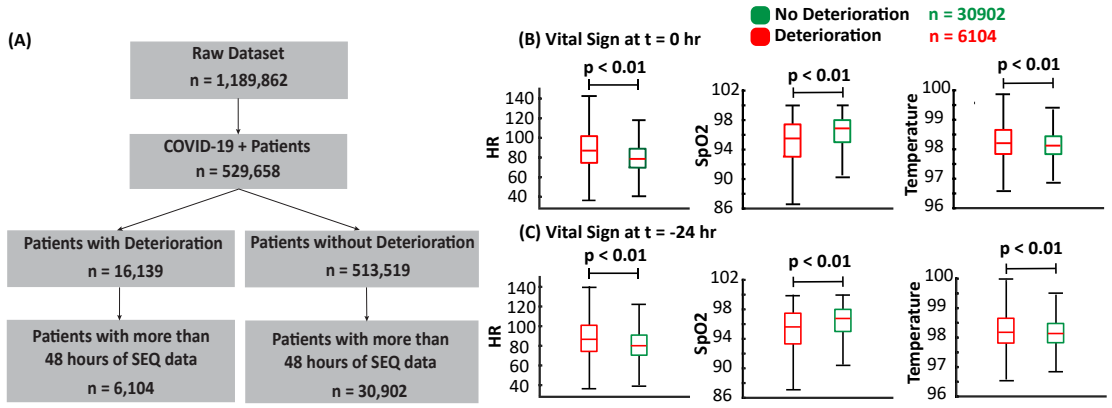


Figure 2.4: **Application of data inclusion and exclusion criteria and distribution of vital signs. (A)** In this flowchart, we illustrate the application of the inclusion and exclusion criteria, where **n** represents the number of patients after each step. **(B)** The boxplot of the vital signs recorded from the patients at the end of the 24-hour input window (t=0) corresponds to the prediction time. **(C)** The boxplot of the vital signs recorded from the patients at the beginning of the 24-hour input window (t=-24). We observe differences between the two groups (evaluated using the T-test), which motivates the design of the proposed temporal model.

In *CSVS-Net*, all of the DC-Conv layers in the R-blocks have 28 filters, and use hyperbolic tangent activation function. Hence, the output dimension of every R-block is $(N \times 28)$, with N being the number of samples for each feature in the vital sign time-series signals. For 24-hour observation period for each feature, since we have 4 samples in each hour, the length of the signal will be $N = 1 + (4 \times 24) = 97$. For analyzing the full length of this signal, the architecture of the *CSVS-Net* should be able to have a receptive field (R_{field}) big enough to analyze the entire signal length. The R_{field} parameter can be computed using Equation (2.1).

$$R_{field} = 1 + 2 \times (K - 1) \times \sum_{i=1}^{B} d_i \tag{2.1}$$

In Equation (2.1), K represents the filter size, B is the number of R-blocks in the architecture, and d_i is the kernel dilation used at each R-block. By choosing $K = 3$, $B = 5$, and $d_i = 2^{(i-1)}$, we will have enabled the model to gain receptive field of 125. It should be noted that any other chosen number of R-blocks will either result in an insufficient or unnecessarily large receptive fields for the model.

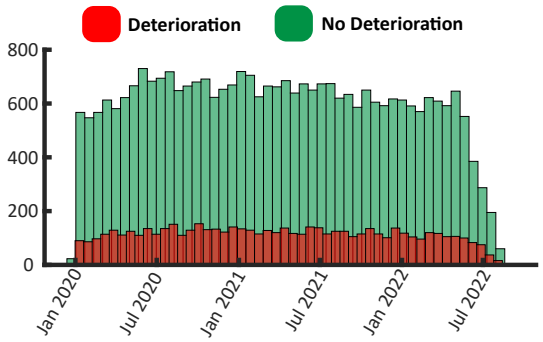


Figure 2.5: **Distribution of samples over time.** We show the number of patients who deteriorated vs. those who did not deteriorate in our final filtered dataset (n=37,006).

Finally, a "slicer" layer is implemented at the end of the R-block chain, which only passes through the last array of the final R-block's output. Hence, the dimension of the slicer layer output is (1×28). Thereafter, this 1×28 vector is processed by an FC layer with dimensionality of 16. The rest of the layers in the *CSVS-Net* are the same as the ones in *RSVS-Net*. The *CSVS-Net* model has **22,709** trainable parameters.

2.1.2.3 Three-Phase Training Strategy

In order to optimize the performance of the proposed network, the training strategy consists of three phases, as described below.

- **Phase 1:** ***Training of SEQ module***
  In the first phase, we train the SEQ module only and freeze all other layers. The output of the FC layer in the SEQ module is connected to another FC layer that computes the prediction. After this training phase, the weights are used to pre-initialize the SEQ module in the next phase, and we remove the second FC layer used to compute predictions during pre-training in the first phase.

- **Phase 2:** ***Training of fusion layer***
  In the second phase, we compute the representations of the SEQ data after initializing the associated module with the weights obtained in the first phase,

and then freeze the SEQ module. We then train the FC layer, FC fusion layer, and the FC output prediction layer using the non-SEQ data.

- **Phase 3: *End-to-end fine-tuning of RSVS-Net and CSVS-Net***
  In the last phase, we initialize the parameters of the entire network using the weights obtained in the first two phases. The network is trained end-to-end, to improve the overall network performance.

2.1.2.4 Model Training and Evaluation

To train and evaluate the models, we divide the entire dataset into training, validation, and testing sets, with percentages of 66.6%, 16.6%, and 16.6%, respectively. Each dataset has the same distribution of positive and negative samples. The final performance results are obtained by evaluating the trained models on the test dataset.

We train the models using the training set for 200 epochs using the three-phase training strategy. For the first and second phases, we choose a learning rate of 0.0001 based on initial experimentation and a learning rate of 0.00001 for the final fine-tuning stage. For all phases, we use the ADAM optimizer [88] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. After training for 200 epochs, the best model is selected based on the epoch with the least validation loss across all epochs. We evaluate the models on the test set. We use the binary focal cross-entropy loss [110] in order to manage class imbalance in the dataset. We evaluate the models' performance using three widely used metrics for binary classification: accuracy (with a 0.5 threshold to convert the predicted probability into binary), AUROC, and Area Under the Precision-Recall Curve (AUPRC).

2.2 Results and Discussion

2.2.1 Patient Cohort

In Figure 2.4(A), we show the application of the inclusion and exclusion criteria. This resulted in 37,006 patient samples, including 6,104 positive samples and 30,902 negative samples. Table 2.1 summarizes the characteristics of the patients. In Figure 2.4(B), we show the differences in vital signs of the two cohorts at $t = 0$,

while in Figure 2.4(C), we show the differences in vital signs between the two cohorts at $t = -24$. Using t-test statistical analysis, it can be seen that the difference between the two cohorts is statistically significant even as early as 24 hours before deterioration. This motivates the use of the proposed DL to decode the hidden pattern differentiating the two cohorts. In Figure 2.5, we show the distribution of the admitted patients over time.

2.2.2 Model Performance

The final model performances are summarized in Table 2.2 and also shown in Figure 2.6 after the three-phase training strategy. The performance of the *RSVS-Net* at the 24 hours time horizon reaches 86.67% accuracy, 0.8278 AUROC, and 0.5805 AUPRC. Moreover, *CSVS-Net* achieves 89.75% accuracy, 0.8844 AUROC, and 0.7227 AUPRC, at the 24-hour time horizon, significantly outperforming the LSTM-based model. As expected the prediction accuracy improves as the prediction horizon reduces. We suspect that the convolution-based model (*CSVS-Net*) outperformed the LSTM-based model (*RSVS-Net*) because the former was able to extract and exploit features from all the vital signs in the observation window relatively equally, while the LSTM model may reduce the contribution of the far away samples. As shown in Figure 2.6, AUROC and AUPRC consistently improve at all prediction horizons after each training phase. By comparing the performance of the three phases of training, it can be observed that the adopted three-phase training strategy boosts the model's performance by forcing the network to extract information initially from the SEQ data and, in the end, from a combination of SEQ and non-SEQ data. The improvement can be seen in accuracy, AUROC, and AUPRC. Although the datasets used in similar studies

2.2.3 Ablation Studies

In order to understand the impact of our design choices within the model architecture, we compare our models to two other networks. The first model, referred to as *Memory-Less Vital Sign Network (MLVS-Net)*, processes the non-SEQ data and only the last set of vital signs collected from the patient, ignoring any sequential information. Hence, instead of using a TD-LSTM or R-blocks, we

Table 2.2: **Model performance.** Performance summary of the proposed networks for three phases of training at all horizons.

Prediction horizon (hours)	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**
RSVS-Net								
Accuracy	**0.882**	0.80	0.873	0.874	0.868	0.868	0.870	0.866
AUROC	**0.875**	0.859	0.848	0.832	0.832	0.822	0.820	0.827
AUPRC	**0.656**	0.649	0.609	0.593	0.590	0.578	0.578	0.580
CSVS-Net								
Accuracy	**0.913**	0.909	0.910	0.906	0.902	0.903	0.900	0.897
AUROC	**0.933**	0.921	0.910	0.897	0.896	0.896	0.884	0.884
AUPRC	**0.805**	0.789	0.775	0.751	0.743	0.746	0.728	0.722

process the vital-sign data (3 features) with an MLP consisting of two FC layers with an output dimensionality of 16 each. The computed representation of the MLP is then concatenated with the representation of the non-SEQ data. We train the model in a similar fashion using the three-phase training strategy, and we freeze the weights of the MLP network in phase two. The second model, referred to as the *non-Sequential Health Status Network (nSHS-Net)*, only considers the non-SEQ data. Hence, the output of the FC layer is processed by a second FC layer to generate the prediction.

We compare the three models in Figure 2.7. First, we observe that *nSHS-Net* performs the worst, implying that incorporating vital signs is crucial for the model prediction. When comparing *MLVS-Net* to *RSVS-Net* and *CSVS-Net*, we observe a better performance with our proposed models across all prediction horizons and evaluation metrics. This implies that incorporating sequential information can significantly improve the model's capability in predicting deterioration relative to using a single measurement of vital signs. As already shown in Figure 2.6, among the two proposed models, the *CSVS-Net* significantly outperforms the *RSVS-Net*. The numerical results of Figure 2.7 are summarized in Appendix A.

2.2.4 Occlusion Analysis

In order to assess the influence of each input variable, we conduct an occlusion analysis on the final optimized model. In particular, we occlude one feature at a

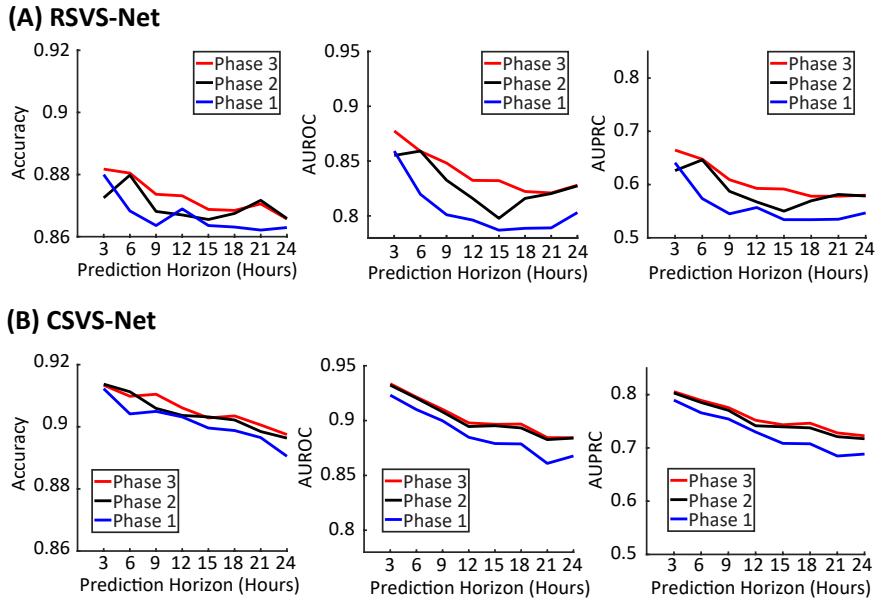


Figure 2.6: Performance results after each phase in the training strategy across all prediction horizons.

time (by setting the corresponding values to zero) and evaluate the performance on the test set. The greater the reduction in the performance metrics upon feature occlusion, the more important the feature is for the prediction. The results of this analysis are shown in Figure 2.8. As shown in Figure 2.8(A), amongst the non-SEQ features, we observe that age plays the most significant role in the model's performance. The other CCC features have less impact once occluded and do not have consistent trends across the different performance metrics and prediction horizons. It should be noted that due to the correlation between the various features, some features may be relevant to the prediction task yet are not considered to be important by this occlusion analysis. For example, if one CCC feature affects the variations in the vital signs over time, then the model would capture the corresponding effect. In this case, the occlusion analysis may show that this CCC feature is unimportant. However, when a feature shows low sensitivity through the occlusion study, the need for that feature to be given to the model as an "independent" input is insignificant. Among the SEQ features, for both models, we observe heart rate as the most important feature, followed by SpO2 and temperature. For *CSVS-Net*, the temperature occlusion shows minuscule effect

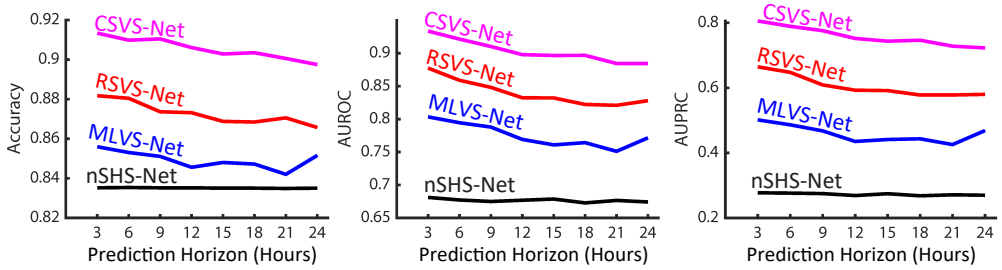


Figure 2.7: **Ablation study results.** Performance results for *RSVS-Net* and *CSVS-Net* (Input: non-SEQ data and SEQ vital sign data), *MLVS-Net* (Input: non-SEQ data and a single set of vital signs), and *nSHS-Net* (Input: non-SEQ only).

in the model performance. This is an interesting observation, possibly because the *CSVS-Net* model was able to capture the information encoded in the change in temperature using the observations on the change in the heart rate and/or SpO2. In other words, the inter-dependencies between the vital signs could be the cause of this observation. It should be highlighted that in Figure 2.8, we have shown that the vital signs are more important than the clinical and comorbidity features, and SEQ vital signs bring additional information than the current vital signs only. The observation here matches the aforementioned analysis. Numerical results of Figure 2.8 are given in Appendix I.

2.2.5 Experimenting with Shorter Observation Windows

In the previous section, we show the importance and significance of the time history of the vital signs for predicting the deterioration of COVID-19 patients, considering an observation window of 24 hours. It is also important to evaluate the behavior of the proposed model for different windows of observation. Thus, here, we analyze system performance for three shorter windows of observation (with lengths of 6 hours, 12 hours, and 18 hours). It should be noted that for this analysis, the proposed *RSVS-Net* and *CSVS-Net* models, which were trained by observing 24 hours of patient data, were "evaluated" for 6, 12, and 18 hours of the observation period. Both *RSVS-Net* and *CSVS-Net* models were not retrained for these shorter windows of observation in order to evaluate the performance of a single model for various input windows. The results of this analysis are shown in Figure 2.9. As can

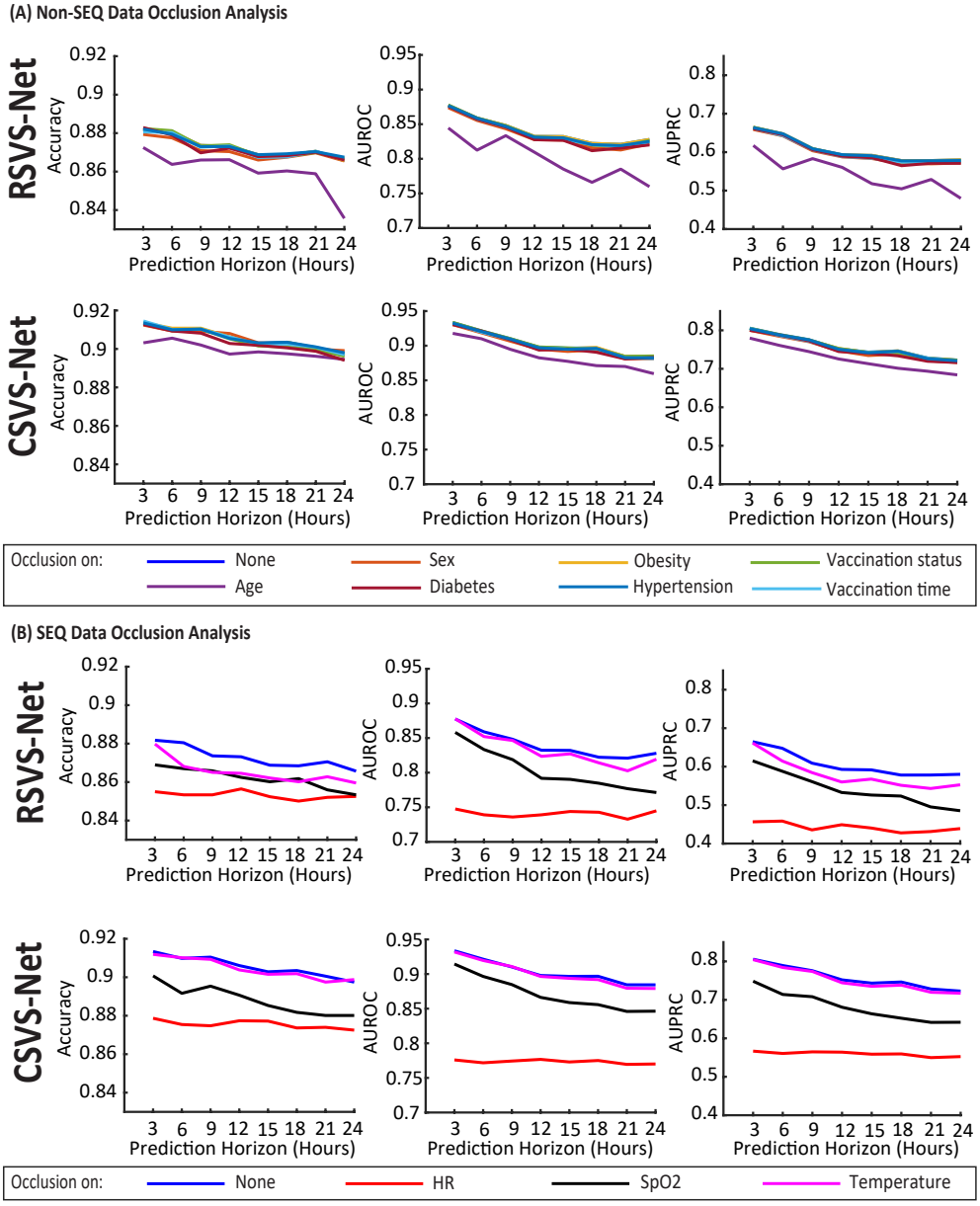


Figure 2.8: **Occlusion analysis: (A)**: Occlusion of the clinical and comorbidity characteristics. Observation: Occlusion of Age decreases performance more significantly than others. **(B)**: Occlusion of SEQ vital sign data. Observation: the HR contributes more significantly to the model performance than the SpO2 and temperature.

be seen, by decreasing the duration of the observation window, the performance

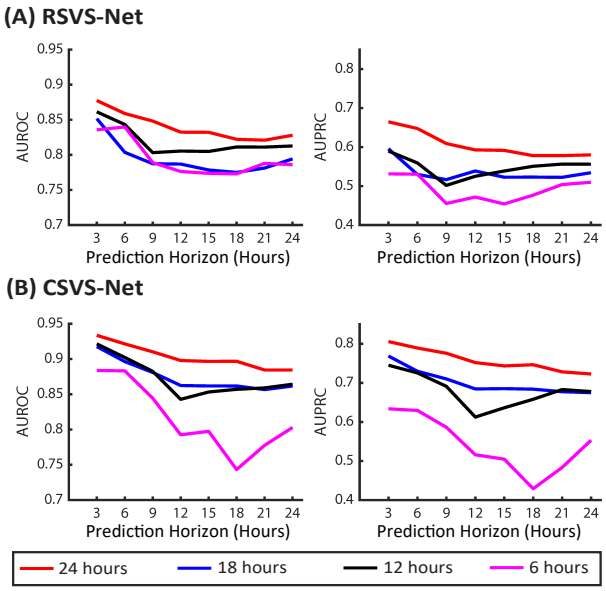


Figure 2.9: **Various observation windows (6 to 24 hours).** The legend of the figures indicates the observation duration. Performance results of the *RSVS-Net* and *CSVS-Net* given different observation duration.

drops. The drop in performance caused by the shorter observation is likely caused by the lack of information used in the model to predict the deterioration event ahead. This analysis further highlights the importance of continual and prolonged monitoring, which may motivate using smart wearable medical systems.

2.2.6 Comparison with COVID-19 Negative Patient Data

In order to further evaluate the performance of *CSVS-Net*, we trained this model on the dataset containing the same set of SEQ and non-SEQ biomarkers derived from the patient cohort without the positive COVID-19 diagnosis (as seen in Figure 2.4(A)). This dataset includes 2508 deteriorated and 48395 non-deteriorated patient data after pre-processing (mentioned in Section 2.1.1). The ratio of non-deteriorated to deteriorated patients in this dataset is noticeably higher than the COVID-19 positive dataset, which corroborates the effect of COVID-19 on patient health deterioration. The performance of the *CSVS-Net* model on this new dataset and the occlusion analysis are presented in Figures 2.10 and 2.11, respectively. It can be seen that although the accuracy of this model is slightly higher than the

CSVS-Net model trained on the COVID-19 positive patient dataset, AUROC follows a lower trend in comparison, and AUPRC is significantly lower. The low AUPRC conveys that the proposed model is not able to secure the same level of sensitivity to differentiate between the to-be-deteriorated and to-be-not-deteriorated subjects, as it is able to do so for the COVID-19-positive patient data. This behavior can stem from the heterogeneity of the COVID-19-negative patient data, which can point out that health deterioration prediction should be investigated using disease-specific models; also, it may highlight that a generic model may not be able to secure the needed sensitivity. In other words, the results show that even though the three proposed biomarkers (heart rate, SpO2, and temperature) can provide a sensitive prediction of deterioration for COVID-19-positive patients, they are not able to secure the same level of sensitivity for a generic patient population.

The results shown in Figure 2.11(A) suggest that among non-sequential biomarkers, occlusion of age has the highest effect on the performance of the model trained on the COVID-19-negative patient cohort. This is the same case for COVID-19-positive patients for the proposed *CSVS-Net* model.

Moreover, in contrast to the *CSVS-Net* model trained on the COVID-19 positive dataset, the most significant SEQ vital sign that affects the model's performance when trained on COVID-19 negative dataset is SpO2, followed by heart rate and temperature, respectively, as seen on Figure 2.11(B). These results, compared with the results presented in Figure 2.8(B), show that the occlusion of heart rate will result in a more significant drop in the performance for COVID-19-positive cases, which may suggest that the temporal dynamics of the heart rate and the corresponding temporal variations are distinctly more critical for detecting health deterioration for COVID-positive patients than for COVID-19-negative patients. Given this observation, the heart rate may be considered as a potentially unique biomarker for COVID-19 health deterioration prediction using the proposed *CSVS-Net* model.

2.3 Conclusion

This study is motivated by the high availability of personal medical devices, such as wearable systems (e.g., smart watches), that can record time-series medical

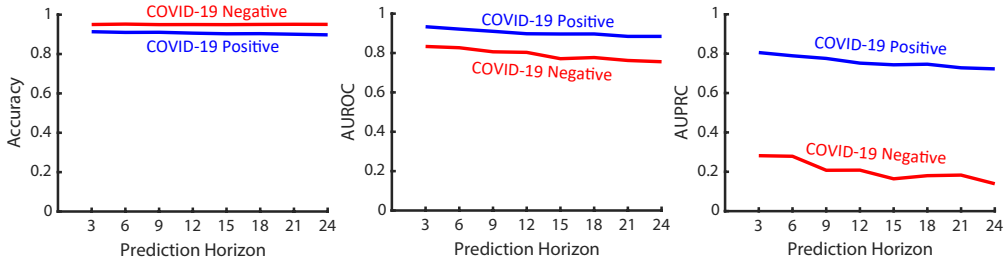


Figure 2.10: **Performance comparison of *CSVS-Net* trained on COVID-19 Positive and COVID-19 Negative cohorts:** Observation: The accuracy and AUROC metrics of the model trained on the COVID-19 negative cohort are comparable to those of the model trained on the COVID-19 positive cohort, while the AUPRC is significantly lower in comparison.

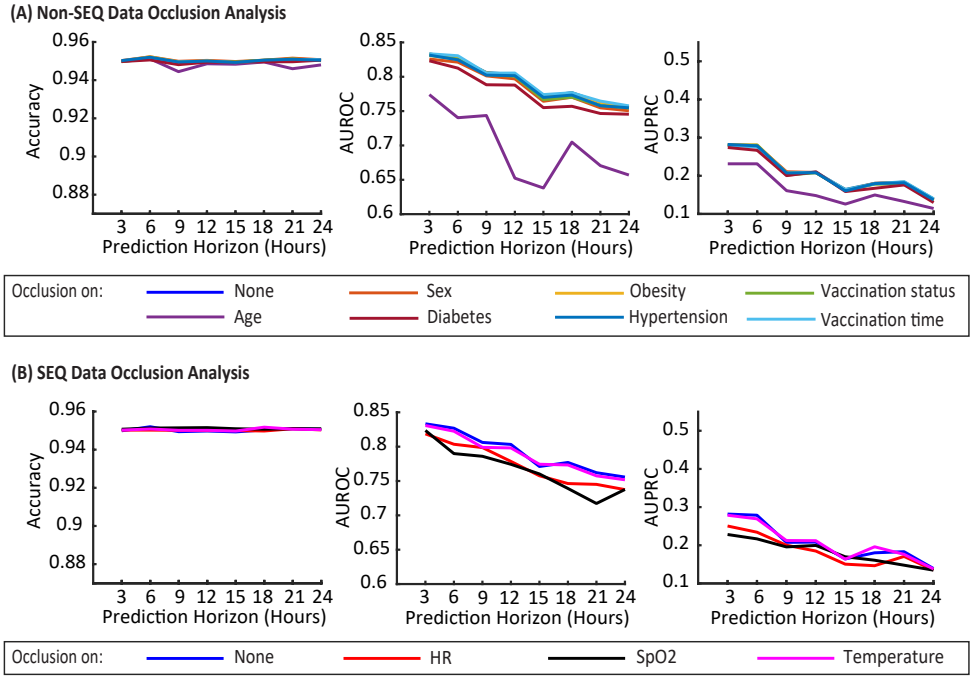


Figure 2.11: **Occlusion analysis for COVID-19 Negative Patient Cohort:** **(A)**: Occlusion of the clinical and comorbidity characteristics. Observation: Occlusion of Age decreases performance more significantly than others. **(B)**: Occlusion of SEQ vital sign data. Observation: SpO2 vital sign has more impact on the model performance than HR and temperature.

data and the prospects of using such devices in the context of telehealth to predict deterioration. In summary, we propose, develop, and evaluate a deterioration

prediction model using a large dataset (n=37,006) collected at NYU Langone Health during the period of January 2020 to September 2022. The *CSVS-Net* achieves a high AUROC of 0.8844-0.9336 in 3 to 24 hours prediction horizons and outperformed the *RSVS-Net*. Our study has several strengths. First, the models use a minimal input feature set consisting of a time series of three routine vital signs, i.e., SpO2, heart rate, and temperature. The models are also provided with basic patient information, including sex, age, vaccination status, vaccination date, and presence of obesity, hypertension, and diabetes, which can be easily collected. Compared to previous work [168] that achieved 0.765 AUORC, our model achieves a better performance. Furthermore, our models were trained and evaluated using much smaller and more accessible data modalities (excluding any sophisticated imaging, such as CT scans). We do note that the model in [168] was trained with significantly fewer data points (number of patients) than ours, and hence this can be another reason on the difference in performance. In addition, we also performed a sensitivity analysis through an occlusion experiment and an ablation study on various inputs including the vital signs, to assess the significance of the various clinical and comorbidity features and the importance of using time-series vital-signs data. The results showed the importance of modeling the temporal variations of vital signs and the possibility of achieving high prognosis accuracy without the need for sophisticated medical imaging. Finally, the proposed framework is scalable as it can be extended for other prediction horizon ranges and observation periods.

The limitations of the proposed work include the following. First, our observation windows for the vital signs are tested only for 6 to 24 hours. We showed that reducing the window lengths leads to degraded performance. In future work, we are interested in evaluating if further increasing the observation window would significantly improve the prediction accuracy. Second, we acknowledge that the final results can be further improved via hyperparameter tuning, including the number of network layers, the number of feature channels, and convolution kernel sizes, and this is an area of future work. To conclude, this chapter highlights the feasibility of an accessible and scalable model to help assist the medical workforce in decision-making. The proposed model's versatility is important, as the data types needed for predicting the deterioration can be easily acquired from patients using wearable sensors and a few clinical data features that can be self-reported.

Part II

Learning Human *Behavior*

Chapter 3

FiMReSt: Finite Mixture of Multivariate Regulated Skew-t Kernels

Recently skew-t mixture models have been introduced as a flexible probabilistic modeling technique taking into account both skewness in data clusters and the statistical degree of freedom (DoF) to improve modeling generalizability, and robustness to heavy tails and skewness. This chapter provides insights into the instability of the DoF, which can result in the divergence of the kernels from the mixture of t-distribution, losing generalizability and power for modeling the outliers. Thus, in this chapter, we propose a regularized iterative optimization process to train the mixture model, enhancing the generalizability and resiliency of the technique. The resulting mixture model is named Finite Mixture of Multivariate Regulated Skew-t (FiMReSt) Kernels, which stabilizes the DoF profile during optimization process of learning. To validate the performance, we have conducted a comprehensive experiment on several real-world datasets and a synthetic dataset. The results highlight (a) superior performance of the FiMReSt, (b) generalizability in the presence of outliers, and (c) convergence of DoF.

In order to expand the previous simplified versions of the mixture models based on skew-t distribution to harness its inherent flexibility, in [108], Lin proposed the use of the Monte Carlo EM (MCEM) algorithm [191] to numerically estimate the expected values needed in the corresponding EM approach. However, the resulting

method suffered from slow convergence and inaccurate expectations due to the numerical nature of MCEM. Finally, in an extended effort, Lee et al. [92] proposed a closed-form EM algorithm for training skew-t mixture model (STMM), using a new formulation for estimating one of the intractable conditional expected values of the latent variables using one-step-late (OSL) estimation approach [64], while using the formulations proposed by Ho et al. [71], to express the other two intractable expected values (using the first and second moments of truncated multivariate t-distribution).

However, in [92], the skewness matrix was assumed to be diagonal. This restriction was solved in [97] by proposing a finite mixture of canonical fundamental skew-t (FM-CFUST) distribution using the OSL-based EM algorithm to incorporate full skewness matrices for the clusters. This model was designed aiming for the highest optimality compared to its counterparts since it exploits the complete skewness matrix as opposed to the STMM model.

Although probabilistic clustering has shown to be effective in modeling multivariate datasets, the numerical methods utilized for fitting such probabilistic models can be suboptimal. In this chapter, we will show that due to the non-convex nature of the existing iterative optimization approaches (such as the one in [97]) for skew-t kernels, some MST kernel's DoF will be monotonically increasing which converges the t-distribusion to a normal distribution. This problem makes the optimization process prone to local minima caused by suboptimal normal or skew-normal kernels. Thus, in this chapter, a new mathematical EM-type algorithm called expectation-regularization-maximization (ERM) is developed for an iterative optimization that can generate a mixture model of skew-t kernels with a "regulated DoF". Subsequently, a finite mixture of regularized skew-t kernels (FiMReSt) distribution is generated, which can model a mixture of skew-t distribution while avoiding ever-increasing values of DoF. The algorithm was tested on asymmetrically scattered datasets to validate the performance. The results show that the proposed approach has a superior behavior taking advantage of a bounded DoF with minimum to no sensitivity to initialization as opposed to the counterparts in the literature.

3.1 Preliminaries

In this chapter, random variables are shown by upper case letters such as X, and the corresponding values are shown in lower case. Also, where z indicates a scalar value, boldened letters $\boldsymbol{z}$ are used to indicate vectors or matrices. In the literature, a weighted summation of multiple multivariate skew-t distributions is shown as seen in Eq. (3.1) [98].

$$f(\boldsymbol{y}|\boldsymbol{\Theta}) = \sum_{i=1}^{g} \omega_i MST_{p,\nu_i}(\boldsymbol{y}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \boldsymbol{\Delta}_i) \tag{3.1}$$

subject to

$$\sum_{i=1}^{g} \omega_i = 1 \tag{3.2}$$

In Eq. (3.1), g is the number of p-dimensional multivariate skew-t distribution clusters in the mixture model, denoted as $MST_{p,\nu}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\Delta})$, and i subscript addresses the parameters associated with the i^{th} cluster in the mixture model. In addition, $\boldsymbol{y}$ is the input vector, and $\boldsymbol{\Theta}$ is the set that contains the defining parameters for all multivariate skew-t kernels in the mixture model. The defining parameters for each cluster include a $p \times 1$ mean vector $\boldsymbol{\mu}$, a $p \times p$ symmetric positive-definite covariance matrix $\boldsymbol{\Sigma}$, a $p \times p$ skewness matrix $\boldsymbol{\Delta}$, and a scalar value ν as DoF.

The multivariate skew-t distribution, denoted as $MST_{p,\nu}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\Delta})$ in Eq. (3.1), is a member of the skew elliptical distribution family [159]. The PDF of this distribution is calculated using Eq. (3.3) as can be seen in

$$MST_{p,\nu}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\Delta}) = 2^p t_{p,\nu}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Omega}) T_{p,\nu'}(\boldsymbol{y}^\star|\boldsymbol{0}, \boldsymbol{\Lambda}). \tag{3.3}$$

In Eq.(3.3), $t_{p,\nu}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Omega})$ is the PDF of the p-dimensional t-distribution (t-PDF) for random vector $\boldsymbol{y}$, where $\boldsymbol{\mu}$ is a $p \times 1$ mean vector, $\boldsymbol{\Omega}$ is a positive-definite symmetrical $p \times p$ covariance matrix, and ν is the scalar DoF parameter which can range from 2 to infinity for multivariate cases. In addition, $T_{p,\nu'}(\boldsymbol{y}^\star|\boldsymbol{0}, \boldsymbol{\Lambda})$ is the p-dimensional t-distribution cumulative density function (t-CDF) for a random vector $\boldsymbol{y}^\star$, given $\boldsymbol{0}$ as the mean, $\boldsymbol{\Lambda}$ as positive-definite symmetrical covariance matrix, and ν' as the DoF.

This distribution represents a general class from which different PDFs can be

obtained. When there is zero skewness (i.e., $\boldsymbol{\Delta} = \mathbf{0}$), the multivariate skew-t PDF represents a multivariate t-distribution, and when DoF becomes a large scalar (i.e., $\nu \to \infty$), the multivariate skew-t PDF converge to the multivariate skew-normal distribution. In the case where there is no skewness and the DoF is large, the multivariate skew-t distribution will become a Gaussian distribution $N_p(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$. (Seen in Fig. 1.2(b)).

The parameters ($\boldsymbol{\Omega}$, $\boldsymbol{y^\star}$, $\boldsymbol{c(y)}$, $\boldsymbol{\Lambda}$, $\boldsymbol{d(y)}$, ν') in Eq. (3.3) defines the multivariate skew-t PDF and are explained as follows.

$$
\begin{aligned}
\boldsymbol{\Omega} &= \boldsymbol{\Sigma} + \boldsymbol{\Delta^T \Delta} \\
\boldsymbol{y^\star} &= \boldsymbol{c(y)} \sqrt{\frac{\nu + p}{\nu + d(\boldsymbol{y})}} \\
\boldsymbol{c(y)} &= \boldsymbol{\Delta^T \Omega^{-1} (y - \mu)} \\
\boldsymbol{\Lambda} &= \boldsymbol{I_p - \Delta^T \Omega^{-1} \Delta} \\
\boldsymbol{d(y)} &= (\boldsymbol{y - \mu})^T \boldsymbol{\Omega}^{-1} (\boldsymbol{y - \mu}) \\
\nu' &= \nu + p
\end{aligned}
$$

In addition, the p-dimensional t-PDF (i.e., $t_{p,\nu}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Omega})$) used in Eq (3), can be calculated as follows. In the mentioned equations above, $d(\boldsymbol{y})$ is the squared Mahalanobis distance between $\boldsymbol{y}$ and $\boldsymbol{\mu}$ with respect to $\boldsymbol{\Omega}$ [19].

$$
t_{p,\nu}(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Omega}) = \frac{\Gamma(\frac{\nu+p}{2})}{\Gamma(\frac{\nu}{2})(\nu\pi)^{p/2}} |\boldsymbol{\Omega}|^{-1/2} \left\{ 1 + \frac{(\boldsymbol{y} - \boldsymbol{\mu})^T \boldsymbol{\Omega}^{-1} (\boldsymbol{y} - \boldsymbol{\mu})}{\nu} \right\}^{-(\nu+p)/2} \tag{3.4}
$$

In Eq. (3.4), $\boldsymbol{y}$ is a random p-dimensional vector, $\boldsymbol{\mu}$ is the $p \times 1$ mean vector, $\boldsymbol{\Omega}$ is the $p \times p$ positive-definite symmetrical covariance matrix, ν is the DoF, and $\Gamma(.)$ denotes the gamma function.

It should be highlighted that there is no analytical closed-form formulation available for t-CDF in Eq. (3.3) [119]. Instead, it is estimated using direct integration of t-PDF or numerical approximations. In this chapter, for estimating the t-CDF, the numerical approximation proposed by Genz et al., [63] is used.

