THE UNIVERSITY OF SYDNEY

Bayesian Perspectives on Conditional Kernel Mean Embeddings: Hyperparameter Learning and Probabilistic Inference

by

Yuan-Shuo Kelvin Hsu

A thesis submitted in fulfillment for the degree of
Doctor of Philosophy

in the
Faculty of Engineering
School of Computer Science

September 2019

Declaration

I hereby certify that, to the best of my knowledge, the content of this thesis is my own work and is the result of original research. This thesis has not been submitted for a higher degree to any other University or Institution.

Kelvin Hsu

September 2019

Abstract

This thesis presents the narrative of a particular journey towards discovering and developing Bayesian perspectives on conditional kernel mean embeddings. It is motivated by the desire and need to learn flexible and richer representations of conditional distributions for probabilistic inference in various contexts. While conditional kernel mean embeddings are able to achieve such representations, it is unclear how their hyperparameters can be learned for probabilistic inference in various settings. These hyperparameters govern the space of possible representations, and critically influence the degree of inference accuracy. At its core, this thesis argues for the notion that Bayesian perspectives lead to principled ways for formulating frameworks that provides a holistic treatment to model, learning, and inference.

The story begins by emulating required properties of Bayesian frameworks via learning theoretic bounds. This is carried through the lens of a probabilistic multiclass setting, resulting in the multiclass conditional embedding framework. Through establishing convergence to multiclass probabilities and deriving learning theoretic and Rademacher complexity bounds, the framework arrives at an expected risk bound whose realizations exhibits desirable properties for hyperparameter learning such as the ever-crucial balance between data-fit error and model complexity, emulating marginal likelihoods. The probabilistic nature of this bound enable batch learning for scalability, and the generality of the model allow for various model architectures to be used and learned end-to-end.

The narrative unfolds into forming approximate Bayesian inference frameworks directly for the likelihood-free Bayesian inference problem, leading to the kernel embedding likelihood-free inference framework. The core motivator centers around the natural suitability of conditional kernel mean embeddings to forming surrogate probabilistic models. By leveraging the likelihood-free Bayesian inference problem structure, surrogate models for both hyperparameter learning and posterior inference are developed.

Finally, the journey concludes with a Bayesian regression framework that aligns the learning and inference to both the problem and the model. This begins by a careful formulation of the conditional mean and the novel deconditional mean problem, thereby revealing the novel deconditional mean embeddings as core elements of the wider kernel mean embedding framework. They can further be established as a nonparametric Bayes' rule with applications towards Bayesian inference. Crucially, by introducing the task transformed regression problem, they can be extended to the novel task transformed Gaussian processes as their Bayesian form, whose marginal likelihood can be used to learn hyperparameters in various forms and contexts.

These perspectives and frameworks developed in this thesis shed light into creative ways conditional kernel mean embeddings can be learned and applied in existing problem domains, and further inspire elegant solutions in novel problem settings.

Acknowledgements

This thesis could not have come to fruition without the support of many others who were there for me throughout this journey.

It is an honour to have worked with excellent mentors during my PhD. To this end, I would like to thank my supervisor, Professor Fabio Ramos. When I was in doubt and wavered, Fabio would share his wisdom to reassure and empower me with courage and confidence. When I was excited and creative, I was able to dash forward knowing he had my back. His guidance was grounded with sharp focus, yet full of trust as I fly and explore my own path.

I would also like to thank Professor Richard Nock for his mentorship, support, and patience. We would discuss and bounce every level of detail in each theorem to our heart's content on a whiteboard. I thoroughly enjoyed those sessions and I feel incredibly lucky to have learned so much from him.

A positive and uplifting research environment can make a whole world of difference. Thank you to my fellow research group for creating such a friendly and comfortable atmosphere on level 5. From exploring crazy and hilarious research ideas to playing board games and sports together, it was such an enjoyable ride.

As the root that sparked the early years of my machine learning journey, NICTA and Data61, CSIRO, holds a special place in my heart. I would like to thank everyone in the spatial inference systems team and machine learning research group. Without you, I would not have discovered my passion for probabilistic machine learning and taken the path that I now walk.

An ensemble of thanks goes to my dear friends, who stood by me at both times of joy and challenges. I would also like to thank my friends in our acoustic band for fostering a warm and lovely community where we express ourselves through music.

Lastly, I am deeply grateful to my dear parents and sister, who showered me with unconditional love and support throughout my life. I could not have done it without you.

Attribution

The contributions presented in this thesis have been published at the following conferences and form the core chapters of this thesis. The author's attribution includes, but is not limited to, the motivation, conceptualization, formalization, derivation, theorization, experimentation, and communication of the papers.

Chapter 3
Hyperparameter Learning for Conditional Kernel Mean Embeddings
with Rademacher Complexity Bounds

Hsu, K., Nock, R., and Ramos, F. (2018). Hyperparameter Learning for Conditional Kernel Mean Embeddings with Rademacher Complexity Bounds. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Volume 11052 of *Lecture Notes in Artificial Intelligence*, pages 227–242, Dublin, Ireland. Springer International Publishing. **Best Student Machine Learning Paper Award**.

Chapter 4
Bayesian Learning of Conditional Kernel Mean Embeddings
for Automatic Likelihood-Free Inference

Hsu, K. and Ramos, F. (2019). Bayesian Learning of Conditional Kernel Mean Embeddings for Automatic Likelihood-Free Inference. In *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, Volume 89 of *Proceedings of Machine Learning Research*, pages 2631–2640, Naha, Okinawa, Japan. Proceedings of Machine Learning Research.

Chapter 5
Bayesian Deconditional Kernel Mean Embeddings

Hsu, K. and Ramos, F. (2019). Bayesian Deconditional Kernel Mean Embeddings. In *Proceedings of the 36th International Conference on Machine Learning*, Volume 97 of *Proceedings of Machine Learning Research*, pages 2830–2838, Long Beach, California, USA. Proceedings of Machine Learning Research.

Contents

List of Figures

List of Tables

Glossary

ABC	approximate Bayesian computation
ARD	automatic relevance determination
ASL-ABC	adaptive synthetic likelihood ABC
AV-ABC	automatic variational ABC
BKR	Bayesian kernel regression
BLR	Bayesian linear regression
BO	Bayesian optimization
CDF	cumulative distribution function
CEN	conditional embedding network
CME	conditional mean embedding
CMO	conditional mean operator
CNN	convolutional neural network
CV	cross validation
DME	deconditional mean embedding
DMO	deconditional mean operator
DR-ABC	distribution regression ABC
ERM	empirical risk minimization
GMCE	Gaussian multiclass conditional embedding
GP	Gaussian process
GPA-ABC	Gaussian process accelerated ABC
GPC	Gaussian process classifier
GPR	Gaussian process regressor
GPS-ABC	Gaussian process surrogate ABC
HS	Hilbert-Schmidt
iid-KML	independent and identically distributed kernel means likelihood

K-ABC	kernel ABC
K2-ABC	double kernel ABC
KBR	kernel Bayes' rule
KDE	kernel density estimation
KELFI	kernel embedding likelihood-free inference
KMCF	kernel Monte Carlo filter
KME	kernel mean embedding
KML	kernel means likelihood
KMP	kernel means posterior
KMPE	kernel means posterior embedding
KR-ABC	kernel recursive ABC
KRR	kernel ridge regression
LFI	likelihood-free inference
LFVI	likelihood-free variational inference
LRR	linear ridge regression
MAP	maximum a posteriori
MCE	multiclass conditional embedding
MCMC	Markov chain Monte Carlo
MCMC-ABC	Markov chain Monte carlo ABC
MDN	mixture density network
MKML	marginal kernel means likelihood
MLH	median length heuristic
MMD	maximum mean discrepancy
MSE	mean squared error
NMSE	normalized mean squared error
OVA	one versus all
OVO	one versus one
PDF	probability density function
RCB	Rademacher complexity bound
REJ-ABC	rejection ABC
RKHS	reproducing kernel Hilbert space
RLSC	regularized least squares classifier
SGD	stochastic gradient descent
SL-ABC	synthetic likelihood ABC
SMC-ABC	sequential Monte Carlo ABC

ST-KML	spatio-temporal kernel means likelihood
SVC	support vector classifier
SVI	stochastic variational inference
SVM	support vector machine
TBKR	transformed Bayesian kernel regression
TBLR	transformed Bayesian linear regression
TTBKR	task transformed Bayesian kernel regression
TTBLR	task transformed Bayesian linear regression
TTGP	task transformed Gaussian process
TTGPR	task transformed Gaussian process regression
TTKRR	task transformed kernel ridge regression
TTLRR	task transformed linear ridge regression
TTR	task transformed regression
VBIL	variational Bayes with intractable likelihood
VBSL	variational Bayes with synthetic likelihood
VI	variational inference

Nomenclature

Chapter 3

$\mathbb{N}_n$	Natural numbers up to n
Ω	Sample space
$\mathbb{P}$	Probability measure on events
$\mathcal{X}$	Input space
$\mathcal{Y}$	Output space
$\mathbb{P}_X$	Probability measure of X on input space
$\mathbb{P}_Y$	Probability measure of Y on output space
x	Generic instance of input
y	Generic instance of output
X	Random variable with values realized in the input space
Y	Random variable with values realized in the output space
k	Kernel on input space
l	Kernel on output space
$\mathcal{H}_k$	Reproducing kernel Hilbert space induced by k
$\mathcal{H}_l$	Reproducing kernel Hilbert space induced by l
f	Generic element in $\mathcal{H}_k$, often also real-valued function on $\mathcal{X}$
g	Generic element in $\mathcal{H}_l$, often also real-valued function on $\mathcal{Y}$
C_{XX}	Second-order mean embedding, also cross-covariance operator
C_{YX}	Second-order mean embedding, also cross-covariance operator
$\mathcal{U}_{Y\|X}$	Conditional kernel mean operator
$\mu_{Y\|X=x}$	Conditional kernel mean embedding
$\hat{C}_{XX}$	Empirical second-order mean embedding
$\hat{C}_{YX}$	Empirical second-order mean embedding
$\hat{\mathcal{U}}_{Y\|X}$	Empirical conditional kernel mean operator
$\hat{\mu}_{Y\|X=x}$	Empirical conditional kernel mean embedding
ϕ	Input feature vector or function
ψ	Output feature vector or function
Φ	Input feature matrix
Ψ	Output feature matrix
K	Input gram matrix
$\mathbf{k}$	Input kernel vector function

δ	Kronecker delta kernel
$\mathbb{1}_c$	Indicator function at c
$\mathcal{H}_\delta$	Reproducing kernel Hilbert space induced by δ
$\mathbf{y}$	One hot encoded label
$\mathbf{Y}$	One hot encoded labels
p_c	Decision probability function for class c
$\mathbf{p}$	Decision probability vector function
$\hat{p}_c$	Empirical decision probability function for class c
$\hat{\mathbf{p}}$	Empirical decision probability vector function
$\tilde{p}_c$	Clip-normalized decision probability function for class c
$\tilde{\mathbf{p}}$	Clip-normalized decision probability vector function
h	Conditional information entropy function
$\tilde{h}$	Clip-normalized conditional information entropy function
u_x	Information gain function for example input x
$\mathbf{u}_x$	Information gain vector function for example input x
$\hat{u}_x$	Extended information gain function for example input x
$\hat{\mathbf{u}}_x$	Extended information gain vector function for example input x
θ	Kernel hyperparameters
λ	Regularization hyperparameter
Θ	Space of kernel hyperparameters
Λ	Space of regularization hyperparameter
$\mathbf{f}$	Decision vector function
$\mathcal{L}$	Cross entropy loss function
$\mathcal{L}_\epsilon$	Modified cross entropy loss function
$\mathcal{R}$	Rademacher complexity
F	Generic function class
σ	Rademacher random variable
r	Rademacher complexity bound
q	Learning objective
φ	Explicit feature map

Chapter 4

ϑ	Parameter space
$\mathcal{X}$	Simulation (data or summary statistics thereof) space
$\mathcal{Y}$	Observation (data or summary statistics thereof) space
$\mathcal{D}$	Data or summary statistics space
$\boldsymbol{\theta}$	Generic instance of parameter
$\mathbf{x}$	Generic instance of simulation
$\mathbf{y}$	Generic instance of observation
$\boldsymbol{\Theta}$	Random variable with values realized in the parameter space
$\mathbf{X}$	Random variable with values realized in the simulation space

$\mathbf{Y}$	Random variable with values realized in the observation space
ϵ	Simulation-to-observation noise hyperparameter
κ_ϵ	ϵ-kernel
$\mathcal{N}$	Gaussian density, univariate or multivariate
$\mathcal{N}(\theta; \mu, \sigma^2)$	Evaluation of $\mathcal{N}$ at θ with mean μ and standard deviation σ
$\mathcal{N}(\boldsymbol{\theta}; \boldsymbol{\mu}, \Sigma)$	Evaluation of $\mathcal{N}$ at $\boldsymbol{\theta}$ with mean $\boldsymbol{\mu}$ and covariance Σ
$p(a\|b)$	Density evaluation, short for $p_{A\|B}(a\|b)$
$p_{\mathbf{X}\|\boldsymbol{\Theta}}$	Simulator density
$p(\mathbf{x}\|\boldsymbol{\theta})$	Evaluation of simulator density
$p_{\mathbf{Y}\|\mathbf{X}}$	Noise density
$p_\epsilon(\mathbf{y}\|\mathbf{x})$	Evaluation of noise density
$p_{\mathbf{Y}\|\boldsymbol{\Theta}}$	Likelihood density
$p_\epsilon(\mathbf{y}\|\boldsymbol{\theta})$	Evaluation of likelihood density
$p_{\boldsymbol{\Theta}}$	Prior density
$p(\boldsymbol{\theta})$	Evaluation of prior density
$p_{\mathbf{Y}}$	Marginal likelihood density
$p_\epsilon(\mathbf{y})$	Evaluation of marginal likelihood density
$p_{\boldsymbol{\Theta}\|\mathbf{Y}}$	Posterior density
$p_\epsilon(\boldsymbol{\theta}\|\mathbf{y})$	Evaluation of posterior density
π	Proposal prior density
$\pi(\boldsymbol{\theta})$	Proposal prior density
D	Number of parameters
d	Index across parameters
n	Number of data points or summary statistics
i	Index across data points or summary statistics
m	Number of simulations
j	Index across simulations
T	Number of prior samples
t	Index across prior samples
S	Number of posterior super-samples
s	Index across posterior super-samples
$\mathbf{x}_j$	j-th simulation data or summary statistics sample
$\boldsymbol{\theta}_j$	j-th simulation parameter sample
$\tilde{\boldsymbol{\theta}}_t$	t-th prior parameter sample
$\hat{\boldsymbol{\theta}}_s$	s-th posterior parameter super-sample
k	Kernel on data or summary statistics space
ℓ	Kernel on parameter space
$\boldsymbol{\alpha}$	Hyperparameters of kernel k
$\boldsymbol{\beta}$	Hyperparameters of kernel ℓ
$\mathcal{H}_k$	Reproducing kernel Hilbert space induced by k
$\mathcal{H}_\ell$	Reproducing kernel Hilbert space induced by ℓ
f	Generic element in $\mathcal{H}_k$, often also real-valued function on $\mathcal{X}$
g	Generic element in $\mathcal{H}_\ell$, often also real-valued function on $\mathcal{Y}$