For fitting a mixture model of multivariate skew-t distributions on a dataset, an MLE method would be needed to maximize the fitness of the model (i.e., *log-likelihood*) over the given dataset by iteratively optimizing the defining parameters

of the kernels. For this process on n observations (datapoints), the log-likelihood can be calculated using Eq. (3.5).

$$l(\boldsymbol{\Theta}, \boldsymbol{y}) = \sum_{j=1}^{n} \log \left(\sum_{i=1}^{g} \omega_i f_{p,\nu_i}(\boldsymbol{y}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \boldsymbol{\Delta}_i) \right) \tag{3.5}$$

In order to maximize this value in the MLE process, an EM algorithm is proposed in the literature [95] to iteratively update the defining parameters until the convergence of the log-likelihood value. The EM algorithm contains two steps:

1. *E-Step*: In this step, the expected value of the mixture model's log-likelihood will be computed given the current defining parameters of the model using Eq. (3.6).

$$Q(\boldsymbol{\Theta}|\hat{\boldsymbol{\Theta}}^{(k)}) = E(l(\boldsymbol{\Theta}, \boldsymbol{y})|\boldsymbol{y}, \hat{\boldsymbol{\Theta}}^{(k)}) \tag{3.6}$$

   In Eq. (3.6), the term $\hat{\boldsymbol{\Theta}}^{(k)}$ denotes the set of mixture model's defining parameter at EM-algorithm's k^{th} iteration, and $E(.)$ is the expectation operator.

2. *M-Step*: In this step, the Q-function calculated in the E-Step will be maximized as in Eq. (3.7).

$$\boldsymbol{\Theta}^{(k+1)} = \arg\max_{\boldsymbol{\Theta}} Q(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) \tag{3.7}$$

In Section 3.2, we will analyze an inherent issue with the EM algorithm for training multivariate skew-t mixture models.

3.2 ERM Algorithm

Although the EM algorithm converges for the mixture models with lower complexity, such as SNMM, we observe that for more complex models such as FM-CFUST, due to non-convex characteristics of the optimization, the convergence of the DoF parameter is not guaranteed. An example is given in Fig. 3.1 for a 2D dataset with 2 clusters. It can be observed that as the number of iterations increases, the log-likelihood will converge to a certain value, while the DoF of one of the FM-CFUST kernels will diverge. This phenomenon is called the DoF instability

in this chapter. Based on the mathematical representation of the multivariate skew-t distribution in Section 3.1, this monotonic increase of DoF will force the associated cluster to become a multivariate "skew-normal" distribution, which is inherently less flexible than the multivariate skew-t counterpart, affecting the generalizability (and robustness to outliers) by reaching a sub-optimal fit over the dataset.

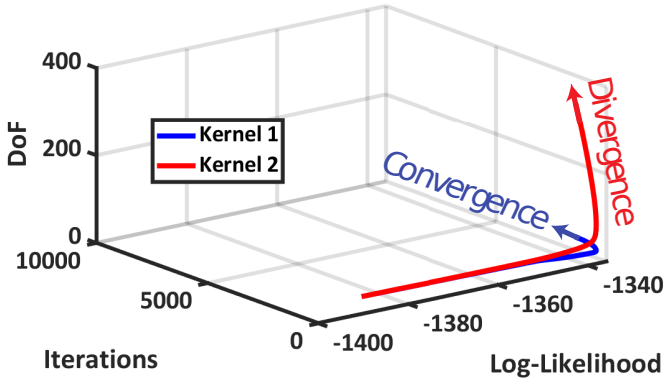


Figure 3.1: Example of DoF instability when fitting a FM-CFUST model over AIS 2D dataset (discussed in Section 3.3.1)

In the existing examples for fitting Skew-t mixture models, this issue is suppressed by either not updating the DoF value in the EM algorithm iterations if it exceeds a pre-defined heuristic value (e.g., 200) [96], or manually fixing/tuning the DoF [121]. This manual manipulation will impact the iterative mathematical process by blocking parts of the designed process. Thus in this work, we propose a regularization step as part of the optimization process to theoretically stabilize the DoF and avoid heuristic fixes while augmenting the generalizability of the proposed framework for unseen data points.

To avoid the DoF instability that can result in local minima generated by a normal or skew-normal kernel, a regularization step is implemented as an intermediate step between the E-Step and M-Step for the mixture of multivariate skew-t distributions with full skewness matrices. In this step, which henceforth will be addressed as *R-Step*, the expected value of the log-likelihood will be regulated using a monotonically increasing radially unbounded regularization function of DoF. We name the resulting iterative algorithm as *Expectation-Regularization-Maximization (ERM)* algorithm, and the following equations represent the mathematical expressions of

each step for the proposed ERM method:

1. *E-Step*:
$$Q(\boldsymbol{\Theta}|\hat{\boldsymbol{\Theta}}^{(k)}) = E(l(\boldsymbol{\Theta}, \boldsymbol{y})|\boldsymbol{y}, \hat{\boldsymbol{\Theta}}^{(k)}) \tag{3.8}$$

2. *R-Step*:
$$Q'(\boldsymbol{\Theta}|\hat{\boldsymbol{\Theta}}^{(k)}) = Q(\boldsymbol{\Theta}|\hat{\boldsymbol{\Theta}}^{(k)}) - \boldsymbol{\alpha}(\boldsymbol{\nu}^{(k)}) \tag{3.9}$$

3. *M-Step*:
$$\boldsymbol{\Theta}^{(k+1)} = \arg\max_{\boldsymbol{\Theta}} Q'(\boldsymbol{\Theta}|\boldsymbol{\Theta}^{(k)}) \tag{3.10}$$

In Eq. (3.9), $\boldsymbol{\alpha}(\boldsymbol{\nu}^{(k)})$ is the regularization function, and in this study, we define it as $\boldsymbol{\alpha}(\boldsymbol{\nu}^{(k)}) = \boldsymbol{B}\boldsymbol{\nu}^{(k)}$, in which $\boldsymbol{B}$ is a $g \times g$ diagonal matrix such that $\boldsymbol{B} = diag(\boldsymbol{\beta})$ where $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_g]$, and $\boldsymbol{\nu}^{(k)} = [\nu_1^{(k)}, \nu_2^{(k)}, \ldots, \nu_g^{(k)}]^T$. We refer to the $\boldsymbol{\beta}$ parameter as the first order *penalty vector*. We refer to the mixture model resulting from ERM algorithm as *finite mixture of regulated skew-t* kernels. The full algorithmic representation of the proposed ERM method is given in Algorithm 1.

3.2.1 E-Step

In order to fit an optimal multivariate skew-t mixture model (based on the MLE process) we should find the solution that maximizes the expected value of the log-likelihood (i.e., the Q-function). To calculate this expected value, it is needed to denote the log-likelihood in terms of the latent random variables of the hierarchical representation of the multivariate skew-t mixture model [101] so that the likelihood of each observation is treated separately and independently of other observations in the dataset. This probabilistic representation is explained in Appendix B. In order to compute the Q-function given the latent variables, five conditional expected values should be computed which will be used thereafter in the M-Step. These conditional expectations are denoted as follows.

$$
\begin{aligned}
z_{ij}^{(k)} &= E_{\boldsymbol{\Theta}^{(k)}}(\boldsymbol{Z}_{ij}|\boldsymbol{y}_j) \\
e_{1,ij}^{(k)} &= E_{\boldsymbol{\Theta}^{(k)}}(\log(W_j)|\boldsymbol{y}_j, z_{ij}=1) \\
e_{2,ij}^{(k)} &= E_{\boldsymbol{\Theta}^{(k)}}(W_j|\boldsymbol{y}_j, z_{ij}=1) \\
\boldsymbol{e}_{3,ij}^{(k)} &= E_{\boldsymbol{\Theta}^{(k)}}(W_j\boldsymbol{U}_j|\boldsymbol{y}_j, z_{ij}=1) \\
\boldsymbol{e}_{4,ij}^{(k)} &= E_{\boldsymbol{\Theta}^{(k)}}(W_j\boldsymbol{U}_j\boldsymbol{U}_j^T|\boldsymbol{y}_j, z_{ij}=1)
\end{aligned}
$$

In the aforementioned probabilistic expressions, $E_{\boldsymbol{\Theta}^{(k)}}$ is the expectation operator given the estimated set of parameters $\boldsymbol{\Theta}^{(k)}$ at k^{th} iteration. Although e_1, $\boldsymbol{e}_3$, and $\boldsymbol{e}_4$ are originally intractable expectations and cannot be written in closed form, by using one-step-late (OSL) approach [65] for e_1 in Eq. (3.12), and expressing $\boldsymbol{e}_3$ and $\boldsymbol{e}_4$ in terms of the first and second moments of the truncated t-distribution (See Appendix C) respectively [72, 93], the E-step can be executed in the closed form using the expressions in Eqs. (3.11)-(3.15).

$$
z_{ij}^{(k)} = \frac{\omega_i^{(k)} f_{p,\nu_i^{(k)}}(\boldsymbol{y}_j^{(k)}|\boldsymbol{\mu}_i^{(k)}, \boldsymbol{\Sigma}_i^{(k)}, \boldsymbol{\Delta}_i^{(k)})}{\sum_{i=1}^{g} \omega_i^{(k)} f_{p,\nu_i^{(k)}}(\boldsymbol{y}_j^{(k)}|\boldsymbol{\mu}_i^{(k)}, \boldsymbol{\Sigma}_i^{(k)}, \boldsymbol{\Delta}_i^{(k)})} \tag{3.11}
$$

$$
e_{1,ij}^{(k)} = e_{2,ij}^{(k)} - \log\left(\frac{\nu_i^{(k)} + d_i^{(k)}(\boldsymbol{y}_j)}{2}\right) - \left(\frac{\nu_i^{(k)} + p}{\nu_i^{(k)} + d_i^{(k)}}\right) - \psi(\frac{\nu_i^{(k)} + p}{2}) \tag{3.12}
$$

$$
e_{2,ij}^{(k)} = \left(\frac{\nu_i^{(k)} + p}{\nu_i^{(k)} + d_i^{(k)}(\boldsymbol{y}_j)}\right) \frac{T_{p,\nu_i^{(k)}+p+2}(\boldsymbol{y}_{2,j}^{(k)}|\boldsymbol{0}, \boldsymbol{\Lambda}_i^{(k)})}{T_{p,\nu_i^{(k)}+p}(\boldsymbol{y}_{1,j}^{(k)}|\boldsymbol{0}, \boldsymbol{\Lambda}_i^{(k)})} \tag{3.13}
$$

$$
\boldsymbol{e}_{3,ij}^{(k)} = e_{2,ij}^{(k)} E_{\boldsymbol{\Theta}^{(k)}}(\boldsymbol{U}_{ij}|\boldsymbol{y}_j) \tag{3.14}
$$

$$
\boldsymbol{e}_{4,ij}^{(k)} = e_{2,ij}^{(k)} E_{\boldsymbol{\Theta}^{(k)}}(\boldsymbol{U}_{ij}\boldsymbol{U}_{ij}^T|\boldsymbol{y}_j) \tag{3.15}
$$

In Eq. (3.12), $\psi(.)$ is the Digamma function, and in Eq. (3.13), for $\boldsymbol{y}_{1,j}$ and $\boldsymbol{y}_{2,j}$ we

have:

$$\boldsymbol{y}_{1,j} = \boldsymbol{c}_i(\boldsymbol{y}_j)\sqrt{\frac{\nu^{(k)}+p}{\nu^{(k)}+d^{(k)}(\boldsymbol{y}_j)}}$$
$$\boldsymbol{y}_{2,j} = \boldsymbol{c}_i(\boldsymbol{y}_j)\sqrt{\frac{\nu^{(k)}+p+2}{\nu^{(k)}+d^{(k)}(\boldsymbol{y}_j)}}$$

In Eq. (3.14) and Eq. (3.15), the latent variable $\boldsymbol{U}_{ij}$ is distributed by truncated t distribution denoted as:

$$\boldsymbol{U}_{ij}|\boldsymbol{y}_j \sim tt_{p,\nu^{(k)}+p+2}\left(\boldsymbol{c}_i^{(k)}(\boldsymbol{y}_j), \left(\frac{\nu^{(k)}+d_i^{(k)}(\boldsymbol{y}_j)}{\nu^{(k)}+p+2}\right)\boldsymbol{\Lambda}_i^{(k)}; \mathbb{R}_p^+\right)$$

where p indicates the dataset dimensionality, $(\nu^{(k)}+p+2)$ is the DoF, $\boldsymbol{c}_i^{(k)}(\boldsymbol{y}_j)$ is the mean, $\left(\frac{\nu^{(k)}+d_i^{(k)}(\boldsymbol{y}_j)}{\nu^{(k)}+p+2}\right)\boldsymbol{\Lambda}_i^{(k)}$ is the covariance, and $\mathbb{R}_p^+$ is the p-dimensional truncation hyperplane (that indicates that the probability only exists in the real positive p-dimensional hyperspace). The details on how to compute the first and second moments of truncated t distribution [135, 139], i.e., $E_{\boldsymbol{\Theta}^{(k)}}(\boldsymbol{U}_{ij}|\boldsymbol{y}_j)$ and $E_{\boldsymbol{\Theta}^{(k)}}(\boldsymbol{U}_{ij}\boldsymbol{U}_{ij}^T|\boldsymbol{y}_j)$, are discussed in Appendix C.

3.2.2 R-Step

To avoid DoF instability, we regulate the Q-function in the R-Step using the monotonically increasing radially unbounded regularization function of DoF (i.e., $\boldsymbol{B}\boldsymbol{\nu}^{(k)}$), as seen in Eq. (3.16). This regularization will act as a penalty term when combined with the expected value of the log-likelihood and thus penalizes the sub-optimal excessive increase in the DoF values throughout the course of the ERM optimization, keeping the model as a multivariate skew-t mixture.

$$Q'(\boldsymbol{\Theta}|\hat{\boldsymbol{\Theta}}^{(k)}) = Q(\boldsymbol{\Theta}|\hat{\boldsymbol{\Theta}}^{(k)}) - \boldsymbol{B}\boldsymbol{\nu}^{(k)} \tag{3.16}$$

The resulting regulated Q'-function will be used in the M-Step of the proposed ERM algorithm for multivariate skew-t based mixture models. This will inject the

regulation needed for the ERM algorithm to modulate the updating dynamics of DoF and to avoid the DoF instability which would result in suboptimal normal or skew-normal clusters in the final mixture model. The effect of the penalty vector on the DoF convergence during the ERM algorithm iterations is discussed in Section 3.3.3, and the results of our study showed the superior performance of the proposed method in comparison to the existing counterparts.

3.2.3 M-Step

For the M-step, using $\boldsymbol{\Theta}^{(k)}$ and the calculations in the E-step, the parameters for the mixture model are updated for the $(k+1)^{th}$ iteration to ensure global maximization of the Q'-function, as seen in Eq. (3.10). This expression yields the following steps in Eqs. (3.17)-(3.20) for updating the mixing proportion, mean vector, covariance matrix, and skewness matrix for each cluster.

$$\omega_i^{(k+1)} = \frac{1}{n}\sum_{j=1}^{n} z_{ij}^{(k)} \tag{3.17}$$

$$\boldsymbol{\mu}_i^{(k+1)} = \frac{\sum_{j=1}^{n} z_{ij}^{(k)} e_{2,ij}^{(k)} \boldsymbol{y}_j - \boldsymbol{\Delta}_i^{(k)} \sum_{j=1}^{n} z_{ij}^{(k)} \boldsymbol{e}_{3,ij}^{(k)}}{\sum_{j=1}^{n} z_{ij}^{(k)} e_{2,ij}^{(k)}} \tag{3.18}$$

$$\boldsymbol{\Delta}_i^{(k+1)} = \left(\sum_{j=1}^{n} z_{ij}^{(k)} (\boldsymbol{y}_j - \boldsymbol{\mu}_i^{(k+1)}) \boldsymbol{e}_{3,ij}^{(k)}\right)\left(\sum_{j=1}^{n} z_{ij}^{(k)} \boldsymbol{e}_{4,ij}^{(k)}\right)^{-1} \tag{3.19}$$

$$\begin{aligned}\boldsymbol{\Sigma}_i^{(k+1)} = \Bigg(\sum_{j=1}^{n} z_{ij}^{(k)} \Big[& e_{ij}^{(k)} (\boldsymbol{y}_j - \boldsymbol{\mu}_i^{(k+1)})(\boldsymbol{y}_j - \boldsymbol{\mu}_i^{(k+1)})^T - \boldsymbol{\Delta}_i^{(k+1)} \boldsymbol{e}_{3,ij}^{(k)} (\boldsymbol{y}_j - \boldsymbol{\mu}_i^{(k+1)})^T \\ & - (\boldsymbol{y}_j - \boldsymbol{\mu}_i^{(k+1)}) \boldsymbol{e}_{3,ij}^{(k)} \boldsymbol{\Delta}_i^{(k+1)} + \boldsymbol{\Delta}_i^{(k+1)} \boldsymbol{e}_{4,ij}^{(k)} \boldsymbol{\Delta}_i^{(k+1)T} \Big] \Bigg)\left(\sum_{j=1}^{n} z_{ij}^{(k)}\right)^{-1}\end{aligned} \tag{3.20}$$

Ultimately, for updating the DoF for each cluster, the maximization process is done by solving Eq. (3.21) for $\nu_i^{(k+1)}$, which is the derivative of the Q'-function with respect to the DoF when equated to zero.

$$\log\left(\frac{\nu_i^{(k+1)}}{2}\right) - \psi\left(\frac{\nu_i^{(k+1)}}{2}\right) + 1 - \frac{\sum_{j=1}^{n} z_{ij}^{(k)} (e_{2,ij}^{(k)} - e_{1,ij}^{(k)})}{\sum_{j=1}^{n} z_{ij}^{(k)}} - \frac{\partial \alpha(\nu_i^{(k+1)})}{\partial \nu_i^{(k+1)}} = 0 \tag{3.21}$$

Algorithm 1 shows how this approach can be implemented to train this distribution based on a given dataset $\boldsymbol{y}$.

Algorithm 1 FiMReS*t* Distribution

Input: $\boldsymbol{y}, g$

Output: $\boldsymbol{\mu}, \boldsymbol{\Delta}, \boldsymbol{\Sigma}, \boldsymbol{\nu}, \boldsymbol{\omega}, l$

Initialisation : $\boldsymbol{\mu}^{(0)}, \boldsymbol{\Delta}^{(0)}, \boldsymbol{\Sigma}^{(0)}, \boldsymbol{\nu}^{(0)}, \boldsymbol{\omega}^{(0)}$ [95]

Set $\delta\boldsymbol{\nu}$, δl, $\boldsymbol{\alpha}(\boldsymbol{\nu})$
$k \leftarrow 1$, $\boldsymbol{t}_1 \leftarrow \delta\boldsymbol{\nu} + 1$, $\boldsymbol{t}_2 \leftarrow \delta l + 1$
while $(|\boldsymbol{t}_1| > \delta\boldsymbol{\nu})$ and $(|\boldsymbol{t}_2| > \delta l)$ **do**
    Compute: Z, $e_1^{(k)}$, $e_2^{(k)}$, $\boldsymbol{e}_3^{(k)}$, $\boldsymbol{e}_4^{(k)}$ using Eqs. (3.11)-(3.15)
    Compute: $\boldsymbol{\omega}^{(k)}, \boldsymbol{\mu}^{(k)}, \boldsymbol{\Delta}^{(k)}, \boldsymbol{\Sigma}^{(k)}$ using Eqs. (3.17)-(3.20)
    Solve Eq. (3.21) to update $\boldsymbol{\nu}^{(k)}$
    Compute: $l^{(k)} = \sum_{j=1}^{n} \log\left(\sum_{i=1}^{g} \omega_i f_{p,\nu_i}(\boldsymbol{y}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \boldsymbol{\Delta}_i)\right)$
    if $(k > 2)$ **then**
        $\boldsymbol{t}_1 \leftarrow (\boldsymbol{\nu}^{(k)} - \boldsymbol{\nu}^{(k-1)})$
        $\boldsymbol{t}_2 \leftarrow (l^{(k)} - l^{(k-1)})$
    end if
    $k \leftarrow k + 1$
end while
return $\boldsymbol{\mu}, \boldsymbol{\Delta}, \boldsymbol{\Sigma}, \boldsymbol{\nu}, \boldsymbol{\omega}, l$

3.3 Results

In this section, we present the results of the FiMReS*t* implementation and the comparison of this new form of mixture modeling with previously proposed techniques (namely GMM, SNMM, STMM, and FM-CFUST) on several datasets (including experimental and synthetic). Also, we evaluate the dependency of the proposed model on the initial values of DoF in addition to the effect of the penalty vector on the DoF convergence.

3.3.1 Datasets

3.3.1.1 Australian Institute of Sport (AIS) Dataset

In this work, we utilize the AIS dataset, which is used frequently in the literature (e.g., [95, 98]) due to the size and inherent asymmetry that would provide a valuable

framework for evaluating the performance of various probabilistic mixture models. This dataset contains 13 different measurements, such as body fat and body mass index, from 202 male and female athletes [46]. In this study, we extract two sub-datasets.The first (that will be referred to as the *AIS 2D dataset* in the rest of this paper) is a bivariate set containing height in centimeters and percentage of body fat. The second (that will be referred to as the *AIS 3D dataset* in the rest of this paper) is a trivariate (3D) set that contains body mass index, lean body mass, and body fat.

3.3.1.2 Neck Muscle Activity Coherence Dataset

We test the performance of the FiMReS*t* model on three different datasets extracted from the muscle activity frequency coherence database acquired by our team at NYU using surface Electromyography (EMG) sensors on perilaryngeal and cranial muscles [136, 140]. This database contains 48 maximum (over frequency) coherence values (ranging from 0 to 1) of EMG muscle activation signals of 14 muscles during various vocal tasks, highlighting co-modulated muscle activations. We extract three different 2D datasets (named here as *Part 1*, *Part 2*, and *Part 3*), each composed of 2 dimensions representing two pairs of muscles as explained in the following:

1. The maximum coherence between the left and right Sternohyoid muscle; and The maximum coherence between left-lower and left-upper Sternocleidomastoid muscles.

2. The maximum coherence between the left front and back Trapezius; and The maximum coherence between the right front and back Trapezius.

3. The maximum coherence between the left and right Masseter; and The maximum coherence between the left and right Omohyoid.

3.3.1.3 Synthetic Dataset

In addition to the real-world experimental datasets, mentioned above, in this paper, we also devised a unitless synthetic dataset with embedded outliers by scattering two random asymmetrically distributed points in a 100×100 square,

Table 3.1: Log-Likelihood of various mixture models over the AIS 2D dataset

Model	Log-Likelihood
GMM	-1351.67
SNMM	-1341.12
STMM	-1340.95 [95]
FM-CFUST	-1335.60
FiMReS*t*	**-1335.20**

with a diagonal separation and outliers at the ends of the separating lines (seen in Fig. 3.6(a)). We refer to this dataset as *Synthetic Dataset* henceforth.

3.3.2 Comparative Study

Fig 3.2 shows the stabilized behavior of the DoF for the proposed FiMReS*t* over the course of ERM iterations, compared to the DoF divergence in fitting FM-CFUST (from the literature) using the EM algorithm. In this figure, the blue lines show the DoF of the kernels in the proposed FiMReS*t* model, and the red lines show the DoF of the kernels in the FM-CFUST model. For the FiMReS*t* model in this case, the penalty vector is chosen to be $\boldsymbol{\beta} = 5 \times \begin{bmatrix} 10^{-6} & 10^{-6} \end{bmatrix}^T$. As expected, we can see that by using the proposed Algorithm 1, the DoF divergence will be avoided, and the kernels will converge as the number of iterations increases.

The resulting convergence, seen in Fig. 3.2, will allow the training to continue while preserving the flexibility of the multivariate skew-t distribution and preventing the kernels from converging to a normal mixture distribution and preventing internal numerical divergence (which would halt the training). This behavior will allow the mixture model to continue the exploration and learning of the underlying physics of the dataset space boosting optimality and generalizability. The log-likelihood comparison from fitting various mixture models over AIS 2D dataset is given in Table 3.1. As can be seen, FiMReS*t* secures the highest log-likelihood when compared with all other mixture models which again supports the performance of the proposed algorithm. Fig 3.3 shows how the kernels of the FiMReS*t* model fit over AIS 2D.

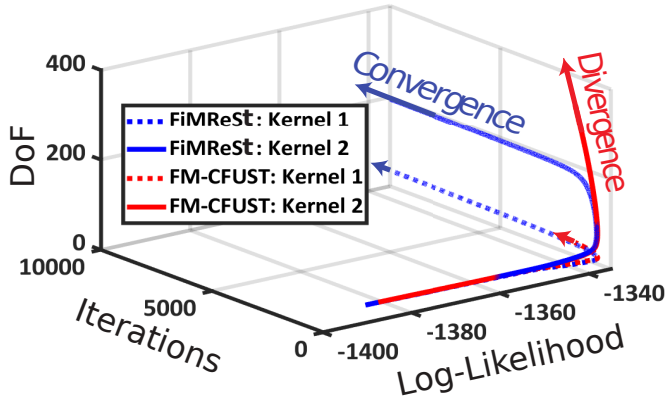


Figure 3.2: 3D plot of DoF and Log-likelihood vs. iteration for fitting the FiMReS*t* and FM-CFUST over the AIS 2D dataset. It can be seen that, unlike FM-CFUST, FiMReS*t* will preserve the converging behavior of the DoFs.

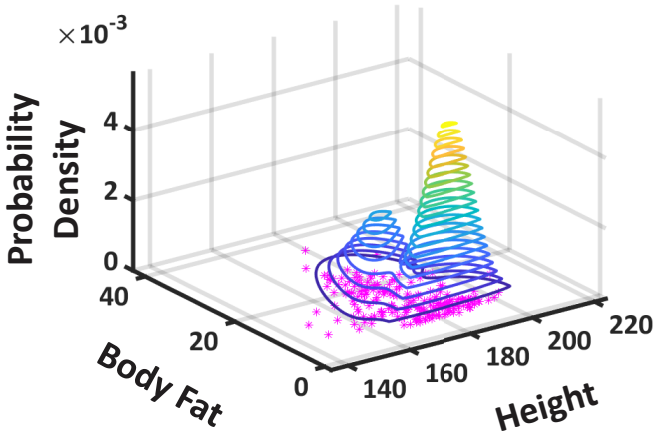


Figure 3.3: Probability density function visualization of the FiMReS*t* model over AIS 2D dataset.

Fig. 3.4 is about the AIS 3D dataset to show the performance of the model on a larger dimension. The figure shows the DoF behavior comparison during the proposed FiMReSt and FM-CFUST modeling. Similar to Fig. 3.2, the lines in Fig. 3.4 show the DoF values of the FiMReSt and FM-CFUST throughout the iterative training process. The same observation can be made in Fig. 3.4 regarding the AIS 3D dataset case that by avoiding the divergence of the DoF values, the FiMReSt model will continue the fitting process and achieve higher log-likelihood in comparison. (refer to Table 3.2). Fig. 3.5 depicts the visualization of the FiMReSt's kernels fitted on the AIS 3D dataset. The trained parameters of the FiMReSt model over the AIS 3D dataset are given in Table 3.3.

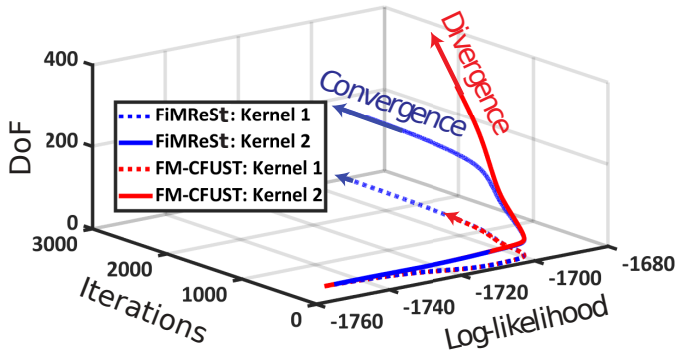


Figure 3.4: Comparison of DoF behavior during fitting iterations of FiMReSt and FM-CFUST over AIS 3D dataset

Table 3.2: Log-Likelihood of the various mixture modeling techniques over AIS 3D dataset

Model	Log-Likelihood
GMM	-1747.20
SNMM	-1726.17
STMM	-1725.01
FM-CFUST	-1700.17 [98]
FiMReSt	**-1692.08**

Fig. 3.6 shows the comparison between GMM, SNMM, STMM, FM-CFUST, and the proposed FiMReSt over the synthetic dataset, and Table 3.4 enumerates

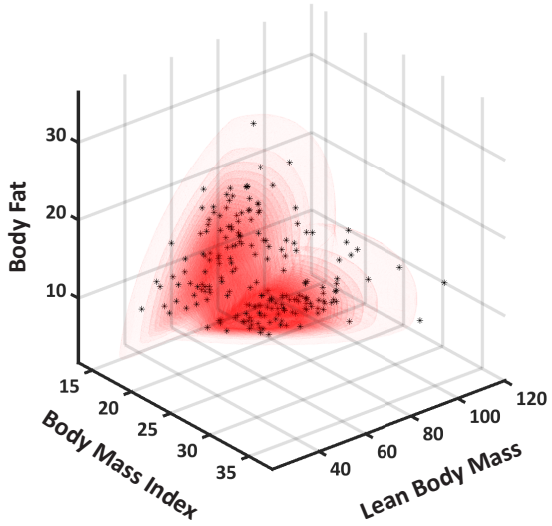


Figure 3.5: Probability density function visualization of the FiMReSt kernels over AIS 3D dataset. Higher probabilities are shown by darker red color in the visualization. The outermost 3D layer indicates 1% of the maximum probability produced by the FiMReSt model in the 3D space.

Table 3.3: Defining parameters of the FiMReSt model given the AIS 3D dataset

i	1	2
$\boldsymbol{\mu}_i$	$\begin{bmatrix} 19.745 \\ 57.698 \\ 11.737 \end{bmatrix}$	$\begin{bmatrix} 20.565 \\ 63.694 \\ 5.773 \end{bmatrix}$
$\boldsymbol{\Delta}_i$	$\begin{bmatrix} 3.616 & -1.979 & 1.090 \\ 5.652 & -8.113 & -1.407 \\ 3.582 & -3.489 & 7.577 \end{bmatrix}$	$\begin{bmatrix} 2.792 & -0.294 & 1.170 \\ 6.544 & 4.343 & 1.021 \\ 0.198 & -0.053 & 3.779 \end{bmatrix}$
$\boldsymbol{\Sigma}_i$	$\begin{bmatrix} 0.046 & -0.377 & -0.116 \\ -0.377 & 7.170 & 2.411 \\ -0.116 & 2.411 & 0.816 \end{bmatrix}$	$\begin{bmatrix} 0.678 & 5.733 & 0.243 \\ 5.733 & 48.535 & 2.055 \\ 0.243 & 2.055 & 0.088 \end{bmatrix}$
ν_i	173.856	7.443
ω_i	0.4806	0.5194
β_i	0.0001	0.0001

the log-likelihood values of the corresponding models. In Fig. 3.6(a), the synthetic dataset is visualized. In Fig. 3.6(b), a GMM with two Gaussian kernels is trained on the synthetic dataset. In Fig. 3.6(c), we can see how the skewness in SNMM will cause the kernels to adapt better to the formation of the data sparsity. In Fig. 3.6(d), the STMM model is shown, and the difference between STMM and SNMM that is caused by changing the skew-normal clusters to skew-t clusters can be seen. Fig. 3.6(e) shows the FM-CFUST model trained over the synthetic dataset benefiting from the full skewness matrix (rather than diagonal) resulting in a better fit when compared with STMM. Fig. 3.6(f) depicts the behavior of the proposed FiMReSt model trained on the synthetic dataset. It can be seen that by stabilizing the DoF parameters of the kernels in the proposed ERM algorithm, avoiding DoF divergence, while exploiting the full potential of the skewness matrix using the math of Skew-t distribution, the FiMReSt model will converge to a higher log-likelihood and secure the best fit, achieving a more conforming shape to the dataset. The kernels in the FiMReSt model are showing more reach towards the outliers of the synthetic dataset, causing the clustering to be more divided in comparison to FM-CFUST and other models.

Table 3.4: Log-Likelihood of the synthetic dataset given different models

Model	Log-Likelihood
GMM	-1390.541
SNMM	-1389.482
STMM	-1389.924
FM-CFUST	-1364.918
FiMReSt	**-1362.914**

Fig. 3.7 shows the comparison of the FiMReSt model with GMM, SNMM, STMM, and FM-CFUST trained on the experimental EMG coherence dataset for all data parts 1, 2 and 3. In Fig. 3.7, the rows indicate which data parts used, and columns show the visualization of the resulting clustering methods, i.e., GMM, SNMM, STMM, FM-CFUST, and FiMReSt, from left to right, respectively. The penalty vector for FiMReSt model for all three parts of the dataset is $\boldsymbol{\beta} = \begin{bmatrix} 10^{-6} & 10^{-6} \end{bmatrix}^T$. In Fig. 3.7, it can be observed that the FiMReSt distribution will adapt more efficiently

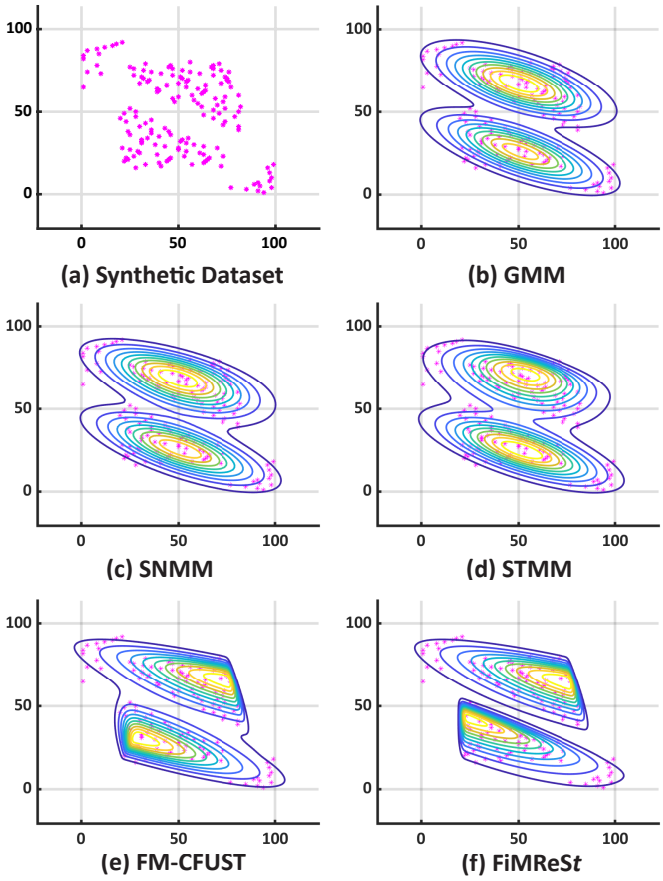


Figure 3.6: Comparison of GMM, SNMM, STMM, FM-CFUST and FiMReSt on the synthetic dataset

to the datasets as the generalizability of the multivariate skew-t distribution is preserved during the ERM algorithm iterations and this is supported by the numerical results given in Table Table 3.5. In addition to this main observation, there are other points to be considered, as explained in the following. In Fig. 3.7, the difference between GMM and SNMM can be seen due to the added skewness. The SNMM model will conform to the dataset more considerably by relaxing the symmetricity assumption of the Gaussian kernels. Thereafter, the replacement of MSN with MST kernels results in a noticeable change in STMM compared to SNMM. Going one step further from STMM, by taking advantage of the full skewness matrix, the FM-CFUST model will show more conformity to the given datasets. Finally, by implementing the proposed ERM method and avoiding the

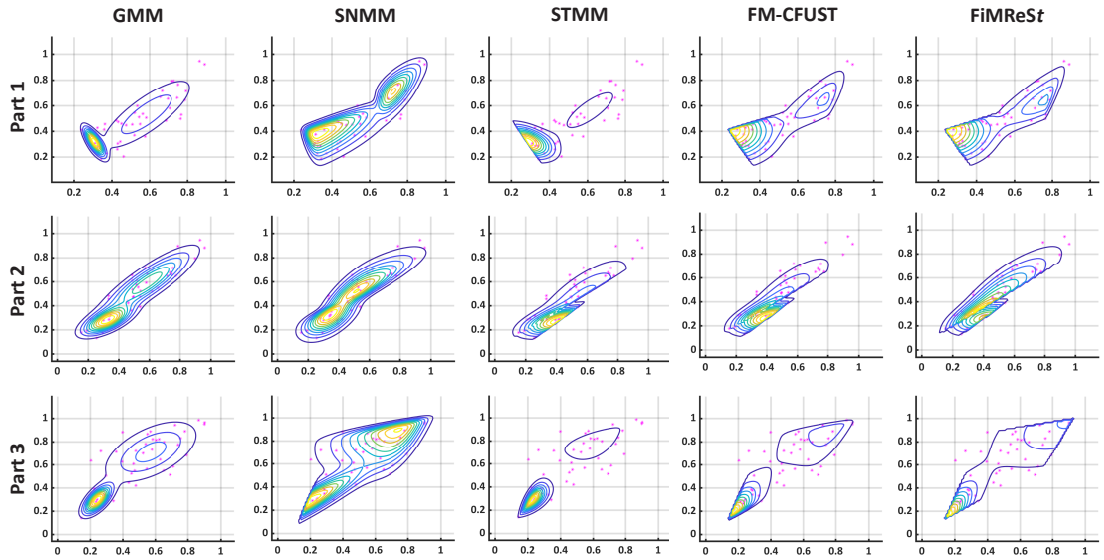


Figure 3.7: The comparison between GMM, SNMM, STMM, FM-CFUST, and FiMReSt over coherence datasets

DoF divergence while taking into account the full skewness, the FiMReSt model further expands towards modeling the outliers in the scattered segments of the data, resulting in a more optimal and generalizable clustering behavior and a higher log-likelihood, given in Table 3.5.

Table 3.5: Log-likelihood of GMM, SNMM, STMM, FM-CFUST, and FiMReSt over coherence parts 1, 2, and 3

Model	Part 1	Part 2	Part 3
GMM	67.0987	64.3616	45.3951
SNMM	69.7102	64.5459	51.0562
STMM	70.4885	71.3382	49.5397
FM-CFUST	75.2098	72.4159	52.3720
FiMReSt	**76.5683**	**72.9244**	**58.1812**

3.3.3 Initial Conditions

In this sub-section, the sensitivity of the proposed model, in terms of the convergence of the DoF, to the initial values of the DoF is investigated. High sensitivity would reduce the practicality of the method. Fig. 3.8 shows the convergence of the DoF values of FiMReSt model for the two clusters of the AIS 3D dataset, considering a large range of initial values for DoF, i.e., $\nu_i \in \{2, 20, 100, 150, 250\}$. We can see that the resulting FiMReSt model has minimal to no dependency on the initial values of DoF, which corroborates the robustness of this mixture model to the DoF initialization. This further supports the practicality of the proposed model.

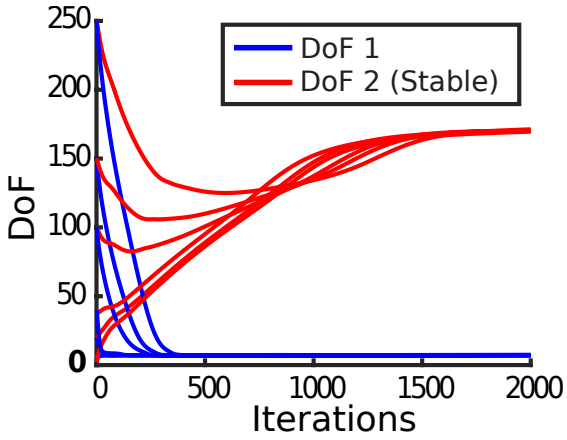


Figure 3.8: DoF convergence of the FiMReSt over the AIS 3D model given different initial values. The red lines show the stabilized DoF.

3.4 Discussions

In this chapter, we proposed an expectation-regularization-maximization algorithm to train a finite mixture of multivariate skew-t distributions. The resulting mixture model is named Finite Mixture of Multivariate Regulated Skew-t (FiMReSt) Kernels, which, unlike its counterparts in the literature, is able to stabilize the DoF profile during the optimization process of learning, increasing generalizability and robustness to outliers. The proposed method is evaluated through a comprehensive comparative study based on real and synthetic data. The results support the

higher performance of the proposed FiMReSt model when compared with existing methods (i.e., GMM, SNMM, STMM, FM-CFUST) in terms of the log-likelihood of fit and the numerical stability of DoF. Additional analysis highlighting the low sensitivity of the proposed method to the initialization of DoF further supports the practicality of the approach.

Chapter 4

SNMR: Skew-Normal Mixture Regression – A flexible probabilistic trajectory generator for Learning from Demonstration

In this chapter, we use Skew-Normal Mixture Regression (SNMR) to infer the most probable human behavior in LfD, modeled by a more flexible mixture model (SNMM). This probabilistic model enables higher flexibility in comparison to the conventional GMM due to its skewness, which can facilitate derivation of more intricate characteristics of the human.

The assumption of the Gaussian distribution of the inherent kernels of data clusters cannot be guaranteed in many cases, especially for complex data space (such as therapeutic responses in the force domain). This assumption cannot be made when data segments have high skewness. In other words, the symmetrical behavior that Gaussian kernels entail dictates a level of rigidity that is not necessarily observed in practical datasets, including on telerobotic demonstration of rehabilitation. This lack of flexibility prevents accurate modeling of demonstration, which challenges the accuracy of machine learning. Thus, it is imperative to relax this constraint using flexible probabilistic functions. This requires novel mathematical derivations.

In order to tackle the issues, different probability density functions (PDFs) can replace the Gaussian kernels of the mixture to form a more flexible representation,

which can encode the skewness and take into account generalizability and robustness. For this purpose, skew-normal PDF can enable more flexibility in modeling when substituted normal PDF in GMM.

In contrast to the Gaussian distribution, which is restrictive to symmetric behavior, skew-normal PDF relaxes the symmetricity by utilizing the skewness (Δ) parameter in addition to the main defining parameters of a Gaussian kernel which are covariance (Σ) and mean (μ).

Thus, if a new iterative EM approach can be derived for the data clusters in the context of learning from demonstration, the resulting mixture model can address the aforementioned challenges of GMM. A comprehensive literature review is conducted for this work, and some efforts have been identified in the literature [36, 61, 107, 109, 121, 148, 185] to generate more flexible mixture models in comparison to GMM. However, each of the existing methods has several restrictions due to the complexity of the mathematical derivation.

To address the issues above, in this chapter, we will improve the probabilistic modeling of the robotic demonstrations using the skew-normal mixture model (SNMM) and use the resulting model to learn the operator's demonstrations in the robot platform by calculating the skew-normal mixture regression (SNMR) model. We continue this section by explaining the preliminaries of SNMM modeling. In the consecutive sections, we will present the different representations and mathematical expressions for the regression modeling, which is followed by the results of this method on robotic data.

4.1 Preliminaries

This section will cover the mathematical preliminaries for SNMM and the corresponding EM algorithm that fits this mixture model onto a given dataset.

4.1.1 Probability Density Function

The SNMM function with G multivariate skew-normal (MSN) clusters that probabilistically models a dataset is defined as a weighted summation of skew-normal PDFs as seen in equation (4.1), in which w_i, $\boldsymbol{\mu}_i$, $\boldsymbol{\Sigma}_i$, and $\boldsymbol{\Delta}_i$, are the mixing

proportion, mean vector, covariance matrix, and skewness matrix of i^{th} skew-normal cluster respectively, and $\boldsymbol{\Theta}$ is the set of all defining properties of all skew-normal clusters in the model.

$$f_{SNMM}(\boldsymbol{y}, \boldsymbol{\Theta}) = \sum_{i=1}^{G} w_i MSN(\boldsymbol{y}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \boldsymbol{\Delta}_i) \tag{4.1}$$

The governing equation of skew-normal PDF seen in the equation above, is provided by Sahu et al. [159] as a member of skew-elliptical distributions. If $\boldsymbol{y}$ is a random p-dimensional vector distributed by a skew-normal distribution with mean vector $\boldsymbol{\mu}$, covariance matrix $\boldsymbol{\Sigma}$, and skewness matrix $\boldsymbol{\Delta}$, then the probability density function would be written as seen in equation (4.2).

$$MSN(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\Delta}) = 2^p \phi_p(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Omega}) \Phi_p(\boldsymbol{\Delta}^T \boldsymbol{\Omega}^{-1}(\boldsymbol{y} - \boldsymbol{\mu})|\mathbf{0}_p, I_p - \boldsymbol{\Delta}^T \boldsymbol{\Omega}^{-1} \boldsymbol{\Delta}) \tag{4.2}$$

In this equation, $\boldsymbol{\Omega} = \boldsymbol{\Sigma} + \boldsymbol{\Delta}\boldsymbol{\Delta}^T$, $\mathbf{0}_p$ is the p-dimensional zero vector, $\boldsymbol{I}_p$ is the $p \times p$ identity matrix, ϕ_p is the p-dimentional normal PDF and Φ_p is the normal CDF.

The MSN PDF can also be represented in a hierarchical format as seen below in expressions (4.3)-(4.5).

$$\boldsymbol{Y}_j|(\boldsymbol{\tau}_j, Z_{ij} = 1) \sim N_p(\boldsymbol{\mu}_i + \boldsymbol{\Delta}_i \boldsymbol{\tau}_j, \boldsymbol{\Sigma}_i) \tag{4.3}$$

$$\boldsymbol{\tau}_j|(Z_{ij} = 1) \sim HN_p(\mathbf{0}_p, \boldsymbol{I}_p) \tag{4.4}$$

$$Z_{ij} \sim \mathbb{M}(1; w_1, w_2, \ldots, w_G) \tag{4.5}$$

In these equations, HN_p denotes the p-dimensional half-normal distribution, $\mathbb{M}$ denotes the multinomial distribution with one trial and cell probabilities the same as mixing proportions of the mixture model. To explain these expressions further, we can state that if $Z_{ij} = 1$, then the j^{th} observation of the dataset solely arises from the i^{th} MSN cluster in the mixture model.

The hierarchical representation of the SNMM is useful for defining the log-likelihood function using the latent variables $\boldsymbol{\tau}$ and Z, which will then be used to optimize the defining parameters of the mixture model to fit the dataset using EM algorithm. The following subsection will explain the mathematical derivation of the iterative EM algorithm for SNMM modeling.

4.1.2 EM Algorithm

The fitness for probabilistic algorithms can be defined as log-likelihood. The log-likelihood is defined for SNMM as seen in equation (4.3). In this equation, n is the number of observations in the dataset $\boldsymbol{y}$.

$$l(\boldsymbol{\Theta}|\boldsymbol{y}) = \sum_{j=1}^{n} log(\sum_{i=1}^{G} w_i MSN(\boldsymbol{y}_j|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \boldsymbol{\Delta}_i)) \tag{4.6}$$

We can expand equation (4.6) based on the hierarchical representation of the mixture model to get to the log-likelihood function as seen in equation (4.7).

$$\begin{aligned} l(\boldsymbol{\Theta}|\boldsymbol{y}, Z, \boldsymbol{\tau}) = \sum_{i=1}^{G}\sum_{j=1}^{n} Z_{ij}\{log(w_i) - \\ \frac{1}{2}log|\boldsymbol{\Sigma}_i| - \frac{1}{2}(y_j - \boldsymbol{\mu}_i - \boldsymbol{\Delta}_i\boldsymbol{\tau}_j)^T\boldsymbol{\Sigma}_i^{-1}(y_j - \boldsymbol{\mu}_i - \boldsymbol{\Delta}_i\boldsymbol{\tau}_j) - \frac{1}{2}\boldsymbol{\tau}_j^T\boldsymbol{\tau}_j\} \end{aligned} \tag{4.7}$$

The EM algorithm is an iterative process that takes place in two consecutive steps, namely, expectation ad maximization. In this approach, first in the expectation step (E-Step), the conditional expected value of the log-likelihood function, given the estimated parameters of the mixture model and the observed data, will be calculated as a function of the latent variables seen in the hierarchical format of the mixture model. Thereafter, in the maximization step (M-Step), this expected value will be maximized over all the defining parameters of the mixture model to update the parameters for the next iteration of the EM algorithm. This process continues until the convergence of the log-likelihood.

4.1.2.1 E-Step

To fit the SNMM, the EM algorithm should maximize the expected value of the log-likelihood function at each step until the convergence of the model. The expected value of the log-likelihood function (also known as the Q-function) can be written as stated in equation (4.8).

$$E(l(\boldsymbol{\Theta}|\boldsymbol{y}, Z, \boldsymbol{\tau})|\boldsymbol{y}, \hat{\boldsymbol{\Theta}}^{(k)}) = Q(\boldsymbol{\Theta}|\hat{\boldsymbol{\Theta}}^{(k)}) =$$
$$\sum_{i=1}^{G}\sum_{j=1}^{n} \hat{z}_{ij}^{(k)}\big\{log(w_i) + \frac{1}{2}|\boldsymbol{\Sigma}_i^{-1}| - \frac{1}{2}(y_j - \boldsymbol{\mu}_i - \boldsymbol{\Delta}_i\hat{\boldsymbol{\eta}}_{ij}^{(k)})^T\boldsymbol{\Sigma}_i^{-1}(y_j - \boldsymbol{\mu}_i - \boldsymbol{\Delta}_i\hat{\boldsymbol{\eta}}_{ij}^{(k)})$$
$$- \frac{1}{2}tr\big(\boldsymbol{\Sigma}_i^{-1}\boldsymbol{\Delta}_i(\hat{\Psi}_{ij}^{(k)} - \hat{\boldsymbol{\eta}}_{ij}^{(k)}\hat{\boldsymbol{\eta}}_{ij}^{(k)^T})\boldsymbol{\Delta}_i^T\big)\big\} \quad (4.8)$$

In this equation, three latent random values are calculated in the k^th iteration of the EM algorithm. These values are:

$$\hat{z}_{ij}^{(k)} = E(\boldsymbol{Z}_{ij}|\boldsymbol{y}_j, \boldsymbol{\Theta}^{(k)})$$
$$\hat{\boldsymbol{\eta}}_{ij}^{(k)} = E(\boldsymbol{\tau}_j|Z_{ij} = 1, \boldsymbol{y}_j, \boldsymbol{\Theta}^{(k)})$$
$$\hat{\Psi}_{ij}^{(k)} = E(\boldsymbol{\tau}_j\boldsymbol{\tau}_j^T|Z_{ij} = 1, \boldsymbol{y}_j, \boldsymbol{\Theta}^{(k)})$$

As mentioned above, in the E-Step, Q-function is to be calculated given the latent random variables of $\hat{z}_{ij}^{(k)}$, $\hat{\boldsymbol{\eta}}_{ij}^{(k)}$, and $\hat{\Psi}_{ij}^{(k)}$. For computing $\hat{z}_{ij}^{(k)}$ we can write the following equation:

$$\hat{z}_{ij}^{(k)} = E(Z_{ij}|\boldsymbol{y}_j, \boldsymbol{\Theta}^{(k)}) = \frac{\hat{w}_i MSN(\boldsymbol{y}_j|\hat{\boldsymbol{\mu}}_i^{(k)}, \hat{\boldsymbol{\Sigma}}_i^{(k)}, \hat{\boldsymbol{\Delta}}_i^{(k)})}{\sum_{m=1}^{G}\hat{w}_m MSN(\boldsymbol{y}_j|\hat{\boldsymbol{\mu}}_m^{(k)}, \hat{\boldsymbol{\Sigma}}_m^{(k)}, \hat{\boldsymbol{\Delta}}_m^{(k)})} \quad (4.9)$$

Using Bayes theorem and the conditional probabilities in (4.3), (4.4), and (4.5), we can prove that:

$$\boldsymbol{\tau}_j|(\boldsymbol{Y}_j = \boldsymbol{y}_j, Z_{ij} = 1) \sim TN_p(\boldsymbol{\Delta}_i^T\boldsymbol{\Omega}_i^{-1}(\boldsymbol{y}_j - \boldsymbol{\mu}_i)|\boldsymbol{0}_p, \boldsymbol{I}_p - \boldsymbol{\Delta}_i^T\boldsymbol{\Omega}_i^{-1}\boldsymbol{\Delta}_i; \mathbb{R}^+) \quad (4.10)$$

which is the p-dimensional truncated normal distribution for the positive real hyperspace. The details of how to compute the expected value of a random variable distributed by a truncated normal distribution are derived mathematically in [107], equations (10) and (11).

4.1.2.2 M-Step

After computing the Q-funciton, we need to maximize this value over the defining parameters of the mixture model. Hence, we update the parameters in the following order:

$$\hat{w}_i^{(k+1)} = \frac{\sum_{j=1}^{n} \hat{z}_{ij}^{(k)}}{n} \tag{4.11}$$

$$\hat{\boldsymbol{\mu}}_i^{(k+1)} = \frac{\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \boldsymbol{y}_j - \hat{\boldsymbol{\Delta}}_i^{(k)} \sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \hat{\boldsymbol{\eta}}_{ij}^{(k)}}{\sum_{j=1}^{n} \hat{z}_{ij}^{(k)}} \tag{4.12}$$

$$\hat{\boldsymbol{\Delta}}_i^{(k+1)} = \Big(\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} (\boldsymbol{y}_j - \hat{\boldsymbol{\mu}}_i^{(k+1)}) \hat{\boldsymbol{\eta}}_{ij}^{(k)^T} \Big) \Big(\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \hat{\boldsymbol{\Psi}}_{ij}^{(k)} \Big)^{-1} \tag{4.13}$$

$$\begin{aligned} \hat{\boldsymbol{\Sigma}}_i^{(k+1)} = \Bigg\{ \sum_{j=1}^{n} \hat{z}_{ij}^{(k)} (\boldsymbol{y}_j - \hat{\boldsymbol{\mu}}_i^{(k+1)} - \hat{\boldsymbol{\Delta}}_i^{(k+1)} \hat{\boldsymbol{\eta}}_{ij}^{(k)}) (\boldsymbol{y}_j - \hat{\boldsymbol{\mu}}_i^{(k+1)} - \hat{\boldsymbol{\Delta}}_i^{(k+1)} \hat{\boldsymbol{\eta}}_{ij}^{(k)})^T + \hat{\boldsymbol{\Delta}}_i^{(k+1)} \\ \times \Big(\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} (\hat{\boldsymbol{\Psi}}_{ij}^{(k)} - \hat{\boldsymbol{\eta}}_{ij}^{(k)} \hat{\boldsymbol{\eta}}_{ij}^{(k)^T}) \Big) \hat{\boldsymbol{\Delta}}_i^{(k+1)^T} \Bigg\} \Bigg(\sum_{j=1}^{n} \hat{z}_{ij}^{(k)} \Bigg)^{-1} \end{aligned} \tag{4.14}$$

4.1.3 Initialization

The process of mixture modeling is highly non-convex, meaning that the final results are prone to be suboptimal. One way to prevent this occurrence is to have an intricate initialization scheme for the mixture model's defining parameters in order to begin the mixture modeling as close as possible to the optimal outcome.

For the SNMM, in the literature [107], it is proposed that the K-means algorithm determines the initial clusters' weight, mean vectors, and covariance matrices, and the skewness will be initialized as a diagonal matrix with values of slightly deviated from zero. In this proposed work, the initialization is more details to ensure a better starting point for the clusters in the mixture model.

First, we model a GMM onto the observed dataset and the resulting parameters of the fitted GMM, namely, clusters' weight, mean vectors, and covariance matrices denoted as $w_{i_{GMM}}$, $\boldsymbol{\mu}_{i_{GMM}}$, and $\boldsymbol{\Sigma}_{i_{GMM}}$ respectively, will be used to initialize the SNMM parameters. To initialize SNMM parameters, we start with the skewness matrices of the clusters and then proceed with initializing the other parameters based on the skewness and the resulting GMM.

To initialize the skewness, we will cluster the data points based on the highest probability achieved from the Gaussian clusters in the GMM model. Eventually, there will be G clusters formed. We denote each cluster as $\boldsymbol{y}^g$ as the g^{th} cluster,

where $g \in \{1, 2, \ldots, G\}$. Equation (4.15) shows how each cluster is formed.

$$\boldsymbol{y}^g = [y_1^g, y_2^g, \ldots, y_m^g] \tag{4.15}$$

where

$$\begin{aligned} & m \in \{1, 2, \ldots, M\} \text{ and } M < N \\ & N(\boldsymbol{y}_m^i | \boldsymbol{\mu}_{i_{GMM}}, \boldsymbol{\Sigma}_{i_{GMM}}) > N(\boldsymbol{y}_m^j | \boldsymbol{\mu}_{j_{GMM}}, \boldsymbol{\Sigma}_{j_{GMM}}) \\ & \text{s.t. } i \neq j \text{ and } i \in \{1, 2, \ldots, G\} \end{aligned}$$

After this hard-assigned clustering, we can determine the skewness matrices as seen in equation (4.16).

$$\boldsymbol{\Delta}_i = \alpha \times Diag(\delta_1, \delta_2, \ldots, \delta_p) \text{ and } \alpha = 0.1 \tag{4.16}$$

where each element δ is the one-dimensional skewness of the i^{th} cluster along each of the dimensions in the p-dimensional hyperspace of the dataset:

$$\delta_k = \frac{E(y_{j_k} - \bar{\mu}_k)^3}{\sigma_k^3} \text{ s.t. } k \in \{1, 2, \ldots, p\}$$

This initialization of skewness matrices will ensure a slight deviation from the symmetricity of the distribution toward where the observed data is leaning. This would give the MSN clusters in SNMM a better head-start when the EM algorithm starts to fit the model.

Based on the mathematical definition of MSN distribution, having a non-zero skewness matrix will expand the distribution's covariance and deviate the expected value of the distribution from the Gaussian counterpart. This means that by initializing the skewness, as mentioned, if we use the same resulting mean vector and covariance matrix of the GMM model, the MSN cluster might be deviated more than needed from its Gaussian counterpart, delaying convergence and also adding to the possibility of suboptimal outcomes. To counter this occurrence, we should ensure that the MSN clusters' starting expected value and covariance parameters are the same as the Gaussian clusters in the GMM. This way, at the start of the EM algorithm, the MSN clusters would have leaned toward the direction the observed data favors. This deviation caused by the skewness will be the initial

driving momentum for the MSN clusters to optimize towards their most optimal form. Based on the mentioned points, we initialize the mean vector and covariance matrices as stated in equations (4.17) and (4.18).

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}_{i_{GMM}} - \sqrt{\frac{2}{\pi}}\boldsymbol{\Delta}_i \mathbf{1}_p \tag{4.17}$$

where $\mathbf{1}_p$ is the p-dimensional vector of ones.

$$\boldsymbol{\Sigma}_i = \boldsymbol{\Sigma}_{i_{GMM}} - \left(1 - \frac{2}{\pi}\right)\boldsymbol{\Delta}_i \boldsymbol{\Delta}_i^T \tag{4.18}$$

It should be noted that choosing $\alpha = 0.1$ in equation (4.16) might lead to $\boldsymbol{\Sigma}_i$ to be non-positive-semidefinite, in which case we can choose a value such as $\alpha = 0.05$ to avoid this issue. As for the mixing proportions, we use the final cluster weights of GMM as the initial values for the SNMM's weights.

$$w_i = w_{i_{GMM}} \tag{4.19}$$

It is important to mention that before fitting an SNMM onto the data, we normalize the data in all dimensions by scaling it down to the range between 0 and 1. This will ensure that the update functions will affect the parameters equally. After the mixture modeling and regression model's trajectory generation, we can scale the observed data and the resulting regression to the dataset's original range.

4.2 Methods

Regression modeling is conventionally used in GMM-based learning from demonstration to generate to most optimal robot trajectory, given the set of task demonstrations [126]. In other words, after the p-dimensional dataset is probabilistically modeled, regression modeling will try to identify the most optimal set of values for the undetermined dimensions if some of the dimensions' values are deterministically known. For instance, if the dataset is two-dimensional with one dimension being time and the other being one joint value of a robot, after the probabilistic modeling is done, using the regression modeling, we can identify what is the most expected joint value (the probabilistic dimension) at a certain point of time (the deterministic

dimension). Based on this definition, regression models are conditional expected values of the probabilistic models.

Regression models of the probabilistic clusterings will be in the form of a weighted summation of all of the PDF clusters' conditional parameters given the determined dimensions. In the case of GMM, for deriving the regression function known as Gaussian Mixture Regression (GMR), the conditional probability of the GMM function should be calculated, which will be formed as seen in equation (4.20).

$$f_{Y_2|Y_1}(\boldsymbol{y_2}) = f_{GMM}(\boldsymbol{Y}_2|\boldsymbol{\Theta}, \boldsymbol{Y}_1 = \boldsymbol{y}_1) \tag{4.20}$$

In equation (4.20), $\boldsymbol{Y}_1$ and $\boldsymbol{Y}_2$ represent the set of deterministic and probabilistic dimensions respectively. Expanding this equation will lead to the expression in equation (4.21):

$$f_{Y_2|Y_1}(\boldsymbol{y_2}) = \sum_{i=1}^{G} \pi_i(y_2) N(y_2|\boldsymbol{\mu}_i^\star, \boldsymbol{\Sigma}_i^\star) \tag{4.21}$$

where

$$\boldsymbol{\mu}_i^\star = \boldsymbol{\mu}_{i_2} + \boldsymbol{\Sigma}_{i_{21}} \boldsymbol{\Sigma}_{i_{11}}^{-1} (\boldsymbol{y}_1 - \boldsymbol{\mu}_{i_1})$$

$$\boldsymbol{\Sigma}_i^\star = \boldsymbol{\Sigma}_{i_{22}} - \boldsymbol{\Sigma}_{i_{21}} \boldsymbol{\Sigma}_{i_{11}}^{-1} \boldsymbol{\Sigma}_{i_{12}}$$

$$\pi_i(\boldsymbol{y}_2) = \frac{w_i N(\boldsymbol{y}_2|\boldsymbol{\mu}_{i_2}, \boldsymbol{\Sigma}_{i_{22}})}{\sum_{g=1}^{G} w_g N(\boldsymbol{y}_2|\boldsymbol{\mu}_{g_2}, \boldsymbol{\Sigma}_{g_{22}})}$$

and for each Gaussian cluster, we have:

$$f_i\left(\begin{bmatrix}\boldsymbol{Y}_1 \\ \boldsymbol{Y}_2\end{bmatrix}\right) = N\left(\begin{bmatrix}\boldsymbol{\mu}_{i_1} \\ \boldsymbol{\mu}_{i_2}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{i_{11}} & \boldsymbol{\Sigma}_{i_{12}} \\ \boldsymbol{\Sigma}_{i_{21}} & \boldsymbol{\Sigma}_{i_{22}}\end{bmatrix}\right)$$

Using equation (4.21), by knowing the values of $\boldsymbol{Y}_1$, we can calculate what would be the expected value and scatteredness of the data on $\boldsymbol{Y}_2$, as seen in equations (4.22) and (4.23).

$$E(\boldsymbol{Y}_2|\boldsymbol{Y}_1 = \boldsymbol{y}_1) = \sum_{i=1}^{G} \pi_i \boldsymbol{\mu}_i^\star = \sum_{i=1}^{G} \pi_i (\boldsymbol{\mu}_{i_2} + \boldsymbol{\Sigma}_{i_{21}} \boldsymbol{\Sigma}_{i_{11}}^{-1} (\boldsymbol{y}_1 - \boldsymbol{\mu}_{i_1})) \tag{4.22}$$

$$Var(\boldsymbol{Y}_2|\boldsymbol{Y}_1 = \boldsymbol{y}_1) = \sum_{i=1}^{G} \pi_i \boldsymbol{\Sigma}_i^{\star} = \sum_{i=1}^{G} \pi_i (\boldsymbol{\Sigma}_{i_{22}} - \boldsymbol{\Sigma}_{i_{21}} \boldsymbol{\Sigma}_{i_{11}}^{-1} \boldsymbol{\Sigma}_{i_{12}}) \tag{4.23}$$

As it is seen in equation (4.22), the outcome of the GMR is the summation of linear functions. This is due to the inherent symmetricity of the Gaussian distribution, which imposes linearity on the conditional expectation. Conversely, in skew-normal distribution, this symmetricity is violated; hence the linearity will not be imposed on the conditional expectation of the MSN clusters in the SNMM. In this study, we aim to exploit this non-linearity to learn the intricate traits of the demonstration given to the robot more flexibly.

In terms of SNMR, we propose two different approaches. 1) Expectation-based SNMR (SNMR-EX), and 2) Maximum probability SNMR (SNMR-Max). In the expectation-based SNMR, we follow the same pattern of calculations as seen in GMR and expand the underlying math to derive this model. However, in SNMR-Max, we are generating the conditional trajectory, which results from a weighted summation of the maximum probability of each MSN cluster. Due to the Gaussian symmetricity in GMR, a kernel's conditional expectation is identical to that cluster's maximum conditional probability. Therefore the expectation-based GMR and maximum probability GMR would result in the same outcome.

4.2.1 SNMR-Ex

The expected value of a simple one-dimensional PDF is calculated as seen in equation (4.24).

$$E(X) = \int_{-\infty}^{+\infty} x f(x) dx \tag{4.24}$$

In multidimensional cases where we have deterministic and probabilistic dimensions, the computation of the conditional expected value will be in the form of equation (4.25). It should be noted that capital letters such as Y address the random variable, and the small notation of the same character y indicates the value of the random variable.

$$E(Y_2|Y_1) = \int_{-\infty}^{+\infty} y_2 f(y_2|y_1) dy_2 \tag{4.25}$$

This study conducts this computation numerically by calculating the integral in equation (4.25) for all the recorded values in deterministic dimensions. It should be noted that the numerical computation is discrete and in the form of equation (4.26), which means, that the higher the precision of the dimensions, the higher the accuracy of the calculated conditional expectation, and the higher the time-consumption of the process.

$$E(Y_2|Y_1) = \sum_{i=min(Y_1)}^{max(Y_1)} y_2 f(y_2|y_{1_i}) \tag{4.26}$$

By calculating the conditional expected value of each MSN kernel in the SNMM, we can proceed with computing the regression model of the SNMM using SNMR by following a modified version of the equation (4.22) for SNMR, as seen in equation (4.27). We denote the resulting multidimensional trajectory as $SNMR_{EX}(\boldsymbol{Y_2})$.

$$SNMR_{EX}(\boldsymbol{Y_2}) = E(\boldsymbol{Y}_2|\boldsymbol{Y}_1 = \boldsymbol{y}_1) = \sum_{i=1}^{G} \pi_i \boldsymbol{\mu}_i^\star \tag{4.27}$$

where $\boldsymbol{\mu}_i^\star$ is the result of equation (4.26) for each MSN cluster, and the weights for this summation can be computed using equation (4.28).

$$\pi_i = \frac{w_i MSN(\boldsymbol{y}_2|\boldsymbol{\mu}_{i_2}, \boldsymbol{\Sigma}_{i_{22}}, \boldsymbol{\Delta_{i_2}})}{\sum_{g=1}^{G} w_g N(\boldsymbol{y}_2|\boldsymbol{\mu}_{g_2}, \boldsymbol{\Sigma}_{g_{22}}, \boldsymbol{\Delta_{g_2}})} \tag{4.28}$$

where

$$\boldsymbol{\Delta_{i_2}} = \begin{bmatrix} \boldsymbol{\Delta_{i_{21}}} & \boldsymbol{\Delta_{i_{22}}} \end{bmatrix}$$

Fig. 4.1 shows an example of expected value trajectory for an MSN kernel, where $\boldsymbol{\mu} = \begin{bmatrix} -5 \\ -5 \end{bmatrix}$, $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $\boldsymbol{\Delta} = \begin{bmatrix} 1 & 5 \\ 5 & 1 \end{bmatrix}$. It can be observed that adding the skewness $\boldsymbol{\Delta}$ will sway the distribution away from the center point of the unskewed Gaussian counterpart with the same $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ values. Another important observation is the non-linear behavior of the expected value curve, which is the main attribute of the MSN kernel in terms of regression curve, enabled by the skewness of the PDF.

4.2.2 SNMR-Max

For SNMR-Max, we tend to explore the probabilistic dimensions for the set of values that acquires the maximum probability given the SNMM model. This means that given each set of values in the deterministic dimensions $\boldsymbol{Y}_1$, the whole $\boldsymbol{Y}_2$ space will be explored for each MSN cluster individually to see which set of $\boldsymbol{y}_2$ values, denoted as $\boldsymbol{y}_{2_{Max}}$, will gain the highest probability. This can be written mathematically as equation (4.29).

$$\boldsymbol{y}_{Max} = \boldsymbol{y}_2^{\star} \quad \textbf{s.t.} \quad f(\boldsymbol{y}_2^{\star}|\boldsymbol{y}_1)) = max(f(\boldsymbol{y}_2|\boldsymbol{y}_1)) \tag{4.29}$$

Now, using the same weighted summation pattern as seen in the previous subsection, we will derive another multidimensional trajectory based on the maximum probability of the MSN clusters in SNMM. We denote this trajectory as $SNMR_{Max}(\boldsymbol{Y_2})$, and calculate it as written in equation (4.30). It should be mentioned that the weights π_i are calculated based on equation (4.28).

$$SNMR_{Max}(\boldsymbol{Y_2}) = \sum_{i=1}^{G} \pi_i \boldsymbol{y}_{Max_i} \tag{4.30}$$

Fig. 4.1 shows the maximum probability of the same MSN kernel overlayed with the expected value curve. The main observation in this depiction is the difference between the two curves. Another important note is that since the trajectory values for max probability and expected values are numerically computed, the final values' precision is limited to the local range of $\boldsymbol{y}_2$.

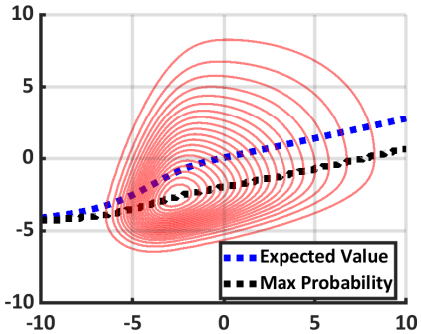


Figure 4.1: Maximum probability trajectory of a 2-dimensional MSN kernel

4.3 Results

Fig. 4.2(A) shows a 2-dimensional synthetic dataset to represent a set of demonstrations of a 2DoF robot's end-effector in the XY plane. The dataset is made in such a way as to represent a triangular no-fly zone during the task demonstration. The data is normalized to range between 0 and 1 in all dimensions. This figure also shows that in all demonstrations, there is a certain point close to $X = 0.5$ and $Y = 0$ that the task required the robot end-effector to meet. Fig. 4.2(B) shows a GMM fit over this dataset with two Gaussian clusters, and Fig. 4.2(C) shows an SNMM fit over the same dataset with two MSN clusters. We can see that by following the EM algorithm and the initialization pattern explained in section 4.1.2, the resulting skewed probabilistic model will conform to the dataset much more reliably, as it is also observed in the higher log-likelihood SNMM achieves.

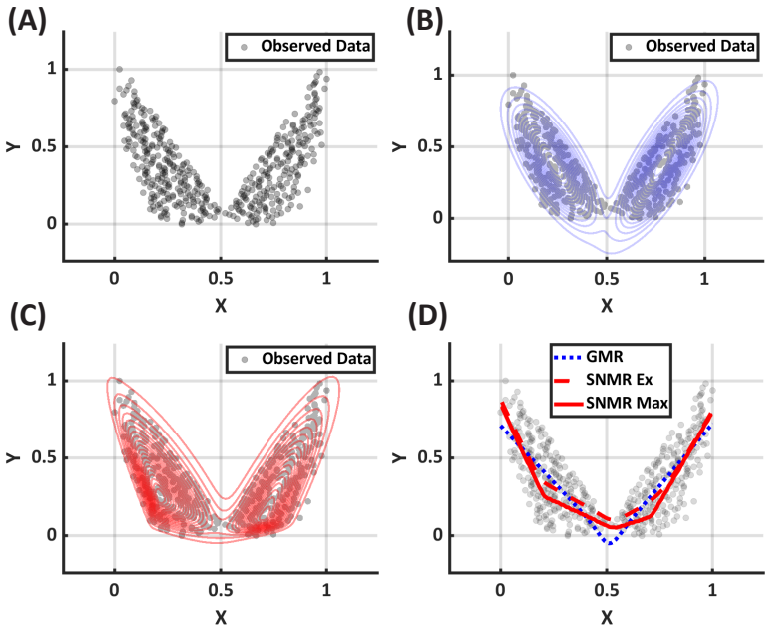


Figure 4.2: (A): Synthetic 2-dimensional Dataset, (B): GMM fit over the dataset, Log-Likelihood = 246.487 (C): SNMM fit over the same dataset, Log-Likelihood = 274.893, (D): GMR, SNMR_{EX}, and SNMR_{Max} comparison.

Using the mathematical representation of the regression models for GMR and SNMR, the resulting trajectories, given X as the deterministic dimension and Y

as the probabilistic dimension, can be calculated subsequently. These results are shown in Fig. 4.2(D). The lack of skewness in Gaussian kernels will hinder the GMM from capturing the unsymmetrical scatteredness of the observed data. As a result, the GMR trajectory fails to arrive at the narrow pathway of demonstrations in the middle of the X-axis. Also, the GMR trajectory starts and ends in positions that are away from the observed data points at both ends of the dataset. Conversely, both types of SNMR trajectories will refrain from undershooting and keep the end-effector within the expected range of motion. SNMR_{Max} will perform better than SNMR_{EX} as it keeps the trajectory closer to the narrow pathway. In addition, the results clearly show the non-linearity in the trajectories based on SNMR, which allows for the trajectories to start and end at points that are comparatively closer to the observed data points' ends.

In Fig. 4.3, we can see the same approach repeated for another synthetic data. Four clusters have been used for both SNMM and GMM models. In this dataset, there are two certain points that all demonstrations have passed through to finish the task, approximately by one-third and two-thirds of the deterministic X dimension. The GMM fit seen in Fig. 4.3(B) is shown to have failed in capturing the unsymmetrical and sharp regions of the observed dataset, and the Gaussian kernels had to expand towards the reach of the dataset to encapsulate the data points as much as GMM could allow. Contrary to that, SNMM has shown a great ability to encompass the dataset around the sharp edges of the dataset, seen in Fig. 4.3(C). This superiority of the SNMM modeling is then shown in Fig. 4.3(D) where the generated regression trajectories are compared. GMR trajectory will fail to cross the attention points in the dataset, while SNMR_{Max} will succeed in meeting those points. In addition, both SNMR_{EX} and SNMR_{Max} will start and end their resulting trajectories more towards the sharp edges of the outer bounds of the dataset, which attests to the SNMR's ability to learn the latent traits of the demonstrations more intuitively, while GMR fails to do so due to the inherent symmetricity of the Gaussian kernels.

Based on the performance of the SNMR_{Max} and how it can outperform SNMR_{EX} when it comes to capturing the hidden trajectory patterns in a given dataset, we can claim that the maximum probability approach is better in addressing the needed dynamics in a set of demonstrations to be learned by the robot. This would

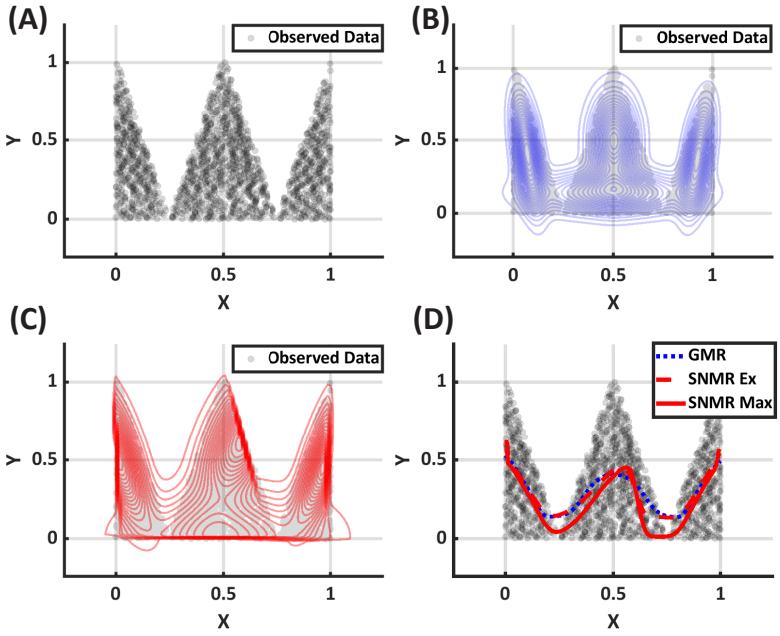


Figure 4.3: (A): Synthetic 2-dimensional Dataset, (B): GMM fit over the dataset, Log-Likelihood = 577.475 (C): SNMM fit over the same dataset, Log-Likelihood = 700.356, (D): GMR, SNMR_{EX}, and SNMR_{Max} comparison.

question the main claim of the GMR-based LfD studies in the literature in which the GMR is operating based on the conditional expected value rather than the maximum probability of each kernel, but due to the identicality of the mentioned properties, the maximum probability is not mentioned as the main computation goal.