λ	Regularization hyperparameter
L	Simulation parameter gram matrix
$\boldsymbol{\ell}$	Simulation parameter vector
$\boldsymbol{\kappa}_\epsilon(\mathbf{y})$	Vector of ϵ-kernel comparisons of simulations to observation
$\mathbf{v}(\mathbf{y})$	Weights of kernel means likelihood evaluated at observation
$q(\mathbf{y}\|\boldsymbol{\theta})$	Evaluation of kernel means likelihood
$q(\mathbf{y})$	Evaluation of kernel means marginal likelihood
$q(\boldsymbol{\theta}\|\mathbf{y})$	Evaluation of kernel means posterior
$C_{\mathbf{X}\|\boldsymbol{\Theta}}$	Conditional mean operator of simulator
$\mu_{\mathbf{X}\|\boldsymbol{\Theta}=\boldsymbol{\theta}}$	Conditional mean embedding of simulator
$\mu_{\boldsymbol{\Theta}}$	Mean embedding of prior
$\mu_{\boldsymbol{\Theta}\|\mathbf{Y}=\mathbf{y}}$	Conditional mean embedding of posterior
$\hat{C}_{\mathbf{X}\|\boldsymbol{\Theta}}$	Empirical conditional mean operator of simulator
$\hat{\mu}_{\mathbf{X}\|\boldsymbol{\Theta}=\boldsymbol{\theta}}$	Empirical conditional mean embedding of simulator
$\tilde{\mu}_{\boldsymbol{\Theta}}$	Empirical mean embedding of prior
$\bar{\mu}_{\boldsymbol{\Theta}\|\mathbf{Y}=\mathbf{y}}$	Kernel mean posterior embedding

Chapter 5

$\mathcal{X}$	Input space of latent function or deconditional mean
$\mathcal{Y}$	Input space of response function or conditional mean
$\mathcal{Z}$	Target space, often the set of real numbers
X	Random variable with values realized in the input space $\mathcal{X}$
Y	Random variable with values realized in the input space $\mathcal{Y}$
Z	Random variable with values realized in the output space $\mathcal{Z}$
x	Generic instance in $\mathcal{X}$
y	Generic instance in $\mathcal{Y}$
z	Generic instance in $\mathcal{Z}$
k	Kernel on input space $\mathcal{X}$
ℓ	Kernel on input space $\mathcal{Y}$
$\mathcal{H}_k$	Reproducing kernel Hilbert space induced by k
$\mathcal{H}_\ell$	Reproducing kernel Hilbert space induced by ℓ
ϕ	Canonical feature function in $\mathcal{H}_k$
ψ	Canonical feature function in $\mathcal{H}_\ell$
f	Generic element in $\mathcal{H}_k$, often also real-valued function on $\mathcal{X}$
g	Generic element in $\mathcal{H}_\ell$, often also real-valued function on $\mathcal{Y}$
p_Y	Prior density
$p_{X\|Y}$	Likelihood density, function of realization and condition
$p_{X\|Y=y}$	Likelihood density, function of realization with fixed condition
$p_{X=x\|Y}$	Likelihood density, function of condition with fixed realization
C_{YY}	Prior mean operator
$C_{X\|Y}$	Conditional mean operator

$C'_{X\|Y}$	Deconditional mean operator
$\mu_{X\|Y=y}$	Conditional mean embedding, function of realization
$\mu'_{X=x\|Y}$	Deconditional mean embedding, function of condition
n	Number of transformation samples
i	Index over transformation samples
m	Number of task samples
j	Index over task samples
x_i	Transformation sample in input space $\mathcal{X}$
y_i	Transformation sample in input space $\mathcal{Y}$
$\tilde{y}_j$	Task sample in input space $\mathcal{Y}$
$\tilde{z}_j$	Task sample in output space $\mathcal{Z}$
$\mathbf{x}$	Collection of transformation samples in input space $\mathcal{X}$
$\mathbf{y}$	Collection of transformation samples in input space $\mathcal{Y}$
$\tilde{\mathbf{y}}$	Collection of task samples in input space $\mathcal{Y}$
$\tilde{\mathbf{z}}$	Collection of task samples in output space $\mathcal{Z}$
Φ	Collection of feature vectors of transformation inputs $\mathbf{x}$
Ψ	Collection of feature vectors of transformation inputs $\mathbf{y}$
$\tilde{\Psi}$	Collection of feature vectors of task inputs $\tilde{\mathbf{y}}$
K	Gram matrix of transformation inputs $\mathbf{x}$
L	Gran matrix of transformation inputs $\mathbf{y}$
$\tilde{L}$	Gram matrix of task inputs $\tilde{\mathbf{y}}$
A	Task transformation matrix
$\mathbf{k}$	Kernel vector function for inputs in $\mathcal{X}$
$\boldsymbol{\ell}$	Kernel vector function for inputs in $\mathcal{Y}$
$\tilde{C}_{YY}$	Empirical prior mean operator
$\hat{C}_{X\|Y}$	Empirical conditional mean operator
$\bar{C}'_{X\|Y}$	Empirical deconditional mean operator
$\hat{\mu}_{X\|Y=y}$	Empirical conditional mean embedding
$\bar{\mu}'_{X=x\|Y}$	Empirical deconditional mean embedding
λ	Regularization hyperparameter for prior mean operator
ϵ	Regularization hyperparameter for evidence mean operator
σ	Noise standard deviation
M	Transformation matrix for transformed regression
Σ	Noise covariance for transformed regression
$\bar{\boldsymbol{\alpha}}$	Weights on kernels in nonparametric view
$\bar{\mathbf{w}}$	Weights on features in parametric view
$\mathcal{N}$	Gaussian density
$\mathbf{u}$	Inducing points

Chapter 1

Introduction

1.1 Motivations

Modern machine learning algorithms are often probabilistic in nature, relying on statistical models for reasoning. In many practical scenarios, the probability distributions involved can be nontrivial and complex. This applies not only for distributions that are used to model stochasticity inherent in real world systems, but also for those that are used to express uncertainty due to limited knowledge. Consequently, they benefit from high complexity models that allow for rich representations of the probability distributions involved.

In particular, relationships between variables of interests are often expressed as conditional distributions. Due to the vastly intricate and complex ways variables can interact in the real world, practical frameworks must be careful to not oversimplify representations that describe the relevant conditional distributions.

Conditional kernel mean embeddings, also referred as conditional kernel means or conditional mean embeddings (CMEs), are kernel models that encode conditional distributions. More precisely, they encode conditional mean operations, meaning that it can be used to compute expectations of functions under the encoded conditional distribution. They are part of the kernel mean embedding (KME) framework, which in general encodes distributions as mean operations via kernels. As many types of inference problems can be expressed as expectations of functions under distributions, this makes KMEs widely applicable to a variety of settings.

These embeddings live in the reproducing kernel Hilbert space (RKHS), and fundamental probability rules are translated into mean operations in this space. While first class citizens of probabilistic rules are density evaluations, first class citizens of RKHS rules are expectations. This makes KMEs extremely attractive in practical scenarios where only samples from distributions of interest are available, such as

observed datasets or simulated examples. Given a finite amount of samples, density estimation is a fundamentally harder problem than estimating expectations, with the latter often amounting to simply taking empirical means. A primary reason is due to the curse of dimensionality affecting the former more than the latter. For instance, when the dimensionality of the problem increases by one, density estimation needs to be re-performed to spread the probability mass across the new dimension. If the number of samples stays constant, samples tend to appear more distant and less space-filling compared to before. Consequently, in general the number of samples required to achieve the same level of estimation quality increases exponentially with dimensionality. On the other hand, estimating expectations often amount to simply including the new dimension when computing the empirical means.

Mean embeddings operate with a philosophy similar to the kernel trick. The kernel trick makes use of the observation that certain learning algorithms only require dot products of features. Consequently, the trick involves computing those dot products via kernels directly, instead of inefficiently computing high dimensional features first before eventually computing dot products anyway. In a similar fashion, since solutions to many inference problems require quantities that can be expressed as expectations of functions, mean embeddings encode the necessary information required to compute expectations of functions directly, instead of computing high dimensional integrals with respect to a distribution. Crucially, when characteristic kernels are used, each distribution has a unique mean embedding. Informally, this means that the ability to represent and compute expectations of functions under the distribution is as useful as knowing the distribution itself.

Perhaps the most attractive aspect of the KME framework is that fundamental RKHS operations that correspond to fundamental probabilistic rules are linear. Combined with the fact that mean operations preserve linearity, the resulting empirical forms can be expressed entirely in terms of linear algebraic operations. In general, this means that the resulting algorithms become delightfully straightforward to implement.

The above considerations make KMEs and CMEs especially attractive for alleviating intractable integrals under complex distributions that are usually required in many probabilistic inference tasks. Consequently, CMEs permit the use of flexible and complex conditional distributions by representing them in the RKHS, while providing a principled framework to perform inference using those representations.

Nevertheless, there remain intricate problems that prevent the wide adoption of CMEs in the machine learning community, despite their apparent aforementioned advantages. This thesis is focused on formulating and developing solutions to these intricate problems. We do this by providing new perspectives and frameworks around CMEs using Bayesian principles.

1.2 Challenges

Despite the powerful representations of probability distributions it provides, CMEs face a number of challenges that prevent their ultimate potential to be fully leveraged. There remain several theoretical and practical challenges to overcome that currently barricade their applicability to a wide variety of problem settings. In particular, in this thesis we focus on the following challenges:

- **Hyperparameter Learning**: How can we learn hyperparameters for CMEs with respect to the problem setting?
- **Probabilistic Inference**: How can we infer relevant probability distributions from CMEs in a given problem setting?
- **Bayesian Computation**: Are there alternative RKHS operators that encode Bayes' rule? How are they related to established operators and models?
- **Uncertainty Quantification**: How can we use CMEs to quantify uncertainties for latent phenomena or models based on CMEs themselves?
- **Interpretability**: Can we establish connections or design techniques that help to interpret empirical artifacts of CME estimation?
- **Scalability**: How can we scale or find alternative ways to apply algorithms that address the above challenges to large datasets?

These challenges motivate the core theme and contributions of this thesis. The frameworks developed in all chapters focuses on addressing hyperparameter learning and probabilistic inference by leveraging Bayesian principles to various degrees, which consequently provides methods to tackle the remaining challenges. Chapter 3 addresses hyperparameter learning, probabilistic inference, and scalability. Chapter 4 addresses hyperparameter learning, probabilistic inference, uncertainty quantification, and interpretability. Chapter 5 addresses hyperparameter learning, probabilistic inference, Bayesian computation, uncertainty quantification, interpretability, and scalability.

1.2.1 Hyperparameter Learning

Perhaps the most important ingredient required for successful adoption and application of CMEs is an appropriate hyperparameter learning algorithm that is suitable for the probabilistic inference task involved. The hyperparameters govern the RKHS for which the feature representations lie within. Consequently, by

developing a principled hyperparameter learning framework, we can learn these appropriate representations for the inference task automatically. We therefore also refer to this process as *automatic representation learning*.

Solutions to kernel methods are often highly dependent on the hyperparameters chosen. They begin by positing a family of kernels, whose parameters become part of their model *hyperparameters*, which may further include other hyperparameters such as noise or regularization hyperparameters. Given a set of hyperparameters, training is often performed by solving a convex or quadratic optimization problem, such as in the case for support vector machines (SVMs) [Schölkopf and Smola, 2002]. In other kernel models, the quadratic programming problem can be solved analytically and expressed as solutions to a set of linear equations, such as in the case for Gaussian process regressors (GPRs) [Rasmussen and Williams, 2006], regularized least squares classifiers (RLSCs) [Rifkin et al., 2003], and most importantly CMEs. Consequently, hyperparameters govern their solution space.

Unfortunately, hyperparameter tuning is not straight forward. Often, cross validation [Song et al., 2013] or median length heuristics [Muandet et al., 2017] remain as standard solutions. However, they are generic approaches that do not refer to the probabilistic inference task directly. For instance, cross validation often uses square losses in the RKHS such as (2.52), which may not necessarily be directly relevant to the task. Furthermore, cross validation can be computationally expensive and sensitive to the selection and number of validation sets. On the other hand, median length heuristics only applies to hyperparameters with a length scale interpretation, so it cannot be used to learn other hyperparameters such as regularization hyperparameters. It also does not make use of the conditional relationship between the two variables, but rather just their respective marginal distributions. Overall, median length heuristics do not make use of important information that CMEs have access to for hyperparameter selection.

One notable success story in this domain are Gaussian process regressors (GPRs) [Rasmussen and Williams, 2006], which employ their marginal likelihood as an objective for hyperparameter learning. The marginal likelihood arises from its Bayesian formulation, and exhibits certain desirable properties – in particular, the ability to automatically balance between data fit and model complexity. On the other hand, CMEs are not necessarily Bayesian, and hence they do not benefit from a natural marginal likelihood formulation, yet such a balance is critical when generalizing the model beyond known examples. A core contribution of this thesis is to develop or approximate Bayesian learning strategies for CMEs in order to achieve this simple yet elegant method for hyperparameter learning.