Fig. 4.4 shows the comparison of 4-cluster SNMR_{Max} trajectory with GMR trajectory produced by GMMs of 4, 5, 6, 7, 8, and 9 Gaussian clusters. It can be seen from the results, that only after 8 Gaussian clusters, GMR trajectory can learn the specific points in the dataset to cross. However, adding more Gaussian clusters to GMM can cause the GMR trajectory to lose its smoothness due to high number of individual trajectories in the weighted summation in equation (4.22). This comparison shows that the SNMR_{Max} trajectory can preserve the smoothness while being able to learn the dynamics of the demonstrations.

Next, we test the SNMR approach on real robot data. The dataset shown in Fig. 4.5(A), is collected using a 2DoF Robot for 35 demonstrations on the target trajectory shown in purple. The task that is demonstrated is to go through the 7

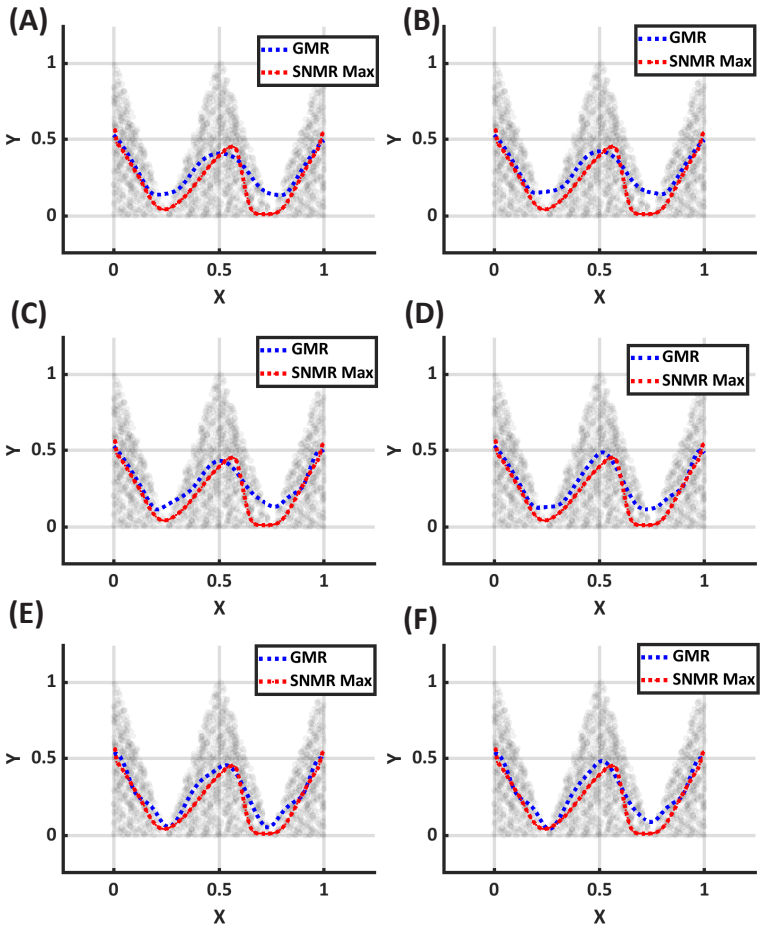


Figure 4.4: The comparison of a 4-cluster SNMR_{Max} trajectory with various number of clusters in GMR; (A): 4-cluster GMR, (B): 5-cluster GMR, (C): 6-cluster GMR, (D): 7-cluster GMR, (E): 8-cluster GMR, (F): 9-cluster GMR

waypoints seen in the trajectory in straight lines. The human error for following the waypoints accurately during the demonstrations can be observed by the visible scatteredness of the data points. These waypoints are marked as green stars on the other subfigures. In Figs. 4.5(B) through 4.5(E), we are comparing the GMR and SNMR_{Max}, with the same number of clusters in each model. In Fig. 4.5(B), we see that by using 3 clusters in mixture modeling, GRM trajectory only meets the first and the last waypoints, while the SNMR trajectory passes through first two and the last two waypoints. In Fig. 4.5(C), by adding one more cluster to the mixture models, we see that GMR trajectory manages to pass through the first three waypoints and the last, and comes close to the sixth waypoint, while SNMR trajectory meets all waypoints but the fifth. By adding another cluster in Fig. 4.5(D) GMR trajectory comes close to capturing the fifth waypoint but still cannot capture the highest waypoint. The SNMR trajectory will still capture 6 out of 7 waypoints, and gets closer to the fifth waypoint. Lastly, by using 6 clusters, GMR and SNMR trajectories will show a similar performance in learning the waypoints. From this example we can observe that the SNMR_{Max}, has significant efficiency in learning the intricacies of the observed demonstrations, unlike GMR trajectory which can only learn the complex demonstration dynamics by increasing the cluster count which jeoperdizes the trajectory smoothness.

We can also evaluate the superiority of SNMR over GMR trajectories on data limitation, and cluster count. Fig. 4.6(A) shows the comparison of GMR and SNMR trajectories on the same dataset seen in Fig. 4.5(A), with same number of demonstrations. In Fig. 4.6(B), the demonstration count in the dataset for SNMR has been reduced to 15 from 35, while GMR is learning the same dataset. In Fig. 4.6(C) the demonstration count is reduced further to 5 demonstrations only. Finally, in Fig. 4.6(D) we add to the number of GMM clusters to compare the GMR with higher cluster count and greater demonstration count with SNMR. As it is observed in all subfigures in Fig. 4.6, SNMR trajectories outperform GMR by showing a closer trend to the target trajectory.

In Fig. 4.7 another comparative set of results are shown. First row of this figure shows the dataset outlook with 5, 15, and 35 demonstrations. The following rows, show the trajectories generated by the GMR and SNMR_{Max}, based on mixture models of 3 and 4 clusters. The GMR and SNMR_{Max} trajectories are shown by blue

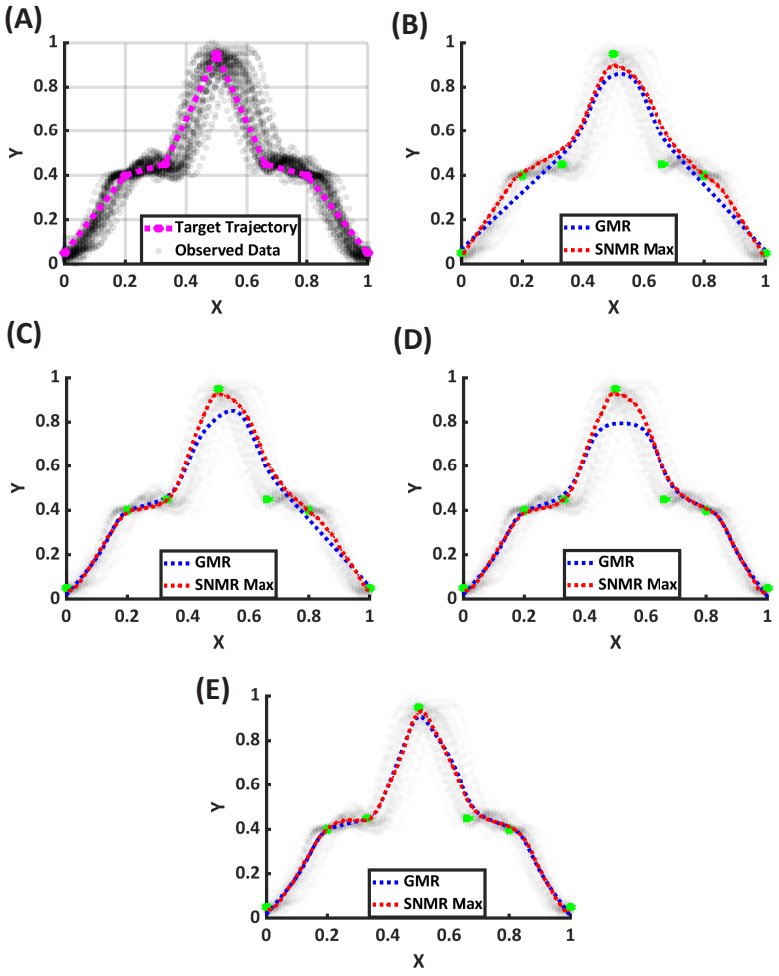


Figure 4.5: (A): Robot Data for 35 demonstrations on the target trajectory. (B): GMR and SNMR comparison (3 clusters) (C): GMR and SNMR comparison (4 clusters) (D): GMR and SNMR comparison (5 clusters) (E): GMR and SNMR comparison (6 clusters)

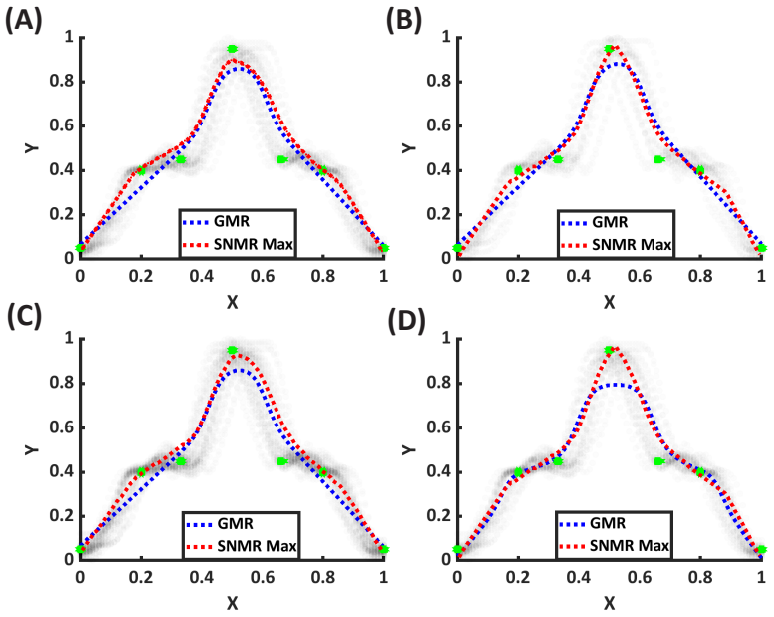


Figure 4.6: (A): GMR (3 clusters, 35 demos) vs. SNMR (3 clusters, 35 demos) (B): GMR (3 clusters, 35 demos) vs. SNMR (3 clusters, 15 demos) (C): GMR (3 clusters, 35 demos) vs. SNMR (3 clusters, 5 demos) (D): GMR (5 clusters, 35 demos) vs. SNMR (3 clusters, 5 demos)

and red dashed lines, respectively. In the case of 5 demonstrations and 3 clusters, GMR can only learn the first down-slide in the XY-plane and will not learn the rest of the motion, while SNMR will capture the begining and the end of the task as indicated by the rise of the red dashed line when X reaches 1. When the number of demonstrations increase from 5 to 15, we see that GMR is still unable to learn the task after $X = 0.2$, but SNMR trajectory has improved significantly by learning the mid-rise on the dataset when $X = 0.5$. For the case of 35-demonstration dataset, GMR learns the rise at the end of the X-axis but at the expense of losing the information about the beginning of the task where $X < 0.2$. This is while SNMR, has improved even more from the previous case, and the trajectory has skewed towards the more dense parts of the dataset, while preserving the knowledge of the mid-rise of the dataset. The comparison on this row of Fig. 4.7 shows the limitation of GMM/GMR approach when faced with low observation count or number of clusters, while SNMM/SNMR algorithm has higher learning capacity in these cases. The same learning superiority of SNMR_{Max} can be seen in the third and forth row

of Fig. 4.7. In the third row (4 cluster mixture models), we see that although both trajectories have learned the mid-rise, SNMR trajectory leans more towards the dense areas of the dataset, emphasizing the importance of those regions over others. This attention to density and flexibility to move the clusters accordingly, is the main trait of SNMR approach that performs more reliably in comparison to GMR. This ability of SNMR will initially helps the regression modeling to identify and learn the dynamics of the demonstrated tasks (such as mid-rise of the dataset in Fig. 4.7), and when such key details of the task has been identified, the density-awareness of SNMR will refine and optimize the trajectory based on the non-uniformity in the dataset scatteredness.

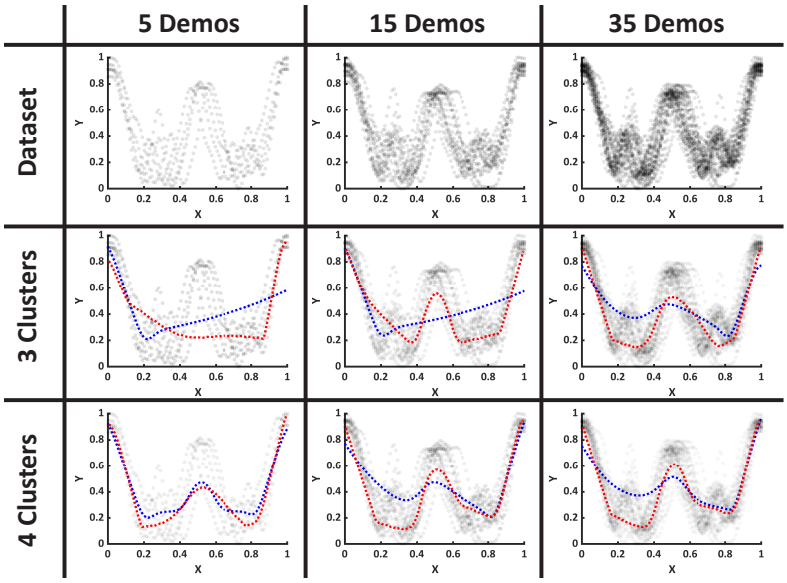


Figure 4.7: SNMR_{Max} and GMR performance comparison using different number of clusters on robot datasets with different number of demonstrations. GMR and SNMR_{Max} trajectories are depicted by blue and red dashed lines respectively.

4.4 Discussions

In this chapter, we have shown the prominence of skewness in relaxing the conventional assumption of symmetricality in observed datasets in robotics. This assumption will impose a certain rigidity when modeling the dataset which will reflect in the subsequent learning process from the demonstrations. We have

shown through multiple examples and comparisons, on synthetic and real robot data, that skewed normal kernels can perform significantly better in modeling the dataset using mixture modeling technique. Hence, the inherent dynamics of the demonstrations are captured more reliably using SNMM, which leads to a more efficient and accurate LfD scheme in robotic platforms.

Chapter 5

Context-Aware Learning from Suboptimal Kinesthetic Demonstrations

In this chapter, we aim to inject apriori information about the *quality* of human demonstrations, given the task requirements, into the LfD paradigms discussed previously. Recently, there have been efforts conducted to modify the maximum likelihood objective function using a curriculum-based LfD which avoids mode-averaging in the behavior space when a mixture of experts provide various solutions for conducting the task, embracing the diversity in task conduction [102]. As a separate line of research, some one-shot learning methods have been introduced, where a single demonstration is leveraged to teach a robot, thus minimizing the demonstrations needed by the human [40, 169]; however, such methods heavily rely on the high quality of a single demonstration. Some methods used generative adversarial learning to exploit sub-optimal demonstrations and learn self-generated solutions to augment/replace those conducted by the trainer [205]. Other methods utilize reinforcement learning or inverse reinforcement learning combined with ranking-based techniques to generate policies based on demonstration with various quality [9, 31, 172, 178, 193, 194].

In this chapter, we address the problem of sub-optimal demonstrations by proposing a new LfD approach that takes into account the quality of the demonstrations by formulating a context-aware EM algorithm, which integrates the concept

of demonstration quality into the mathematical derivation of probabilistic modeling. The proposed method incorporates the task context into the EM algorithm by solving the corresponding optimization problem when each data point is associated with a given weight, defining the quality, which can be variable throughout the demonstration or within demonstrations, highlighting the importance of the provided data within the task context. In this chapter, the proposed context-aware EM algorithm is embedded into the training of a GMM-GMR model to learn from various quality demonstrations and their data points. Through experimental validation in an obstacle avoidance task, we show that this approach can be used to find optimal trajectories even when demonstrations are limited to only a few observations. In an extreme example, we show that the proposed approach could rely on the input context and learn the policy only when a single observation is provided. The results show superior performance compared to the baseline GMM-GMR algorithm.

5.1 Preliminaries

In this section, the conventional GMM training process is summarized (more details can be found in [37, 146] and references therein). The math is reformulated in the next section to describe our approach. For a basic GMM algorithm, we consider a set of D kinesthetic demonstrations as $\mathcal{T} = \{(\tau_d)\}_{d=1}^{D}$, where each demonstration contains the trajectory as a sequence of states $\tau_d = (x_0, x_1, ..., x_{N_d})$ such as the cartesian pose of the robot's end-effector. In Gaussian Mixture Modeling, we would estimate the distribution of these data points (states) as a finite weighted summation of Gaussian functions. The parameters of the Gaussian functions are the prior π_k, mean μ_k, and covariance matrix Σ_k for $k \in \{1, ..., K\}$ clusters. The probability density function for the GMM is given as:

$$P(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \tag{5.1}$$

where the sum of the priors is 1 ($\sum_{k=1}^{K} \pi_k = 1$). The EM algorithm [50] is typically used in the literature to maximize the log-likelihood function:

$$l(\Theta) = \sum_{n=1}^{N} \log(P(x_n)) \tag{5.2}$$

where $\Theta = \{(\pi_1, \mu_1, \Sigma_1), ..., (\pi_K, \mu_K, \Sigma_K)\}$ is the set of all Gaussian parameters in the GMM, and N is the number of data points. The EM algorithm will iteratively maximize the log-likelihood in Eq. (5.2) by consecutively alternating between expectation and maximization steps (E-Step and M-Step, respectively) until convergence of the log-likelihood. In the E-Step, the conditional expected value of the GMM's latent variables will be calculated, and in the M-Step, the GMM parameters will be updated to maximize the log-likelihood based on the calculated expectation in the E-Step.

E-Step: Computes an $N \times K$ matrix Z, in which any element Z_{nk} is a latent variable representing the probability that an observation x_n comes from cluster k.

$$Z_{nk} = P(k|x_n) = \frac{P(k)P(x_n|k)}{p(x_n)} = \frac{\pi_k \mathcal{N}(x_n|\mu_k, \Sigma_k)}{\sum_{i=1}^{K} \pi_i \mathcal{N}(x_n|\mu_i, \Sigma_i)} \tag{5.3}$$

M-Step: Updates the Gaussians' parameters to maximize the log-likelihood using the latent variables from the E-step:

$$\pi_k = \left(\sum_{n=1}^{N} Z_{nk} \right) / N \tag{5.4}$$

$$\mu_k = \left(\sum_{n=1}^{N} Z_{nk} x_n \right) \left(\sum_{n=1}^{N} Z_{nk} \right)^{-1} \tag{5.5}$$

$$\Sigma_k = \left(\sum_{n=1}^{N} Z_{nk} (x_n - \mu_k)(x_n - \mu_k)^T \right) \left(\sum_{n=1}^{N} Z_{nk} \right)^{-1} \tag{5.6}$$

Considering that the states in our demonstration represent a combination of inputs u and outputs y such that $x = [u \ \ y]^T$, the posterior mean estimate $P(y|u)$ can be calculated (Gaussian Mixture Regression) [87]:

$$y = \sum_{k=1}^{K} \frac{P(k)P(u|k)}{\sum_{i=1}^{K} P(i)P(u|i)} \left(\mu_{y,k} + \Sigma_{uy,k} (\Sigma_{uu,k})^{-1} (u - \mu_{u,k}) \right) \tag{5.7}$$

$$\text{s.t. } \mu_k = \begin{bmatrix} \mu_{u,k} \\ \mu_{y,k} \end{bmatrix} \text{ and } \Sigma_k = \begin{bmatrix} \Sigma_{uu,k} & \Sigma_{uy,k} \\ \Sigma_{yu,k} & \Sigma_{yy,k} \end{bmatrix}$$

Therefore, the demonstration data can be used to create a probability density function as a Gaussian Mixture Model fitted to maximize the log-likelihood using the EM algorithm. Gaussian Mixture Regression can then calculate the posterior mean of a desired set of outputs y given observed inputs u.

5.2 Methods

Note that the log-likelihood function in (5.2) equally considers the probability of all datapoints in the demonstration dataset. In order to incorporate the quality of each datapoint (that can be defined by the quality or task context), we proposed a new log-likelihood function as given in (5.8).

$$l(\Theta, \alpha) = \sum_{n=1}^{N} \alpha_n \log(P(x_n)) \text{ s.t. } \alpha \in (0, 1] \tag{5.8}$$

The non-negative factor of $0 < \alpha_n \leq 1$ is associated with the *quality* of each datapoint based on the task context. For example, this quality factor can be prescribed by distance from obstacles or goal locations or based on speed and smoothness of demonstration. Thus, $\alpha_n = 0$ and $\alpha_n = 1$ indicate the low and high quality of the corresponding data point. The mathematical derivation of the new context-aware EM is re-designed in the next sub-section to analytically diminish the impact of low-quality data points by basing the design on a new log-likelihood function that incorporates data quality.

5.2.1 Context-Aware EM Algorithm

Given the new definition of the log-likelihood function (5.8), the context-aware EM algorithm mathematically guarantees the maximization of this function during each iteration.

For the E-Step (which updates the GMM's latent variables given π, μ, and Σ), since Z and α are independent variables and α is deterministic and not subject to change throughout the EM iterations, Z will have the same update policy as that of the conventional EM provided in (5.3). Hence, the expected value of the log-likelihood function known as the Q-function can be written as follows, given the latent variables:

$$Q(\Theta, Z, \alpha) = \sum_{n=1}^{N} \alpha_n \sum_{k=1}^{K} Z_{nk} \left(\log(\pi_k) - \frac{\log(|\Sigma_k|)}{2} - \frac{M_{nk}}{2} \right) \tag{5.9}$$

where M_{nk} is the squared Mahalanobis distance of x_n and μ_k given Σ_k which is $M_{nk} = (x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k)$.

The M-step equations maximize the Q-function in (5.9), derived from the new log-likelihood function in (5.8). The updates for each parameter are obtained by setting the Q-function derivatives (with respect to the corresponding parameter) equal to zero. For the mixing proportions π, utilizing the Lagrangian approach, we incorporate the constraint $\sum_{k=1}^{K} \pi_k = 1$ into the update equation to ensure the constraint remains satisfied after updating. Hence, the partial derivative is applied to $Q'(\Theta, Z, \alpha) = Q(\Theta, Z, \alpha) + \lambda(1 - \sum_{k=1}^{K} \pi_k)$, where λ is the Lagrange multiplier. By defining $r_{nk} = \alpha_n Z_{nk}$ as the responsibility of datapoint n being originated from cluster k given the quality value α_n, the π update equation is derived as follows:

$$\frac{\partial Q'(\Theta, Z, \alpha)}{\partial \pi_k} = \frac{\partial}{\partial \pi_k} \big[\sum_{n=1}^{N} \alpha_n \sum_{k=1}^{K} Z_{nk} \log(\pi_k) + \lambda(1 - \sum_{k=1}^{K} \pi_k) \big]$$

$$= \frac{1}{\pi_k} \sum_{n=1}^{N} r_{nk} - \lambda = 0 \Rightarrow \pi_k = \frac{\sum_{n=1}^{N} r_{nk}}{\lambda} \quad (*)$$

$$\frac{\partial Q'(\Theta, Z, \alpha)}{\partial \lambda} = \frac{\partial}{\partial \lambda} \big[\lambda(1 - \sum_{k=1}^{K} \pi_k) \big] \Rightarrow 1 - \sum_{k=1}^{K} \pi_k = 0$$

$$\Rightarrow 1 - \frac{1}{\lambda} \sum_{k=1}^{K} \sum_{n=1}^{N} r_{nk} = 0 \Rightarrow \lambda = \sum_{k=1}^{K} \sum_{n=1}^{N} r_{nk} \quad (**)$$

Substituting $(**)$ into $(*)$ $\Rightarrow$ $\boxed{\pi_k = \frac{\sum_{n=1}^{N} r_{nk}}{\sum_{k=1}^{K} \sum_{n=1}^{N} r_{nk}}}$

(5.10)

As mentioned, for updating the mean vectors and covariance matrices, the partial derivation of the Q-function with respect to μ and Σ should be calculated and solved for zero.

$$\frac{\partial Q(\Theta, Z, \alpha)}{\partial \mu_k} = \frac{\partial}{\partial \mu_k}\Big[\sum_{n=1}^{N} \alpha_n \sum_{k=1}^{K} -\frac{Z_{nk} M_{nk}}{2}\Big]$$
$$= \sum_{n=1}^{N} \alpha_n \sum_{k=1}^{K} Z_{nk} \frac{\partial}{\partial \mu_k}\Big[\frac{-(x_n - \mu_k)^T \Sigma_k^{-1}(x_n - \mu_k)}{2}\Big]$$
$$= \Sigma_k^{-1} \sum_{n=1}^{N} \alpha_n Z_{nk}(x_n - \mu_k) = \sum_{n=1}^{N} r_{nk} x_n - \sum_{n=1}^{N} r_{nk} \mu_k = 0$$
$$\Rightarrow \boxed{\mu_k = \frac{\sum_{n=1}^{N} r_{nk} x_n}{\sum_{n=1}^{N} r_{nk}}} \tag{5.11}$$

$$\frac{\partial Q(\Theta, Z, \alpha)}{\partial \Sigma_k} = \frac{\partial}{\partial \Sigma_k}\Bigg[\sum_{n=1}^{N} \alpha_n \sum_{k=1}^{K} \frac{-Z_{nk}}{2}\Big(\log(|\Sigma_k|) + M_{nk}\Big)\Bigg]$$
$$= \sum_{n=1}^{N} r_{nk}\Big(-\frac{1}{2}\Sigma_k^{-1} + \frac{1}{2}\Sigma_k^{-1}(x_n - \mu_k)(x_n - \mu_k)^T \Sigma_k^{-1}\Big)$$
$$= \Sigma_k\Big(\sum_{n=1}^{N} r_{nk}\Big) - \Big(\sum_{n=1}^{N} r_{nk}(x_n - \mu_k)(x_n - \mu_k)^T\Big) = 0$$
$$\Rightarrow \boxed{\Sigma_k = \frac{\sum_{n=1}^{N} r_{nk}(x_n - \mu_k)(x_n - \mu_k)^T}{\sum_{n=1}^{N} r_{nk}}} \tag{5.12}$$

Hence, (5.10) is the updated analytical solution for π_k replacing (5.4), (5.11) replaces the calculation of μ_k in (5.5), and (5.12) replaces the solution for Σ_k in (5.6). As can be seen, the new update policies for the context-aware EM, now take into account the quality of each data point throughout the course of optimization. Based on the new EM formulation, the resulting GMR generates a trajectory that more closely resembles the higher-quality data points from the demonstration dataset. As can be seen in the equations above, if "all" data points are considered equally significant (for $n = 1 : N$ we have $\alpha_n = 1$), then the analytical solutions for the M step would converge to the conventional M step. Thus, the proposed math suggests a new generalized form of EM that can incorporate the quality of data points that may represent the quality or task context.

5.2.2 Context-based α-functions

This work introduces an α-function to encode task-specific context within LfD. The α-function translates task features into weights for individual demonstrations or sub-demonstrations. For example, the function could modulate data point quality based on trajectory smoothness, task completion time, proximity to the target, or distance from obstacles. Note that some obstacles may only be introduced after the initial demonstration. The question here is how to learn from demonstrations in a way that incorporates these practical features through the design of α-functions. This allows adapting the learned model based on measures of demonstration quality and expected difficulty during task execution. The choice of α-functions depends on the task context and can be designed by the trainer. We categorize α-functions into two categories: (1) per-demonstration α-functions that act on the entire demonstration and can be seen as a 'reward' for whether a goal was achieved or condition was kept, and (2) per-datapoint α-functions which act on each datapoint and represent the quality of each datapoint irrespective of the overall demonstration. Different α-functions can be combined together to incorporate several objectives into a single quality measure (i.e., by multiplication of α-functions followed by a normalization and scaling).

Examples of per-demonstration α-functions include measurements that evaluate the entire trajectory such as speed, smoothness, and whether the trajectory contains any collisions or reaches the goal location. These are in the form $\alpha_d = \alpha(\tau_d)$, and all datapoints in the demonstration are assigned the same α_d value. Examples of per datapoint α-functions include measurements that evaluate the distance of datapoints from an obstacle or target location. These are in the form $\alpha_n = \alpha(x_n)$, and each datapoint has its own associated value.

5.3 Evaluation

We examine the proposed method and different α-functions through a comparative experimental study using the Franka robot. We conduct a real-world obstacle avoidance task and collect demonstrations via kinesthetically teaching the robot. For the experimental setup, there are three fixed cylindrical obstacles, a start location, and a goal location. The trainer navigates the end-effector of the

robot from the start location to the goal location while avoiding the obstacles. The demonstration dataset is comprised of 20 demonstrations with varying speeds, collision cases (13 without collision and 7 with at least one collision), and smoothness, as shown in Fig. 5.3(A). The demonstrations were designed to cover diverse cases that could occur during a kinesthetic training of the robot. They mimic real-world situations, including suboptimal demonstrations due to human error or differences between human teachers. In order to prescribe α for the proposed context-aware EM algorithm, we implemented two different strategies, explained below as "Per-demonstration α-function" and "Per-datapoint α-function". The former would assign α in a way that it is different between demonstrations but similar within each demonstration. The latter case would assign α based on the coordinates in the workspace and relative distance to targets and obstacles.

Per-demonstration α-function: Two components were considered in the Per-demonstration α-function to represent the quality of the demonstration according to the number of collisions and overall smoothness of the demonstrated trajectory. The component for the collisions is given as $\alpha_{\tau_d,c} = \frac{1}{1+N_c}$, where N_c is the number of collisions with obstacles for a given demonstration. For the overall smoothness, the component is considered to be the average of the inverse summation of the differences between end-effector velocity in the XY-plane at each datapoint in the demonstration: $\alpha_{\tau_d,v} = \frac{1}{N_d-1}\big(\sum_{n=1}^{N_d-1} |(\dot{x}_n - \dot{x}_{n+1})|\big)^{-1}$. Using the mentioned components, we define each demonstration quality as the normalized product of $\alpha_{\tau_d,v}$ and $\alpha_{\tau_d,c}$ as seen below:

$$\alpha_{\tau_d,d} = \frac{\alpha_{\tau_d,c}\alpha_{\tau_d,v}}{\max(\alpha_{\tau_i,c}\alpha_{\tau_i,v})} \text{ s.t. } i \in \{1,2,\ldots,D\} \tag{5.13}$$

The calculated per-demonstration α uniformly modulates the quality factor of all datapoints in the corresponding given demonstration, and this factor can vary between demonstrations.

Per-datapoint α-function: The previous method did not account for varying data quality within a demonstration. Even if some datapoints in a low-quality demonstration were favorable for learning, they suffered the same quality reduction as the rest. This may require collecting more demonstrations according the trainer's experience. However, the per-datapoint approach, explained in the following, can utilize partial demonstrations and high-quality "sub-demonstrations." This

is supported by the (5.10)-(5.12) of the proposed context-aware EM algorithm which allows for the modulation of the quality factor for each and every datapoint. To assess the functionality, in this experimental example, we created an α-map on the robot task space given the context of the task (e.g., avoid obstacles and convergence), and each point of the demonstration set can be given an individual quality factor.

For this purpose, five different components for α-function are considered here in this experimental validation to define the quality of datapoints in any demonstration.

Among the five, three components represent the Euclidean distance of the datapoint from one of the three obstacles. (B.4) shows the equation for the corresponding component given the i^{th} obstacle location and radius. Likewise, two components represent the negative of the Euclidean distance from the start and goal locations (attracting regions). (B.5) shows the equation for the corresponding component given the i^{th} attracting location and the radius.

$$\alpha_{n,o} = \max(10^{-6}, ||x_n - o_i||^2 - r_i) \text{ s.t. } i \in \{1, 2, 3\} \tag{5.14}$$

$$\alpha_{n,a} = -\max(10^{-6}, ||x_n - a_i||^2) \text{ s.t. } i \in \{1, 2\} \tag{5.15}$$

$$\alpha_{n,f} = \frac{\alpha_{n,o}\alpha_{n,a}}{\max(\alpha_{i,o}\alpha_{i,a})} \text{ s.t. } i \in \{1, 2, \ldots, N\} \tag{5.16}$$

The overall field of α-map in (5.16) is created from normalizing the element-wise product of all five mentioned α-maps in (B.4) and (B.5). The resulting α-map is shown in Fig. 5.1(F) which is the normalized element-wise product of the other five α-maps shown in Fig. 5.1(A)-(E). Thus, in this experimental protocol, by utilizing the aforementioned α-field, we can associate quality factors to each datapoint based on the obstacle repulsions and target attractions. Different maps can be generated based on different features defining the quality and context.

Fig. 5.2(A) represents the per-demonstration quality (α) values given the dataset in Fig. 5.3(A). It can be observed that the α of the demonstrations have been reduced from unity based on the demonstration quality of the task. Additionally, Fig. 5.2(B) shows the per-point α values for the same dataset. It can be observed that the α varies within demonstrations indicating that the quality of the points can be subject to local task contexts/changes; accounting for these enable increased

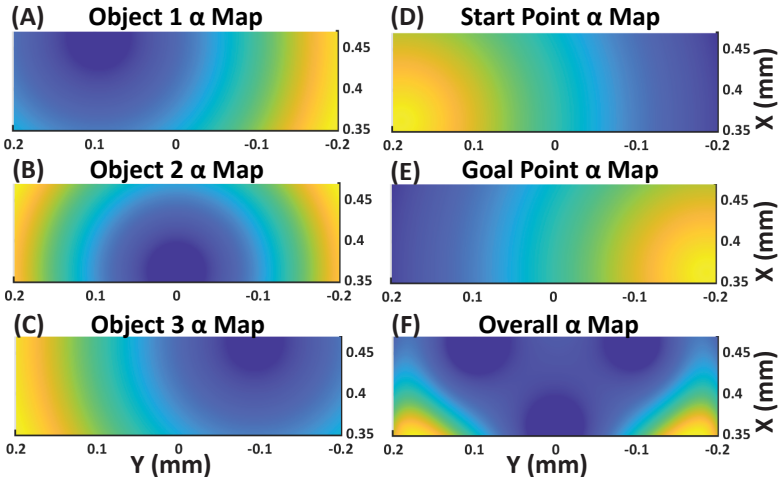


Figure 5.1: The α maps resulting from: (A): the distance from the first object on the left. (B): the distance from the second object in the middle. (C): the distance from the third object on the right. (D): the starting point of the task.(E): the goal of the task. (F): the normalized element-wise product of all α-maps.

dynamic variances for robot learning.

Randomizing obstacle locations: To further challenge and test the algorithm's ability to generalize to trajectories containing obstacle locations not present in the demonstration dataset, we randomized the obstacle locations, which resulted in new α-functions. Therefore, we evaluate whether, by only knowing the context of the task (start, new obstacle, and goal locations), the proposed algorithm can fit a probabilistic model that generates a trajectory learned from the demonstration but not shown within the original demonstration dataset, eliminating the need for data re-collection, and boosting generalizability of the robot learning. In other words, if the robot can still learn from the demonstrations provided while adapting to the new context of the task.

5.4 Results

The total demonstration dataset consisted of optimal and suboptimal trajectories based on collision, speed, and smoothness. Here, the per-demonstration and per-datapoint α-function approaches are compared to conventional GMR as a baseline.

In Fig. 5.3(B), we compare GMR to the per-demonstration α-function approach.

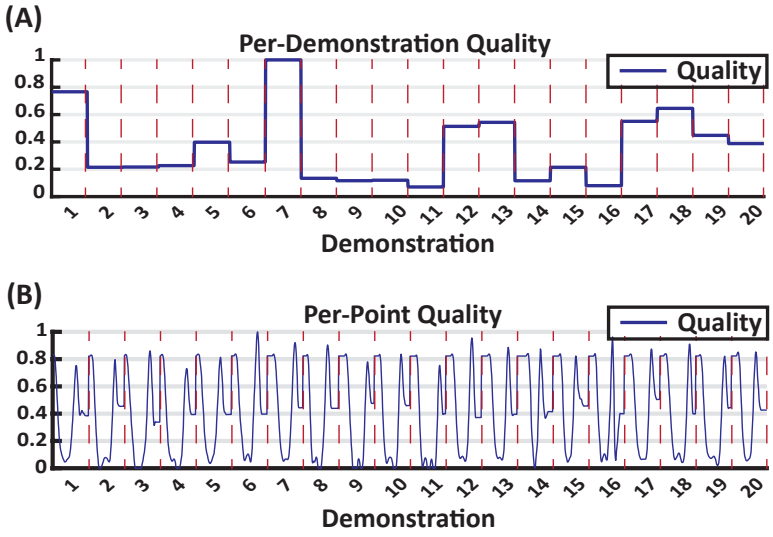


Figure 5.2: (A): The plot of per-demonstration α function for each demonstration in the dataset. (B): Visualization of the per-point α function for each point within each demonstration.

The dataset used in this sub-experiment had eight trajectories (four with collisions, four without). Since it contains many suboptimal trajectories, baseline GMR also produced a colliding trajectory (with the first obstacle). However, the per-demonstration approach avoided all collisions despite many provided demonstrated trajectories containing them. This is because trajectories with collisions get lower weights in the context-aware EM algorithm, decreasing their contribution to the probabilistic model optimization. In contrast, baseline GMR considers all demonstrations equally when fitting the model, thus reflecting aspects of the suboptimal policy of the trainer.

In Fig. 5.3(C), we compare the baseline GMR with the per-demonstration and per-datapoint α-function approaches. The demonstration dataset had five trajectories (four with collisions, one without) reflecting overall low quality. The baseline GMR produced a colliding trajectory. The per-demonstration approach led to a trajectory with a very close distance to the left obstacle. The per-datapoint approach generated a trajectory that avoided all obstacles. This experiment represents an overall poor demonstration quality and implications of the per-demonstration method. The per-datapoint method differentiates locally between high and low-quality data points, maintaining contributions from high-quality

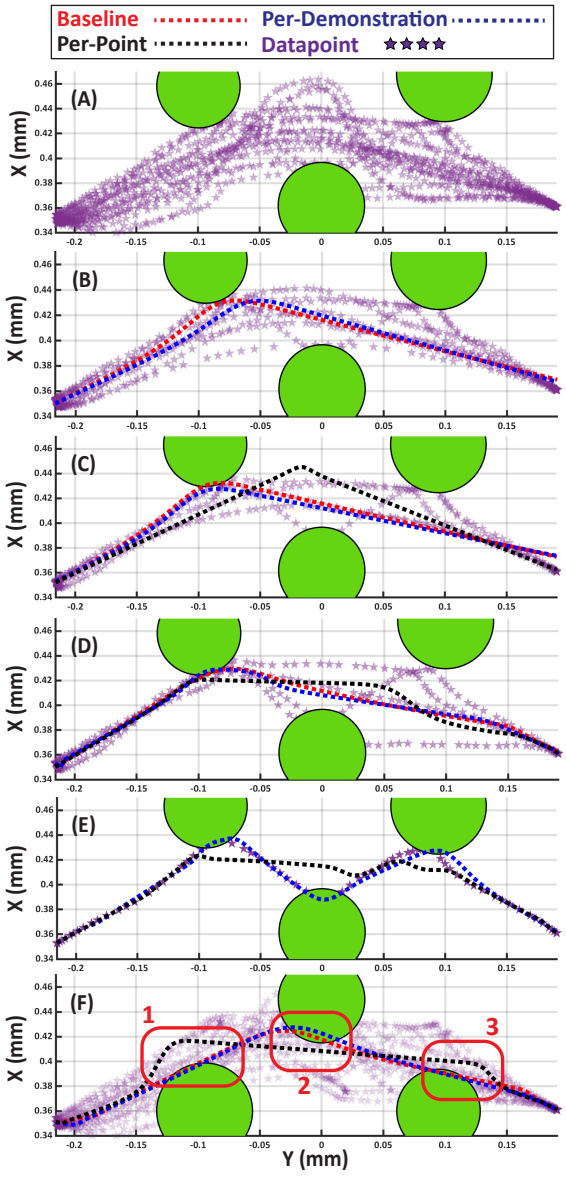


Figure 5.3: **(A):** Example set of 20 demonstrations. The green circles indicate the obstacles in the task space. **(B):** Per-demonstration approach. The dataset consists of four with collision cases and four without. **(C):** Per-demonstration approach (blue), Per-datapoint approach (black), and baseline GMR (red). The dataset consists of four trajectories with collision cases and one without. **(D):** Few-Shot Learning. The dataset consists of four colliding trajectories. **(E):** One-Shot Learning. The dataset contains only one demonstration, which collides with all three obstacles. **(F):** Randomized Obstacles. The obstacle locations are moved to new locations. The per-datapoint approach generates trajectories that avoid new obstacles in the new task setting.

points while reducing the effect of low-quality points on the generated policy.

In Fig. 5.3(D), we reduced the provided demonstrations to only four trajectories, all containing collisions. This is an extreme case when the trainer provided a poor quality of demonstration. The baseline GMR and per-demonstration α-function approach failed to generate collision-free trajectories. Since all provided demonstrations had collisions, the per-demonstration approach down-weighted them all similarly, resulting in a suboptimal trajectory similar to that generated by the baseline GMR. However, the per-datapoint approach generated a collision-free trajectory despite the collision-only dataset. This is because it assigns quality factor to each datapoint using the α-map, enabling local context-based quality assessment and improving policy generation in the presence of low-quality demonstration.

In Fig. 5.3(E), only one demonstration was provided for training, in which the robot collided with all three obstacles, simulating a very complex task for the robot to infer. Since the baseline GMR and per-demonstration approach are identical with one demonstration, only the per-demonstration trajectory is shown. This example illustrates the clear effect of the task space α-map on learning. The per-demonstration approach "learned" to collide with obstacles as that was the only dataset provided. In contrast, the per-point approach utilizes the task context information to consider datapoints with higher α values (no collision) and significantly lessen the contribution of data points with lower α values (collision points), resulting in a policy that safely maneuvers the robot in between obstacles. The approach learns the overall behavior from the provided demonstration and augments that knowledge with information provided by the α-function encoding task context.

In Fig. 5.3(F), the obstacle locations were moved to an unseen configuration after demonstrations were provided, inducing a mismatch between the original and new task contexts. This caused most demonstration trajectories to collide with the new obstacle locations. As a result, utilizing conventional models would require re-collecting demonstrations in the new setting to encode the new policy; otherwise, the generated trajectories would collide with the new obstacles. In particular, the baseline EM and per-demonstration approaches collided with all new obstacles, failing to adapt to the changed context when no new demonstration was provided. However, the per-datapoint approach augmented the trained policy with

the new task context, avoiding all collisions. Thus, by encoding α-map values, the proposed context-aware EM could generate policies that are flexible with the induced changes in the task context without the need for re-demonstration. *Therefore, the proposed context-aware EM algorithm can learn a policy (i.e., navigate start to goal while avoiding obstacles) from the original demonstrations and adapt the learned policy based on the new context (using the alpha-map) without needing new demonstrations, thereby augmenting scalability, increasing flexibility, and relaxing the need for extensive recalibration when task parameters change.*

5.5 Discussions

In this chapter, we have introduced a context-aware EM algorithm for kinesthetic robot learning from demonstrations that considers local data quality and task context. Kinesthetic training can result in suboptimal demonstrations containing jerky motions, collisions, and trajectory deviations due to dynamical discrepancies between humans and robots, uncompensated impedance, stability concerns with low impedance control, delays, and other factors. These imperfections present a key challenge because they can adversely affect the quality of the demonstrations and thus the learned policy if all demonstration segments are valued equally during the optimization process. To address this, we have derived a mathematical formulation for a context-aware EM algorithm that assigns a localized quality factor to each data point based on the task context and demonstration quality. This quality factor tunes the data point's contribution when fitting the parameters of a model using a new formulation of the EM approach. We have validated the advantages of our algorithm by providing a set of suboptimal kinesthetic demonstrations containing collisions and trajectory errors. In contrast to traditional methods that fail on such demonstrations, our context-aware approach can generate smooth, optimal trajectories by selectively down-weighing low-quality portions of the demonstration data. This highlights significant improvements over conventional learning from demonstration solutions that treat all segments equivalently. Our method can potentially enhance robot learning in manufacturing and healthcare applications where a trainer can provide imperfect demonstrations kinesthetically or when a group of trainers with different expertise levels provides various-quality

demonstrations. It should also be added that the proposed context-aware learning approach can allow for the adaptation of optimized policies from a small number of suboptimal demonstrations when task conditions change, increasing the data-efficiency of learning.

Part III

Learning Human *Intent*

Chapter 6

Cost Function Estimation Using Inverse Reinforcement Learning with Minimal Observations

We present an iterative inverse reinforcement learning algorithm to infer optimal cost functions in continuous spaces. Based on a popular maximum entropy criteria, our approach iteratively finds a weight improvement step and proposes a method to find an appropriate step size that ensures learned cost function features remain similar to the demonstrated trajectory features. In contrast to similar approaches, our algorithm can individually tune the effectiveness of each observation for the partition function based on the current estimate of the cost function parameters, guiding the algorithm towards better estimates in the following iterations. In addition, it does not need a large sample set, enabling faster learning. We generate sample trajectories by solving an optimal control problem instead of random sampling, leading to more informative trajectories. The performance of our method is compared to two state of the art algorithms to demonstrate its benefits in several simulated environments.

Prior work have tried to solve the IRL problem by either using a maximum margin formulation [149], or considering a task related probability distribution to quantify the occurrence of the expert demonstration with highest probability given the cost function [206] using maximum entropy principle [75]. Initially, in [206], for small discrete domains dynamic programming was used to estimate the distribution

accurately. Further in [30], this approach was modified to optimize the embedded maximum likelihood estimation given the relative entropy of the samples. Since then, efforts have been made to generalize these method to higher dimensions and continuous domains using various techniques for estimating the distribution such as approximation [100] and various sampling methods [6, 58, 80].

Although these approaches have shown great potential for learning the cost function, they still suffer from various shortcomings. For example, local sampling around expert demonstration, as used in [6, 80, 82], might fail to contain informative and diverse non-optimal demonstrations to represent task space as a whole. Furthermore, requiring initial sample set can be challenging as well. It either necessitates to use specialized sampling techniques, or to use simpler uniform sampling while risking inaccuracy. Aside from the need for initial samples, most approaches require to use all the samples iteratively for accurate estimation of the trajectory space, which can be computationally demanding. In addition, a majority of IRL methods tend to control convergence based on probabilistic features such as maximum or relative entropy. Hence, the cost function will not be estimated by how optimal it can generate a new trajectory close to the expert demonstration. Some of these issues can be mitigated with IOC approaches, where gradient-based methods are used to solve the problem, hence foregoing the need of trajectory sampling. However, IOC approaches tend to be computationally expensive as they require to solve a bi-level optimization problem which includes an OC problem in its inner loop [125].

Another challenge in regards to approximating the trajectory space in the sampling-based approaches, is to incorporate *how* non-optimal is a sampled trajectory within the trajectory set. To capture this measure of non-optimality, in [80], the trajectories are sampled in the local vicinity of the optimal trajectory, considering their non-optimality as uniform within that vicinity. Closer to our approach, in [30] and [58], non-optimal samples are individually rescaled. While these efforts improve the trajectory space approximations, they do not rely on the policy parameters the IRL is optimizing, but rather on the system dynamics uncertainties and state-action transition probability distributions.

To address these challenges, this paper proposes an iterative sampling-based IRL algorithm based on maximum entropy principles for continuous task spaces.

The approach seeks to iteratively find improvements in the cost estimate while ensuring it would generate trajectories close to the demonstration. We reformulate the probability distribution of the demonstrations by individually tuning the effectiveness of each sample based on the the previous iteration's estimate of the cost function parameters.

Moreover, sampling is conducted using an OC solver, where at each iteration we generate new trajectories based on the current cost function estimation. This generates meaningful samples consistent with the the intended use of the cost function and relaxes the need for an ad-hoc sampling strategy. We demonstrate empirically that our approach leads to significantly faster convergence and higher quality estimates when compared to state of the art methods.

6.1 Preliminaries

Our method relies on the probabilistic approach of [206], which writes the probability of the expert's demonstration (referred to from now on as the optimal trajectory)

$$P(\tau^\star|\bar{\boldsymbol{\tau}}) = \frac{1}{Z}\exp(-C(\tau^\star)) \tag{6.1}$$

with *partition function*

$$Z = \exp(-C(\tau^\star)) + \sum_{\tau_i \in \bar{\boldsymbol{\tau}}} \exp\left(-C(\tau_i)\right) \tag{6.2}$$

where $\tau^\star$ is the optimal trajectory. Observed trajectories are denoted as $\tau = \{\boldsymbol{x}_0, \boldsymbol{u}_0, \boldsymbol{x}_1, \boldsymbol{u}_1, \ldots, \boldsymbol{x}_T\}$ where $\boldsymbol{x}_t$ and $\boldsymbol{u}_t$ are state and control inputs at time t. T is the length of the trajectory. $C(\tau)$ is the cost of trajectory τ, and $\bar{\boldsymbol{\tau}} = \{\tau_1, \tau_2, \ldots, \tau_K\}$ is the set of all observed trajectories, excluding $\tau^\star$. With this formulation, trajectories with higher cost are exponentially less likely to occur and vice versa.

The optimal expert demonstration $\tau^\star$ should then have the highest probability compared to all other possible trajectories. Hence, IRL approaches based on (6.1) aim to maximize the probability of the optimal trajectory, under any set of trajectory observations $\bar{\boldsymbol{\tau}}$

$$C^\star = \arg\min_C -\log(P(\tau^\star|\bar{\boldsymbol{\tau}})) \tag{6.3}$$

A key challenge in such methods is estimating the partition function, i.e. generating a representative set of non-optimal trajectories that will help discriminate cost functions when solving (6.3). In the following we propose a method for generating trajectories used in the partition function Z that allows the IRL algorithm to have a reasonable understanding of how much each observed trajectory should be incorporated into the partition function. This approach allows for a better representation of the partition function, and even relaxes the need to have all observed trajectories used for the trajectory space representation, hence reducing computational cost.

6.2 Methods

As mentioned above, we are interested in finding cost functions represented by a linear combination of explicit features. This representation is more constrained than IRL approaches based on neural networks but has two benefits: it allows a seamless integration of the learned cost with optimal control solvers for model-predictive control (MPC) and provides interpretable cost functions, which is important for human movement understanding or human-robot interaction tasks. Further, such cost functions are capable of generating diverse movements and achieving many tasks [117]. We hence search for costs represented as a weighted sum of features

$$C(\tau, \boldsymbol{w}) = \boldsymbol{w}^T \Phi(\tau) \tag{6.4}$$

$$\Phi(\tau) = \sum_{t=0}^{T-1} \phi_s(\boldsymbol{x}_t, \boldsymbol{u}_t)\Delta t + \phi_T(\boldsymbol{x}_T) \tag{6.5}$$

where $\Phi(\tau)$ is a vector of time integrated features of τ. ϕ_s and ϕ_T are vectors of stage and terminal cost features, respectively. For brevity, the features of the i^{th} trajectory is denoted Φ_i, and $\Phi^\star$ is the optimal trajectory's feature vector.

6.2.1 Problem Statement

Given the linearity assumption of the cost function, the IRL problem becomes

$$\boldsymbol{w}^\star = \arg\min_{\boldsymbol{w}} -\log(P(\tau^\star|\boldsymbol{w}, \bar{\boldsymbol{\tau}})) \tag{6.6}$$

$$\text{s.t.} \quad \boldsymbol{w} \geq \mathbf{0} \tag{6.7}$$

This problem aims to find weights that maximize the probability of the optimal trajectory for a set of observed trajectories. As a consequence, the recovered weights should then produce the same optimal trajectory when used in an OC solver. However, maximizing the probability of the optimal trajectory does not guarantee this. Indeed, the partition function cannot be computed for all possible trajectories and will be computed from a finite, often small, set of non optimal trajectories. Therefore, there exist trajectories not included in the set that can have higher probability than the demonstration. This issue is typical of all IRL approaches based on (6.1). In this paper, we mitigate this issue by proposing a way to minimize the difference between the estimated costs and the features of the optimal ($\Phi^\star$) and OC trajectory (denoted henceforth as $\tilde{\Phi}$).

We propose to solve the problem iteratively by taking small steps Δw to improve the cost candidate and propose a procedure to accept such a step at each iteration to ensure it also improves the trajectory that will be generated by the OC solver.

6.2.2 Computing a step direction

We can explicit the minimizing of (6.6) as

$$\boldsymbol{w}^\star = \arg\min_{\boldsymbol{w}} -\log \frac{1}{1 + \sum_{\tau \in \bar{\boldsymbol{\tau}}} e^{-\boldsymbol{w}^T(\Phi_i - \Phi^\star)}} \tag{6.8}$$

Since we seek to find an improvement step, we write the weights at iteration $n+1$ as $\boldsymbol{w}_{n+1} = \boldsymbol{w}_n + \Delta \boldsymbol{w}_n$ which leads to the following problem

$$\Delta \boldsymbol{w} = \arg\min_{\Delta \boldsymbol{w}} -\log \frac{1}{1 + \sum_{\tau \in \bar{\boldsymbol{\tau}}} \gamma_i e^{-\Delta \boldsymbol{w}_t^T(\Phi_i - \Phi^\star)}} \tag{6.9}$$

$$\text{s.t.} \quad \Delta \boldsymbol{w}_t > -\boldsymbol{w}_t \tag{6.10}$$

where

$$\gamma_i = e^{-\boldsymbol{w}_t^T(\Phi_i - \Phi^\star)} \tag{6.11}$$

can be understood as a weight on each trajectory to give more importance to non-optimal trajectories that are more probable. This means that trajectories that are farther from optimality will weigh less on the 'changes' in weights and vice versa. The benefits of assigning a weight to each trajectory when computing the partition function has already been discussed in [30, 58], where an importance sampling argument was used to specifically compute these weights. In our approach, these weights, which are dependent on both trajectory features and cost weights, appear naturally as we search for an improvement direction Δw.

6.2.3 Step Acceptance Method

Once a step candidate $\Delta \boldsymbol{w}$ has been computed by solving problem (7.3), we need to determine the actual step size to ensure that the change of weight will help find a cost function that will lead to trajectories closer to the optimal one. Reminiscent to line search methods in optimization, we generate a trajectory with an OC solver using $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha \Delta \boldsymbol{w}_t$ where $\alpha = 1$, and check if this trajectory is closer to the optimal trajectory in either feature space or cost using the following two merit functions

$$m_1(\boldsymbol{w}_{t+1}) = m_1(\boldsymbol{w}_t + \alpha \Delta \boldsymbol{w}) = \frac{1}{2}(\boldsymbol{w}^T \Phi^\star - \boldsymbol{w}^T \tilde{\Phi})^2 \tag{6.12}$$

$$m_2(\boldsymbol{w}_{t+1}) = m_2(\boldsymbol{w}_t + \alpha \Delta \boldsymbol{w}) = ||\Phi^\star - \tilde{\Phi}||_2 \tag{6.13}$$

We will accept a step if it leads to an improvement on either of these merit functions. For m_1, as we can compute its derivative with respect to Δw, we use Wolfe conditions to measure acceptable improvement. For m_2 we simply require that it is lower than its value at the previous iteration. We then add the OC solver's resulting trajectory with the new weight estimate to $\bar{\boldsymbol{\tau}}$ and iterate again. Otherwise, we decrease α and try again. In our algorithm, if we did not find an acceptable step-size after 10 iterations, the algorithm terminates.

The procedure above also explains our approach to sampling, as we add trajectories computed with an OC solver in our dataset. Initially, the algorithm has bad weight estimation and it results in trajectory samplings that are far from optimal. As iterations proceed, generated trajectories become closer to the optimal one. This way, our trajectory sample set will contain both "good" and "bad" non-optimal

trajectories.

6.2.4 Sub-sampling Trajectories

We propose to augment our dataset by adding trajectories of reduced length from the current trajectories. For each full length trajectory, with T timestamps per trajectory, in the partition function, we create a sub trajectory starting from $t = d$ and ending at $t = T$. We generate several subset of trajectories by using different values of $d \in D$. Our minimization is then adapted to

$$\Delta \boldsymbol{w} = \arg\min_{\Delta \boldsymbol{w}} \sum_{d \in D} -\theta_d \log \frac{1}{1 + \sum_{\tau \in \bar{\boldsymbol{\tau}}} \gamma_i e^{-\Delta \boldsymbol{w}^T (\Phi_{i,d} - \Phi^\star_d)}} \tag{6.14}$$

where $\Phi_{i,d}$ and $\Phi^\star_d$ denote features truncated from $t = d$ and D is the set of timestamps equidistantly chosen between 0 and T. For example, for $N = 3$, in addition to the original set we will have two more subsets, containing trajectories with a one-third and two-thirds of their original lengths, i.e., if $T = 100$, then $D = \{0, 33, 66\}$. To give more importance to longer trajectories, we scale each subset's likelihood by a coefficient $\theta_d = (T - d + 1)/(T + 1)$. This can be useful for lower dimensional feature spaces, and lower length trajectories where the trajectory cost can be less sensitive to weight changes.

6.2.5 Regularization

We also add L1 and L2 regularization to form an *elastic net* regularization, to lower weights magnitude and favor sparsity. The induced sparsity in $\Delta \boldsymbol{w}$ causes the most important features to change weight instead of all at the same time. We thus minimize

$$\begin{aligned} \Delta \boldsymbol{w} = \arg\min_{\Delta \boldsymbol{w}} \sum_{d \in D} -\theta_d \log \frac{1}{1 + \sum_{\tau \in \bar{\boldsymbol{\tau}}} \gamma_i e^{-\Delta \boldsymbol{w}^T \bar{\Phi}_i d}} \\ + \lambda |\Delta \boldsymbol{w}| + \frac{\beta}{2} ||\Delta \boldsymbol{w}||_2^2 \end{aligned} \tag{6.15}$$

Regularization is useful when feature space is small, and learning process can be challenging especially when opposing features are present as seen in the experiments below.

6.2.6 Moving Window

Since our approach restricts the steps taken to improve the weights and further scales the importance of each trajectory in the partition function individually, it is not necessary to keep a large history of trajectories to compute the partition function. Indeed, trajectories far from the optimal one will have a negligible weight. Therefore we will only use the last L trajectories at each iteration, discarding older trajectories.

This approach is in contrast to several maximum entropy based IRL algorithms which typically require many trajectories to work well. This further helps reduce computational cost and empirically we notice that it leads to faster convergence as can be seen in Sections 6.3.1.2 and 6.3.2.2.

The method is summarized in Algorithm 2. To initialize the trajectory set $\bar{\tau}$, we solve the OC problem given an initial guess of cost weights which we set to small non-zero values. The resulting trajectory τ_0 is the initial set of the non-optimal trajectories. $\bar{\boldsymbol{\tau}}_{all}$ represents the set of all sampled trajectories throughout the iterations.

6.3 Results

In this section, we evaluate our approach (MO-IRL for Minimum Observation IRL) on two different environments and compare it with state of the art approaches. The first environment is a simple goal-reaching and obstacle avoidance task with a point mass model. The second environment is a reaching task with obstacle avoidance with a simulated Kuka IIWA 14 robot. We compare our results against the approaches proposed in [82] and [6], which will henceforth be referred to as PI^2-IRL and IS-IRL (for Iterative Scaling), respectively. These methods sample local trajectories based on the method provided in [78]. We used Pinocchio [39] together with the Croccoddyl framework [117] and the MiM_Solver nonlinear SQP solver [77] to solve OC problems. We used MuJoCo [180] for the robot simulation. The demonstrated trajectory in all subsequent examples is generated by setting cost weights and using the SQP solver. We do not use the optimal weights anywhere in the tests.

Algorithm 2 MO-IRL

Input: $\tau^\star, L, N$
Output: $\boldsymbol{w}_{irl}$

$t \leftarrow 0$, $\boldsymbol{w}_0 \leftarrow [\mathbf{0.01}]$ # `Non-zero small values`
$D = \{0, \frac{1}{N}T, \frac{2}{N}T, \dots \frac{N-1}{N}T\}$ # `for N sub-samples`
$\tau_0 = OC(\boldsymbol{w}_0)$, $\bar{\boldsymbol{\tau}} = \{\tau_0\}$, $\bar{\boldsymbol{\tau}}_{all} = \bar{\boldsymbol{\tau}}$
$M_1 \leftarrow \infty$, $M_2 \leftarrow \infty$
while not converged **do**
  Find $\Delta \boldsymbol{w}$ using (6.15)
  $\alpha \leftarrow 1$
  while Step not found for 10 trials **do**
    if $m_1(\boldsymbol{w}_t + \alpha\Delta\boldsymbol{w}) < M_1$ or $m_2(\boldsymbol{w}_t + \alpha\Delta\boldsymbol{w}) < M2$ **then**
      $\boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t + \alpha\Delta\boldsymbol{w}$ # `Step Found`
      $M_1 \leftarrow m_1(\boldsymbol{w}_t + \alpha\Delta\boldsymbol{w})$
      $M_2 \leftarrow m_2(\boldsymbol{w}_t + \alpha\Delta\boldsymbol{w})$
    else
      $\alpha \leftarrow \alpha/4$ # `Step Not Found`
    end if
  end while
  $\tau_{t+1} = OC(\boldsymbol{w}_{t+1})$
  $\bar{\boldsymbol{\tau}}_{all} = \{\tau_0, \tau_1, \dots, \tau_{t+1}\}$
  if $t + 1 > L$ **then**
    $\bar{\boldsymbol{\tau}} = \bar{\boldsymbol{\tau}}_{all}$
  else
    $\bar{\boldsymbol{\tau}} = \{\tau_{t-L+1}, \tau_{t-L+2}, \dots, \tau_{t+1}\}$
  end if
  $t \leftarrow t + 1$
end while

6.3.1 Point Mass Environment

6.3.1.1 Problem setup

Our first environment is a 2D point mass that reaches a target reaching and obstacle avoidance. The task is to reach the green square, while avoiding the gray circles. The full trajectories are $1.5s$ long for in the one obstacle case, and $2.5s$ seconds long for the other cases with $\Delta t = 0.05$. We use the following cost features: Goal tracking ('G') is a quadratic cost on both state distance to the goal and non-zero velocity, state and control regularization (Namely 'XReg' and 'UReg') is the squared summation of the corresponding vectors, obstacle avoidance cost for N_{obs} obstacles ('Obs') is zero if the point mass is farther from an obstacle by

an activation margin l, and otherwise increases quadratically with the distance to the obstacle's center O_i. We use the same set of features (excluding UReg) for the terminal costs.

The initial trajectory set for PI^2-IRL and IS-IRL were generated by rolling out 20 noisy control inputs in the parametrization of the optimal trajectory. For MO-IRL, however, there is no need for an initial set, and only an OC solution with small non-zero weights will suffice as the 'set' of non-optimal trajectories. For the moving window of the MO-IRL we used $L = 1$, meaning that at each iteration only the previously generated trajectory is used for updating the weights. Lastly, we used $N = 20$ sub-samples for the trajectories, and regularized the maximum likelihood estimation by choosing $\lambda = 10^{-6}, \beta = 10^{-2}$.

6.3.1.2 Results

Fig.6.1 shows the performance of MO-IRL in comparison with PI^2-IRL and IS-IRL. We test the algorithms in 3 different point mass environments with 1, 4, and 5 obstacles (referred to as 'PM1', 'PM2', and 'PM3', respectively), and show the OC solution for the resulting cost function for the point mass starting from different locations on the map. For MO-IRL, convergence happens after 6, 2, and 2 iterations for PM1, PM2, and PM3 respectively, with samplings getting closer to the optimal trajectory faster than the other methods. We can also see, that with changes in the task (starting from other points) the point mass still avoids obstacles and reaches the goal.

PI^2-IRL however, fails to learn a cost that avoids obstacles. This mismatch of goal association comes from the fact that 'XReg' and 'G' features are inherently opposing each other, making it a challenge to tune them well. During optimization, we observed that PI^2-IRL fails to lower the 'XReg' weights in the terminal features, which results in the attraction of the point mass towards the starting point when approaching the terminal state. IS-IRL samples trajectories at each iteration by analyzing the entire sampled set, and based on the minimum cost difference of the trajectories, increases the *temperature* of the exponentiated cost in the maximum likelihood estimation process. This tends to render learning more sensitive to smaller changes in the feature space as samples get closer to optimality. Although this approach has the advantage of monitoring the IRL improvement, it has some

potential issues. First, the stopping criteria only rests on temperature passing a threshold, and not how the weights converge. As a result, IS-IRL will increase the weights to much higher values than needed by the last iterations, losing sensitivity to some features, which can lead to some issues. For example, we see in Fig.6.1 that in PM2, all trajectories starting behind the obstacles converge to a local minimum and never reach the goal. Indeed, the loss of sensitivity to certain features led the algorithm to give too large weight on the obstacle avoidance costs to the detriment of target reaching. Secondly, IS-IRL intends to change the effectiveness of the sampled trajectories, but it does so by changing the temperature for the whole set, meaning that the entire set is needed to form the partition function. This can be computationally heavy as iterations proceed. Conversely, in our method the influence of the trajectories is individually tuned given the previous weights, which relaxes the need for having the full set at all times.

Fig.6.2 shows the performance of the algorithms on the point mass tests. MO-IRL concludes the training much earlier than IS-IRL, even though it terminates on a higher trajectory costs. IS-IRL usually starts the iterations where PI^2-IRL finishes the learning, which is expected since without iterative scaling of the partition function, IS-IRL and PI^2-IRL are practically very close in nature. Table 6.1, shows the costs of the trajectories generated from the learned weights from each algorithm using the original optimal cost function. Our approach produces a trajectory with a cost closer to the optimal cost. In addition, computation time is provided in Table 6.2 and stresses the performance of the method to find a suitable cost function. While PI2-IRL is faster, it does not converge to good cost functions. We implemented each algorithm in a similar way but our results need to be taken with care as it is always possible to improve an implementation.

To better understand the importance of each part of the algorithm, we conducted experiments with PM2 with versions of the algorithm that did not include the step acceptance strategy, the regularization or sub-sampling. The results are summarized in Fig. 6.3. We notice that without either of these approaches, the algorithm fails completely at recovering a good cost. Adding sub-sampling leads to some trajectories reaching the target. Adding the step acceptance strategy further leads to trajectories that always reach the target several are colliding with the obstacles. Regularization similarly improves the behavior of the algorithm, with

similar obstacle collision issues. The best results are found when everything is used.

6.3.2 Robot Simulation

6.3.2.1 problem Setup and Initialization

In this example, we study a goal reaching task with obstacle avoidance for a simulated torque-controlled 7-DOFs Kuka iiwa robot. We use the same cost features (albeit in a higher dimensional space) and do the following modifications. The goal cost is computed in end-effector frame. We further create capsules around the last four links of the robot and use the distance between the capsules and the obstacles to compute obstacle feature costs (one feature per pair of capsule / obstacle). The full trajectories are one second long, with $\Delta t = 0.01$. The environment consists of $N_{obs} = 4$ box obstacles that the robot has to maneuver around in order to get to the green sphere target. For stage cost, there are 19 features (16 for obstacle avoidance, XReg, UReg, and G), and 18 for the terminal (All but UReg), making the total of 37 cost features. The initial trajectory set for PI^2-IRL and IS-IRL were generated by 20 noisy DMP rollouts. We noticed that when the feature numbers are relatively high, it is more beneficial to the learning if the weights are bounded as mentioned in 6.2.4 so that no *overweighting* would occur. Although this weight binding is a helpful improvement for MO-IRL, it is necessary for IS-IRL since it weighs the obstacle avoidance features high and forgets other cost features, resulting in robot not moving at all. Hence, for the results presented in this section, we used weight bounds for all approaches ($\mathbf{1} \geq \boldsymbol{w}_t \geq \mathbf{0}$). In the robot simulation scenario, we choose parameters $L = 1$, $N = 20$, $\lambda = 10^{-6}$, and $\beta = 10^{-5}$ for MO-IRL.

Table 6.1: Point Mass costs given optimal cost function

Cost $\times 10^4$	$\boldsymbol{w}^{\star T}\Phi_{MO}$	$\boldsymbol{w}^{\star T}\Phi_{IS}$	$\boldsymbol{w}^{\star T}\Phi_{PI^2}$	$\boldsymbol{w}^{\star T}\Phi^{\star}$
PM1	0.5085	0.5121	1.8025	0.5036
PM2	0.6061	6.0114^{+}	6.3888^{+}	0.5091
PM3	0.5613	0.5779	6.0938^{+}	0.4970

$^{+}$Did not converge to target.

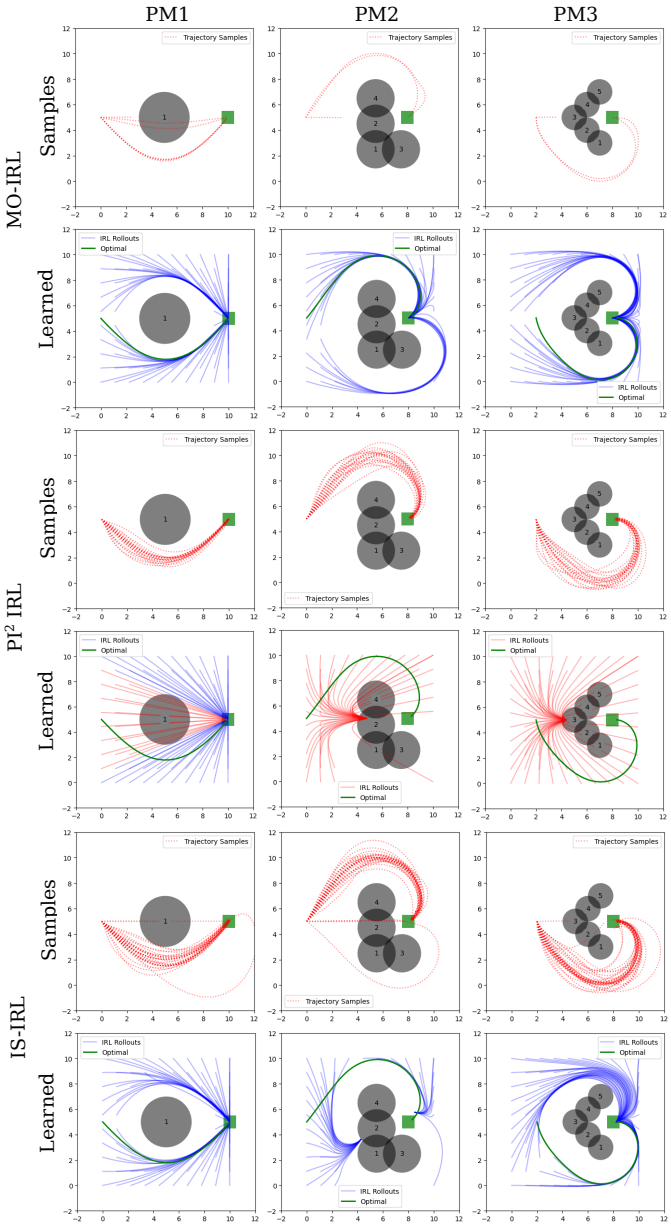


Figure 6.1: The performance of the IRL algorithms on three point mass environments (one per column). Results are shown in pairs of rows for MO-IRL, PI^2-IRL, and IS-IRL. Top row of each pair shows the sampled trajectory set for each algorithm. Bottom rows of pairs show the set of OC rollouts based on the learned cost function. The initial optimal trajectory is shown in green in the bottom row. In the bottom rows, rollouts with obstacle collision are shown in red.

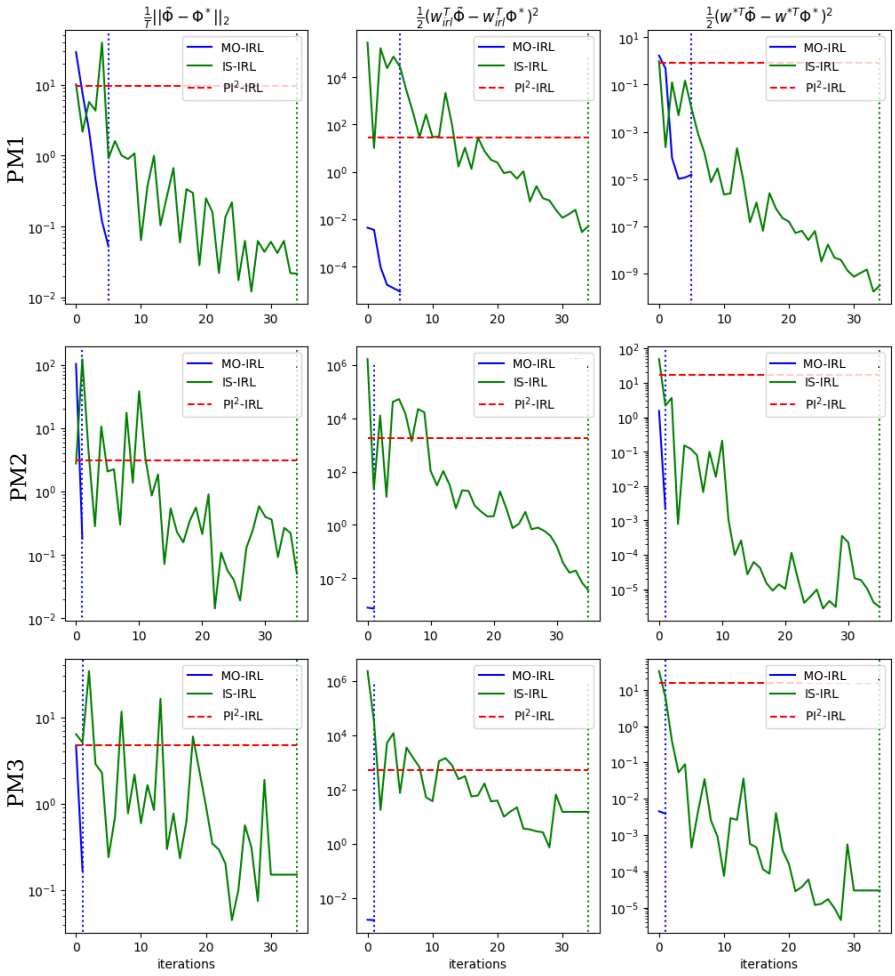


Figure 6.2: Performance of MO-IRL and IS-IRL throughout their iterations. First column indicates the deviation of the resulting trajectory from optimality, second column shows the cost differences of the optimal and the solved trajectory given the estimated weight set, and the third column shows the cost difference of the optimal and solved trajectory given the optimal weights at each iteration. Vertical dashed lines show the termination of associated algorithms. PI^2-IRL's performance is shown by a horizontal red dashed line.

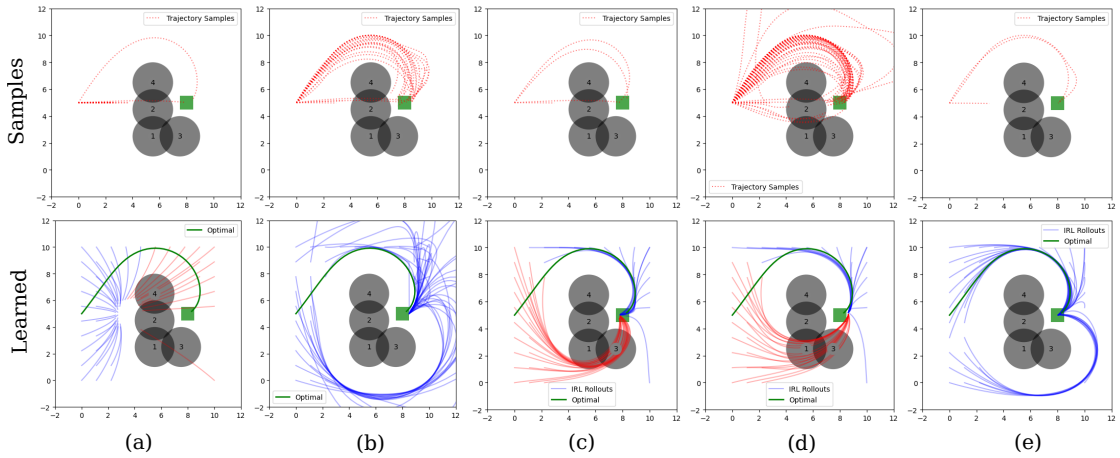


Figure 6.3: Comparison of each improving step for modifying the IRL performance. **(a)** shows the algorithm without any step acceptance, regularization, and sub-sampling. **(b)** is the same as (a) with sub-sampling. **(c)** shows (b) with added step acceptance method. **(d)** is (b) but with regularization. **(e)** shows the full version with sub-sampling, regularization, and step acceptance method.