Motivated by the log marginal likelihood objective for hyperparameter learning of GPRs, Flaxman et al. [2016] developed a log marginal likelihood objective for marginal mean embeddings. This is done by placing a GP prior on the true

mean embedding itself, and constructing a Gaussian likelihood from the true mean embedding to the empirical mean embedding. Similar to the GPR, with a GP prior and a Gaussian likelihood, inference becomes tractable and a closed form solution to the log marginal likelihood can be obtained up to a slight change of variable factor. Nevertheless, it is not obvious how this can be extended to CMEs.

Furthermore, hyperparameter learning for CMEs can be comparatively more challenging than marginal mean embeddings. Firstly, there are two distinct kernels on two spaces which play different roles, one for the "output" and one for the "input", as opposed to one overall kernel for marginal mean embeddings. Effects of changing hyperparameters for these two kernels can interact and affect the CME in complex ways. Secondly, empirical CMEs have a regularization hyperparameter. While the regularization hyperparameter is introduced to prevent overfitting [Song et al., 2009], as we can see from (2.49) it serves to prevent overfitting specifically with respect to the square loss in the RKHS by shrinking the Hilbert-Schmidt (HS) norm of the conditional mean operators (CMOs). This is not necessarily the shrinkage or regularization that is relevant depending on the problem setting. Consequently, it may not be obvious how to interpret it or construct objectives to learn it appropriately. Furthermore, empirically it serves to stabilize the inversion of gram matrices, which are often only positive semidefinite without the regularization hyperparameter. As a result, the inversion may explode as the regularization hyperparameter decreases such that the matrix to be inverted approaches singularity. This is another practical concern when learning the regularization hyperparameter via optimization techniques.

In this thesis, we derive and develop algorithms to learn hyperparameters for CMEs in three different contexts. Chapter 3 presents a learning theoretic approach to motivate and construct a complexity balanced learning objective that emulate a marginal likelihood in the probabilistic multiclass classification context, chapter 4 approximates a marginal likelihood surrogate for likelihood-free inference (LFI), and chapter 5 formulates a marginal likelihood by establishing a Bayesian regression view.

1.2.2 Probabilistic Inference

Inferring probability distributions using KMEs can be a challenging task. In particular, this often involved the recovery of encoded distributions from their mean embeddings, which is generally an unsolved problem. A core aspect of the KME framework is to map distributions $\mathbb{P}_X$ to its mean embedding μ_X and perform inference solely in the RKHS via RKHS machinery and operations. While the mapping from distributions to their mean embeddings is a simple kernel mean and is injective for characteristic kernels, the reverse mapping do not have a readily known solution. That is, it is not easy to recover the distribution $\mathbb{P}_X$ that

was embedded as a mean embedding μ_X once it is embedded, even though this mapping is one-to-one.

This is known as the *pre-image* problem, where the challenge is to find what the distribution looked like (hence, *image*) before it was embedded as a mean embedding (hence, *pre*).

Before discussing the pre-image problem further, it is worthwhile to point out that a core part of the spirit of KMEs is to circumvent the need and requirement of explicitly working with distributions in the usual probability space, such as densities. For instance, density estimation is much more prone to the curse of dimensionality than kernel mean estimation, since their convergence rates do not directly involve the dimensionality of the samples as per theorem 2.7 and theorem 2.9, since samples have been mapped into an infinite dimensional RKHS. As such, it is seemingly counterproductive to seek pre-images from mean embeddings, since if explicit probability distributions are required, then frameworks other than the KME could be more suitable.

However, there exists scenarios where one would be nevertheless interested in the pre-image of certain mean embeddings.

The first scenario is when marginal likelihoods is to be recovered for hyperparameter learning. In section 1.2.1 we alluded to the idea that hyperparameter learning of CMEs can potentially be achieved via formulating or emulating a marginal likelihood objective. However, marginalization integrals involved for the marginal likelihood is often hard to compute in the usual probability space. Yet, since marginalization integrals are expectations, and KMEs encode expectations, it is natural to consider using the KMEs framework to sidestep the difficult integral and instead obtain the mean embedding of the marginal likelihood. Nevertheless, suppose we are somehow able to derive the mean embedding of the marginal likelihood, we still need to recover the encoded marginal likelihood so that we can optimize it with respect to the hyperparameters. In this thesis, chapter 4 presents solutions to approximate marginal likelihoods from mean embeddings, and chapter 5 formulates marginal likelihoods of the mean embeddings themselves.

The second scenario is when interpretable inference results are required. For instance, in classification settings decision boundaries are often determined by decision probabilities, and in Bayesian inference settings the posterior itself is of interest to quantify uncertainty. Therefore, after the mean embeddings of these distributions are obtained, it is of ultimate interest to convert them back to distributions, either in the form of probabilities, densities, or samples, so that these results can be interpreted for use in their respective problem setting. In this thesis, chapter 3 addresses the classification setting, and chapter 4 addresses the Bayesian inference setting.

In general, techniques to recover distributions from mean embeddings would enable a synergistic framework where RKHS operations with mean embeddings and probabilistic operations with distributions can be used in conjunction to take advantages from both worlds. Ideally, when inference requires intractable integrals which are too difficult in the usual probability space, we can embed distributions into the RKHS as mean embeddings, perform the equivalent RKHS operations, and recover the distribution the resulting RKHS object encodes. Unfortunately, this recovery of distributions from mean embeddings do not have a closed form solution.

Furthermore, it is also possible that the pre-image may not even exist. While the mapping from distributions to mean embeddings is injective, it is not necessarily guaranteed to be surjective. This means that if one collects all the mean embeddings obtained from embedding all possible distributions, it still may not fill up the whole RKHS. While this is not a problem for true mean embeddings, as they have a pre-image by construction, it can become a problem once the mean embedding undergo RKHS operations, effectively bringing the mean embedding to another point in the RKHS, which may not correspond to any distribution.

As an intuitive illustration, consider the analogy where probability distributions are akin to human individuals and their mean embeddings are akin to their gene. All human individuals have unique gene profiles. However, not all genes belong to humans, since some may belong to dogs, cats, or other species. That is, the space of all possible genes is in some sense bigger than the space of all possible humans.

This problem is even more pronounced for empirical CMEs. Suppose a kernel was chosen such that true CMEs always evaluate to nonnegative values. The empirical CME may evaluate to negative values even if its true CME never evaluates to negative values, due to the presence of a matrix inversion. This means that there are no distributions in the space of distributions considered that would map to such an empirical CMEs.

Current approaches to solve the pre-image problem usually begin by assuming a particular form to the density of the pre-image. For instance, one can approximate the pre-image with a mixture of Gaussians [McCalman et al., 2013] or a mixture smoothing kernels [Kanagawa and Fukumizu, 2014] which are also often Gaussians. Nevertheless, recovering distributions from mean embeddings for probabilistic inference remain a challenging problem in many problem settings.

In this thesis, we recover distributions from CMEs for probabilistic inference in three different contexts. Chapter 3 recovers class decision probability estimates for probabilistic multiclass classification, chapter 4 approximates surrogate densities of likelihoods and posteriors for LFI, and chapter 5 reveals posterior predictive processes by establishing a Bayesian regression view.

1.2.3 Bayesian Computation

Bayesian inference often requires computation of the posterior $\mathbb{P}_{Y|X}$ when given the likelihood $\mathbb{P}_{X|Y}$ and the prior $\mathbb{P}_Y$. When density evaluations exist, the Bayes' rule provides their relationship as $p_{Y|X}(\cdot|x) = \frac{p_{X|Y}(x|\cdot)p_Y(\cdot)}{\int_{\mathcal{Y}} p_{X|Y}(x|y)p_Y(y)dy}$. We refer to this computation as Bayesian computation.

Importantly, Bayesian inference and Bayesian computation are separate concepts. Bayesian inference is focused on inferring distributions on latent or unknown variables of interest, thereby capturing a sense of uncertainty in our knowledge of these variables. It often requires Bayesian computation using the Bayes' rule to obtain a posterior distribution of the latent or unknown variables given the observed variables. Bayesian inference is a philosophy of approach. On the other hand, Bayesian computation refers to the process of obtaining representations of $\mathbb{P}_{Y|X}$ from representations of $\mathbb{P}_{X|Y}$ and $\mathbb{P}_Y$, regardless of their meaning. If these representations are densities or probabilities, then Bayes' rule can be applied. The Bayes' rule is simply a mathematical statement that is true regardless of the nature of the random variables. Note that these representations of distributions do not have to be densities. For example, since mean embeddings are unique representations of distributions, should there be a rule that specifies the mean embedding of $\mathbb{P}_{Y|X}$ as a function of the mean embeddings of $\mathbb{P}_{X|Y}$ and $\mathbb{P}_Y$, then such a rule can be used for Bayesian computation.

In this section, we discuss the application of KMEs towards Bayesian computation, and defer the discussion of Bayesian inference specifically to the next section.

In Bayesian computation using the usual Bayes' rule, several levels of intractability may arise. The first is when both likelihood and prior density evaluations are tractable but the evidence integral $\int_{\mathcal{Y}} p_{X|Y}(x|y)p_Y(y)dy$ is intractable, leading to literatures such as variational inference (VI) [Blei et al., 2017] and Markov chain Monte Carlo (MCMC) [Hastings, 1970]. The next is when only likelihood evaluations are intractable but sampling is possible, leading to literatures such as likelihood-free inference (LFI) and approximate Bayesian computation (ABC) [Marin et al., 2012]. More rarely, only prior evaluations are intractable but available via sampling, leading to literatures in implicit priors. The last is when both the likelihood and prior evaluations are intractable but available via sampling, leading to newer literatures such as implicit generative models.

While there are many approaches that addresses posterior approximations in each of these scenarios, the underlying limitation that is shared across all these settings is that Bayes' rule requires density evaluations that are difficult to approximate in high dimensions from samples. Instead, if relationships between the posterior, likelihood, and prior can be captured without using density evaluations, but directly by using mean embeddings, this issue could be more naturally sidestepped.

The KME framework can be used to encode probability rules in the RKHS. Canonical examples include the *kernel sum rule*, *kernel chain rule*, and *kernel Bayes' rule (KBR)*, which encodes the sum rule, chain rule, and Bayes' rule respectively [Fukumizu et al., 2013, Muandet et al., 2017, Song et al., 2013]. We will collectively refer to them as *kernel distribution rules*.

Kernel distribution rules provide an elegant set of machinery for performing probabilistic inference without operating directly in the usual probability space, such as using densities. They relate mean embeddings via linear operations in the RKHS, instead of density evaluations via products and integrals. In particular, the KBR is powerful in that it circumvents intractabilities such as marginalization integrals that may occur in the computation of the posterior in the usual Bayes' rule. The usual Bayes' rule takes a likelihood density and a prior density, and outputs a posterior density. Analogously, the KBR takes a CMO encoding the likelihood and a marginal mean embedding encoding the prior, and outputs a CMOs encoding the posterior [Fukumizu et al., 2013]. Since its empirical form only requires samples from the distributions instead of a parametrized density, and its complexity increases with the number of samples, we refer to this type of encoding of Bayes' rule as a *nonparametric Bayes' rule*.

Importantly, because a distribution can be encoded as mean embeddings of multiple orders, there are different versions for these kernel distribution rules depending on the order of the tensors used to encode particular distributions. In particular, for KBR, there are two versions, which uses different ordered tensors for the likelihood and prior. Crucially, on top of this, they further use two different ordered tensors for the likelihood and prior within each version. This can cause unnatural consequences in the nonparametric version of the empirical KBRs, such as having the effect of the prior mean embedding vanish as hyperparameters change. Since the same probability rule can be encoded at multiple orders in the RKHS, it remains an open question whether there are more natural alternatives for a nonparametric Bayes' rule, and what assumptions or consequences they bring.

In this thesis, chapter 5 establishes the proposed operator as a nonparametric Bayes' rule, and discusses its connections to KBR.

1.2.4 Uncertainty Quantification

Uncertainty quantification is a core aspect of Bayesian inference. There are two settings in which uncertainty quantification is important to CMEs.

The first is when CMEs are used in approximate Bayesian inference problems. This setting is concerned with approximating the posterior of a given inference problem, such as the inference of parameters of a simulator model, which in turn

provides a sense of uncertainty for the inference of such quantities. The second is when uncertainty needs to be quantified on the CMEs themselves. This involves establishing the task for which CMEs solve and quantifying the uncertainty on the solution the CMEs provides. Importantly, this requires treating such a solution as a latent function or object, and performing Bayesian inference on it.

The two settings are distinct. In the second setting, the Bayes' rule acts between observations and latents involved in the CME itself. In contrast, in the first setting the Bayes' rule acts between the observations and latents involved in the given inference problem, where the CME serves to encode the distributions involved. In essence, in the first setting the CME seen as a tool to perform Bayesian computation for Bayesian inference on another problem setting, whereas in the second setting the CME is the object of interest that inference is performed on.

In this thesis, chapters 4 and 5 address the first and second settings respectively.

1.2.5 Scalability

Computational efficiency and scalability is always a pressing practical concern that determines the usability of the framework. Many kernel methods like CMEs require the inversion of a regularized gram matrix of size $n \times n$, where n is the number of data points. Consequently, the computational complexity is dominated by $O(n^3)$ in time and $O(n^2)$ in space. This means that kernel approaches like CMEs can become prohibitively expensive, especially compared to linear approaches in the feature space, which often have linear time complexity of $O(n)$.

In this thesis, chapter 3 addresses scalability by requiring only stochastic or batch updates, while chapter 5 addresses scalability via theoretical connections to GPs.

1.2.6 Interpretability

A common theme throughout the discussion above is the difficulty in interpreting aspects of the KME framework. Mean embeddings do not have the same level of interpretability as usual probability distribution, which can be interpreted directly with notions of frequency or uncertainty. Furthermore, empirical estimations of CMEs often introduce artifacts that can impact interpretability such as regularization hyperparameters for operator inversions or non-positive mean embeddings despite the use of positive kernels.

In this thesis, chapter 4 addresses the interpretability of negatively valued CMEs through surrogate densities and sampling strategies for recovering the desired distribution, while chapter 5 addresses the interpretability of the regularization hyperparameters via establishing alternative regression views to CMEs.