6.3.2.2 Results

Fig.6.4 portrays the sampling set for IRL performances in the Kuka simulation. It can be seen that the trajectory sampling of MO-IRL is significantly different from that of the other approaches. The samples needed for MO-IRL are not local unlike the other samplings seen for PI^2-IRL and IS-IRL. In addition, due to this particular sampling, MO-IRL can converge with a lower number of trajectory samples. For PI^2-IRL the number of samples does not change (20 trajectories). For IS-IRL, this number rose to 55, and for MO-IRL, total sample count was only 14. Although the

Table 6.2: Point Mass IRL computation time (sec)

	PM1	PM2	PM3
MO-IRL	20.14 ± 3.60	24.69 ± 18.76	29.50 ± 9.51
IS-IRL	40.77 ± 3.54	93.07 ± 17.45	90.17 ± 8.79
PI^2-IRL	4.13 ± 3.07	4.72 ± 3.71	7.84 ± 2.91

*Results are achieved under the same setting on a single device, with an Intel Core i7-10750H CPU @ 2.60GHz

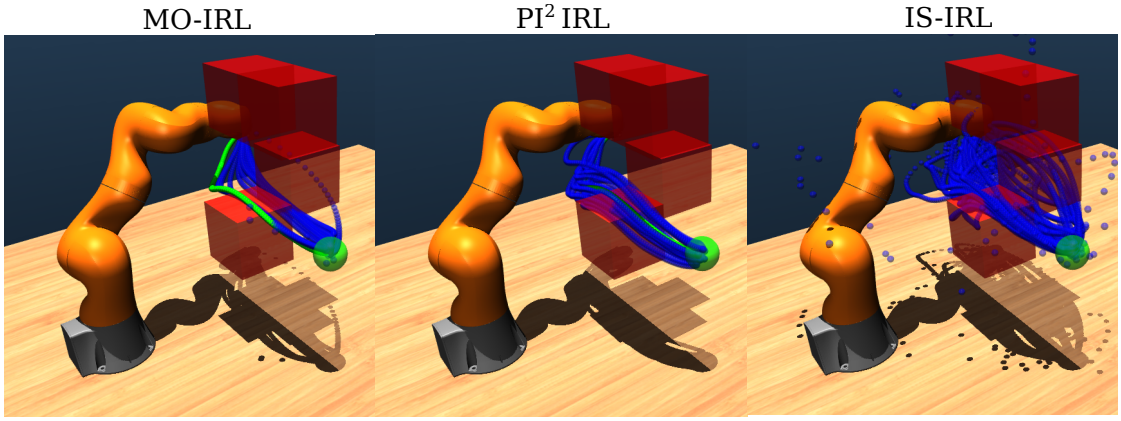


Figure 6.4: Sampled trajectory sets for the IRL approaches. MO-IRL carries out a different way of sampling that is not necessarily local to the optimal trajectory. PI^2 IRL uses only the initial noisy local rollouts. IS-IRL concludes the iterations with a trajectory set containing both initial local rollouts, and OC's solutions. The green path seen on the left figure is the optimal trajectory.

samples for MO-IRL are not as diverse as the ones for IS-IRL, there are trajectories that have gone through the obstacles that could not have been achieved by local rollouts. This ability of MO-IRL to generate informative bad demonstrations is extremely important for exposing the robot to states that it should learn not to go towards. For example, in the initial stage where there are only τ_0 and $\tau^\star$ sampled, MO-IRL optimize a cost that create trajectories going straight to the goal. This results in sampling a trajectory that hits the obstacles along its way, gaining more knowledge about the environment in subsequent iterations.

We compare the iterative performance of the algorithms for robot simulation in Fig.6.5. Similar to point mass tests, MO-IRL terminates after a lower number of iterations with higher optimal trajectory cost difference in comparison to IS-IRL. Unlike in the point mass tests however, MO-IRL produces the most optimal trajectory in terms of the features (left sub-figure). Moreover, since we used only the latest trajectory in MO-IRL, the computation for weight change is less demanding, and the convergence happened twice as fast as IS-IRL.

Fig. 6.6 shows how the robot conducted the task given the learned weights. PI^2-IRL in this test case performed poorly, as the same failure occurred for tuning the XReg against G, therefore robot stands mostly still. For that reason we cannot

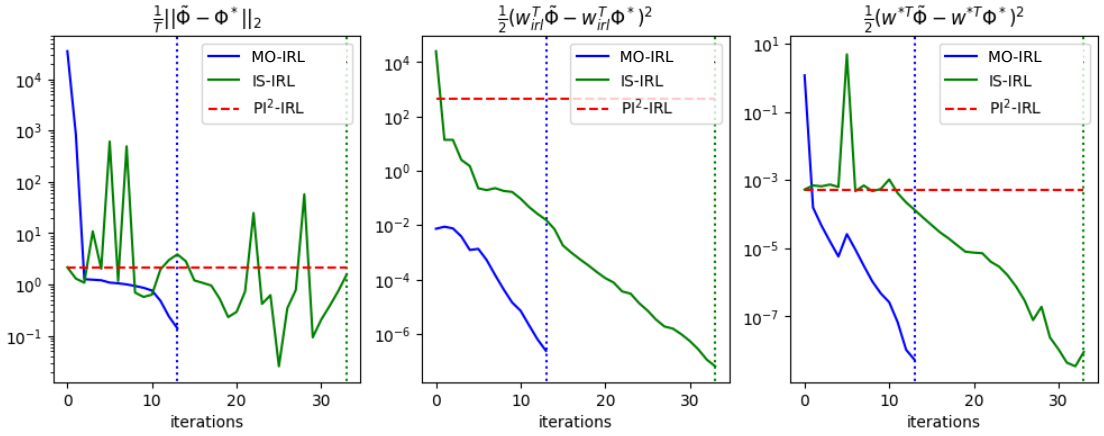


Figure 6.5: Iterative performance comparison between MO-IRL and IS-IRL. PI^2-IRL's performance is indicated by the horizontal dashed red line.

portray any movement resulting from PI^2-IRL in Fig. 6.6. MO-IRL and IS-IRL however showed good performance for the same task and when the goal position was changed. Apart from the environment with the same goal position (left sub-figure) where IS-IRL slightly grazes the lower obstacle, both algorithms showed nearly the same behavior.

To empirically investigate how the learned cost would fare in a different environment, we moved the obstacles and the goal in such a way that the robot needs to needle through the obstacles to get to the target. Fig. 6.7 shows the results for this experiment. Like in the previous environment, the robot shows performance with lower cost closer to the optimal one. However, we notice that the resulting behavior is different from the one generated with the optimal cost. The optimal cost (which we tuned for the first environment) generates a trajectory that hits the obstacles while the trajectory generated by our approach is different and does not hit the obstacles. We hypothesize that since our approach is designed to find small weight changes Δw, it tends to find smaller, well-balanced weights overall which might lead to better balanced feature selection. This will need, however, to be further investigated.

Table 6.3 shows the comparison between costs of the trajectory generated by the weights learned by MO-IRL, IS-IRL, and PI^2-IRL with respect to the desired cost function. We provide computational times to convergence as well as cost per

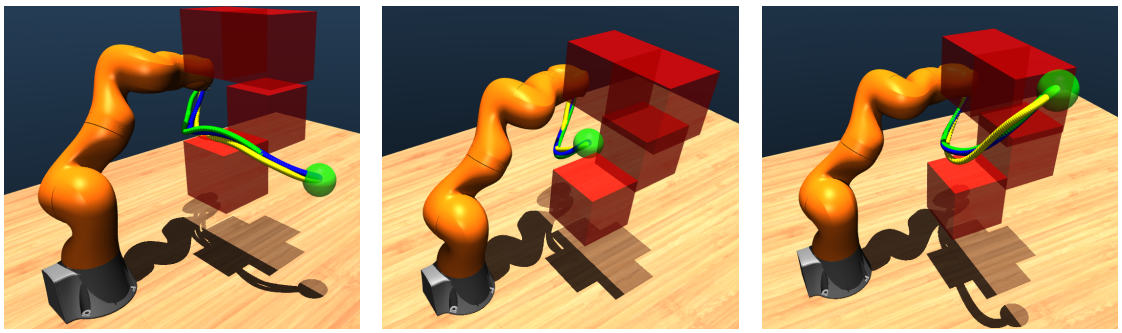


Figure 6.6: Comparison of robot behavior in the same obstacle environment with different goal locations. MO-IRL, IS-IRL, and optimal solution given the desired weights, are shown in blue, yellow, and green respectively.

iteration. MO-IRL is very efficient compared to IS-IRL (PI2-IRL did not find good costs).

6.4 Discussions

We presented the MO-IRL algorithm, which iteratively improves the estimate of the optimal cost function by representing the trajectory space with minimal samples. We have shown that the approach can outperform state of the art IRL algorithms. Our method assigns individual tuning to trajectory samples for the partition function at each iteration depending on the most recent weight estimation with respect to the optimal trajectory. This means that any subset of samples in the set using this method can provide a reasonable space representation, relaxing

Table 6.3: Robot Simulation costs given optimal cost function

Cost $\times 10^4$	$\boldsymbol{w}^{\star T}\Phi_{MO}$	$\boldsymbol{w}^{\star T}\Phi_{IS}$	$\boldsymbol{w}^{\star T}\Phi_{PI^2}$	$\boldsymbol{w}^{\star T}\Phi^{\star}$
Same Task	8.635	8.949	322.703^{+}	7.607
Changed Env.	365.221	394.249	380.1662^{+}	182.584
Duration (s)	20.8 ± 10.5	73.6 ± 28.5	45.9 ± 30.5	N/A
1 Iteration (s)	6.16 ± 5.11	2.07 ± 0.70	N/A	N/A

$^{+}$Did not converge to target.

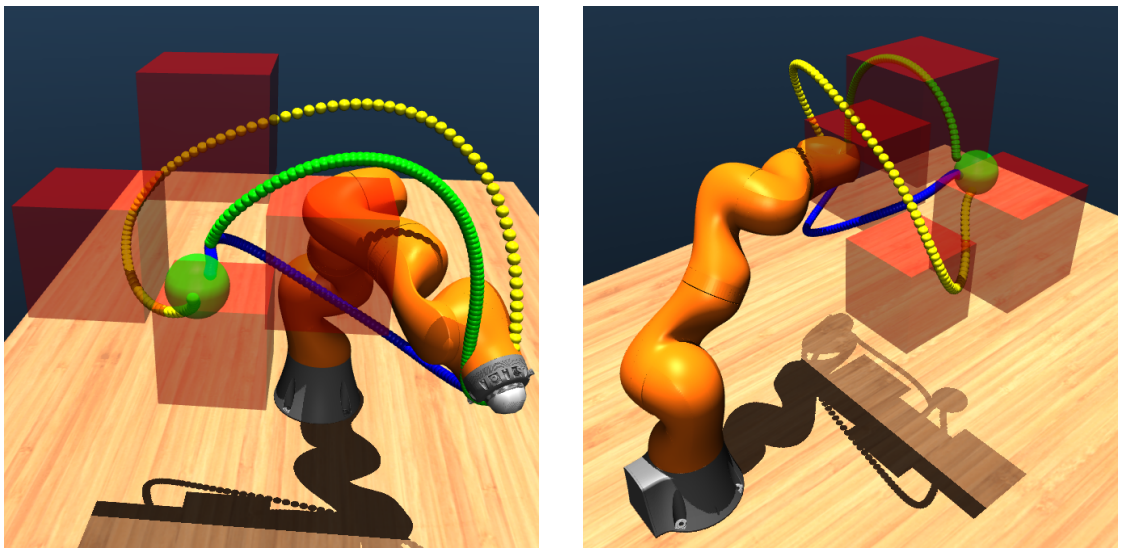


Figure 6.7: Comparison of robot behavior in different challenging obstacle environment and goal location. The images are for the same test from different angles for better visibility. The color-coding is the same as in Fig. 6.6.

the need for subjective tunings in the partition function, or necessity of including all samples. Due to the independence of MO-IRL to the whole sampled trajectory set, even one trajectory can be sufficient for learning the weights properly. This helps both convergence and computational complexity. Moreover, MO-IRL does not rely on local samplings and consequently does not require a policy parametrization. Instead, the samples are drawn from the OC solver at each iteration where a new cost function is estimated.

In the future, we aim to exploit the algorithm and its advantageous computational complexity towards online cost learning and MPC implementations. This would be beneficial for human-in-the-loop scenarios where an expert is presently trying to interactively teach a task to a robot.

Chapter 7

Human Motion Prediction using Inverse Reinforcement Learning with Minimal Observations

This chapter investigates whether a single, unified cost function can explain and predict human reaching movements, in contrast with existing approaches that rely on subject- or posture-specific optimization criteria. Using the Minimal Observation Inverse Reinforcement Learning (MO-IRL) algorithm, together with a seven-dimensional set of candidate cost terms, we efficiently estimate time-varying cost weights for a standard planar reaching task. MO-IRL provides orders-of-magnitude faster convergence than bilevel formulations, while using only a fraction of the available data, enabling the practical exploration of time-varying cost structures.

Three levels of generality are evaluated:

1. Subject-Dependent Posture-Dependent
2. Subject-Dependent Posture-Independent
3. Subject-Independent Posture-Independent

Across all cases, time-varying weights substantially improve trajectory reconstruction, yielding an average 27% reduction in RMSE compared to the baseline. The inferred costs consistently highlight a dominant role for joint-acceleration regulation, complemented by smaller contributions from torque-change smoothness.

Overall, a single subject- and posture-agnostic time-varying cost function is shown to predict human reaching trajectories with high accuracy, supporting the existence of a unified optimality principle governing this class of movements.

IRL offers a probabilistic alternative that reduces the number of nested optimizations, but its performance depends strongly on the quality of the trajectory-space approximation. Existing approaches attempt to improve this approximation through local sampling around demonstrations [79, 81] or through trajectory augmentation [59], but sampled trajectories often remain too close to the observed ones and must all be included in the likelihood computation, which increases computational cost.

To overcome these limitations, we turn to Minimal Observation Inverse Reinforcement Learning (MO-IRL) [123]. This method approximates the trajectory space by iteratively scaling the contribution of each observed trajectory based on its estimated optimality. Even with a small observation set, MO-IRL provides improved weight updates and achieves significantly faster convergence.

In this context, the goal of this paper is to investigate, for the first time, whether time-varying cost weights can be used to recover a single unified cost function that explains human reaching motions across different subjects and initial postures. Addressing this question requires an inverse reinforcement learning method that remains efficient even when the number of parameters grows significantly. To this end, we extend Minimal Observation Inverse Reinforcement Learning (MO-IRL) so that it can learn tasks involving time-varying cost weights, and we formulate an estimation procedure that jointly leverages both position and velocity information from human demonstrations. We validate our approach on a widely used reference dataset in the human motor control community [24].

7.1 Methods

7.1.1 Experimental protocol and mechanical model

The human data used in this study were kindly provided by Berret et al. [24]. These data were used in numerous other studies since their publication and are considered a reference. They consist of motion capture 3D marker positions, including markers located at the shoulder, elbow, and wrist level. Data from 15

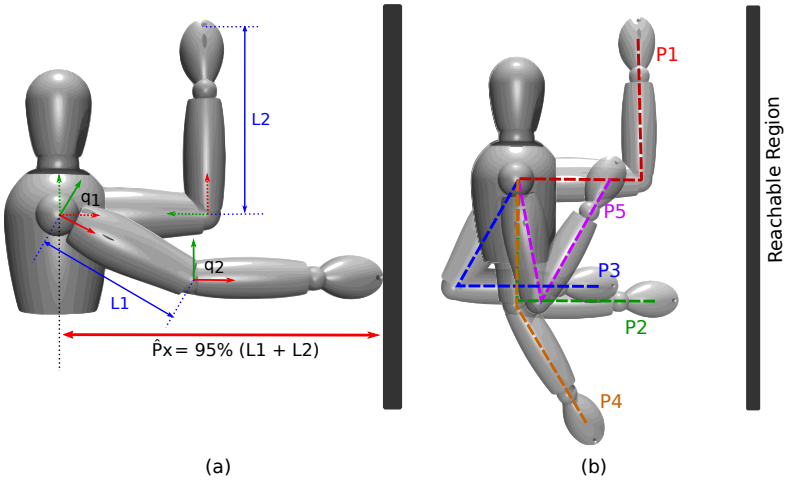


Figure 7.1: (a) Biomechanical model definition, showing the beginning and the end of the pointing task. (b) Five different initial postures for the pointing task [24].

right-handed naive subjects were provided. Subjects performed the pointing task depicted in Fig. 7.1.a. While seated, participants were instructed to perform a series of pointing movements toward a vertical target bar positioned in front of the participant. Only shoulder and elbow flexion/extension were permitted during the task, as wrist movement was restricted. The shoulder-to-bar horizontal distance was set to 85% of the participant's total arm length ($L = L_1 + L_2$, with L_1 and L_2 representing upper arm and forearm lengths, respectively; Fig. 7.1.a). Five initial arm postures, labeled P1 through P5 and shown in Fig. 7.1.b, were defined. Each individual performed 20 trials for each posture.

A planar biomechanical model (Fig.7.1.a) was developed to represent flexion/extension movements at the shoulder (q_1) and elbow (q_2) joints. The model's base frame was located at the shoulder joint, and the relative positions of successive joints in their parent frames were computed using segment lengths L_1 and L_2, estimated from marker data. Inertial parameters were calculated using anthropometric tables [52].

7.1.2 Optimal control problem

In the context of pointing or reaching movements, inspired by the literature in human motor control [16, 22, 25, 26, 60, 128, 129, 132, 183], we propose a set of

$N_\Phi = 7$ candidate cost functions, as detailed in Table 7.1. Although the task may appear elementary, we posit that individuals do not adhere to a single cost function throughout the entire movement. To account for such time-varying motor strategies, each recorded trajectory of duration T was segmented into N_w equal time windows, each comprising N_s samples. This segmentation was chosen based on consistent inflection points observed in the majority of trajectories. To model the temporal evolution of movement strategies, we introduced a weight matrix $\boldsymbol{\omega} \in \boldsymbol{R}^{N_\Phi \times N_w}$, which allows distinct cost function contributions across different movement phases. The full trajectory $\boldsymbol{x} \in \boldsymbol{R}^{N_s \times N_w}$ was defined as the concatenation of state vectors $\boldsymbol{x}_s = (\boldsymbol{q}_s, \dot{\boldsymbol{q}}_s)$ for each section $s \in \{1, \cdots, N_w\}$. Accordingly, $\boldsymbol{u}_s$ was defined as the control torque input to the human model joints for each section. The associated Direct Optimal Control (DOC) problem was then formulated to reconstruct the observed human motion over this multi-phase framework.

$$
\begin{aligned}
\boldsymbol{x}^* = \ & \arg\min_{\boldsymbol{u}} \quad \sum_{s=1}^{N_w} \sum_{j=1}^{N_\Phi} \omega_{s,j} \Phi_j(\mathbf{x}_s, \boldsymbol{u}_s) \\
& \text{s.t.} \quad \boldsymbol{x}(t+1) = f(\boldsymbol{x}(t), \boldsymbol{u}(t)), \\
& \qquad P_X(T) = \hat{P}_X, \\
& \qquad \boldsymbol{q}^- \leq \boldsymbol{q} \leq \boldsymbol{q}^+ \\
& \qquad \boldsymbol{q}(0) = \boldsymbol{q_0} \\
& \qquad ||\dot{\boldsymbol{q}}|| \leq \dot{\boldsymbol{q}}^+
\end{aligned}
\tag{7.1}
$$

where $\boldsymbol{x}(t)$ and $\boldsymbol{u}(t)$ are the state and control at discrete time t; f is the Euler time-discretized dynamics; $P(t)$ is the position of the hand obtained from forward kinematics, and $\hat{P}_X$ is the horizontal goal position (Fig. 7.1.a); $\boldsymbol{q_0}$ is the initial human joint configuration; $\boldsymbol{q}^-$, $\boldsymbol{q}^+$ are the lower and upper joint boundaries, respectively; $\dot{\boldsymbol{q}}^+$ is the maximal joint velocity.

Modeling of the human body was done using Pinocchio [39], and the Croccoddyl framework [117] and the MiM_Solver nonlinear CSQP solver [77] were used to define and solve the constrained DOC in Eq.(7.1).

Table 7.1: (Discretized) Biomechanical cost functions

Label	Name	Equation	Reference
Φ_1	Cartesian velocity (CV)	$\sum_{t=0}^{T} \dot{P}(t)^T \dot{P}(t)\, dt$	[60]
Φ_2	Energy (Eng)	$\sum_{t=0}^{T} \left\lvert \dot{\mathbf{q}}(t)^T \boldsymbol{\tau}(t) \right\rvert\, dt$	[25, 132]
Φ_3	Geodesic (Geo)	$\sum_{t=0}^{T} \dot{\mathbf{q}}(t)^T M \dot{\mathbf{q}}(t)\, dt$	[26]
Φ_4	Joint acceleration (JA)	$\sum_{t=0}^{T} \ddot{\mathbf{q}}(t)^T \ddot{\mathbf{q}}(t)\, dt$	[22]
Φ_5	Joint torque change (JTC)	$\sum_{t=0}^{T} \dot{\boldsymbol{\tau}}(t)^T \dot{\boldsymbol{\tau}}(t)\, dt$	[128, 183]
Φ_6	Joint velocity (JV)	$\sum_{t=0}^{T} \dot{\mathbf{q}}(t)^T \dot{\mathbf{q}}(t)\, dt$	[16]
Φ_7	Joint torque (Tau)	$\sum_{t=0}^{T} \boldsymbol{\tau}(t)^T \boldsymbol{\tau}(t)\, dt$	[129]

7.1.3 Minimum Observation IRL (MO-IRL)

As previously mentioned, we aim to solve an IOC problem to derive the optimal cost function as a linear combination of explicit features. For this purpose, an augmented MO-IRL[123] framework was modified to accommodate learning for several time windows with different weights. As in most IRL algorithms, MO-IRL aims to maximize the probability of the optimal demonstrations (i.e. the human trajectory)

$$\begin{aligned} \boldsymbol{\omega}^* &= \arg\max_{\boldsymbol{\omega}} P(\boldsymbol{x}^*|\boldsymbol{\omega}, \bar{\boldsymbol{x}}) \\ \text{where} \quad P(\boldsymbol{x}^*|\boldsymbol{\omega}, \bar{\boldsymbol{x}}) &= \frac{e^{-\boldsymbol{\omega}^T \boldsymbol{\Phi}^*}}{\sum_{i=1}^{K} e^{-\boldsymbol{\omega}^T \boldsymbol{\Phi}_i}} \\ \boldsymbol{\omega} &\geq 0 \end{aligned} \tag{7.2}$$

in which $\bar{\boldsymbol{x}}$ is the set of K observed trajectories. For clarity, we write $\boldsymbol{\omega}$ the concatenated weight vector for the cost features and $\boldsymbol{\Phi}$ the concatenated feature vector. In the following, $\boldsymbol{\Phi}(\boldsymbol{x}_i, \boldsymbol{u}_i)$ which is the feature costs for the i^{th} trajectory is henceforth referred to as $\boldsymbol{\Phi}_i$. As observed in (7.2), all the sub-optimal trajectories are being incorporated and contribute equally in the denominator, irrespective of how close they are to optimality. This lack of distinction can create numerical issues for the optimizer. Therefore, it is important to scale them to emphasize their effectiveness so the optimizer will have better information about what the approximated trajectory set represents in terms of optimality.

MO-IRL solves Eq. (7.2) by iteratively improving $\boldsymbol{\omega}$ rather than optimizing it in one shot. Considering an update of cost weights at each iteration $n+1$ in the form $\boldsymbol{\omega}_{n+1} = \boldsymbol{\omega}_n + \Delta\boldsymbol{\omega}_n$, the original probability distribution can be rewritten to

instead find the best $\Delta\boldsymbol{\omega}_n$:

$$\begin{aligned}\Delta\boldsymbol{\omega}_n^* &= \arg\min_{\Delta\boldsymbol{\omega}_n} -\log\frac{1}{1+\sum_{\boldsymbol{x}_i\in\bar{\boldsymbol{x}}}\gamma_i e^{-\Delta\boldsymbol{w}_t^T(\boldsymbol{\Phi}_i-\boldsymbol{\Phi}^*)}}\\ \text{s.t.}\quad & \Delta\boldsymbol{\omega}_n > -\boldsymbol{\omega}_n\\ & \gamma_i = e^{-\boldsymbol{\omega}_n^T(\boldsymbol{\Phi}_i-\boldsymbol{\Phi}^*)}\end{aligned} \tag{7.3}$$

In this case, sampled trajectories are automatically scaled depending on their cost in the previous iteration. MO-IRL solves Eq. (7.3) and then seeks to find an update of the form $\omega_{t+1} = \omega_t + \alpha\Delta\omega$ where α is selected using a merit function (similar to a line search procedure). Starting with $\alpha = 1$, the algorithm checks if the resulting trajectory is closer to the optimal demonstration by evaluating the merit function. If the merit function value has not been decreased with the added change to the weight, MO-IRL scales down α by factor of 0.25, and tries again for a maximum of 10 trials. If by the 10^{th} trial there was no improvement, the algorithm stops. Otherwise, the accepted trajectory is added to the observed trajectory set $\bar{\boldsymbol{x}}$, and MO-IRL moves on to the next iteration.

In the literature, algorithms to learn human motion trajectories usually minimize the gap between the estimated and the optimal trajectories in joint space without considering velocities. In this study, however, we propose to minimize the gap in both joint position and joint velocity concurrently. Therefore, the estimation improvement was based on the full state vector $\boldsymbol{x} = [q_1, q_2, \dot{q}_1, \dot{q}_2]$ with $m(\boldsymbol{x}) = \frac{1}{T}||\boldsymbol{x}^* - \boldsymbol{x}||_2^2$ as merit function.

MO-IRL formulation was extended for learning multiple weight sections from multiple demonstrations as follows:

$$\begin{aligned}&\Delta\omega_t^* = \arg\min_{\Delta\omega_t}\\ &\sum_{d=1}^{D}\left(-\log\frac{1}{1+\sum_{\boldsymbol{x}\in\bar{\boldsymbol{x}}}\gamma_i e^{-C(\boldsymbol{x}_i,\Delta\boldsymbol{\omega}_t)}}\right)+\frac{\beta}{2}||\Delta\omega_t||_2^2\\ &\text{s.t.}\quad \Delta\boldsymbol{\omega}_t > -\boldsymbol{\omega}_t\\ &\quad \gamma_i = e^{-C(\boldsymbol{x}_i,\omega_t)}\\ &\quad C(\boldsymbol{x}_i,\Delta\omega_t) = \sum_{s=1}^{N_\omega}\Delta\boldsymbol{\omega}_{st}^T(\Phi_{si}-\Phi_{sd}^*)\end{aligned} \tag{7.4}$$

where D is the number of provided optimal demonstrations, and N_ω is the number of weight sections as mentioned before. We also use a small $L2$ regularizer ($\beta = 10^{-9}$) for the optimization to prevent high changes in weights and overfitting. When learning from multiple demonstrations, the merit function for step acceptance is changed to $m(\boldsymbol{x}) = \frac{1}{D}\sum_{d=1}^{D}(\frac{1}{N}||\boldsymbol{x}_d^* - \boldsymbol{x}||_2^2)$. In case the input demonstrations are from various initial postures, MO-IRL will generate one trajectory per initial posture to evaluate the merit function value against human demonstrations starting from the same posture. Eventually, all generated trajectories from different initial postures will be added to $\bar{\boldsymbol{x}}$ when the step gets accepted.

7.1.4 Learning and Validation

In this study, the proposed approach was validated across three cases ranging from subject and posture specific cost functions generation to a generic one. For each demonstration in the input set, a single–shooting optimal control problem was generated using

$$N_\omega = \left\lfloor \tfrac{1}{2} T_d \right\rfloor,$$

which corresponds to the maximum number of weight segments allowable while still enabling MO–IRL to identify meaningful time variations in the cost structure.

To evaluate the weights estimated by MO–IRL, the DOC was solved for each demonstration using its initial configuration $\boldsymbol{q}_0$ and movement duration T. The RMSE between the resulting predicted trajectory and the recorded trajectory was then computed. RMSE values are reported for each joint as well as for the overall average across joints. For each case, a cost function was considered better if it would regenerate human motions that closely matched the recorded demonstrations provided by Berret et al. [24].

7.1.4.1 Subject-Dependent Posture-Dependent (SDPD)

Similarly to Berret et al. [24], SDPD analysis aims to estimate the optimal cost function specified for each subject for each initial posture. The input non-optimal set in this case consists of 3 random demonstrations for each subject and each posture. Overall, there will be 15(subject) $\times$ 5(postures) = 75 estimated cost functions resulting from this test case.

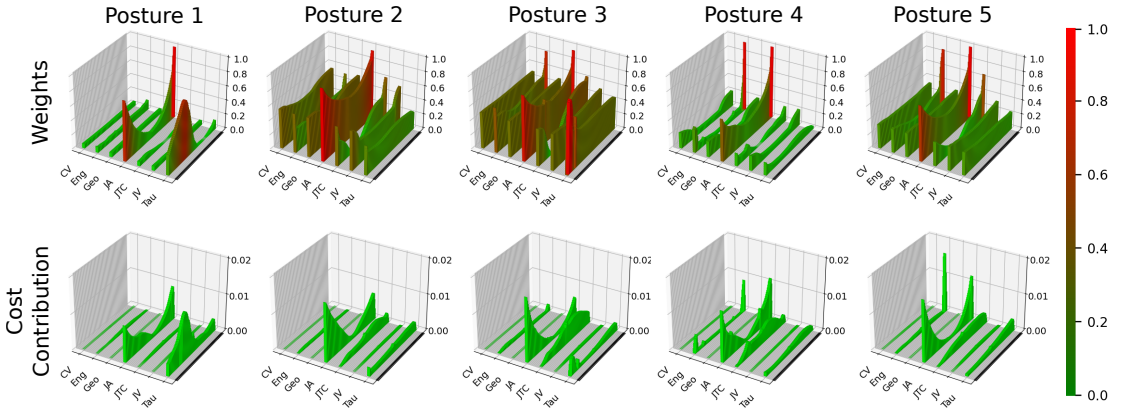


Figure 7.2: Averaged estimated weights, and cost contribution, across all subjects for initial postures 1-5, produced through SDPD analysis. Plots show a prominent importance of the Joint Acceleration feature in the cost function (Φ_4).

7.1.4.2 Subject-Dependent Posture-Independent (SDPI)

The SDPI case is attempting to evaluate if optimal weights can be estimated irrespectively of the initial posture of the human arm. In other words, we aim to find subject specific weights for the reaching task, and to see if they have similarities we can rely on to produce a subject independent cost function. For SDPI, a set of 5 demonstrations is used as input for MO-IRL, consisting of one random demonstration per each initial posture. Overall, there will be 15(subject) estimated cost functions resulting from this test case.

7.1.4.3 Subject-Independent Posture-Independent (SIPI)

The SIPI case aims to estimate an optimal cost function irrespective of the subject and initial posture. In this case, the input demonstration set consists of 5 random demonstration from each subject (one per initial posture), totaling 75. This test case will produce one set of estimated cost weights.

7.2 Results

Fig. 7.2 presents the average estimated weights for all subjects and postures in the SDPD condition. A consistent pattern emerges across all initial postures,

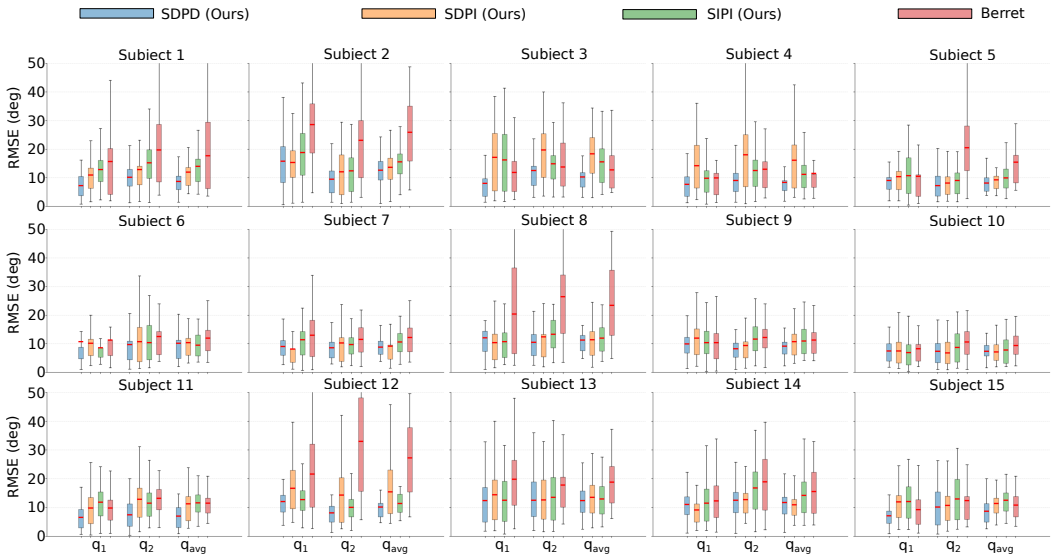


Figure 7.3: RMSE values boxplot comparison of all investigated cases against the baseline, averaged across all subjects. Red line in each boxplot indicates the mean value.

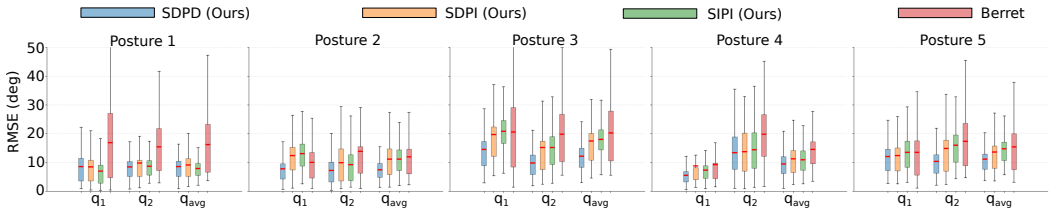


Figure 7.4: RMSE values (deg) boxplot comparison of all methods against the baseline, averaged across all initial postures. Red line in each boxplot indicates the mean value.

with the acceleration cost term Φ_4 dominating the overall contribution. The estimated acceleration weight is particularly high near the beginning and end of the reach, while remaining lower during mid–movement. The resulting predicted joint trajectories exhibit and average RMSE values of 9.59 ± 5.20 deg, as evident in Table 7.2. This is significantly lower than values reported in the literature using a single weight set [24, 105, 177]. Using the fixed weights, obtained from a bi-level approach, for the same pointing data, Berret et al. obtained an average RMSE of 15.44 ± 10.57 deg.

Figs. 7.3 and 7.4 further demonstrate that time varying weights systematically improve prediction accuracy across subjects and postures.

Initial Posture	MO-IRL SDPD (Ours)			Baseline		
	q_1	q_2	Avg	q_1	q_2	Avg
P1	8.56	8.47	**8.51**	16.92	15.40	16.16
P2	7.77	7.17	**7.47**	10.06	13.84	11.95
P3	14.46	9.76	**12.11**	20.60	19.83	20.21
P4	5.52	13.33	**9.43**	9.28	19.84	14.56
P5	11.99	10.36	**11.17**	13.54	17.35	15.45
All	9.50	9.68	**9.59**	13.89	16.99	15.44

Table 7.2: RMSE results of SDPD weights for reaching task generated by MO-IRL, in comparison with the baseline.

In the SDPI condition, a single time–varying weight profile is learned per subject across all postures. The resulting weights again show an obvious rise and fall in Φ_4, confirming the central role of acceleration regularization independently of the initial configuration. RMSE values for each posture are summarized in Table 7.3, showing that SDPI outperforms the baseline in almost all joints and postures. This is an other clue supporting that allowing time–varying weights provides a more flexible representation of human motor patterns than a static cost structure.

Initial Posture	MO-IRL SDPI (Ours)			Baseline		
	q_1	q_2	Avg	q_1	q_2	Avg
P1	8.39	9.80	**9.10**	16.92	15.40	16.16
P2	12.40	9.94	**11.17**	10.06	13.84	11.95
P3	19.71	15.17	**17.44**	20.60	19.83	20.21
P4	8.75	13.76	**11.26**	9.28	19.84	14.56
P5	12.32	14.83	**13.57**	13.54	17.35	15.45
All	11.99	10.36	**11.17**	13.89	16.99	15.44

Table 7.3: RMSE results of SDPI weights for reaching task generated by MO-IRL, in comparison with the baseline.

The SIPI condition investigates whether MO–IRL can recover a single, fully general cost structure for the reaching task, independent of both subjects and initial postures. The estimated weight profile and cost contributions are shown in Fig. 7.5. The same temporal pattern for Φ_4 is present at the population level, suggesting a strong commonality in how humans regulate acceleration throughout the reach.

The SIPI results in Table 7.4 provide several important insights into the existence of a unified cost function for human reaching. First, the MO–IRL weights consistently outperform the baseline across all initial postures, despite being learned without access to subject- or posture-specific structure. The average RMSE is reduced by 27.65% when compare to the baseline, which consider a subject-dependent cost function. This demonstrates that the temporal profile of the inferred cost

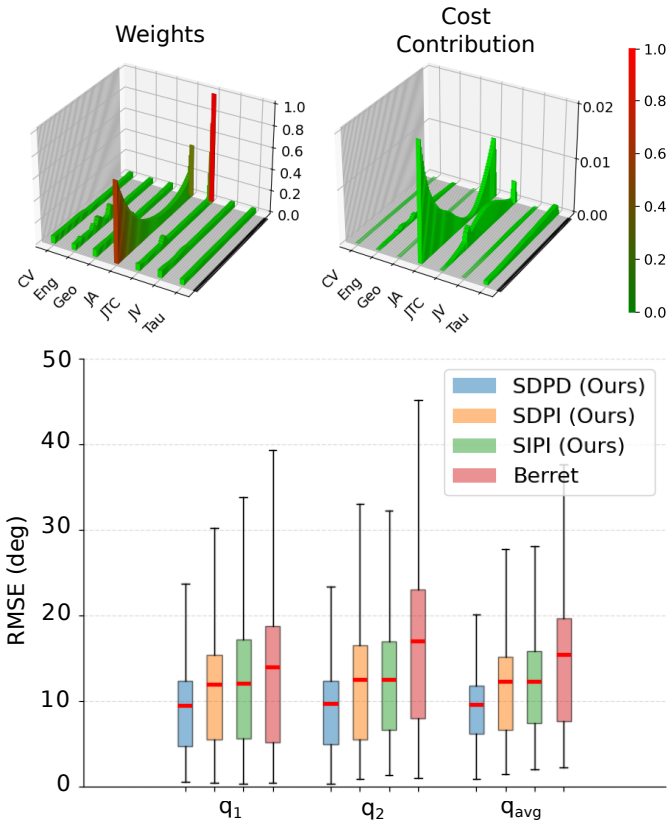


Figure 7.5: Weights and cost contributions of general cost weights proposed for the reaching task in SIPI case. The RMSE comparison boxplots between all existing demonstrations and predictions are presented in the figure at the bottom.

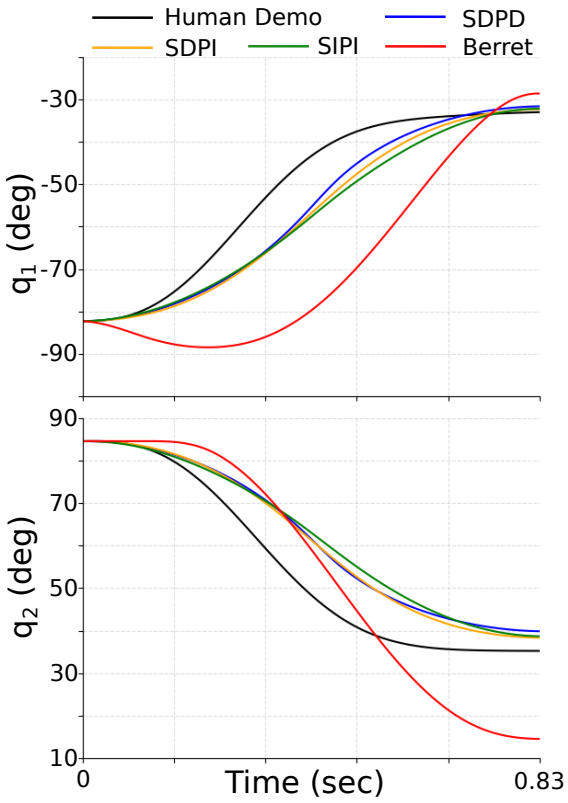


Figure 7.6: Comparison of SDPD, SDPI, SIPI, and baseline predictions of the subject's joint values, against the human demonstration.

function captures fundamental aspects of human motor behavior that generalize across individuals and kinematic configurations.

Second, the reductions in RMSE are particularly substantial for postures P1 and P4. These postures involve relatively extended initial arm configurations, where small variations in joint torques likely lead to more predictable and less noisy task-space motions. In these cases, the general time-varying cost function appears sufficient to reproduce the smooth, bell-shaped trajectories characteristic of human reaches [60], suggesting that the dominant control principles governing these movements are shared across subjects.

Posture P1 illustrates this point most strongly: the baseline method yields an average error of 16.16 deg, while the SIPI cost function reduces this to 7.83 deg. This large improvement indicates that static cost weights, such as those proposed in [24], fail to capture the temporal modulation of control effort and smoothness

Initial Posture	MO-IRL SIPI (Ours)			Baseline		
	q_1	q_2	Avg	q_1	q_2	Avg
P1	6.98	8.69	**7.83**	16.92	15.40	16.16
P2	13.09	9.26	**11.17**	10.06	13.84	11.95
P3	20.85	15.23	**18.04**	20.60	19.83	20.21
P4	7.37	14.36	**10.86**	9.28	19.84	14.56
P5	13.55	16.00	**14.77**	13.54	17.35	15.45
All	11.99	10.36	**11.17**	13.89	16.99	15.44

Table 7.4: RMSE results of SIPI weights for reaching task generated by MO-IRL, in comparison with the baseline.

that humans naturally apply.

Third, posture P3 yields the highest errors for both the baseline and SIPI conditions, reflecting the inherent difficulty of this configuration. Its more flexed-elbow posture requires larger joint excursions for small task-space motions, amplifying joint-space variability and the effects of signal-dependent motor noise [66]. Despite this, the SIPI weights still offer a clear improvement over the baseline, indicating that the general cost function captures consistent behavioral structure even under unfavorable dynamics.

Beyond the dominant influence of joint acceleration, the estimated weights also reveal a meaningful contribution of the joint torque change term (Φ_5), visible in Fig. 7.5 through elevated mid-movement values.

Fig. 7.6 shows the comparison of predictions using SDPD, SDPI, and SIPI methods against the human data, for an example demonstration conducted by Subject 1 initiated from Posture 2. This example shows the effect of the gradual decline of specificity from SDPD to SDPI and ultimately SIPI, as the predicted trajectory deviates from the human demonstration further. However, the results in Fig. 7.6 show a significant improvement in comparison with the baseline in the joint space.

7.3 Discussions

Across all three evaluation cases, a consistent and robust pattern emerges in the inferred cost structure for human reaching. The joint acceleration term Φ_4 is the dominant contributor, with a characteristic increase at movement onset and termination. This behavior aligns with classical motor–control findings: penalizing

large accelerations avoids impulsive motor commands and reduces signal–dependent noise [66], while the terminal increase reflects the need for precise endpoint stabilization, consistent with submovement corrections and bell–shaped velocity profiles [60, 133].

A second key result is the non-negligible contribution of the joint torque change term Φ_5 during the mid-reach phase. This observation complements and extends the acceleration-based explanation. The minimum–torque–change model of Uno et al. [183] established that penalizing rapid variations in torques generates smooth and human-like multi-joint trajectories. More recent studies have further shown that stabilizing torque fluctuations helps reduce motor noise, improves robustness to interaction torques, and limits neuromuscular effort [51, 66]. For our reaching scenario the joint torque change appearance at the middle of the motion is consistent with the need to maintain stable coordination while the segments are accelerating most rapidly, a period where motor noise and interaction torques are largest. The combined presence of Φ_4 and Φ_5 therefore suggests that human reaching optimizes both kinematic smoothness (acceleration regulation) and actuation smoothness (torque-change regulation), revealing a multi-criteria optimality principle that adapts over the course of the movement.

The limited influence of energy-related terms stands in contrast to the interpretation proposed in [24]. Several factors may explain this discrepancy. First, the pivot-based normalization used in [24] imposes strong constraints on the relative magnitudes of the cost weights, whereas the present formulation allows them to evolve freely over time. Second, the very large RMSE values reported when using Berret's weights (Table 7.4) suggest that the attribution of a dominant role to joint energy may simply not be reliable. When prediction errors are so high, many different combinations of cost terms can yield equally poor fits to the data, making it difficult to draw firm conclusions about the underlying optimality principle. In this sense, the explanation based on joint energy may simply not be identifiable within the sensitivity limits of the DOC problem.

An other methodological difference from previous IOC and IRL studies concerns the use of both joint positions and joint velocities in the learning process. Most approaches in the literature rely exclusively on joint positions when estimating cost weights, either because velocities exhibit higher trial-to-trial variability or

because position-only models simplify the inverse problem [177]. However, excluding velocity information can obscure important aspects of the movement dynamics. Human reaching is characterized by highly stereotyped velocity profiles, and these profiles carry essential information about smoothness, timing, and coordination. Incorporating velocities therefore provides richer constraints on the underlying cost structure and reduces the ambiguity inherent in position-only formulations, where many weight combinations can reproduce similar end-point trajectories. In the present work, we believe that, the inclusion of velocity data contributes to the clear emergence of both acceleration and torque-change related terms, and likely improves the identifiability of time-varying weights.

Taken together, these findings support the interpretation that the central nervous system relies on time-varying adjustments of a small set of key control principles, rather than on fixed task- or subject-specific cost parameters. The consistency of the inferred weights across heterogeneous initial postures highlights the ability of MO–IRL to uncover this shared temporal structure.

Finally, the SIPI results demonstrate that a single time-varying cost function can reconstruct human trajectories across all subjects and postures with markedly higher accuracy than classical single-weight models. Achieving the performance reported in Table 7.4 with a fully general cost function is noteworthy, and provides, to our knowledge, the first evidence that human reaching may be governed not by static cost weights, but by a temporally structured cost landscape balancing effort, smoothness, and accuracy throughout the movement.

7.4 Conclusion

This work introduced a modified MO–IRL framework for learning time-varying cost functions that predict human reaching movements with substantially higher accuracy than classical fixed-weight approaches. The inferred cost structure consistently highlights the dominant role of joint acceleration, complemented by a smaller contribution of torque-change smoothness in the middle of the motion, reflecting a multi-criteria optimality principle that adapts throughout the movement. For the first time a single subject- and posture-independent cost function was identified and its ability to predict motion demonstrated with real experimental data. It

suggests that reaching may be governed by a shared temporal organization rather than by fixed task-specific parameters.

A central methodological contribution of this study is the use of both joint positions and joint velocities during weight estimation. Incorporating velocity information reduces the ambiguity inherent in position-only formulations and improves the identifiability of dynamic cost terms. This points toward a broader conclusion: future IOC and IRL frameworks may benefit from exploiting the full state and input trajectories, including joint torques, external forces, and interaction-based cues, especially when dealing with inherently multimodal human motion.

An important next step is to assess whether the proposed cost function generalizes to more realistic reaching scenarios involving full 3D arm motion. Extending the framework to higher-dimensional models will allow testing whether the temporal structure uncovered here persists beyond planar tasks, and whether additional cost components emerge when multi-joint coordination become more complex.

From a robotics perspective, the inferred time-varying cost structure provides a principled foundation for designing bio-inspired controllers and generating human-like trajectories in humanoid and collaborative robots. A key advantage of MO–IRL is its ability to learn generalizable cost functions from only a small number of demonstrations. This greatly reduces the data-collection burden typically associated with human motion modeling and could enable the rapid construction of large-scale synthetic motion datasets. Such datasets are increasingly valuable for training robot imitation-learning policies, for benchmarking perception or prediction modules, and for providing diverse, human-consistent motion samples in simulation.

Embedding the learned cost functions into optimal control or Model Predictive Control frameworks allows robots not only to reproduce human-like movement patterns, but also to anticipate and respond to human actions in an optimal and temporally coordinated manner. This opens the way to more natural physical interaction, improved intent prediction, and adaptive assistance strategies that align robotic behaviour with the underlying principles of human motor control.

Chapter 8

Conclusions and Future Work

Understanding humans—from physiological state to external behavior to underlying intent—is central to advancing embodied intelligence and enabling robots to interact meaningfully with people. However, human-centered learning is fundamentally challenged by data scarcity, noise, suboptimality, and inter-individual variability. This thesis addressed these challenges through a multi-layered progression of contributions unified by one methodological principle: *leveraging structure to overcome limited data*.

In Part I, we demonstrated that clinically significant human-state estimation can be achieved from extremely limited measurements. By modeling the temporal evolution of only three vital signs, our LSTM- and TCN-based prognostic frameworks accurately predicted deterioration in COVID-19 patients across multiple horizons. Beyond pandemic relevance, these results illustrate that high-value physiological insights do not require dense biomedical datasets; instead, carefully designed temporal models can extract latent state information even under severe data constraints. As future directions for this line of work, the proposed modeling structures can be extended to multiple diseases to develop a unified, disease-agnostic framework. Beyond COVID-19, time-series biomarker datasets exist for a wide range of syndromes, for which accurate prognosis can be critically valuable in mitigating subsequent adverse outcomes. This work demonstrates that the AI community can leverage relatively simple and low-dimensional datasets to achieve high-performance predictive models, thereby enabling the development of reliable clinical decision-support platforms with reduced burden on the medical workforce.

In Part II, we developed probabilistic modeling tools capable of representing complex, asymmetric, and context-dependent human motion. The proposed FiMReSt model stabilized skew-t mixture learning and enabled expressive behavioral representations, while SNMR provided a flexible regression framework that generalizes well in few-shot settings. Incorporating optimality-aware weighting into Learning from Demonstration further addressed suboptimal human input, allowing robots to learn meaningful behavior even when demonstrations are imperfect. Together, these contributions show that rich behavioral understanding is possible when statistical structure is exploited, even with minimal or noisy data. Several avenues remain open for future extensions of this work. The regression formulation of the FiMReSt model has yet to be systematically evaluated across diverse datasets to examine how mixture models based on kernels with greater expressiveness and reach can improve behavior learning from human demonstrations. Furthermore, the context-aware EM algorithm introduced in Chapter 5 has demonstrated strong potential as a lightweight behavior cloning framework based on GMM/GMR that explicitly accounts for task objectives. Replacing Gaussian kernels in this framework with more expressive alternatives—such as SNMM/SNMR and the FiMReSt model and its regression formulation—is expected to further enhance behavior learning performance, particularly given the gains already observed using lower-flexibility kernels.

In Part III, we shifted from descriptive to principled models of behavior by recovering why humans act as they do. MO-IRL demonstrated that inverse reinforcement learning can be made tractable and data-efficient, enabling cost-function recovery from sparse and partially observed trajectories. The method successfully inferred both task-specific and subject-independent cost structures, providing new insights into human motor control and enabling accurate motion prediction. These findings suggest that human intent—often considered inaccessible due to ambiguity and under-determination—can be systematically estimated through structured optimization and informed sampling strategies. The studies presented in Part III can be extended along two complementary directions. The first is the development of a lightweight, low-horizon variant of MO-IRL that operates in real time, enabling a robot to infer and update its cost function as its behavior is being corrected by a human expert during physical human–robot interaction (pHRI). In this setting,

external corrective inputs must be locally compensated through immediate updates to the cost function, allowing the robot to incorporate the objective modifications implicitly intended by the human operator as soon as they are observed. Such an approach has the potential to substantially improve objective learning in scenarios where discrepancies arise between the robot's learned intent and the operator's true intent. These discrepancies may occur, for example, when a robot trained by one human is subsequently operated by another, or when the robot encounters a novel environment that requires behavioral adaptation without access to additional demonstration data, making full offline MO-IRL—as presented in Chapter 6 impractical.

The second direction concerns the modeling of human intent itself. The human intent learning framework introduced in Chapter 7 demonstrated strong predictive performance by assuming multiple weight sections within a task. Extending this formulation to a broader range of human behaviors would enable a systematic investigation of multi-phase task execution, with the goal of identifying more general principles underlying human decision-making. Moreover, integrating these insights into pHRI frameworks could improve safety, adaptability, and collaboration by aligning robot behavior more closely with human expectations and intent.

Across all three parts, this thesis established that data efficiency is not a limitation but a design axis, which can lead to more interpretable, robust, and human-aligned algorithms. By combining deep temporal modeling, expressive probabilistic learning, and structured IRL, we created a coherent set of tools for understanding human state, behavior, and intent in ways that support safe and intelligent human-robot interaction. Looking ahead, these methods open several paths for future research: integrating behavior-intent models into unified inference pipelines, deploying probabilistic behavior models in bilateral physical human-robot interactions, and extending IRL to online adaptive human-robot settings. Ultimately, the contributions of this thesis aim to bring robots closer to operating not just around humans, but truly with them, guided by an understanding of how humans feel, how they act, and why they choose their actions.

Appendix A

Ablation Study Results

Here we provide numerical results with respect to Figure 7 in Table A.1, and results with respect to Figure 8 in Tables A.2 and A.3.

Table A.1: Numerical comparison of *CSVS-Net*, *RSVS-Net*, *MLVS-Net*, and *nSHS-Net* for 3-24 hours of prediction horizons

Prediction Horizon (hours)	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**
Accuracy								
CSVS-Net	**0.913**	**0.910**	**0.911**	**0.906**	**0.903**	**0.904**	**0.901**	**0.898**
RSVS-Net	0.883	0.881	0.873	0.874	0.869	0.869	0.871	0.867
MLVS-Net	0.856	0.853	0.851	0.846	0.848	0.847	0.842	0.852
nSHS-Net	0.835	0.835	0.835	0.835	0.835	0.835	0.835	0.835
AUROC								
CSVS-Net	**0.934**	**0.922**	**0.910**	**0.898**	**0.897**	**0.897**	**0.884**	**0.884**
RSVS-Net	0.875	0.860	0.848	0.833	0.832	0.823	0.821	0.828
MLVS-Net	0.803	0.795	0.788	0.769	0.761	0.764	0.751	0.772
nSHS-Net	0.681	0.677	0.675	0.677	0.679	0.673	0.677	0.674
AUPRC								
CSVS-Net	**0.806**	**0.789**	**0.776**	**0.752**	**0.743**	**0.746**	**0.728**	**0.723**
RSVS-Net	0.657	0.649	0.609	0.594	0.591	0.579	0.578	0.581
MLVS-Net	0.502	0.486	0.468	0.435	0.441	0.443	0.426	0.469
nSHS-Net	0.278	0.277	0.275	0.269	0.275	0.268	0.271	0.270

Table A.2: Numerical comparison of all the occlusions on clinical and comorbidity characteristics and SEQ vital sign data for 3-24 hours of prediction horizons for *RSVS-Net*

Prediction Horizon (hours)	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**
Accuracy								
None	0.883	0.881	0.873	**0.874**	**0.869**	0.869	**0.871**	0.867
Sex	0.879	0.878	0.871	0.870	0.866	0.868	0.870	0.866
Obesity	0.882	0.881	**0.874**	0.873	**0.869**	0.869	**0.871**	0.866
Age	0.873	0.864	0.866	0.866	0.859	0.860	0.859	0.836
Diabetes	**0.883**	0.879	0.870	0.872	0.868	0.868	0.870	0.866
Hypertension	0.882	0.880	0.873	0.873	**0.869**	**0.869**	0.870	**0.868**
Vac. Time	0.882	**0.881**	0.873	0.874	0.868	0.868	0.870	0.867
Vac. Status	0.881	0.880	0.873	0.873	0.867	0.868	**0.871**	0.866
HR	0.855	0.853	0.853	0.856	0.852	0.850	0.852	0.853
SpO2	0.869	0.867	0.866	0.862	0.860	0.862	0.856	0.853
Temperature	0.880	0.868	0.865	0.865	0.862	0.860	0.863	0.860
AUROC								
None	0.875	**0.860**	0.848	**0.833**	**0.832**	**0.823**	0.821	0.828
Sex	0.874	0.855	0.843	0.828	0.827	0.816	0.813	0.821
Obesity	0.877	0.859	**0.848**	0.832	0.832	0.822	**0.821**	**0.828**
Age	0.845	0.813	0.833	0.809	0.785	0.766	0.785	0.760
Diabetes	0.876	0.857	0.846	0.828	0.827	0.812	0.816	0.820
Hypertension	0.877	0.859	0.847	0.831	0.830	0.820	0.819	0.825
Vac. Time	**0.878**	0.859	0.848	0.832	0.830	0.820	0.817	0.827
Vac. Status	0.876	0.857	0.846	0.830	0.828	0.820	0.819	0.824
HR	0.748	0.739	0.736	0.739	0.744	0.743	0.733	0.745
SpO2	0.858	0.833	0.819	0.792	0.790	0.785	0.777	0.771
Temperature	0.878	0.852	0.847	0.824	0.827	0.814	0.803	0.819
AUPRC								
None	0.657	**0.649**	**0.610**	**0.594**	0.591	**0.579**	**0.578**	**0.581**
Sex	0.659	0.642	0.604	0.588	0.583	0.571	0.570	0.572
Obesity	0.665	0.648	0.609	0.593	**0.592**	0.578	0.578	0.580
Age	0.617	0.557	0.583	0.560	0.518	0.505	0.529	0.480
Diabetes	0.661	0.645	0.606	0.590	0.586	0.565	0.571	0.571
Hypertension	0.664	0.648	0.609	0.593	0.591	0.577	0.578	0.578
Vac. Time	**0.664**	0.647	0.609	0.592	0.588	0.575	0.574	0.578
Vac. Status	0.661	0.643	0.606	0.589	0.584	0.574	0.573	0.575
HR	0.457	0.458	0.436	0.449	0.440	0.428	0.431	0.439
SpO2	0.615	0.588	0.561	0.533	0.526	0.524	0.495	0.485
Temperature	0.661	0.615	0.585	0.560	0.568	0.552	0.543	0.553

Table A.3: Numerical comparison of all the occlusions on clinical and comorbidity characteristics and SEQ vital sign data for 3-24 hours of prediction horizons for *CSVS-Net*

Prediction Horizon (hours)	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**
Accuracy								
None	0.913	0.910	0.911	0.906	0.903	**0.904**	0.901	0.898
Sex	0.912	0.909	0.909	**0.908**	**0.903**	0.903	0.900	**0.899**
Obesity	0.914	**0.911**	**0.911**	0.906	0.902	**0.904**	0.901	0.897
Age	0.903	0.906	0.902	0.897	0.899	0.898	0.896	0.895
Diabetes	0.913	0.909	0.908	0.903	0.902	0.901	0.899	0.894
Hypertension	0.914	0.910	0.910	0.905	**0.903**	**0.904**	**0.901**	0.898
Vac. Time	**0.915**	0.910	0.910	0.906	0.903	0.903	0.901	0.897
Vac. Status	0.913	0.910	0.910	0.905	0.902	0.901	0.900	0.896
HR	0.879	0.875	0.875	0.877	0.877	0.874	0.874	0.873
SpO2	0.901	0.892	0.895	0.891	0.885	0.882	0.880	0.880
Temperature	0.912	0.910	0.909	0.904	0.902	0.902	0.898	0.899
AUROC								
None	0.934	**0.922**	0.910	0.898	0.897	0.897	0.884	0.884
Sex	0.930	0.919	0.907	0.895	0.892	0.894	0.881	0.881
Obesity	0.933	0.921	0.910	0.898	0.896	**0.898**	**0.885**	0.885
Age	0.918	0.910	0.895	0.883	0.877	0.871	0.870	0.860
Diabetes	0.931	0.921	0.909	0.894	0.895	0.891	0.881	0.883
Hypertension	0.933	0.921	0.910	0.897	0.895	0.896	0.883	0.883
Vac. Time	0.932	0.920	0.909	0.897	0.895	0.896	0.882	0.881
Vac. Status	**0.934**	**0.922**	0.910	**0.898**	**0.897**	0.895	**0.885**	**0.885**
HR	0.776	0.772	0.774	0.777	0.773	0.775	0.770	0.770
SpO2	0.914	0.897	0.885	0.866	0.859	0.856	0.846	0.846
Temperature	0.932	0.920	**0.911**	0.897	0.894	0.892	0.880	0.879
AUPRC								
None	**0.806**	**0.789**	**0.776**	0.752	**0.743**	0.746	**0.728**	**0.723**
Sex	0.800	0.785	0.770	0.747	0.735	0.738	0.721	0.716
Obesity	0.804	0.788	0.775	0.753	0.743	**0.747**	0.727	0.723
Age	0.780	0.761	0.745	0.726	0.713	0.701	0.694	0.684
Diabetes	0.800	0.788	0.773	0.745	0.741	0.734	0.720	0.716
Hypertension	0.804	0.788	0.775	0.751	0.742	0.746	0.726	0.721
Vac. Time	0.805	0.786	0.772	0.750	0.742	0.745	0.728	0.719
Vac. Status	0.805	0.789	0.775	**0.753**	0.742	0.741	0.727	0.722
HR	0.567	0.561	0.565	0.564	0.559	0.559	0.550	0.552
SpO2	0.748	0.714	0.708	0.681	0.664	0.652	0.641	0.642
Temperature	0.805	0.784	0.774	0.744	0.735	0.738	0.720	0.717

Appendix B

Hierarchical Representation of the Multivariate Skew-t Distribution

In the MLE process, the key assumption is that the likelihood of each observation is independent of the likelihood of other observations. Therefore, to satisfy this assumption, the multivariate skew-t mixture model PDF in Eq. (3.1) is represented in the hierarchical format [101, 159] for the ERM algorithm, as seen in Eq. (3.3).

$$\begin{aligned}
&\boldsymbol{Y}_j|\boldsymbol{u}_j, w_j, z_{ij}=1 \sim N_p(\boldsymbol{\mu}_i+\boldsymbol{\Delta}_i\boldsymbol{u}_j, \frac{1}{w_j}\boldsymbol{\Sigma}_i)\\
&\boldsymbol{U}_j|w_j, z_{ij}=1 \sim HN_p(\boldsymbol{0}, \frac{1}{w_j}\boldsymbol{I}_p)\\
&W_j|z_{ij}=1 \sim \Gamma(\frac{\nu_i}{2}, \frac{\nu_i}{2})\\
&\boldsymbol{Z}_j \sim \mathcal{M}_g(1; \omega_1, \omega_2, \ldots, \omega_g)
\end{aligned} \tag{B.1}$$

In Eq. (B.1), g is the number of multivariate skew-t kernels in the mixture model, j subscript addresses the corresponding parameter to the j^{th} observed point in the dataset, such that $j=1,\ldots,n$, and i subscript addresses the parameter related to the i^{th} multivariate skew-t kernel, where $i=1,\ldots,g$. In addition, $HN_p(.,.)$ and $\Gamma(.,.)$ denote the p-dimensional half-normal, and Gamma distributions, respectively. For a multivariate skew-t mixture model with g clusters, a $g\times n$ matrix $\boldsymbol{Z}$ is defined to be associated with how much each cluster is responsible for each data point in the set, hence $\sum_{i=1}^{g} z_{ij}=1$. This means that if $z_{ij}=1$, $\boldsymbol{y}_j$ belongs solely to the cluster i. Utilizing the aforementioned definition, we can see that each random

vector $\boldsymbol{Z}_j = [z_{1j}, z_{2j}, \ldots, z_{gj}]^T$ is distributed by a multinomial distribution with one trial, i.e., $\boldsymbol{Z}_j \sim \mathcal{M}_g(1; \omega_1, \omega_2, \ldots, \omega_g)$, and the cell probabilities ω_i are the mixing proportions of the mixture model.

The hierarchical representation of the mixture model, utilizes random values that are unseen in the mathematical representation of the mixture model distribution in Eq. (3.1), namely $\boldsymbol{Z}$, $\boldsymbol{W}$, and $\boldsymbol{U}$. These parameters are the so-called latent variables of the mixture model, and the ERM algorithm relies on the computation of these latent variables.

By expanding Eq. (3.5), we can decompose the log-likelihood as seen in Eq. (B.2).

$$l(\boldsymbol{\Theta}|\boldsymbol{y}, \boldsymbol{u}, w, \boldsymbol{z}) = L_1(\boldsymbol{\Theta}|\boldsymbol{z}) + L_2(\boldsymbol{\Theta}|w, \boldsymbol{z}) + L_3(\boldsymbol{\Theta}|\boldsymbol{y}, \boldsymbol{u}, w, \boldsymbol{z}) \tag{B.2}$$

where each term can be re-written as a function of the defining parameters of the multivariate skew-t clusters in the mixture model and the latent variables of the hierarchical distribution function of the mixture model following Eqs. (B.3)-(B.5).

$$L_1(\boldsymbol{\Theta}|\boldsymbol{z}) = \sum_{i=1}^{g}\sum_{j=1}^{n} z_{ij}\log(\omega_i) \tag{B.3}$$

$$L_2(\boldsymbol{\Theta}|w, \boldsymbol{z}) = \sum_{i=1}^{g}\sum_{j=1}^{n} z_{ij}\Big\{(\frac{\nu_i}{2})\log(\frac{\nu_i}{2}) + (\frac{\nu_i}{2} + p - 1)\log(w_j) - \log\Gamma(\frac{\nu_i}{2}) - \frac{w_j}{2}\nu_i\Big\} \tag{B.4}$$

$$\begin{aligned} L_3(\boldsymbol{\Theta}|\boldsymbol{y}, \boldsymbol{u}, w, \boldsymbol{z}) = \sum_{i=1}^{g}\sum_{j=1}^{n} z_{ij}\Big\{ &- p\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}_i| \\ &- \frac{w_j}{2}\Big[d_i(\boldsymbol{y_j}) + (\boldsymbol{u}_j - \boldsymbol{c}_i(\boldsymbol{y}_j))^T\boldsymbol{\Lambda}_i^{-1}(\boldsymbol{u}_j - \boldsymbol{c}_i(\boldsymbol{y}_j))\Big]\Big\} \end{aligned} \tag{B.5}$$

The parameters seen in Eqs. (B.3)-(B.5). are derived using the expressions seen below:

$$\begin{aligned} d_i(\boldsymbol{y}_j) &= (\boldsymbol{y}_j - \boldsymbol{\mu}_i)^T\Omega_i^{-1}(\boldsymbol{y}_j - \boldsymbol{\mu}_i) \\ \boldsymbol{c}_i(\boldsymbol{y}_j) &= \boldsymbol{\Delta}_i^T\Omega_i^{-1}(\boldsymbol{y}_j - \boldsymbol{\mu}_i) \\ \boldsymbol{\Lambda}_i &= I_p - \boldsymbol{\Delta}_i^T\Omega_i^{-1}\boldsymbol{\Delta}_i \\ \boldsymbol{\Omega}_i &= \boldsymbol{\Sigma}_i + \boldsymbol{\Delta}_i\boldsymbol{\Delta}_i^T \end{aligned}$$

Appendix C

Conditional Expected Values of Truncated t-Distribution

In the E-step of the ERM algorithm, the conditional expected value of two random entities should be calculated, which are $\boldsymbol{e}_3$ and $\boldsymbol{e}_4$ as seen in Eq. (3.14) and Eq. (3.15). These expressions can be calculated as the product of e_1 and the first and second moments of the truncated t distribution of vector $\boldsymbol{U}$ respectively. For brevity, we denote the defining parameters of the truncated t distribution as $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\mathbb{A}$, and ν as seen in Eq. (C.1).

$$\boldsymbol{U} \sim tt_{p,\nu}(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \mathbb{A}) \tag{C.1}$$

where the truncated hyperplane is $\mathbb{A} = \{\boldsymbol{x} > \boldsymbol{a}, \boldsymbol{a} \in \mathbb{R}^p\}$. The truncation used in this PDF is left-truncation, meaning that each element of the vector $\boldsymbol{x}$ is greater than the associated element in the truncation vector $\boldsymbol{a}$. The density of this left truncated distribution is given by Eq. (C.2).

$$f_{\mathbb{A}}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, \nu) = T^{-1}_{p,\nu}(\boldsymbol{a}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) t_{p,\nu}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{x} \in \mathbb{A} \tag{C.2}$$

For evaluation of the first and second moments of this distribution, O'Hagan [135] first provided the explicit representation of the formulae based on moments of untruncated t-distribution in the univariate case of $tt_{p,\nu}(0, \sigma^2; a)$. Later O'Hagan [139] analyzed the multivariate case in the centralized case. Thereafter, Lee et al. [95] in 2014 generalized the representation of these moments to non-central

cases. Table C.1 shows the characteristics of certain notations used in the series of mathematical steps for computing the first and second moment of the truncated t-distribution.

For clarity of the equations mentioned subsequently, a brief description of mathematical notations used in the equations is provided in Table C.1. In this table, $\boldsymbol{x}$ is a $p \times 1$ vector and $\boldsymbol{X}$ is a $p \times p$ matrix.

Table C.1: Description of the notations

Notation	Type	Size	Description
x_i	Scalar	1×1	ith element of the vector $\boldsymbol{x}$
$\boldsymbol{x}_{-i}$	Vector	$(p-1) \times 1$	$\boldsymbol{x}$ without its ith element
$\boldsymbol{x}_{-ij}$	Vector	$(p-2) \times 1$	$\boldsymbol{x}$ without its ith and jth element
$\boldsymbol{X}_{ij}$	Matrix	2×2	$\boldsymbol{X}_{ij} = \begin{bmatrix} x_{ii} & x_{ij} \\ x_{ji} & x_{jj} \end{bmatrix}$
$\boldsymbol{X}_{-i}$	Matrix	$(p-1) \times (p-1)$	Matrix $\boldsymbol{X}$ without its ith row and column
$\boldsymbol{X}_{-ij}$	Matrix	$(p-2) \times (p-2)$	Matrix $\boldsymbol{X}$ without its ith and jth rows and columns
$\boldsymbol{X}_{(i)}$	Vector	$(p-1) \times 1$	Vector consisting ith column of matrix $\boldsymbol{X}$ without its ith element
$\boldsymbol{X}_{(ij)}$	Matrix	$(p-2) \times 2$	Matrix consisting ith and jth columns of matrix $\boldsymbol{X}$ without its ith and jth rows

$E(\boldsymbol{X})$ can be represented as the first moment of PDF in Eq. (C.3) with:

$$E(\boldsymbol{X}) = \boldsymbol{\mu} + \boldsymbol{\epsilon} \tag{C.3}$$

$$\boldsymbol{\epsilon} = c_1^{-1} \boldsymbol{\Sigma} \boldsymbol{\xi} \tag{C.4}$$

$$c_1 = T_{p,\nu}(\boldsymbol{\mu} - \boldsymbol{a} | \boldsymbol{0}, \boldsymbol{\Sigma}) \tag{C.5}$$

$$\xi_i = (2\pi\sigma_{ii})^{-\frac{1}{2}} \Big(\frac{\nu}{\nu + \sigma_{ii}^{-1}(\mu_i - a_i)^2} \Big)^{(\frac{\nu-1}{2})} \sqrt{\frac{\nu}{2}} \frac{\Gamma(\frac{\nu-1}{2})}{\Gamma(\frac{\nu}{2})} T_{p-1,\nu-1}(\boldsymbol{a}^\star | \boldsymbol{\mu}, \boldsymbol{\Sigma}^\star) \tag{C.6}$$

where

$$\boldsymbol{a}^{\star} = (\boldsymbol{\mu}_{-i} - \boldsymbol{a}_{-i}) - (\mu_i - a_i)\sigma_{ii}^{-1}\boldsymbol{\Sigma}_{(i)} \tag{C.7}$$

$$\boldsymbol{\Sigma}^{\star} = \Big(\frac{\nu + \sigma_{ii}^{-1}(\mu_i - a_i)^2}{\nu - 1}\Big)(\boldsymbol{\Sigma}_{-i} - \sigma_{ii}^{-1}\boldsymbol{\Sigma}_{(i)}\boldsymbol{\Sigma}_{(i)}^T) \tag{C.8}$$

However, For the second moment, the equations become more complex as seen in Eq. (C.9).

$$E(\boldsymbol{X}\boldsymbol{X}^T) = \boldsymbol{\mu}\boldsymbol{\mu}^T + \boldsymbol{\mu}\boldsymbol{\epsilon}^T + \boldsymbol{\epsilon}\boldsymbol{\mu}^T - c_1^{-1}\boldsymbol{\Sigma}\boldsymbol{H}\boldsymbol{\Sigma} + c_2 c_1^{-1}\big(\frac{\nu}{\nu - 2}\big)\boldsymbol{\Sigma} \tag{C.9}$$

$$c_2 = T_{p,\nu-2}\big(\boldsymbol{\mu} - \boldsymbol{a}|\boldsymbol{0}, \boldsymbol{\Sigma}\big) \tag{C.10}$$

where H is a $p \times p$ matrix for which we first calculate the off-diagonal elements as the following:

$$h_{ij} = \frac{1}{2\pi\sqrt{\sigma_{ii}\sigma_{jj} - \sigma_{ij}^2}}\big(\frac{\nu}{\nu - 2}\big)\big(\frac{\nu}{\nu^{\star}}\big)^{(\frac{\nu-2}{2})}T_{p-2,\nu-2}(\boldsymbol{a}^{\star\star}|\boldsymbol{0}, \boldsymbol{\Sigma}^{\star\star}), i \neq j \tag{C.11}$$

and the diagonal elements are calculated as such:

$$h_{ii} = \sigma_i^{-1} i\bigg((\mu_i - a_i)\xi_i - \sum_{i \neq j}\sigma_{ij}h_{ij}\bigg) \tag{C.12}$$

where

$$\nu^{\star} = \nu + (\boldsymbol{\mu}_{ij} - \boldsymbol{a}_{ij})^T\boldsymbol{\Sigma}_{ij}^{-1}(\boldsymbol{\mu}_{ij} - \boldsymbol{a}_{ij}) \tag{C.13}$$

$$\boldsymbol{a}^{\star\star} = (\boldsymbol{\mu}_{-ij} - \boldsymbol{a}_{-ij}) - \boldsymbol{\Sigma}_{(ij)}\boldsymbol{\Sigma}_{ij}^{-1}(\boldsymbol{\mu}_{ij} - \boldsymbol{a}_{ij}) \tag{C.14}$$

$$\boldsymbol{\Sigma}^{\star\star} = \frac{\nu^{\star}}{\nu - 2}(\boldsymbol{\Sigma}_{-ij} - \boldsymbol{\Sigma}_{(ij)}\boldsymbol{\Sigma}_{ij}^{-1}\boldsymbol{\Sigma}_{(ij)}^T) \tag{C.15}$$

Note that in the case of univariate datasets, $T_{p-1,v-1}(\boldsymbol{a}^{\star}|\boldsymbol{0}, \boldsymbol{\Sigma}^{\star}) = 1$, and in the case of univariate and bivariate datasets, $T_{p-2,v-2}(\boldsymbol{a}^{\star\star}|\boldsymbol{0}, \boldsymbol{\Sigma}^{\star\star}) = 1$.

Bibliography

[1] De-identified COVID-19 NYU Langone Database. https://med.nyu.edu/our-community/health-technology/events/covid-19-data-challenge. Accessed: 07-06-2023.

[2] De-identified COVID-19 NYU Langone Database. https://med.nyu.edu/our-community/health-technology/events/covid-19-data-challenge/dataset. Accessed: 07-06-2023.

[3] WHO. https://covid19.who.int. Accessed: 2022-10-04.

[4] N. Ab Azar, A. Shahmansoorian, and M. Davoudi. From inverse optimal control to inverse reinforcement learning: A historical review. *Annual Reviews in Control*, 50:119–138, 2020.

[5] S. Adams, T. Cody, and P. A. Beling. A survey of inverse reinforcement learning. *Artificial Intelligence Review*, 55(6):4307–4346, 2022.

[6] N. Aghasadeghi and T. Bretl. Maximum entropy inverse reinforcement learning in continuous state spaces with path integrals. In *2011 IEEE/RSJ international conference on intelligent robots and systems*, pages 1561–1566. IEEE, 2011.

[7] S. Ahuja, B. K. Panigrahi, N. Dey, V. Rajinikanth, and T. K. Gandhi. Deep transfer learning-based automated detection of covid-19 from lung ct scan slices. *Applied Intelligence*, 51(1):571–585, 2021.

[8] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[9] R. Akrour, M. Schoenauer, and M. Sebag. Preference-based policy learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2011, Athens, Greece, September 5-9, 2011. Proceedings, Part I 11*, pages 12–27. Springer, 2011.