1.3 Contributions

By exploring the theme of adopting Bayesian principles to overcome challenges that barricade the applicability of CMEs, the core contributions of this thesis involve holistic hyperparameter learning and probabilistic inference frameworks for three general problem settings – classification (chapter 3), inference (chapter 4), and regression (chapter 5).

Chapter 3 presents the multiclass conditional embedding (MCE) framework. Contrary to most classification frameworks whose formulation begins as a binary classification problem that is later extended to a multiclass form, the MCE is formulated naturally as a probabilistic and multiclass classifier, by leveraging the generality of CMEs (section 3.3). It further provides a multiclass information entropy measure for quantifying the uncertainty in its predictions (section 3.4).

Importantly, the hyperparameter learning framework for MCEs are based on learning theoretic bounds, which reveal the Rademacher complexity bound (RCB) as a data-dependent complexity measure for multiclass settings (section 3.5). Together with a modified cross entropy loss, these learning theoretic bounds express generalization risk and can be used to derive risk bounds as a function of hyperparameters, leading to learning objectives for balancing data fit error against model complexity. Due to the generality and applicability of these bounds, by realizing the bound using a subset of the data, we establish a scalable stochastic batch gradient-based hyperparameter learning algorithm through stochastic batch realizations of the learning theoretic bound (section 3.6).

In order to establish MCEs as approximations to probabilistic classifiers, we also provide convergence guarantees (section 3.7). With the model output being approximate probabilistic predictions, we open up the realm of learning the MCEs under the cross entropy loss. Critically, to arrive at the learning theoretic bounds that would help prevent overfitting, bounds that are statements regarding a function class must be translated to statements regarding a particular function with particular hyperparameters (section 3.8).

With its roots in CMEs, MCEs can benefit from using various different model architectures as to form features and thus kernels. One notable and widely useful instance is the conditional embedding network (CEN) and its variants, providing a general family of MCEs formed from deep neural network architectures that can be trained end-to-end under the scalable hyperparameter learning algorithm (section 3.9).

Chapter 4 presents the kernel embedding likelihood-free inference (KELFI) framework. Current likelihood-free inference (LFI) methods do not directly address the calibration or learning of hyperparameters for the specific method, including the

hyperparameter ϵ that governs the trade-off between computational requirement and inference accuracy. In contrast, the KELFI framework performs automatic likelihood-free Bayesian inference, where the hyperparameter learning algorithm directly addresses the inference task (section 4.3).

The KELFI framework can be understood via three parts - model, learning, and inference. It begins with the kernel means likelihood (KML) as a consistent likelihood surrogate *model* that modularizes approximate likelihood evaluations from simulation calls (section 4.3.2). The marginal kernel means likelihood (MKML) then acts as a Bayesian *learning* objective for hyperparameters that improves inference accuracy through optimal surrogate representations (section 4.3.3). The kernel means posterior (KMP) then approximates the true posterior as a posterior surrogate density for approximate Bayesian *inference* on simulator parameters, which is then embedded back into the corresponding Hilbert space as kernel means posterior embedding (KMPE) so that performing super-sampling would in a set of fast-converging posterior samples of simulator parameters (section 4.3.4). Finally, we provide proofs of convergence guarantees for kernel embedding likelihood-free inference (KELFI) (section 4.4).

The KELFI framework is further extended by the formulation of the spatio-temporal kernel means likelihood (ST-KML) and independent and identically distributed kernel means likelihood (iid-KML) as classes of KML that are suitable for spatio-temporal data and *iid* data respectively, alleviating requirements of designing summary statistics (sections 4.5 and 4.6). We also discuss a trick for translating a LFI problem into one that involves a Gaussian prior, which enables KELFI to use closed-formed solutions (section 4.7).

Chapter 5 presents the deconditional mean embedding (DME) and task transformed Gaussian process (TTGP) framework. We first begin by emphasizing CMEs directly as solutions to the conditional mean problem (section 5.3; proofs in section 5.11). Starting from the lens of recovering functions from their conditional means to complement CMEs in the KME framework, this work introduces and formulates DMEs as a new fundamental member of the KME framework via parallel formalization of the deconditional mean problem against the conditional mean problem (section 5.4; proofs in section 5.12). We then move onto a regression view of DMEs by establishing them as solutions to chained losses by introducing the task transformed regression task, and present both the parametric and non-parametric forms of the DME solution. We then introduce and formalize TTGPs, and establish them as Bayesian regression views of the DME (section 5.5; proofs in section 5.13). This reveals a marginal likelihood for learning hyperparameters of DMEs and CMEs as a sub-case.

By considering the distributions encoded by DMEs, we establish that DMEs can be seen as a nonparametric Bayes' rule, encoding Bayes' rule through samples

without prescribing to any particular parametric form (section 5.6; proofs in section 5.14). In particular, DMEs possess a natural second order form that makes it distinct to kernel Bayes' rule (KBR) which uses third order tensors that lead to unstable empirical estimates (section 5.7). In various ways, DMEs complete the framework of KMEs and have interesting relationships with techniques in the literature (section 5.8).

Importantly, as a consequence of establishing a marginal likelihood of TTGPs for the task transformed regression problem, they can also be used for sparse representation learning via establishing the learning of inducing points as a sub-case of this problem (section 5.9). On the other hand, as a consequence of establishing DMEs as a nonparametric Bayes' rule, DMEs can be used for likelihood-free inference (LFI), opening up another avenue for kernel-based technique in this area (section 5.9; proofs in section 5.15).

The rest of this thesis is structured as follows. Chapter 2 begins with an introduction to the framework of kernel mean embeddings (KMEs) with a specific focus on conditional mean embeddings (CMEs). Chapters 3 to 5 presents our main contributions above, and chapter 6 concludes the thesis with a summary of the contributions, the overall theme, and a forward discussion on future work.

Chapter 2

Background

The framework of kernel mean embeddings (KMEs) is concerned with representations and transformations of highly flexible and general probability distributions in the context of performing probabilistic inference with these distributions. It represents distributions in reproducing kernel Hilbert spaces (RKHSs) as mean embeddings, where complex operations required for probabilistic inference become linear in the RKHS. The central object from which the RKHS is founded upon is the kernel, a symmetric function that measures the similarity between any two objects. Empirical forms of these representations and operators are able to take full advantage of kernel machinery, leading to a general nonparametric framework for representing and transforming distributions directly with data. Consequently, KMEs enable nonparametric expressions of probability distributions that alleviate common intractability or inflexibility issues encountered in probabilistic inference.

In this section we review KMEs in a general fashion and introduce the necessary background required for the core chapters of this thesis. In particular, we present this review in a bottom-up fashion. We first introduce probability theory and RKHSs in section 2.1 and section 2.2 separately. These form the foundations for an introduction to mean embeddings and conditional mean embeddings (CMEs) in section 2.3 and section 2.4 respectively. We then present their empirical and nonparametric forms in section 2.5 and section 2.6 respectively.

Part of the contributions of this thesis involves either formulating KMEs in a different or non-standard setting, or reformulating the framework from a different perspective. Consequently, we will also briefly review CMEs again in a manner that is more directly relevant for each core chapter, in order to introduce the setup or formulation for that particular chapter.

2.1 Probability Theory

Probability is at the heart of learning and inference. We begin with a minimal review of probability theory with special attention to notation and shorthand.

Probability Space. We first introduce a probability space, which is a specific type of a measure space. In the following paragraph, we use terminologies from probability theory, and include the corresponding terminologies from measure theory in brackets for reference.

Let the tuple $(\Omega, \mathcal{W}, \mathbb{P})$ be a probability space (measure space), where Ω is the sample space (set), $\mathcal{W}$ is a σ-field (σ-algebra) of subsets of Ω, and $\mathbb{P} : \mathcal{W} \to [0, 1]$ a probability measure (measure). An element W in $\mathcal{W}$ is called an event (measurable set), which are by definition subsets of Ω. An element ω in Ω is called a sample point (point).

In other words, the probability measure $\mathbb{P}$ assigns a probability between 0 and 1, inclusive, for a given event W, where events are elements of the σ-field $W \in \mathcal{W}$ or equivalently are subsets of the samples space $W \subseteq \Omega$.

Random Variables. A random variable X is formally a map from the sample space Ω to some Borel measurable domain $\mathcal{X}$ of interest.

Definition 2.1 (Random Variable)**.** A measurable function $X : \Omega \to \mathcal{X}$ for some domain $\mathcal{X}$ is called a *random variable*.

When $\mathcal{X}$ is a space of scalars, we call X a *scalar-valued* random variable. When $\mathcal{X}$ is a space of vectors, we call X a *vector-valued* random variable, or sometimes a random vector, with d dimensions. We will formally define scalars and vectors in the next section. When we develop frameworks for a general $\mathcal{X}$, we will usually denote the random variable as X, where it is left general for either or other cases. When the vectorial nature of vector-valued random variable is to be emphasized, we denote the random variable in bold as $\mathbf{X}$. While random variables are formally a mapping, they are so named to emphasize their interpretation as a variable that can be instantiated to different value(s) under different events. An instantiation of a random variable is denoted with a corresponding lower case letter x or $\mathbf{x}$.

Distributions. A distribution describes the probability for which the random variable will take on values from a particular subset of the domain.

Definition 2.2 (Distribution)**.** Let $(\Omega, \mathcal{W}, \mathbb{P})$ be a probability space. The *distribution* of a random variable $X : \Omega \to \mathcal{X}$ is the Borel measure $\mathbb{P}_X : \mathcal{A} \to [0, 1]$ defined by

$$\mathbb{P}_X[A] := \mathbb{P}[X \in A], \tag{2.1}$$

where $\mathcal{A}$ is a σ-field of subsets on $\mathcal{X}$ and $A \subseteq \mathcal{X}$ is a subset of the domain. The event $\{X \in A\}$ is a shorthand notation for the inverse image, also referred as the pre-image, of $A \subseteq \mathcal{X}$ under the random variable X,

$$\{X \in A\} := X^{-1}[A] := \{\omega \in \Omega : X(\omega) \in A\} \in \mathcal{W}. \tag{2.2}$$

The distribution is also a probability measure itself, as it satisfies the defining properties of a probability measure, such that the tuple $(\mathcal{X}, \mathcal{A}, \mathbb{P}_X)$ is also a probability space. It differs from $\mathbb{P}$ in the sense that it operates in a different domain. Instead of operating directly on the events, it operates on the values the random variable takes for those events.

Cumulative Distribution Function. For *real-valued* random variables, we can consider probability measures on the real line.

Definition 2.3 (Cumulative Distribution Function)**.** The *cumulative distribution function (CDF)* $P_X : \mathbb{R} \to [0, 1]$ of a *real-valued* random variable $X : \Omega \to \mathbb{R}$ and random vector $\mathbf{X} : \Omega \to \mathbb{R}^d$ is defined as

$$\begin{aligned} P_X(x) &:= \mathbb{P}_X[(-\infty, x]] = \mathbb{P}[X \leq x], \\ P_{\mathbf{X}}(\mathbf{x}) &:= \mathbb{P}_{\mathbf{X}}[(-\infty, \mathbf{x}]] = \mathbb{P}[\mathbf{X} \leq \mathbf{x}]. \end{aligned} \tag{2.3}$$

For conciseness, we define the following shorthand,

$$\begin{aligned} \{X \leq x\} &:= \{X \in (-\infty, x]\}, \\ \{\mathbf{X} \leq \mathbf{x}\} &:= \bigcap_{j=1}^{d} \{X_j \in (-\infty, x_j]\} \in \mathcal{W}, \\ (-\infty, \mathbf{x}] &:= (-\infty, x_1] \times \cdots \times (-\infty, x_d]. \end{aligned} \tag{2.4}$$

Probability Density Function. For domains $\mathcal{X}$ where a Lebesgue measure is defined, we can further consider densities of probability measures. Intuitively, the Lebesgue measure provides a sense of volume for the domain, and probability measures attempt to measure the portion of mass distributed within a given volume in that domain, giving rise to a notion of density throughout the domain.

Theorem 2.1 (Radon-Nikodym)**.** *If the distribution $\mathbb{P}_X$ is absolutely continuous with respect to some measure ν on $\mathcal{X} = \mathbb{R}^d$, then the Radon-Nikodym theorem implies the existence of a density for $\mathbb{P}_X$ with respect to ν. We call this the distribution density of X with respect to ν, or more commonly known as the probability density function (PDF), denoted by $p_X : \mathcal{X} \to [0, \infty)$. The PDF is the function that satisfies*

$$\mathbb{P}[X \in A] = \int_A p_X d\nu \quad \forall A \subseteq \mathcal{X}. \tag{2.5}$$

If the distribution ν is a Lebesgue measure ν_d on some Euclidean space $\mathbb{R}_d$, then the PDF $p_X : \mathbb{R}^d \to [0, \infty)$ is the function that satisfies the familiar form

$$\mathbb{P}[X \in A] = \int_A p_X d\nu_d \equiv \int_A p_X(x)dx \quad \forall A \subseteq \mathbb{R}^d. \tag{2.6}$$

Importantly, CDF P_X and PDF p_X are functions of $x \in \mathcal{X}$ and their evaluations are denoted as $P_X(x)$ and $p_X(x)$ respectively. It is common practice that when it is clear from context what random variable we are referring to, such as when the point of evaluation x is the lower case letter of the random variable X, we can drop the subscript and denote them as $P(x)$ and $p(x)$ respectively.

Function Expectations. For a random variable $X : \Omega \to \mathcal{X}$ and a function $f : \mathcal{X} \to \mathbb{R}$. The notation in probability theory and statistics for the expectation of a function of a random variable would be $\mathbb{E}[f(X)]$, as if X is a variable and not a function, while in measure theory the more precise notation is $\mathbb{E}[f \circ X]$, as $\mathbb{E}$ is a functional of functions on the event space. In practice, $\mathbb{E}[f(X)]$ becomes a shorthand for $\mathbb{E}[f \circ X]$, and thus the former is used more often.

Definition 2.4 (Function Expectation)**.** Let $f : \mathcal{X} \to \mathbb{R}$ be some arbitrary real-valued function over the domain $\mathcal{X}$, then the function expectation is defined as

$$\mathbb{E}[f(X)] := \int_{\mathcal{X}} f d\mathbb{P}_X = \int_{\Omega} (f \circ X) d\mathbb{P} =: \mathbb{E}[f \circ X]. \tag{2.7}$$

Since random variables $X : \Omega \to \mathcal{X}$ are functions of events , we have that the composition $(f \circ X) : \Omega \to \mathbb{R}$ assigns a real-value output to the events directly.

When the density exists with respect to the Lebesgue measure ν_d as per theorem 2.1, then we can use the more common and familiar form for expectations,

$$\mathbb{E}[f(X)] = \int_{\mathbb{R}^\Gamma} f p_X d\nu_d = \int_{\mathbb{R}^\Gamma} f(x) p_X(x) dx. \tag{2.8}$$

Functions and Functionals. While both measures and regular functions are mappings, we make a distinction between mappings whose domain is a σ-algebra, such as measures, and mappings whose domain is not, such as regular functions. This distinction is emphasized by the notation of how a measure ν or a regular function f takes its arguments. A measure takes its arguments with square brackets, as in $\nu[\cdot]$. A regular function takes its arguments with round brackets, as in $f(\cdot)$. For functionals $\mathbb{F}$, or mappings that take regular functions as an input, square brackets are also used, such as $\mathbb{F}[\cdot]$.

Typical examples of measures are the probability measure $\mathbb{P} : \mathcal{W}_\Omega \to [0, 1]$ on the sample space Ω and the Lebesgue measure $\nu_d : \mathcal{A}_{\mathbb{R}^d} \mapsto [0, \infty)$ on Euclidean space $\mathbb{R}^d$, where $\mathcal{W}_\Omega$ and $\mathcal{A}_{\mathbb{R}^d}$ are σ-algebras of subsets of their respective subscripts.

Typical examples of regular functions involve real-valued functions that act on a domain $f : \mathcal{X} \to \mathbb{R}$ and random variables $X : \Omega \to \mathcal{X}$.

Typical examples of functionals are the expectation operator $\mathbb{E}$ that take in a function or function composition whose domain is a σ-field of subsets of the sample space Ω. For example, it takes in a function that maps from Ω to $\mathbb{R}$ such as a random variable, and outputs a real number in $\mathbb{R}$. A Lebesgue integral $\int_B \cdot d\nu$ with respect to a measure ν on some set B is also a functional.

Indicator Measures and Functions. Another example of a measure is the indicator measure $\mathbb{1} : \mathcal{W} \to \{0, 1\}$. We define $\mathbb{1}_A(X) \equiv \mathbb{1}_A \circ X := \mathbb{1}[X \in A]$ and $\mathbb{1}_A(x) := \mathbb{1}[x \in A]$. The latter is the indicator function, and is a regular function $\mathbb{1}_A : \mathcal{X} \to \{0, 1\}$, although it involves some abuse of notation since the shorthand for $x \in A$ is only defined if x is a random variable (and thus a mapping). Formally, we define the indicator function as

$$\mathbb{1}_A(x) := \begin{cases} 1 & : x \in A, \\ 0 & : x \notin A. \end{cases} \tag{2.9}$$

Probabilities of events W can be written as expectations of indicator measures $\mathbb{P}[W] = \mathbb{E}[\mathbb{1}[W]]$ for all events $W \in \mathcal{W}$. For example, the CDF for a vector-valued random variable $\mathbf{X} : \Omega \to \mathbb{R}^d$ is

$$\begin{aligned} P_{\mathbf{X}}(\mathbf{x}) &= \mathbb{P}[\mathbf{X} \leq \mathbf{x}] = \mathbb{E}[\mathbb{1}[\mathbf{X} \leq \mathbf{x}]] = \mathbb{E}[\mathbb{1}_{(-\infty, \mathbf{x}]}(\mathbf{X})] \\ &= \int_\Omega \mathbb{1}_{(-\infty, \mathbf{x}]} \circ \mathbf{X} d\mathbb{P} = \int_{\mathbb{R}^d} \mathbb{1}_{(-\infty, \mathbf{x}]} d\mathbb{P}_{\mathbf{X}} \\ &= \int_{(-\infty, \mathbf{x}]} d\mathbb{P}_{\mathbf{X}}. \end{aligned} \tag{2.10}$$

Datasets. A dataset consisting of n observations of d-dimensional vector quantities is often denoted using an upper case, such as X. It is a matrix of size $n \times d$ so that $X \in \mathbb{R}^{n \times d}$. As such, we often write $X = \{x_{ij}\}_{i=1,j=1}^{n,d} \equiv \{x_{ij}\}_{i,j=1}^{n,d}$, where x_{ij} is the j-th element of the i-th observation. We also denote each data point by

$$\mathbf{x}_i = \{x_{ij}\}_{j=1}^{d} \equiv \{x_{i1}, \ldots, x_{id}\} \equiv \begin{bmatrix} x_{i1} & \cdots & x_{id} \end{bmatrix}^T, \tag{2.11}$$

for all $i \in \mathbb{N}_n := \{1, \ldots, n\}$, and thus

$$X = \{\mathbf{x}_i\}_{i=1}^{n} \equiv \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \equiv \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_n \end{bmatrix}^T. \tag{2.12}$$

We use the notation $\{\cdot\}$ to stack components in the first axis. Consequently, stacking d scalars would turn it into a $\mathbb{R}^d$ vector (2.11), while stacking n $\mathbb{R}^d$ vectors will turn it into a $\mathbb{R}^{n\times d}$ matrix (2.12).

Rather than denoting a set, the notation $\{a_i\}_{i=1}^n$ denotes a generalized array since the stacking has a particular order determined by the enumerations specified from its subscript and superscript. We also define an alternative notation for $\mathbb{N}_n := \{1,\dots,n\} = \{i\}_{i=1}^n$ with the shorthand $[n] := \{1,\dots,n\} = \{i\}_{i=1}^n$, denoting the set of positive integers from 1 to the integer n in the subscript or bracket.

Unfortunately, the convention of using upper case letters for datasets clashes with that of random variables. Therefore, we will often avoid this notation for datasets and simply enumerate the dataset whenever possible such as by $\{\mathbf{x}_i\}_{i=1}^n$, even though it may introduce extra clutter. When appropriate, alternative choices for these notations will be made depending on the context.

2.2 Reproducing Kernel Hilbert Spaces

A reproducing kernel Hilbert space (RKHS) is space of functions whose properties, such as smoothness or stationarity, are governed by its respective kernel. Part of the beauty of RKHS theory is that functions can be treated and identified as vectors, whose notions and properties are directly transferable and applicable.

To build this intuition, consider for instance an usual vector $\mathbf{v} = \{v_j\}_{j=1}^d$ whose elements v_j are indexed by $j \in \mathbb{N}_d := \{1,\dots,d\}$. While vectors are identified by the collection of its elements, it can also be thought of as a function $v : \mathbb{N}_d \to \mathbb{F}$ with $v : j \mapsto v_j$, mapping an index j to its corresponding element v_j. In particular, the index positions of the vector view becomes the inputs of the function view. In this sense, the usual view of a vector is simply a full enumeration of all possible outputs of the function, indexed by the corresponding function inputs.

In the same way, a scalar-valued function $f : \mathcal{X} \to \mathbb{F}$ or $f : x \mapsto f(x)$ can also be thought of as an usual vector by enumerating all of its possible function outputs indexed by its corresponding function inputs $\{f(x)\}_{x\in\mathcal{X}}$. If $\mathcal{X}$ is a finite set, say $\mathcal{X} = \{x_l\}_{l=1}^m$, then the vector form is simply $\mathbf{f} = \{f(x_l)\}_{l=1}^m \in \mathbb{F}^m$. However, if $\mathcal{X}$ is a countably infinite set, say the set of all integers, or an uncountably infinite set, say the set of real numbers or a general Euclidean space, then it may not be immediately obvious if the vector view still stands as a valid concept, suggesting a more formal characterization of this intuition.

The above discussion sheds light and connects two representations or views, as an enumeration (vector) or a mapping (function), of the same particular object of

interest. Rather than the specifics of their representations, however, formal mathematical objects are characterized by their properties and applicable operations.

This leads us to the notion of a *vector space* $\mathcal{V}$ over a *field* $\mathbb{F}$.

Definition 2.5 (Field and Vector Space)**.** A *field* $\mathbb{F}$ is a set with two operations, addition $+$ and multiplication $\cdot$, and has two special elements, 0 and 1, such that for all $a, b, c \in \mathbb{F}$ the *field axioms* hold,

$$
\begin{aligned}
a + b &\in \mathbb{F}, && \text{(Closure under Addition)} \\
a \cdot b &\in \mathbb{F}, && \text{(Closure under Multiplication)} \\
a + b &= b + a, && \text{(Commutativity of Addition)} \\
(a + b) + c &= a + (b + c), && \text{(Associativity of Addition)} \\
\exists 0 \in \mathbb{F} \quad \text{s.t.} \quad a + 0 &= a, && \text{(Existence of Additive Identity)} \\
\exists (-a) \in \mathbb{F} \quad \text{s.t.} \quad a + (-a) &= 0, && \text{(Existence of Additive Inverse)} \\
a \cdot b &= b \cdot a, && \text{(Commutativity of Multiplication)} \\
(a \cdot b) \cdot c &= a \cdot (b \cdot c), && \text{(Associativity of Multiplication)} \\
\exists 1 \in \mathbb{F} \quad \text{s.t.} \quad 1 \cdot a &= a, && \text{(Existence of Multiplicative Identity)} \\
a \cdot (b + c) &= a \cdot b + a \cdot c. && \text{(Distributivity)}
\end{aligned}
$$

A *vector space* $\mathcal{V}$ over a field $\mathbb{F}$, notated as a tuple $(\mathcal{V}, \mathbb{F})$, is a set which satisfies the following properties for all vectors $u, v, w \in \mathcal{V}$ in the vector space and for all scalars $a, b \in \mathbb{F}$ in the associated field,

$$
\begin{aligned}
(u + v) + w &= u + (v + w), && \text{(Associativity of Superposition)} \\
u + v &= v + u, && \text{(Commutativity of Superposition)} \\
\exists 0 \in \mathcal{V} \quad \text{s.t.} \quad v + 0 &= v, && \text{(Existence of Zero Vector)} \\
\exists (-v) \in \mathcal{V} \quad \text{s.t.} \quad v + (-v) &= 0, && \text{(Existence of Opposite Vector)} \\
\exists 1 \in \mathbb{F} \quad \text{s.t.} \quad 1v &= v, && \text{(Existence of Unit Scalar)} \\
a(bv) &= (ab)v, && \text{(Associativity of Scaling)} \\
(a + b)v &= av + bv, && \text{(Distributivity of Superposed Scaling)} \\
a(u + v) &= au + av. && \text{(Distributivity of Scaled Superposition)}
\end{aligned}
$$

Typical example of a field include the set of real numbers $\mathbb{R}$ or the set of complex numbers $\mathbb{C}$. In this thesis, we will usually consider vectors spaces over real numbers and real-valued functions.

From definition 2.5, we can see that both the space of usual vectors $\mathbf{v} \in \mathbb{F}^d$ and functions $f : \mathcal{X} \to \mathbb{F}$ satisfy properties of a vector space. In particular, for spaces

of usual vectors such as $\mathcal{V} = \mathbb{F}^d$, they satisfy vector space properties in an element-wise fashion. For spaces of functions such as $\mathcal{V} = \{f | f : \mathcal{X} \to \mathbb{F}\}$, they satisfy vector space properties in a point-wise fashion.

Intuitively, just as adjacent elements of a vector have nothing to do with each other and need not need to be close, functions values of "adjacent" input points need not to be close either. Recall that index positions on an usual vector is akin to inputs to an usual function. For example, for an usual vector $\mathbf{v} \in \mathbb{F}^3$, the values of the first and second elements v_1 and v_2 need not be closer than that of the first and third elements v_1 and v_3, despite their index positions being closer. Similarly, for a function $f : \mathcal{X} \to \mathbb{R}$ from a vector space of functions, the values $f(x_1)$ and $f(x_2)$ need not be closer to the values $f(x_1)$ and $f(x_3)$, even if x_1 and x_2 are closer than x_1 and x_3. After all, the space of functions can satisfy vector space properties by merely satisfying them point-wise. Yet, this sense of "closeness" or "continuity" is something we would often like to endow on the space of functions we are considering. This is especially so when algorithms are to learn from a finite set of examples, and good generalization often depend on a sense of "continuity" of a function, where "close" inputs should result in "close" outputs.

The above consideration focuses on relationship between constituents – elements for vectors or point evaluations for functions – of one particular member in the vector space. Another important consideration is concerned with the overall structure of the vector space itself and how objects in that space relate to one another. These considerations motivates the notion of *distance metrics* for a sense of "closeness", *norms* for a sense of "size", and *inner products* for a sense of "similarity" as progressively desirable properties of a vector space, especially for vector space of functions. These notions lead to the construction of *metric spaces*, *normed spaces*, and *inner product spaces* respectively. It further motivates the notion of *completeness* for a sense of "continuity", intuitively describing spaces with no "punctures" in it. These notions lead to the construction of *complete metric spaces*, *Banach spaces*, and *Hilbert spaces* as *completed* versions of *metric spaces*, *normed spaces*, and *inner product spaces* respectively.

We now formally introduce these notions.

Any pair of two objects in a set, including vector spaces, can be endowed with a sense of "distance" through a *metric* or *distance* function in that space.

Definition 2.6 (Metric Space)**.** A *metric space* $(\mathcal{X}, d)$ is a non-empty set $\mathcal{X}$ with a metric $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that satisfies the following properties for all $x, y, z \in \mathcal{X}$,

$$\begin{aligned} &d(x,y) \geq 0, && \text{(Distance is Nonnegative)} \\ &d(x,y) = 0 \iff x = y, && \text{(Distance Vanishes for Equal Elements)} \\ &d(x,y) = d(y,x), && \text{(Distance is Symmetric)} \\ &d(x,y) \leq d(x,z) + d(y,z). && \text{(Triangle Inequality)} \end{aligned}$$

To identify "punctures" in a set, we consider every sequence of objects that get progressively closer and closer to each other as measured by a metric. If they all eventually converge somewhere in the space and not outside the space, then we can consider the set to be completely "filled" without any "punctures".