[10] A. M. Alqudah. An enhanced method for real-time modelling of cardiac related biosignals using gaussian mixtures. *Journal of medical engineering & technology*, 41(8):600–611, 2017.

[11] M. Alzubaidi, H. D. Zubaydi, A. A. Bin-Salem, A. A. Abd-Alrazaq, A. Ahmed, and M. Househ. Role of deep learning in early detection of covid-19: Scoping review. *Computer methods and programs in biomedicine update*, 1:100025, 2021.

[12] E. Angeli, S. Dalto, S. Marchese, L. Setti, M. Bonacina, F. Galli, E. Rulli, V. Torri, C. Monti, R. Meroni, et al. Prognostic value of ct integrated with clinical and laboratory data during the first peak of the covid-19 pandemic in northern italy: A nomogram to predict unfavorable outcome. *European journal of radiology*, 137:109612, 2021.

[13] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[14] D. Assaf, Y. Gutman, Y. Neuman, G. Segal, S. Amit, S. Gefen-Halevi, N. Shilo, A. Epstein, R. Mor-Cohen, A. Biber, et al. Utilization of machine-learning models to accurately predict the risk for critical covid-19. *Internal and emergency medicine*, 15(8):1435–1443, 2020.

[15] D. Assaf, Y. Gutman, Y. Neuman, G. Segal, S. Amit, S. Gefen-Halevi, N. Shilo, A. Epstein, R. Mor-Cohen, A. Biber, et al. Utilization of machine-learning models to accurately predict the risk for critical covid-19. *Internal and emergency medicine*, 15(8):1435–1443, 2020.

[16] C. G. Atkeson and J. M. Hollerbach. Kinematic features of unrestrained vertical arm movements. *J. Neurosci.*, 5:2318–2330, 1985.

[17] A. Azzalini and A. Capitanio. Distributions generated by perturbation of symmetry with emphasis on a multivariate skew t-distribution. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(2):367–389, 2003.

[18] S. Bai, J. Z. Kolter, and V. Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[19] A. Basharat, A. Gritai, and M. Shah. Learning object motion patterns for anomaly detection and improved object detection. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[20] F. Bečanović, V. Bonnet, R. Dumas, K. Jovanović, and S. Mohammed. Force sharing problem during gait using inverse optimal control. *IEEE Robotics and Automation Letters*, 8(2):872–879, 2022.

[21] F. Bečanović, J. Miller, V. Bonnet, K. Jovanović, and S. Mohammed. Assessing the quality of a set of basis functions for inverse optimal control via projection onto global minimizers. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 7598–7605. IEEE, 2022.

[22] S. Ben-Itzhak and A. Karniel. Minimum acceleration criterion with constraints implies bang-bang control as an underlying principle for optimal trajectories of arm reaching movements. *Neural Comput.*, 20:779–812, 2008.

[23] N. Bernstein. The coordination and regulation of movements. *(No Title)*, 1967.

[24] B. Berret, E. Chiovetto, F. Nori, and T. Pozzo. Evidence for composite cost functions in arm movement planning: an inverse optimal control approach. *PLoS Comput. Biol.*, 7(10):e1002183, 2011.

[25] B. Berret, C. Darlot, F. Jean, T. Pozzo, C. Papaxanthis, and J. P. Gauthier. The inactivation principle: mathematical solutions minimizing the absolute work and biological implications for the planning of arm movements. *PLoS Comput. Biol.*, 4:e1000194, 2008.

[26] A. Biess, D. G. Liebermann, and T. Flash. A computational model for redundant human three-dimensional pointing movements: integration of independent spatial and temporal motor plans simplifies movement dynamics. *J. Neurosci.*, 27:13045–13064, 2007.

[27] A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Survey: Robot programming by demonstration. Technical report, Springrer, 2008.

[28] H. Bitar and S. Alismail. The role of ehealth, telehealth, and telemedicine for chronic disease patients during covid-19 pandemic: A rapid systematic review. *Digital health*, 7:20552076211009396, 2021.

[29] S. Bolourani, M. Brenner, P. Wang, T. McGinn, J. S. Hirsch, D. Barnaby, T. P. Zanos, N. C.-. R. Consortium, et al. A machine learning prediction model of respiratory failure within 48 hours of patient admission for covid-19: model development and validation. *Journal of medical Internet research*, 23(2):e24246, 2021.

[30] A. Boularias, J. Kober, and J. Peters. Relative entropy inverse reinforcement learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 182–189. JMLR Workshop and Conference Proceedings, 2011.

[31] D. Brown, W. Goo, P. Nagarajan, and S. Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pages 783–792. PMLR, 2019.

[32] R. P. Browne and P. D. McNicholas. A mixture of generalized hyperbolic distributions. *Canadian Journal of Statistics*, 43(2):176–198, 2015.

[33] R. P. Browne, P. D. McNicholas, and M. D. Sparling. Model-based learning using a mixture of mixtures of gaussian and uniform distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):814–817, 2011.

[34] A. Byravan, M. Monfort, B. D. Ziebart, B. Boots, and D. Fox. Graph-based inverse optimal control for robot manipulation. In *Ijcai*, volume 15, pages 1874–1890, 2015.

[35] C. R. B. Cabral, V. H. Lachos, and M. O. Prates. Multivariate mixture modeling using skew-normal independent distributions. *Computational Statistics & Data Analysis*, 56(1):126–142, 2012.

[36] C. R. B. Cabral, V. H. Lachos, and M. O. Prates. Multivariate mixture modeling using skew-normal independent distributions. *Computational Statistics & Data Analysis*, 56(1):126–142, 2012.

[37] S. Calinon. A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics*, 9(1):1–29, 1 jan 2016.

[38] S. Cao, Z. Luo, and C. Quan. Online inverse optimal control for time-varying cost weights. *Biomimetics*, 9(2):84, 2024.

[39] J. Carpentier, G. Saurel, G. Buondonno, J. Mirabel, F. Lamiraux, O. Stasse, and N. Mansard. The pinocchio c++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *2019 IEEE/SICE International Symposium on System Integration (SII)*, pages 614–619. IEEE, 2019.

[40] M. Chang and S. Gupta. One-shot visual imitation via attributed waypoints and demonstration augmentation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5055–5062, 2023.

[41] G. Chen and S. Luo. Robust bayesian hierarchical model using normal/independent distributions. *Biometrical Journal*, 58(4):831–851, 2016.

[42] S. Chernova and A. L. Thomaz. *Robot learning from human teachers*. Morgan & Claypool Publishers, 2014.

[43] M. A. Cifci. Deep learning model for diagnosis of corona virus disease from ct images. *Int. J. Sci. Eng. Res*, 11(4):273–278, 2020.

[44] J. P. Cohen, L. Dao, K. Roth, P. Morrison, Y. Bengio, A. F. Abbasi, B. Shen, H. K. Mahsa, M. Ghassemi, H. Li, et al. Predicting covid-19 pneumonia severity on chest x-ray with deep learning. *Cureus*, 12(7), 2020.

[45] J. Colombel, D. Daney, and F. Charpillet. On the reliability of inverse optimal control. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8504–8510. IEEE, 2022.

[46] R. D. Cook and S. Weisberg. *An introduction to regression graphics*, volume 405. John Wiley & Sons, 2009.

[47] A. Das, U. R. Acharya, S. S. Panda, and S. Sabut. Deep learning based liver cancer detection using watershed transform and gaussian mixture model techniques. *Cognitive Systems Research*, 54:165–175, 2019.

[48] A. F. de Moraes Batista, J. L. Miraglia, T. H. R. Donato, and A. D. P. Chiavegatto Filho. Covid-19 diagnosis prediction in emergency care patients: a machine learning approach. *MedRxiv*, 2020.

[49] M. P. Deisenroth, D. Fox, and C. E. Rasmussen. Gaussian processes for data-efficient learning in robotics and control. *IEEE transactions on pattern analysis and machine intelligence*, 37(2):408–423, 2013.

[50] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

[51] J. Dideriksen and F. Negro. Feedforward modulation of gamma motor neuron activity can improve motor command accuracy. *Journal of Neural Engineering*, 18(4):046068, 2021.

[52] R. Dumas, L. Cheze, and J.-P. Verriest. Adjustments to mcconville et al. and young et al. body segment inertial parameters. *J. Biomech.*, 40:543–553, 2007.

[53] D. A. Duque, F. A. Prieto, and J. G. Hoyos. Trajectory generation for robotic assembly operations using learning by demonstration. *Robotics and Computer-Integrated Manufacturing*, 57:292–302, 2019.

[54] D. A. Duque, F. A. Prieto, and J. G. Hoyos. Trajectory generation for robotic assembly operations using learning by demonstration. *Robotics and Computer-Integrated Manufacturing*, 57:292–302, 2019.

[55] H. Estiri, Z. H. Strasser, J. G. Klann, P. Naseri, K. B. Wagholikar, and S. N. Murphy. Predicting covid-19 mortality with electronic medical records. *NPJ digital medicine*, 4(1):1–10, 2021.

[56] J. Fan, K. Zhang, Y. Huang, Y. Zhu, and B. Chen. Parallel spatio-temporal attention-based tcn for multivariate time series prediction. *Neural Computing and Applications*, pages 1–10, 2021.

[57] W. Fan, L. Yang, and N. Bouguila. Unsupervised grouped axial data modeling via hierarchical bayesian nonparametric models with watson distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9654–9668, 2021.

[58] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58. PMLR, 2016.

[59] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58. PMLR, 2016.

[60] T. Flash and N. Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *J. Neurosci.*, 5:1688–1703, 1985.

[61] S. Frühwirth-Schnatter and S. Pyne. Bayesian inference for finite mixtures of univariate and multivariate skew-normal and skew-t distributions. *Biostatistics*, 11(2):317–336, 2010.

[62] M. Gasior, J. Jaroszewicz, K. Wita, D. Cieśla, and B. Hudzik. High post-discharge mortality in hospitalized covid-19 patients with cardiovascular comorbidities. *Pol. Arch. Intern. Med*, 131(7-8):749–751, 2021.

[63] A. Genz and F. Bretz. Comparison of methods for the computation of multivariate t probabilities. *Journal of Computational and Graphical Statistics*, 11(4):950–971, 2002.

[64] P. J. Green. On use of the em algorithm for penalized likelihood estimation. *Journal of the Royal Statistical Society: Series B (Methodological)*, 52(3):443–452, 1990.

[65] P. J. Green. On use of the em algorithm for penalized likelihood estimation. *Journal of the Royal Statistical Society: Series B (Methodological)*, 52(3):443–452, 1990.

[66] C. M. Harris and D. M. Wolpert. Signal-dependent noise determines motor planning. *Nature*, 394(6695):780–784, 1998.

[67] J. A. Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[68] S. Hassantabar, M. Ahmadi, and A. Sharifi. Diagnosis and detection of infected tissue of covid-19 patients based on lung x-ray image using convolutional neural network approaches. *Chaos, Solitons & Fractals*, 140:110170, 2020.

[69] Y. He and J. Zhao. Temporal convolutional networks for anomaly detection in time series. In *Journal of Physics: Conference Series*, volume 1213, page 042050. IOP Publishing, 2019.

[70] E. E.-D. Hemdan, M. A. Shouman, and M. E. Karar. Covidx-net: A framework of deep learning classifiers to diagnose covid-19 in x-ray images. *arXiv preprint arXiv:2003.11055*, 2020.

[71] H. J. Ho, T.-I. Lin, H.-Y. Chen, and W.-L. Wang. Some results on the truncated multivariate t distribution. *Journal of Statistical Planning and Inference*, 142(1):25–40, 2012.

[72] H. J. Ho, T.-I. Lin, H.-Y. Chen, and W.-L. Wang. Some results on the truncated multivariate t distribution. *Journal of Statistical Planning and Inference*, 142(1):25–40, 2012.

[73] Y. Hong, M. Kim, H. Lee, J. J. Park, and D. Lee. Early fault diagnosis and classification of ball bearing using enhanced kurtogram and gaussian mixture model. *IEEE Transactions on Instrumentation and Measurement*, 68(12):4746–4755, 2019.

[74] Y. Huang, L. Rozo, J. Silvério, and D. G. Caldwell. Kernelized movement primitives. *The International Journal of Robotics Research*, 38(7):833–852, 2019.

[75] E. T. Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

[76] Z. Jiang, M. Hu, Z. Gao, L. Fan, R. Dai, Y. Pan, W. Tang, G. Zhai, and Y. Lu. Detection of respiratory infections using rgb-infrared sensors on portable device. *IEEE Sensors Journal*, 20(22):13674–13681, 2020.

[77] A. Jordana, S. Kleff, A. Meduri, J. Carpentier, N. Mansard, and L. Righetti. Stagewise implementations of sequential quadratic programming for model-predictive control. *Subm. IEEE TRO*, 2023.

[78] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *2011 IEEE international conference on robotics and automation*, pages 4569–4574. IEEE, 2011.

[79] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *2011 IEEE international conference on robotics and automation*, pages 4569–4574. IEEE, 2011.

[80] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal. Learning objective functions for manipulation. In *2013 IEEE International Conference on Robotics and Automation*, pages 1331–1336. IEEE, 2013.

[81] M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal. Learning objective functions for manipulation. In *2013 IEEE International Conference on Robotics and Automation*, pages 1331–1336. IEEE, 2013.

[82] M. Kalakrishnan, E. Theodorou, and S. Schaal. Inverse reinforcement learning with pi 2. In *The Snowbird Workshop, submitted to*. Citeseer, 2010.

[83] M. Karami and L. Wang. Fault detection and diagnosis for nonlinear systems: A new adaptive gaussian mixture modeling approach. *Energy and Buildings*, 166:477–488, 2018.

[84] B. Keller, M. Draelos, K. Zhou, R. Qian, A. N. Kuo, G. Konidaris, K. Hauser, and J. A. Izatt. Optical coherence tomography-guided robotic ophthalmic microsurgery via reinforcement learning from demonstration. *IEEE Transactions on Robotics*, 36(4):1207–1218, 2020.

[85] A. I. Khan, J. L. Shah, and M. M. Bhat. Coronet: A deep neural network for detection and diagnosis of covid-19 from chest x-ray images. *Computer methods and programs in biomedicine*, 196:105581, 2020.

[86] A. M. Khan, H. El-Daly, and N. M. Rajpoot. A gamma-gaussian mixture model for detection of mitotic cells in breast cancer histopathology images. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 149–152. IEEE, 2012.

[87] S. M. Khansari-Zadeh and A. Billard. Learning stable nonlinear dynamical systems with gaussian mixture models. *IEEE Transactions on Robotics*, 27(5):943–957, 2011.

[88] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[89] H. Ko, H. Chung, W. S. Kang, K. W. Kim, Y. Shin, S. J. Kang, J. H. Lee, Y. J. Kim, N. Y. Kim, H. Jung, et al. Covid-19 pneumonia diagnosis using a simple 2d deep learning framework with a single chest ct image: model development and validation. *Journal of medical Internet research*, 22(6):e19569, 2020.

[90] D. Kulić and Y. Nakamura. Incremental learning and memory consolidation of whole body human motion primitives. *Adaptive Behavior*, 17(6):484–507, 2009.

[91] C. Lauretti, F. Cordella, E. Guglielmelli, and L. Zollo. Learning by demonstration for planning activities of daily living in rehabilitation and assistive robotics. *IEEE Robotics and Automation Letters*, 2(3):1375–1382, 2017.

[92] S. Lee and G. J. McLachlan. On the fitting of mixtures of multivariate skew t-distributions via the em algorithm. *arXiv preprint arXiv:1109.4706*, 2011.

[93] S. Lee and G. J. McLachlan. On the fitting of mixtures of multivariate skew t-distributions via the em algorithm. *arXiv preprint arXiv:1109.4706*, 2011.

[94] S. Lee and G. J. McLachlan. Finite mixtures of multivariate skew t-distributions: some recent and new results. *Statistics and Computing*, 24(2):181–202, 2014.

[95] S. Lee and G. J. McLachlan. Finite mixtures of multivariate skew t-distributions: some recent and new results. *Statistics and Computing*, 24(2):181–202, 2014.

[96] S. X. Lee and G. J. McLachlan. Emmix-uskew: an r package for fitting mixtures of multivariate skew t-distributions via the em algorithm. *arXiv preprint arXiv:1211.5290*, 2012.

[97] S. X. Lee and G. J. McLachlan. Finite mixures of canonical fundamenal skew $$$$-disribuions. *Statistics and computing*, 26(3):573–589, 2016.

[98] S. X. Lee and G. J. McLachlan. Finite mixures of canonical fundamenal skew $$$$-disribuions. *Statistics and computing*, 26(3):573–589, 2016.

[99] Y. Lee, Y.-S. Kim, D.-i. Lee, S. Jeong, G.-H. Kang, Y. S. Jang, W. Kim, H. Y. Choi, J. G. Kim, and S.-h. Choi. The application of a deep learning system developed to reduce the time for rt-pcr in covid-19 detection. *Scientific reports*, 12(1):1–10, 2022.

[100] S. Levine and V. Koltun. Continuous inverse optimal control with locally optimal examples. *arXiv preprint arXiv:1206.4617*, 2012.

[101] R. Levy. Probabilistic models in the study of language. *Online Draft, Nov*, 2012.

[102] M. X. Li, O. Celik, P. Becker, D. Blessing, R. Lioutikov, and G. Neumann. Curriculum-based imitation of versatile skills. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2951–2957, 2023.

[103] Y. Li, Z. Wang, T. Zhao, and S. Wanqing. Research on a pattern recognition method of cyclic gmm-fcm based on joint time-domain features. *IEEE Access*, 9:1904–1917, 2020.

[104] Z. Li, S. Zhao, Y. Chen, F. Luo, Z. Kang, S. Cai, W. Zhao, J. Liu, D. Zhao, and Y. Li. A deep-learning-based framework for severity assessment of covid-19 with ct images. *Expert Systems with Applications*, 185:115616, 2021.

[105] J. F.-S. Lin, P. Carreno-Medrano, M. Parsapour, M. Sakr, and D. Kulić. Objective learning from human demonstrations. *Annu. Rev. Control.*, 51:111–129, 2021.

[106] T. I. Lin. Maximum likelihood estimation for multivariate skew normal mixture models. *Journal of Multivariate Analysis*, 100(2):257–265, 2009.

[107] T. I. Lin. Maximum likelihood estimation for multivariate skew normal mixture models. *Journal of Multivariate Analysis*, 100(2):257–265, 2009.

[108] T.-I. Lin. Robust mixture modeling using multivariate skew t distributions. *Statistics and Computing*, 20(3):343–356, 2010.

[109] T.-I. Lin. Robust mixture modeling using multivariate skew t distributions. *Statistics and Computing*, 20(3):343–356, 2010.

[110] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[111] W. Liu, J. Zhong, R. Wu, B. L. Fylstra, J. Si, and H. H. Huang. Inferring human-robot performance objectives during locomotion using inverse reinforcement learning and inverse optimal control. *IEEE Robotics and Automation Letters*, 7(2):2549–2556, 2022.

[112] Z. Liu, L. Yu, J. H. Hsiao, and A. B. Chan. Primal-gmm: Parametric manifold learning of gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(6):3197–3211, 2021.

[113] M. Loey, F. Smarandache, and N. E. M. Khalifa. Within the lack of chest covid-19 x-ray dataset: a novel detection model based on gan and deep transfer learning. *Symmetry*, 12(4):651, 2020.

[114] M. Maaref, A. Rezazadeh, K. Shamaei, R. Ocampo, and T. Mahdi. A bicycle cranking model for assist-as-needed robotic rehabilitation therapy using learning from demonstration. *IEEE Robotics and Automation Letters*, 1(2):653–660, 2016.

[115] G. J. Maeda, G. Neumann, M. Ewerton, R. Lioutikov, O. Kroemer, and J. Peters. Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks. *Autonomous Robots*, 41:593–612, 2017.

[116] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[117] C. Mastalli, R. Budhiraja, W. Merkt, G. Saurel, B. Hammoud, M. Naveau, J. Carpentier, L. Righetti, S. Vijayakumar, and N. Mansard. Crocoddyl: An efficient and versatile framework for multi-contact optimal control. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2536–2542. IEEE, 2020.

[118] H. Mayer, I. Nagy, and A. Knoll. Skill transfer and learning by demonstration in a realistic scenario of laparoscopic surgery. In *Proceedings of the IEEE International Conference on Humanoids CD-ROM*, 2003.

[119] G. J. McLachlan and T. Krishnan. *The EM algorithm and extensions*. John Wiley & Sons, 2007.

[120] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake. Finite mixture models. *Annual review of statistics and its application*, 6:355–378, 2019.

[121] G. J. McLachlan, S. X. Lee, and S. I. Rathnayake. Finite mixture models. *Annual review of statistics and its application*, 6:355–378, 2019.

[122] G. J. McLachlan and D. Peel. Robust cluster analysis via mixtures of multivariate t-distributions. In *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*, pages 658–666. Springer, 1998.

[123] S. Mehrdad, A. Meduri, and L. Righetti. Cost function estimation using inverse reinforcement learning with minimal observations, 2025. Submitted to IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2025.

[124] X. Mei, H.-C. Lee, K.-y. Diao, M. Huang, B. Lin, C. Liu, Z. Xie, Y. Ma, P. M. Robson, M. Chung, et al. Artificial intelligence–enabled rapid diagnosis of patients with covid-19. *Nature medicine*, 26(8):1224–1228, 2020.

[125] K. Mombaur, A. Truong, and J.-P. Laumond. From human to humanoid locomotion—an inverse optimal control approach. *Autonomous robots*, 28:369–383, 2010.

[126] M. Najafi, M. Sharifi, K. Adams, and M. Tavakoli. Robotic assistance for children with cerebral palsy based on learning from tele-cooperative demonstration. *International Journal of Intelligent Robotics and Applications*, 1(1):43–54, 2017.

[127] M. Najafi, M. Sharifi, K. Adams, and M. Tavakoli. Robotic assistance for children with cerebral palsy based on learning from tele-cooperative demonstration. *International Journal of Intelligent Robotics and Applications*, 1(1):43–54, 2017.

[128] E. Nakano, H. Imamizu, R. Osu, Y. Uno, H. Gomi, T. Yoshioka, and M. Kawato. Quantitative examinations of internal representations for arm trajectory planning: minimum commanded torque change model. *J. Neurophysiol.*, 81:2140–2155, 1999.

[129] W. L. Nelson. Physical principles for economies of skilled movements. *Biol. Cybern.*, 46:135–147, 1983.

[130] A. Y. Ng, S. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[131] Y. Nguyen, F. Corre, V. Honsel, S. Curac, V. Zarrouk, C. P. Burtz, E. Weiss, J.-D. Moyer, T. Gauss, J. Gregory, et al. A nomogram to predict the risk of unfavourable outcome in covid-19: a retrospective cohort of 279 hospitalized patients in paris area. *Annals of medicine*, 52(7):367–375, 2020.

[132] J. Nishii. Energetic optimicality of arm trajectory. In *Proc. Int. Conf. on Biomechanics of Man, 2002*, 2002.

[133] K. Novak, L. Miller, and J. Houk. The use of overlapping submovements in the control of rapid hand movements. *Experimental brain research*, 144(3):351–364, 2002.

[134] Y. Oh, S. Park, and J. C. Ye. Deep learning covid-19 features on cxr using limited training data sets. *IEEE transactions on medical imaging*, 39(8):2688–2700, 2020.

[135] A. O'hagan. Bayes estimation of a convex quadratic. *Biometrika*, 60(3):565–571, 1973.

[136] R. O'Keeffe, Y. Shirazi, S. Mehrdad, T. Crosby, A. M. Johnson, and S. F. Atashzar. Perilaryngeal functional muscle network in patients with vocal hyperfunction-a case study. *bioRxiv*, pages 2023–01, 2023.

[137] S. Omboni, R. S. Padwal, T. Alessa, B. Benczúr, B. B. Green, I. Hubbard, K. Kario, N. A. Khan, A. Konradi, A. G. Logan, et al. The worldwide impact of telemedicine during covid-19: current evidence and recommendations for the future. *Connected health*, 1:7, 2022.

[138] Ş. Öztürk, U. Özkaya, and M. Barstuğan. Classification of coronavirus (covid-19) from x-ray and ct images using shrunken features. *International Journal of Imaging Systems and Technology*, 31(1):5–15, 2021.

[139] A. O'Hagan. Moments of the truncated multivariatet distribution, 1976.

[140] R. O'Keeffe, S. Y. Shirazi, S. Mehrdad, T. Crosby, A. M. Johnson, and S. F. Atashzar. Perilaryngeal-cranial functional muscle network differentiates vocal tasks: A multi-channel semg approach. *IEEE Transactions on Biomedical Engineering*, 69(12):3678–3688, 2022.

[141] A. M. Panchea, N. Ramdani, V. Bonnet, and P. Fraisse. Human arm motion analysis based on the inverse optimization approach. In *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob)*, pages 1005–1010. IEEE, 2018.

[142] A. Paraschos, C. Daniel, J. Peters, and G. Neumann. Using probabilistic movement primitives in robotics. *Autonomous Robots*, 42:529–551, 2018.

[143] Y. Pathak, P. K. Shukla, A. Tiwari, S. Stalin, and S. Singh. Deep transfer learning based classification model for covid-19 disease. *Irbm*, 2020.

[144] C. J. Pérez-del Pulgar, J. Smisek, I. Rivas-Blanco, A. Schiele, and V. F. Muñoz. Using gaussian mixture models for gesture recognition during haptically guided telemanipulation. *Electronics*, 8(7):772, 2019.

[145] A. Pervez, A. Ali, J.-H. Ryu, and D. Lee. Novel learning from demonstration approach for repetitive teleoperation tasks. In *2017 IEEE World Haptics Conference (WHC)*, pages 60–65. IEEE, 2017.

[146] E. Pignat and S. Calinon. Bayesian gaussian mixture model for robotic policy imitation. *IEEE Robotics and Automation Letters*, 4(4):4452–4458, 2019.

[147] S. Pyne, X. Hu, K. Wang, E. Rossin, T.-I. Lin, L. M. Maier, C. Baecher-Allan, G. J. McLachlan, P. Tamayo, D. A. Hafler, et al. Automated high-dimensional flow cytometric data analysis. *Proceedings of the National Academy of Sciences*, 106(21):8519–8524, 2009.

[148] S. Pyne, X. Hu, K. Wang, E. Rossin, T.-I. Lin, L. M. Maier, C. Baecher-Allan, G. J. McLachlan, P. Tamayo, D. A. Hafler, et al. Automated high-dimensional flow cytometric data analysis. *Proceedings of the National Academy of Sciences*, 106(21):8519–8524, 2009.

[149] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich. Maximum margin planning. In *Proceedings of the 23rd international conference on Machine learning*, pages 729–736, 2006.

[150] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard. Recent advances in robot learning from demonstration. *Annual review of control, robotics, and autonomous systems*, 3:297–330, 2020.

[151] P. Remy. Temporal convolutional networks for keras. https://github.com/philipperemy/keras-tcn, 2020.

[152] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn. Speaker verification using adapted gaussian mixture models. *Digital signal processing*, 10(1-3):19–41, 2000.

[153] D. A. Reynolds and R. C. Rose. Robust text-independent speaker identification using gaussian mixture speaker models. *IEEE transactions on speech and audio processing*, 3(1):72–83, 1995.

[154] H. Ritchie, E. Mathieu, L. Rodés-Guirao, C. Appel, C. Giattino, E. Ortiz-Ospina, J. Hasell, B. Macdonald, D. Beltekian, and M. Roser. Coronavirus pandemic (covid-19). *Our World in Data*, 2020. https://ourworldindata.org/coronavirus.

[155] R. C. Rose and D. A. Reynolds. Text independent speaker identification using automatic acoustic segmentation. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 293–296. IEEE, 1990.

[156] L. D. Rozo, P. Jiménez, and C. Torras. Learning force-based robot skills from haptic demonstration. In *CCIA*, pages 331–340, 2010.

[157] L. Rozo Castañeda, P. Jimenez Schlegl, and C. Torras. Sharpening haptic inputs for teaching a manipulation skill to a robot. In *1st IEEE International Conference on Applied Bionics and Biomechanics*, pages 331–340, 2010.

[158] M. Sabbah, F. Bečanović, S. Mehrdad, L. Righetti, B. Watier, and V. Bonnet. Minimal observations inverse reinforcement learning for predicting human

box-lifting motions. In *2025 IEEE-RAS 24th International Conference on Humanoid Robots (Humanoids)*, pages 491–498. IEEE, 2025.

[159] S. K. Sahu, D. K. Dey, and M. D. Branco. A new class of multivariate skew distributions with applications to bayesian regression models. *Canadian Journal of Statistics*, 31(2):129–150, 2003.

[160] A. Salama, A. Darwsih, and A. E. Hassanien. Artificial intelligence approach to predict the covid-19 patient's recovery. In *Digital Transformation and Emerging Technologies for Fighting COVID-19 Pandemic: Innovative Approaches*, pages 121–133. Springer, 2021.

[161] A. W. Salehi, P. Baglat, and G. Gupta. Review on machine and deep learning models for the detection and prediction of coronavirus. *Materials Today: Proceedings*, 33:3896–3901, 2020.

[162] K. Santosh, D. Das, and U. Pal. Truncated inception net: Covid-19 outbreak screening using chest x-rays. *PREPRINT (Version 1) available at Research Square*, 3, 2020.

[163] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3(6):233–242, 1999.

[164] M. Schneider and W. Ertel. Robot learning by demonstration with local gaussian process regression. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 255–260. IEEE, 2010.

[165] A. Sedik, A. M. Iliyasu, A. El-Rahiem, M. E. Abdel Samea, A. Abdel-Raheem, M. Hammad, J. Peng, A. El-Samie, E. Fathi, A. El-Latif, et al. Deploying machine and deep learning models for efficient data-augmented detection of covid-19 infections. *Viruses*, 12(7):769, 2020.

[166] D. P. K. Sethy and S. K. Behera. Detection of coronavirus disease (covid-19) based on deep features. 2020.

[167] F. Shahid, A. Zameer, and M. Muneeb. Predictions for covid-19 with deep learning models of lstm, gru and bi-lstm. *Chaos, Solitons & Fractals*, 140:110212, 2020.

[168] F. E. Shamout, Y. Shen, N. Wu, A. Kaku, J. Park, T. Makino, S. Jastrzebski, J. Witowski, D. Wang, B. Zhang, et al. An artificial intelligence system for predicting the deterioration of covid-19 patients in the emergency department. *NPJ digital medicine*, 4(1):1–11, 2021.

[169] A. Shek, B. Y. Su, R. Chen, and C. Liu. Learning from physical human feedback: An object-centric one-shot adaptation method. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9910–9916, 2023.

[170] J. Silvério, Y. Huang, F. J. Abu-Dakka, L. Rozo, and D. G. Caldwell. Uncertainty-aware imitation learning using kernelized movement primitives. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 90–97. IEEE, 2019.

[171] P. Spurek. General split gaussian cross–entropy clustering. *Expert Systems with Applications*, 68:58–68, 2017.

[172] H. Sugiyama, T. Meguro, and Y. Minami. Preference-learning based inverse reinforcement learning for dialog control. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[173] P. Sun, X. Lu, C. Xu, W. Sun, and B. Pan. Understanding of covid-19 based on current evidence. *Journal of medical virology*, 92(6):548–551, 2020.

[174] T. Sun, Q. Hu, P. Gulati, and S. F. Atashzar. Temporal dilation of deep lstm for agile decoding of semg: Application in prediction of upper-limb motor intention in neurorobotics. *IEEE Robotics and Automation Letters*, 6(4):6212–6219, 2021.

[175] A. Supady, J. R. Curtis, D. Abrams, R. Lorusso, T. Bein, J. Boldt, C. E. Brown, D. Duerschmied, V. Metaxa, and D. Brodie. Allocating scarce intensive care resources during the covid-19 pandemic: practical challenges to theoretical frameworks. *The Lancet Respiratory Medicine*, 9(4):430–434, 2021.

[176] N. Sylla, V. Bonnet, G. Venture, N. Armande, and P. Fraisse. Human arm optimal motion analysis in industrial screwing task. In *5th IEEE RAS/EMBS international conference on biomedical robotics and biomechatronics*, pages 964–969. IEEE, 2014.

[177] N. Sylla, V. Bonnet, G. Venture, N. Armande, and P. Fraisse. Human arm optimal motion analysis in industrial screwing task. In *5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 964–969, 2014.

[178] V. Tangkaratt, B. Han, M. E. Khan, and M. Sugiyama. Variational imitation learning with diverse-quality demonstrations. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9407–9417, 2020.

[179] Z. M. Temesgen, D. C. DeSimone, M. Mahmood, C. R. Libertin, B. R. V. Palraj, and E. F. Berbari. Health care after the covid-19 pandemic and the influence of telemedicine. In *Mayo Clinic Proceedings*, volume 95, pages S66–S68. Elsevier, 2020.

[180] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[181] S. Toraman, T. B. Alakus, and I. Turkoglu. Convolutional capsnet: A novel artificial neural network approach to detect covid-19 disease from x-ray images using capsule networks. *Chaos, Solitons & Fractals*, 140:110122, 2020.

[182] J. F. Torres, D. Hadjout, A. Sebaa, F. Martínez-Álvarez, and A. Troncoso. Deep learning for time series forecasting: a survey. *Big Data*, 9(1):3–21, 2021.

[183] Y. Uno, M. Kawato, and R. Suzuki. Formation and control of optimal trajectory in human multijoint arm movement. *Biol. Cybern.*, 61:89–101, 1989.

[184] I. Vrbik and P. McNicholas. Analytic calculations for the em algorithm for multivariate skew-t mixture models. *Statistics & Probability Letters*, 82(6):1169–1174, 2012.

[185] I. Vrbik and P. McNicholas. Analytic calculations for the em algorithm for multivariate skew-t mixture models. *Statistics & Probability Letters*, 82(6):1169–1174, 2012.

[186] A. Waheed, M. Goyal, D. Gupta, A. Khanna, F. Al-Turjman, and P. R. Pinheiro. Covidgan: data augmentation using auxiliary classifier gan for improved covid-19 detection. *Ieee Access*, 8:91916–91923, 2020.

[187] C. Wang, P. W. Horby, F. G. Hayden, and G. F. Gao. A novel coronavirus outbreak of global health concern. *The lancet*, 395(10223):470–473, 2020.

[188] C. Wang, P. W. Horby, F. G. Hayden, and G. F. Gao. A novel coronavirus outbreak of global health concern. *The lancet*, 395(10223):470–473, 2020.

[189] S. Wang, Y. Zha, W. Li, Q. Wu, X. Li, M. Niu, M. Wang, X. Qiu, H. Li, H. Yu, et al. A fully automatic deep learning system for covid-19 diagnostic and prognostic analysis. *European Respiratory Journal*, 56(2), 2020.

[190] Y. Wang, M. Hu, Q. Li, X.-P. Zhang, G. Zhai, and N. Yao. Abnormal respiratory patterns classifier may contribute to large-scale screening of people infected with covid-19 in an accurate and unobtrusive manner. *arXiv preprint arXiv:2002.05534*, 2020.

[191] G. C. Wei and M. A. Tanner. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American statistical Association*, 85(411):699–704, 1990.

[192] Y. Wei, Y. Tang, and P. D. McNicholas. Flexible high-dimensional unsupervised learning with missing data. *IEEE transactions on pattern analysis and machine intelligence*, 42(3):610–621, 2018.

[193] C. Wirth, J. Fürnkranz, and G. Neumann. Model-free preference-based reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[194] Y.-H. Wu, N. Charoenphakdee, H. Bao, V. Tangkaratt, and M. Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pages 6818–6827. PMLR, 2019.

[195] H.-C. Yan, J.-H. Zhou, and C. K. Pang. Gaussian mixture model using semisupervised learning for probabilistic fault diagnosis under new data categories. *IEEE Transactions on Instrumentation and Measurement*, 66(4):723–733, 2017.

[196] L. Yan, H.-T. Zhang, Y. Xiao, M. Wang, C. Sun, J. Liang, S. Li, M. Zhang, Y. Guo, Y. Xiao, et al. Prediction of survival for severe covid-19 patients with three clinical features: development of a machine learning-based prognostic model with clinical data in wuhan. *medRxiv*, 2020.

[197] H. Yao, N. Zhang, R. Zhang, M. Duan, T. Xie, J. Pan, E. Peng, J. Huang, Y. Zhang, X. Xu, et al. Severity detection for the coronavirus disease 2019 (covid-19) patients using a machine learning model based on the blood and urine tests. *Frontiers in cell and developmental biology*, page 683, 2020.

[198] L. Yao and Y. Guan. An improved lstm structure for natural language processing. In *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, pages 565–569. IEEE, 2018.

[199] S. H. Yoo, H. Geng, T. L. Chiu, S. K. Yu, D. C. Cho, J. Heo, M. S. Choi, I. H. Choi, C. Cung Van, N. V. Nhung, et al. Deep learning-based decision-tree classifier for covid-19 diagnosis from chest x-ray imaging. *Frontiers in medicine*, 7:427, 2020.

[200] L. Yu, T. Yang, and A. B. Chan. Density-preserving hierarchical em algorithm: Simplifying gaussian mixture models for approximate inference. *IEEE transactions on pattern analysis and machine intelligence*, 41(6):1323–1337, 2018.

[201] A. Zargari Khuzani, M. Heidari, and S. A. Shariati. Covid-classifier: An automated machine learning model to assist in the diagnosis of covid-19 infection in chest x-ray images. *Scientific Reports*, 11(1):1–6, 2021.

[202] K. E. Zelik and A. D. Kuo. Mechanical work as an indirect measure of subjective costs influencing human movement. *PloS one*, 7(2):e31143, 2012.

[203] A. Zeroual, F. Harrou, A. Dairi, and Y. Sun. Deep learning methods for forecasting covid-19 time-series data: A comparative study. *Chaos, Solitons & Fractals*, 140:110121, 2020.

[204] J. S. Zhu, P. Ge, C. Jiang, Y. Zhang, X. Li, Z. Zhao, L. Zhang, and T. Q. Duong. Deep-learning artificial intelligence analysis of clinical variables predicts mortality in covid-19 patients. *Journal of the American College of Emergency Physicians Open*, 1(6):1364–1373, 2020.

[205] Z. Zhu, K. Lin, B. Dai, and J. Zhou. Learning sparse rewarded tasks from sub-optimal demonstrations. *arXiv preprint arXiv:2004.00530*, 2020.

[206] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.