Definition 2.7 (Cauchy Sequence)**.** A sequence (x_i), $i \in \mathbb{N}$, in a space $\mathcal{X}$ is said to be *Cauchy* or *fundamental* with respect to a metric d if for any $\epsilon > 0$ there exists n such that

$$d(x_i, x_j) < \epsilon \quad \forall i, j > n. \tag{2.13}$$

Definition 2.8 (Complete Metric Space)**.** A metric space $(\mathcal{X}, d)$ is said to be *complete* if every Cauchy sequence in $\mathcal{X}$ with respect to d converges in $\mathcal{X}$.

Vectors can be endowed with a sense of size or length through a norm operator.

Definition 2.9 (Normed Space)**.** A *norm* on $\mathcal{V}$ is a real-valued function $\|\cdot\| : \mathcal{V} \to \mathbb{R}$ satisfying the following properties for all $u, v \in \mathcal{V}$ and $a \in \mathbb{F}$,

$$\begin{aligned} \|v\| = 0 &\iff v = 0, && \text{(Zero Vector has Zero Norm)} \\ \|v\| > 0 &\iff v \neq 0, && \text{(Non-Zero Vectors has Positive Norm)} \\ \|av\| &= |a|\|v\|, && \text{(Norm of Scaled Vector is Scaled Norm of Vector)} \\ \|u+v\| &\leq \|u\| + \|v\|. && \text{(Triangle Inequality)} \end{aligned}$$

A *normed space* $(\mathcal{V}, \mathbb{F}, \|\cdot\|)$ is a vector space with a norm $\|\cdot\|$.

Definition 2.10 (Banach Space)**.** A normed space $(\mathcal{V}, \mathbb{F}, \|\cdot\|)$ is a *Banach* space if $(\mathcal{V}, d)$ is a complete metric space with the distance function $d : (u, v) \mapsto \|u - v\|$.

Often, for brevity we refer to Banach spaces $(\mathcal{V}, \mathbb{F}, \|\cdot\|)$ by just $\mathcal{V}$ when it is clear from context what the associate field $\mathbb{F}$ and norm $\|\cdot\|$ are. We also often notate Banach spaces with $\mathcal{B}$ instead of $\mathcal{V}$.

Any pair of two vectors can be endowed with a sense of similarity through an inner product. This describes the geometry of the space by introducing the notions of projections and angles.

Definition 2.11 (Inner Product Space)**.** An *inner product* on a vector space $(\mathcal{V}, \mathbb{F})$ is a mapping $\langle \cdot, \cdot \rangle : \mathcal{V} \times \mathcal{V} \to \mathbb{F}$ satisfying the following properties for all $u, v, w \in \mathcal{V}$ and $a, b \in \mathbb{F}$,

$$\begin{aligned} \langle au + bv, w \rangle &= a\langle u, w \rangle + b\langle b, w \rangle, && \text{(Linearity in First Argument)} \\ \langle u, v \rangle &= \overline{\langle v, u \rangle}, && \text{(Conjugate Symmetry)} \\ \langle v, v \rangle &\geq 0, && \text{(Positivity)} \\ \langle v, v \rangle &= 0 \iff v = 0. && \text{(Zero Vector)} \end{aligned}$$

An *inner product space* or a *pre-Hilbert space* $(\mathcal{V}, \mathbb{F}, \langle \cdot, \cdot \rangle)$ is a vector space $(\mathcal{V}, \mathbb{F})$ with an inner product $\langle \cdot, \cdot \rangle$.

Definition 2.12 (Hilbert Space)**.** An inner product space $(\mathcal{V}, \mathbb{F}, \langle \cdot, \cdot \rangle)$ is a *Hilbert space* if $(\mathcal{V}, d)$ is a complete metric space with the distance function $d : (u, v) \mapsto \|u - v\|$ and the norm $\| \cdot \| : v \mapsto \sqrt{\langle v, v \rangle}$. That is, the metric $d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}$ is $d : (u, v) \mapsto \sqrt{\langle u - v, u - v \rangle}$.

Often, for brevity we refer to Hilbert spaces $(\mathcal{V}, \mathbb{F}, \langle \cdot, \cdot \rangle)$ by just $\mathcal{V}$ when it is clear from context what the associate field $\mathbb{F}$ and inner product $\langle \cdot, \cdot \rangle$ are. We also often notate Hilbert spaces with $\mathcal{H}$ instead of $\mathcal{V}$. We further notate inner products $\langle \cdot, \cdot \rangle$ as $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ when the associated Hilbert space $\mathcal{H}$ is to be emphasized or clarified.

We are now ready to introduce kernels. In this thesis, we use the following definition of a kernel.

Definition 2.13 (Kernel)**.** A mapping $k : \mathcal{X} \times \mathcal{X} \to \mathbb{F}$ is *kernel* on domain $\mathcal{X}$ if there exists a mapping $\phi : \mathcal{X} \to \mathcal{H}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle$ for all $x, x' \in \mathcal{X}$ where $(\mathcal{H}, \mathbb{F}, \langle \cdot, \cdot \rangle)$ is a Hilbert space. The mapping ϕ is called a *feature map* induced by k on the domain $\mathcal{X}$.

In this thesis we primarily consider real-valued kernels, as opposed to kernels that evaluates to a general field, such as complex-valued kernels, or those that evaluates to a general tensor, such as vector-valued kernels. That is, we assume that $\mathbb{F} = \mathbb{R}$. With the field always being the set of real-numbers, we no longer need to specify the field and from now on simply say that a Hilbert space $\mathcal{H}$ is endowed with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$. Since $\mathbb{F} = \mathbb{R}$, using the conjugate symmetry property from definition 2.11 on definition 2.13 immediately shows that real-valued kernels must be a symmetric functions of its two variables.

Kernels are functions of two variables that measure the similarity of its two inputs. To do this, inputs $x \in \mathcal{X}$ are represented as features $\phi(x) \in \mathcal{H}$ in a feature space $\mathcal{H}$. By assuming that the feature space $\mathcal{H}$ is a Hilbert space, kernels use the associated inner product to measure the notion of similarity in the sense that higher $\langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$ means that x and x' are more similar.

It is worthwhile to note that for a given kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the feature map ϕ may not be unique. That is, there could be multiple feature maps ϕ for which $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$. Polynomial kernels are typical examples of this. Consider the case where $\mathbf{x} \in \mathcal{X} = \mathbb{R}^2$. One can check that there are at least two feature maps defined by $\phi_1(\mathbf{x}) = \begin{bmatrix} x_1^2 & x_2^2 & \sqrt{2}x_1x_2 \end{bmatrix}^T$ and $\phi_2(\mathbf{x}) = \begin{bmatrix} x_1^2 & x_2^2 & x_1x_2 & x_2x_1 \end{bmatrix}^T$ that correspond to the same kernel $k(\mathbf{x}, \mathbf{x}') = \langle \phi_1(\mathbf{x}), \phi_1(\mathbf{x}') \rangle = \langle \phi_2(\mathbf{x}), \phi_2(\mathbf{x}') \rangle = (x_1x_1')^2 + (x_2x_2')^2 + 2x_1x_1'x_2x_2'$ where the inner products are the standard dot products in $\mathbb{R}^3$ and $\mathbb{R}^4$ respectively. Notice that in this example ϕ_2 was constructed by simply scaling the third feature of ϕ_1 into x_1x_2 and repeating it as the third and fourth feature of ϕ_2 by making sure the features are scaled correctly to result in the same kernel. With the same process, one could construct very high dimensional features that still result in the same kernel.

Moreover, unlike the previous example, there are also kernels whose feature maps are infinite dimensional. This includes the ubiquitous Gaussian kernel $k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^T\Sigma^{-1}(\mathbf{x} - \mathbf{x}')\right)$ and Matérn kernels.

In practice, when learning algorithms only make use of feature maps $\phi(x)$ via inner products $\langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$, we can instead simply compute the inner product directly using a kernel. This is known as the *kernel trick*. When the features are high dimensional, this saves computational time and memory by simply computing the inner product, which is a scalar, directly from inputs, instead of first computing the high dimensional features from the inputs only to finally reduce them into the inner product. More importantly, when the features are infinite dimensional, this enables inner product computations to become tractable in the case where it was otherwise infeasible.

Consequently, to "kernelize" a learning algorithm, a kernel instead of a feature map is selected, designed, or learned, and the corresponding feature map and feature space is only implicitly defined.

Since the feature map ϕ is specified only implicitly for a given kernel k and there may be multiple feature maps induced by that kernel k, it may be natural to question whether there exists a special feature map amongst them with convenient properties for learning or representing functions, a fundamental task in machine learning. After all, functions and vectors are simply two views of the object characterized by being members of a vector space. If we restrict the vector space to be a Hilbert space, we may be able to take advantage of the representational power of kernels, since the induced feature map ϕ defines a feature space $\mathcal{H}$ that is also a Hilbert space in which the function could reside in.

Let $\mathcal{H}$ be a Hilbert space of real-valued functions on domain $\mathcal{X}$. For a certain class of kernels known as *reproducing kernels*, such a convenient feature map exists.

Definition 2.14 (Reproducing Kernel)**.** A kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a *reproducing kernel* of a Hilbert space $\mathcal{H}$ if and only if

$$\begin{aligned} k(x, \cdot) &\in \mathcal{H} && \forall x \in \mathcal{X}, \\ \langle k(x, \cdot), f \rangle_{\mathcal{H}} &= f(x) && \forall x \in \mathcal{X} \quad \forall f \in \mathcal{H}. \end{aligned} \tag{2.14}$$

The latter criterion of which is also known as the *reproducing property*.

The dot notation $\cdot$ is a place-holder for an argument, so that the remaining object is a function of that argument. For instance, $k(x, \cdot) : \mathcal{X} \to \mathbb{R}$ or $k(x, \cdot) : x' \mapsto k(x, x')$.

Definition 2.15 (Reproducing Kernel Hilbert Space)**.** A Hilbert space $\mathcal{H}$ is a *reproducing kernel Hilbert space (RKHS)* if it has a reproducing kernel k.

When a Hilbert space $\mathcal{H}$ has a reproducing kernel, its evaluation functional $\delta_x : \mathcal{H} \to \mathbb{R}$ characterized by $\delta_x[f] = f(x)$ is simply $\delta_x = \langle k(x, \cdot), \cdot \rangle_{\mathcal{H}}$.

In particular, since $k(x, \cdot) \in \mathcal{H}$, the kernel value between two points can be evaluated as the inner product between the feature functions at those points,

$$\langle k(x, \cdot), k(x', \cdot) \rangle_{\mathcal{H}} = k(x, x'). \tag{2.15}$$

The above property (2.15) motivates the choice $\phi(x) = k(x, \cdot) \in \mathcal{H}$ as the canonical feature map when reproducing kernels of a Hilbert space exist, since it satisfies the definition of a feature map as per definition 2.13.

Definition 2.16 (Canonical Feature Map)**.** The *canonical feature map* ϕ of a RKHS $\mathcal{H}$ with reproducing kernel k is obtained by partially applying its reproducing kernel,

$$\phi(x) = k(x, \cdot) \in \mathcal{H}. \tag{2.16}$$

Since a reproducing kernel k determines a unique canonical feature map ϕ which in turn defines the feature space $\mathcal{H}$, we have that a reproducing kernel k defines or induces a canonical feature space $\mathcal{H}$. We therefore say that the RKHS is induced by k and use the notation $\mathcal{H}_k$ for the RKHS to emphasize this, and abbreviate the associated inner product as $\langle \cdot, \cdot \rangle_{\mathcal{H}_k} \equiv \langle \cdot, \cdot \rangle_k$.

Importantly, the canonical features $\phi(x) = k(x, \cdot)$ form a set of basis for its RKHS.

Theorem 2.2 (Xu and Zhang [2009])**.** *The RKHS $\mathcal{H}_k$ induced by its reproducing kernel k is the closure span of its canonical features,*

$$\mathcal{H}_k = \overline{\operatorname{span}\{k(x, \cdot) : x \in \mathcal{X}\}}. \tag{2.17}$$

In particular, this means that linear combinations of canonical features are in the RKHS, $\sum_{i=1}^{n} a_i k(x_i, \cdot) \in \mathcal{H}_k$ for $a_i \in \mathbb{R}$, although not all functions in the RKHS can be expressed in this form.

Usually, the kernel k is parametrized by a set of hyperparameters θ. Whenever it is relevant or helpful to make this explicit, we include the hyperparameters as a subscript k_θ. Note that while there are often multiple hyperparameters such that θ is often a vector, we will often still use a non-bold θ to emphasize and communicate the generality that kernel hyperparameters can come in non-vectorial forms too. In certain contexts where the symbol θ is already used for other concepts, we use α for k_α and β for ℓ_β.

2.3 Mean Embeddings

We now are ready to introduce *mean embeddings*, which are objects in the RKHS that represent and encode probability distributions.

Let $X : \Omega \to \mathcal{X}$ be a random variable taking values in $\mathcal{X}$. Similarly, let $Y : \Omega \to \mathcal{Y}$ be a random variable taking values in $\mathcal{Y}$. Let $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ be reproducing kernels defined on $\mathcal{X}$ and $\mathcal{Y}$ respectively (definition 2.14). Let ϕ and ψ be canonical feature maps induced by k and ℓ respectively, where $\phi(x) = k(x, \cdot)$ and $\psi(y) = \ell(y, \cdot)$ (definition 2.16). Let $f \in \mathcal{H}_k$ and $g \in \mathcal{H}_\ell$ be real-valued functions in the RKHSs $\mathcal{H}_k$ and $\mathcal{H}_\ell$ induced by k and ℓ respectively (definition 2.15).

Definition 2.17 (Mean Embedding)**.** The mean embedding of a distribution $\mathbb{P}_X$ (resp. $\mathbb{P}_Y$) under a kernel k (resp. ℓ) is defined as the expectation of the canonical feature in its induced RKHS $\mathcal{H}_k$ (resp. $\mathcal{H}_\ell$) with respect to said distribution,

$$\begin{aligned} \mu_X &:= \mathbb{E}[\phi(X)] \in \mathcal{H}_k, \\ \mu_Y &:= \mathbb{E}[\psi(Y)] \in \mathcal{H}_\ell. \end{aligned} \tag{2.18}$$

We can alternatively emphasize that the mean embedding $\mu_X \equiv \mu_X(\cdot)$ (resp. $\mu_Y \equiv \mu_Y(\cdot)$) is a function on its respective domain $\mu_X : \mathcal{X} \to \mathbb{R}$ (resp. $\mu_Y : \mathcal{Y} \to \mathbb{R}$),

$$\begin{aligned} \mu_X(\cdot) &:= \mathbb{E}[k(X, \cdot)] \in \mathcal{H}_k, \\ \mu_Y(\cdot) &:= \mathbb{E}[\ell(Y, \cdot)] \in \mathcal{H}_\ell. \end{aligned} \tag{2.19}$$

Since it is the expectation of the kernel with respect to one of its argument, it is also called the *kernel mean*, or the kernel mean embedding (KME). Alternatively,

if the integral exists, this definition is equivalent to

$$\begin{aligned}\mu_X(x) &:= \mathbb{E}[k(X,x)] = \int_{\mathcal{X}} k(\cdot,x)d\mathbb{P}_X,\\ \mu_Y(y) &:= \mathbb{E}[\ell(Y,y)] = \int_{\mathcal{Y}} \ell(\cdot,y)d\mathbb{P}_Y.\end{aligned} \tag{2.20}$$

The dot notation $\cdot$ is a place-holder for an argument. In equation (2.18) it emphasizes that the mean embedding $\mu_X(\cdot) : \mathcal{X} \to \mathbb{R}$ (resp. $\mu_Y(\cdot) : \mathcal{Y} \to \mathbb{R}$) is a function in the RKHS. In equation (2.20) the place holder in equation (2.19) is filled with a specific evaluation $x \in \mathcal{X}$ (resp. $y \in \mathcal{Y}$). Note that the dot notation $\cdot$ in equation (2.20) is for a different argument than equation (2.19), as this argument is to be integrated out in the Bochner integral written out above.

In some literature, to make explicit of the fact that the mean embedding is an embedding of a distribution measure $\mathbb{P}_X$, the mean embedding is instead denoted by $\mu_{\mathbb{P}_X}$. Nevertheless, since a random variable X and its distribution $\mathbb{P}_X$ are specified together, either notation is equally informative.

When densities exist as per theorem 2.1, then mean embeddings can be written as

$$\begin{aligned}\mu_X &:= \mathbb{E}[k(X,\cdot)] = \int_{\mathcal{X}} k(x,\cdot)p_X(x)dx,\\ \mu_Y &:= \mathbb{E}[\ell(Y,\cdot)] = \int_{\mathcal{Y}} \ell(y,\cdot)p_Y(y)dy.\end{aligned} \tag{2.21}$$

Importantly, mean embeddings equation (2.18) do not rely on the existence of densities. In this sense, mean embeddings can be applied in general settings where quantities of interest can be expressed as expectations, alleviating the need for performing often intractable integrals with respect to densities. A canonical example include finding expectations of functions under these encoded distributions.

Theorem 2.3 (Function Expectation)**.** *The expectation of a function $f \in \mathcal{H}_k$ (resp. $g \in \mathcal{H}_\ell$) of a random variable X (resp. Y) with distribution $\mathbb{P}_X$ (resp. $\mathbb{P}_Y$) can be evaluated as the inner product between the corresponding mean embedding and the function,*

$$\begin{aligned}\langle \mu_X, f\rangle_k &= \mathbb{E}[f(X)],\\ \langle \mu_Y, g\rangle_\ell &= \mathbb{E}[g(Y)].\end{aligned} \tag{2.22}$$

Proof. Consider the inner product,

$$\begin{aligned}\langle \mu_X, f\rangle_k &= \langle \mathbb{E}[\phi(X)], f\rangle_k && \text{(definition 2.17)}\\ &= \mathbb{E}[\langle \phi(X), f\rangle_k] && \text{(bilinearity of } \langle\cdot,\cdot\rangle \text{ and linearity of } \mathbb{E}[\cdot])\\ &= \mathbb{E}[f(X)]. && \text{(definition 2.14)}\end{aligned}$$

The remaining equation in (2.22) follows by a parallel derivation. □

We call the kernel *characteristic* if the mapping from probability measures $\mathbb{P}_X$ (resp. $\mathbb{P}_Y$) to their mean embeddings μ_X (resp. μ_Y) in the associated RKHS $\mathcal{H}_k$ (resp. $\mathcal{H}_\ell$) is injective. In this sense, the mean embedding is said to encode or represent the corresponding distribution in the RKHS, since distributions can be characterized by its mean embedding. Intuitively, having the ability to compute expectations of any function in the RKHS under the distribution of interest, as well as knowing the kernel inducing the RKHS, is as informative and useful as knowing the distribution itself.

The characteristic property of kernels is heavily interlinked to the notions of universality and positive definiteness, although implications between them vary depending on the class of kernels chosen [Sriperumbudur et al., 2011]. Loosely, universality is concerned with whether the RKHS induced by the kernel in consideration is rich enough to approximate any target function arbitrarily well, while positive definiteness enables desirable properties such as symmetry and invertibility of operators. In this thesis, we will often assume and employ kernel classes that have all these properties.

2.4 Conditional Mean Embeddings

We now introduce conditional mean embeddings (CMEs), which are objects in the RKHS that represent and encode conditional probability distributions. By definition, they are a type of mean embedding whose encoded distribution is conditional, rather than marginal. As such, results that pertain to mean embeddings hold directly for CMEs. When we want to emphasize our discussions specifically on mean embeddings that encode marginal distributions, as opposed to conditional distributions, then we use the terminology *marginal mean embeddings*.

Given that CMEs are a type of mean embeddings, what interesting properties do they have that we could not obtain by viewing them simply as a type of mean embedding? After all, the mean embedding of $\mathbb{P}_X$ is μ_X, so the mean embedding of $\mathbb{P}_{X|Y=y}$ is $\mu_{X|Y=y}$, and we simply call this the CME.

Frameworks based on the CME become interesting in the case when the conditioned variable is seen also as a variable of interest, as opposed to fixed. This includes instances where instead of having one particular distribution $\mathbb{P}_{X|Y=y}$, we have a family of distributions $\mathbb{P}_{X|Y} := \{\mathbb{P}_{X|Y=y}\}_{y\in\mathcal{Y}}$ indexed by the conditioned variable. In other words, we consider distributions conditioned across a family of events, and not just a single event. Another interesting instance is when the conditioned variable itself also has a distribution $\mathbb{P}_Y$ specified completely separately

or independently from the family of conditional distributions $\mathbb{P}_{X|Y}$, describing the distribution of the family of events being conditioned.

Consequently, it is very helpful and constructive to form operator views of CMEs. In this view, instead of seeing them as mean embeddings of conditional distributions, they can be further decomposed into components formed from mean embeddings of marginal distributions only, where these mean embeddings can be seen as higher order operators. This is analogous to the way conditional densities can be decomposed into components formed from only marginal densities such as $p_{X|Y}(x|y) = p_{XY}(x, y)/p_Y(x)$. In fact, these analogies can be made formal to express probability rules in the RKHS with such operators, and is a key motivation and element of the KME framework.

In this section, we will review CMEs in a fashion that is faithful to the manner they are presented in the literature [Muandet et al., 2017, Song et al., 2013, 2009]. Importantly, in each subsequent chapter, we will briefly review CMEs again within the context of their problem setting. This is because our contributions begin by carefully formulating perspectives to the problem setting such that CMEs become their natural solutions. In particular, chapter 5 will reformulate CMEs slightly, using a different set of definitions which result in an equivalent set of properties for the CME. This slight reformulation would motivate insights that are important to the development and contributions presented in chapter 5.

We begin with an introduction to cross-covariance operators and operator notations in general, especially for operators formed from second-order tensors.

Definition 2.18 (Tensor Products as Operators)**.** Suppose $a \in \mathcal{H}_1$ and $b \in \mathcal{H}_2$. Applying the tensor product $(a \otimes b) \in \mathcal{H}_1 \otimes \mathcal{H}_2$ on $c \in \mathcal{H}_2$ results in an element in $\mathcal{H}_1$ defined by

$$(a \otimes b)c := a\langle b, c\rangle_{\mathcal{H}_2} = \langle b, c\rangle_{\mathcal{H}_2} a \in \mathcal{H}_1. \tag{2.23}$$

Definition 2.19 (Inner Products of Tensor Products)**.** Suppose $a, c \in \mathcal{H}_1$ and $b, d \in \mathcal{H}_2$. The inner product of the two tensor products are defined to be the product of the corresponding component inner products,

$$\langle a \otimes b, c \otimes d\rangle_{\mathcal{H}_1 \otimes \mathcal{H}_2} := \langle a, c\rangle_{\mathcal{H}_1} \langle b, d\rangle_{\mathcal{H}_2}. \tag{2.24}$$

We will refer to the order of the tensor product by the number of vectors, in the Hilbert space sense, that is used to form the tensor product. For instance, $a \otimes b \otimes c \otimes d$ would be a fourth order tensor. When only second-order tensors are involved, tensors act analogously to usual matrices and vectors in the general Hilbert space sense act as usual vectors. Consequently, when only second-order tensors are involved and the Hilbert spaces involved are clear from context, we use the slight abuse of notation $ab^T = a \otimes b$ and $b^T c = \langle b, c\rangle_{\mathcal{H}}$. This becomes

helpful for writing linear combinations of tensor products succinctly using matrix notation, which will be appear when empirical estimates are introduced.

Definition 2.20 (Uncentered Cross-Covariance Operators)**.** The uncentered cross-covariance operators, or cross-covariance operators for short, are second-order mean embeddings,

$$\begin{aligned} C_{XX} &:= \mathbb{E}[\phi(X) \otimes \phi(X)] \in \mathcal{H}_k \otimes \mathcal{H}_k, \\ C_{XY} &:= \mathbb{E}[\phi(X) \otimes \psi(Y)] \in \mathcal{H}_k \otimes \mathcal{H}_\ell, \\ C_{YX} &:= \mathbb{E}[\psi(Y) \otimes \phi(X)] \in \mathcal{H}_\ell \otimes \mathcal{H}_k, \\ C_{YY} &:= \mathbb{E}[\psi(Y) \otimes \psi(Y)] \in \mathcal{H}_\ell \otimes \mathcal{H}_\ell. \end{aligned} \tag{2.25}$$

Since $\phi(x) := k(x, \cdot)$ and $\psi(y) := \ell(y, \cdot)$ as per definition 2.16, we can alternatively emphasize that the uncentered cross-covariance operators are functions of two variables on the tensor products of two domains,

$$\begin{aligned} C_{XX}(\cdot, \cdot) &:= \mathbb{E}[k(X, \cdot) \otimes k(X, \cdot)] \in \mathcal{H}_k \otimes \mathcal{H}_k, \\ C_{XY}(\cdot, \cdot) &:= \mathbb{E}[k(X, \cdot) \otimes \ell(Y, \cdot)] \in \mathcal{H}_k \otimes \mathcal{H}_\ell, \\ C_{YX}(\cdot, \cdot) &:= \mathbb{E}[\ell(Y, \cdot) \otimes k(X, \cdot)] \in \mathcal{H}_\ell \otimes \mathcal{H}_k, \\ C_{YY}(\cdot, \cdot) &:= \mathbb{E}[\ell(Y, \cdot) \otimes \ell(Y, \cdot)] \in \mathcal{H}_\ell \otimes \mathcal{H}_\ell. \end{aligned} \tag{2.26}$$

Alternatively, this definition is equivalent to

$$\begin{aligned} C_{XX}(x, x') &:= \mathbb{E}[k(X, x)k(X, x')] = \int_{\mathcal{X}} k(\cdot, x)k(\cdot, x')d\mathbb{P}_X, \\ C_{XY}(x, y) &:= \mathbb{E}[k(X, x)\ell(Y, y)] = \int_{\mathcal{X}\times\mathcal{Y}} k(\cdot, x)\ell(\cdot, y)d\mathbb{P}_{XY}, \\ C_{YX}(y, x) &:= \mathbb{E}[\ell(Y, y)k(X, x)] = \int_{\mathcal{Y}\times\mathcal{X}} \ell(\cdot, y)k(\cdot, x)d\mathbb{P}_{YX}, \\ C_{YY}(y, y') &:= \mathbb{E}[\ell(Y, y)\ell(Y, y')] = \int_{\mathcal{Y}} \ell(\cdot, y)\ell(\cdot, y')d\mathbb{P}_Y. \end{aligned} \tag{2.27}$$

In the above, the placeholders $\cdot$ are integrated out under the probability measure. When densities exist as per theorem 2.1, then uncentered cross-covariance operators can be written as

$$\begin{aligned} C_{XX}(\cdot, \cdot) &:= \mathbb{E}[k(X, \cdot)k(X, \cdot)] = \int_{\mathcal{X}} k(x, \cdot)k(x, \cdot)p_X(x)dx, \\ C_{XY}(\cdot, \cdot) &:= \mathbb{E}[k(X, \cdot)\ell(Y, \cdot)] = \int_{\mathcal{X}}\int_{\mathcal{Y}} k(x, \cdot)\ell(y, \cdot)p_{XY}(x, y)dydx, \\ C_{XY}(\cdot, \cdot) &:= \mathbb{E}[\ell(Y, \cdot)k(X, \cdot)] = \int_{\mathcal{Y}}\int_{\mathcal{X}} \ell(y, \cdot)k(x, \cdot)p_{YX}(y, x)dxdy, \\ C_{YY}(\cdot, \cdot) &:= \mathbb{E}[\ell(Y, \cdot)\ell(Y, \cdot)] = \int_{\mathcal{Y}} \ell(y, \cdot)\ell(y, \cdot)p_Y(y)dy. \end{aligned} \tag{2.28}$$

In the above, placeholders $\cdot$ are left open for evaluation. For instance, evaluating C_{XY} gives $C_{XY}(x', y') := \mathbb{E}[k(X, x')\ell(Y, y')] = \int_{\mathcal{X}} \int_{\mathcal{Y}} k(x, x')\ell(y, y')p_{XY}(x, y)dydx$. While $p_{XY} \neq p_{YX}$ since $p_{XY}(a, b) \neq p_{YX}(a, b)$, we have that $p_{XY}(a, b) = p_{YX}(b, a)$ so that $p_{XY}(x, y) = p_{YX}(y, x)$. Here $\neq$ denotes "is not necessarily equal to".

Since cross-covariance operators are simply second-order mean embeddings, we also occasionally notate them as as mean embedding such as $\mu_{XX} \equiv C_{XX}$ and $\mu_{XY} \equiv C_{XY}$. Nevertheless, they are most useful when viewed as operators.

Lemma 2.1. *$C_{XY} \in \mathcal{H}_k \otimes \mathcal{H}_\ell$ can be identified as the operator $C_{XY} : \mathcal{H}_\ell \to \mathcal{H}_k$ where for all $g \in \mathcal{H}_\ell$,*

$$C_{XY}g = \mathbb{E}[g(Y)\phi(X)] \in \mathcal{H}_k. \tag{2.29}$$

Proof. The result follows from treating C_{XY} as an operator under definition 2.18,

$$\begin{aligned} C_{XY}g &= \mathbb{E}[\phi(X) \otimes \psi(Y)]g && \text{(definition 2.20)} \\ &= \mathbb{E}[(\phi(X) \otimes \psi(Y))g] && \text{(linearity of } \mathbb{E}[\cdot]) \\ &= \mathbb{E}[\phi(X)\langle\psi(Y), g\rangle_\ell] && \text{(definition 2.18)} \\ &= \mathbb{E}[g(Y)\phi(X)]. && \text{(definition 2.14)} \end{aligned}$$

□

Corollary 2.1. *Consequently, $C_{XY} : \mathcal{H}_\ell \to \mathcal{H}_k$ encodes the uncentered cross-covariance between any pair of functions $f \in \mathcal{H}_k$ and $g \in \mathcal{H}_\ell$,*

$$\langle C_{XY}g, f\rangle_k = \mathbb{E}[f(X)g(Y)]. \tag{2.30}$$

Proof. The result follows from direct application of the inner product,

$$\begin{aligned} \langle C_{XY}g, f\rangle_k &= \langle \mathbb{E}[g(Y)\phi(X)], f\rangle_k && \text{(lemma 2.1)} \\ &= \mathbb{E}[g(Y)\langle\phi(X), f\rangle_k] && \text{(bilinearity of } \langle\cdot,\cdot\rangle) \\ &= \mathbb{E}[f(X)g(Y)]. && \text{(definition 2.14)} \end{aligned}$$

□

Lemma 2.2. *Alternatively, since $C_{XY} \in \mathcal{H}_k \otimes \mathcal{H}_\ell$, the inner product of C_{XY} and an outer product of functions $f \otimes g \in \mathcal{H}_k \otimes \mathcal{H}_\ell$ result in the same uncentered cross-covariance,*

$$\langle C_{XY}, f \otimes g\rangle_{k\otimes\ell} = \mathbb{E}[f(X)g(Y)], \tag{2.31}$$

where $\langle\cdot,\cdot\rangle_{k\otimes\ell}$ is shorthand for $\langle\cdot,\cdot\rangle_{\mathcal{H}_k\otimes\mathcal{H}_\ell}$.

Proof. The result follows from definition 2.19 of inner products on tensor products,

$$\begin{aligned}
\langle C_{XY}, f \otimes g \rangle_{k\otimes\ell} &= \langle \mathbb{E}[\phi(X) \otimes \psi(Y)], f \otimes g \rangle_{k\otimes\ell} && \text{(definition 2.20)} \\
&= \mathbb{E}[\langle \phi(X) \otimes \psi(Y), f \otimes g \rangle_{k\otimes\ell}] && \text{(bilinearity of } \langle\cdot,\cdot\rangle) \\
&= \mathbb{E}[\langle \phi(X), f \rangle_k \langle \psi(Y), g \rangle_\ell] && \text{(definition 2.19)} \\
&= \mathbb{E}[f(X)g(Y)]. && \text{(definition 2.14)}
\end{aligned}$$

□

Together, corollary 2.1 and lemma 2.2 reveal multiple ways the cross-covariance operator C_{XY} encodes the cross-covariance of functions,

$$\begin{aligned}
\langle C_{XY} g, f \rangle_k = \langle C_{XY}, f \otimes g \rangle_{k\otimes\ell} = \langle C_{YX} f, g \rangle_\ell \\
= \mathbb{E}[f(X)g(Y)].
\end{aligned} \tag{2.32}$$

where the second equality is obtained through symmetry. In particular, corollary 2.1 provides the view that cross-covariance operators are operators from one space to the other, while lemma 2.2 presents the view that they are mean embeddings in the joint or tensor space.

Nevertheless, there are relationships that cross-covariance operators provide that are only evident in the operator view. This is the case where we move our focus to functions formed through expectations.

Theorem 2.4 (Fukumizu et al. [2009])**.** *Define f in relation to g as follows and assume $f \in \mathcal{H}_k$,*

$$f := \mathbb{E}[g(Y)|X = \cdot] \in \mathcal{H}_k. \tag{2.33}$$

Then, the following result hold,

$$C_{XX} f = C_{XY} g. \tag{2.34}$$

Further, if the inverse of the operator C_{XX} exists, then we have that

$$f := \mathbb{E}[g(Y)|X = \cdot] = C_{XX}^{-1} C_{XY} g. \tag{2.35}$$

Proof. The result follows from treating C_{XX} as an operator under definition 2.18,

$$\begin{aligned}
C_{XX}f &= \mathbb{E}[\phi(X) \otimes \phi(X)]f && \text{(definition 2.20)} \\
&= \mathbb{E}[(\phi(X) \otimes \phi(X))f] && \text{(linearity of } \mathbb{E}[\cdot]) \\
&= \mathbb{E}[\phi(X)\langle \phi(X), f\rangle_k] && \text{(definition 2.18)} \\
&= \mathbb{E}[\phi(X)f(X)] && \text{(definition 2.14)} \\
&= \mathbb{E}[\phi(X)\mathbb{E}[g(Y)|X]] && \text{(equation (2.33))} \\
&= \mathbb{E}[\mathbb{E}[\phi(X)g(Y)|X]] && (X \text{ is given}) \\
&= \mathbb{E}[\phi(X)g(Y)] && \text{(tower property)} \\
&= C_{XY}g. && \text{(lemma 2.1)}
\end{aligned}$$

If the inverse of the operator C_{XX} exists, then we can write the following,

$$\begin{aligned}
C_{XX}f &= C_{XY}g, && \text{(equation (2.34))} \\
f &= C_{XX}^{-1}C_{XY}g, && (C_{XX}^{-1} \text{ exists}) \\
\mathbb{E}[g(Y)|X = \cdot] &= C_{XX}^{-1}C_{XY}g. && \text{(equation (2.33))}
\end{aligned}$$

□

As a function in $\mathcal{H}_k$ which contains real-valued functions on $\mathcal{X}$, the evaluation of $f : \mathbb{E}[g(Y)|X = \cdot]$ on x is $f(x) = \mathbb{E}[g(Y)|X = x]$.

Definition 2.21 (Conditional Mean Embedding)**.** The conditional mean embedding (CME) of a conditional distribution $\mathbb{P}_{Y|X=x}$ (resp. $\mathbb{P}_{X|Y=y}$) under a kernel ℓ (resp. k) is defined as the expectation of the canonical feature in its induced RKHS $\mathcal{H}_\ell$ (resp. $\mathcal{H}_k$) with respect to said distribution,

$$\begin{aligned}
\mu_{Y|X=x} &:= \mathbb{E}[\psi(Y)|X = x], \\
\mu_{X|Y=y} &:= \mathbb{E}[\phi(X)|Y = y].
\end{aligned} \tag{2.36}$$

We can alternatively emphasize that the CME $\mu_{Y|X=x} \equiv \mu_{Y|X=x}(\cdot)$ (resp. $\mu_{X|Y=y} \equiv \mu_{X|Y=y}(\cdot)$) is a function on its respective domain $\mu_{Y|X=x} : \mathcal{Y} \to \mathbb{R}$ (resp. $\mu_{X|Y=y} : \mathcal{X} \to \mathbb{R}$),

$$\begin{aligned}
\mu_{Y|X=x}(\cdot) &:= \mathbb{E}[\ell(Y, \cdot)|X = x], \\
\mu_{X|Y=y}(\cdot) &:= \mathbb{E}[k(X, \cdot)|Y = y].
\end{aligned} \tag{2.37}$$

Since it is the conditional expectation of the kernel with respect to one of its argument, it is also called the *conditional kernel mean*, or the conditional kernel mean embedding. Alternatively, this definition is equivalent to

$$\begin{aligned}
\mu_{Y|X=x}(y) &:= \mathbb{E}[\ell(Y, y)|X = x] = \int_{\mathcal{Y}} \ell(\cdot, y) d\mathbb{P}_{Y|X=x}, \\
\mu_{X|Y=y}(x) &:= \mathbb{E}[k(X, x)|Y = y] = \int_{\mathcal{X}} k(\cdot, x) d\mathbb{P}_{X|Y=y}.
\end{aligned} \tag{2.38}$$

When densities exist as per theorem 2.1, then CMEs can be written as

$$\begin{aligned}\mu_{Y|X=x} &:= \mathbb{E}[\ell(Y,\cdot)|X=x] = \int_{\mathcal{Y}} \ell(y,\cdot)p_{Y|X}(y|x)dy,\\ \mu_{X|Y=y} &:= \mathbb{E}[k(X,\cdot)|Y=y] = \int_{\mathcal{X}} k(x,\cdot)p_{X|Y}(x|y)dx.\end{aligned} \tag{2.39}$$

Theorem 2.5 (Function Conditional Expectation)**.** *The conditional expectation of a function can be evaluated as the inner product between the corresponding CME and the function, which can also be obtained by applying cross-covariance operators to the function and then evaluating that function,*

$$\begin{aligned}\langle \mu_{Y|X=x}, g\rangle_{\ell} &= \mathbb{E}[g(Y)|X=x] = (C_{XX}^{-1}C_{XY}g)(x),\\ \langle \mu_{X|Y=y}, f\rangle_{k} &= \mathbb{E}[f(X)|Y=y] = (C_{YY}^{-1}C_{YX}f)(y).\end{aligned} \tag{2.40}$$

Proof. The proof follows from the reproducing property and theorem 2.4.

$$\begin{aligned}\langle \mu_{Y|X=x}, g\rangle &= \langle \mathbb{E}[\psi(Y)|X=x], g\rangle && \text{(definition 2.21)}\\ &= \mathbb{E}[\langle \psi(Y), g\rangle|X=x] && \text{(bilinearity of } \langle\cdot,\cdot\rangle)\\ &= \mathbb{E}[g(Y)|X=x] && \text{(definition 2.14)}\\ &= (\mathbb{E}[g(Y)|X=\cdot])(x) && \text{(notation: } f\equiv f(\cdot))\\ &= (C_{XX}^{-1}C_{XY}g)(x). && \text{(theorem 2.4)}\end{aligned}$$

□

Definition 2.22 (Conditional Mean Operator)**.** The conditional mean operator (CMO) $C_{Y|X}$ is defined by the operator that maps the canonical feature of the conditioned variable to the conditional mean embedding,

$$\mu_{Y|X=x} = C_{Y|X}\phi(x). \tag{2.41}$$

Theorem 2.6 (Conditional Mean Operator)**.** *Assuming C_{XX}^{-1} exists, the conditional mean operator $C_{Y|X}$ can be expressed in terms of cross-covariance operators,*

$$C_{Y|X} = C_{YX}C_{XX}^{-1}. \tag{2.42}$$

Proof. The proof relies on the tower property $\mathbb{E}[\mathbb{E}[A|B]] = \mathbb{E}[A]$. Importantly, $\mathbb{E}[A|B]$ is a random variable with its distribution determined by B. We use the notation $\mu_{Y|X}$ in a similar manner, which describes a random variable $\omega \mapsto \mu_{Y|X=X(\omega)}$

TABLE 2.1: Mean embeddings and their encoded expectations. Switch $X \leftrightarrow Y$ for all combinations. Since the bottom two rows do not apply for the first column, additional equivalences for the last column are provided instead. Let $g, g' \in \mathcal{H}_\ell$ and $f \in \mathcal{H}_k$ be generic example functions in their respective RKHSs.

Random Variable	Y $: \Omega \to \mathcal{Y}$	(Y, Y) $: \Omega \to \mathcal{Y} \times \mathcal{Y}$	(X, Y) $: \Omega \to \mathcal{X} \times \mathcal{Y}$	$X\|Y = y$ $: \Omega \to \mathcal{X}$
Density Function	$p_Y \in \mathcal{P}_\mathcal{Y}$ $p_Y(y) \in \mathbb{R}^+$	$p_{YY} \in \mathcal{P}_{\mathcal{Y}\times\mathcal{Y}}$ $p_{YY}(y, y') \in \mathbb{R}^+$	$p_{XY} \in \mathcal{P}_{\mathcal{X}\times\mathcal{Y}}$ $p_{XY}(x, y) \in \mathbb{R}^+$	$p_{X\|Y=y} \in \mathcal{P}_\mathcal{X}$ $p_{X\|Y=y}(x) \in \mathbb{R}^+$
Mean Map Definition	$\mu_Y :=$ $\mathbb{E}[\ell(Y, \cdot)]$	$\mu_{YY} :=$ $\mathbb{E}[\ell(Y, \cdot) \otimes \ell(Y, \cdot)]$	$\mu_{XY} :=$ $\mathbb{E}[k(X, \cdot) \otimes \ell(Y, \cdot)]$	$\mu_{X\|Y=y} =$ $\mathbb{E}[k(X, \cdot)\|Y = y]$
Mean Embedding	$\mu_Y \in \mathcal{H}_\ell$ $\mu_Y(y) \in \mathbb{R}$	$\mu_{YY} \in \mathcal{H}_{\ell\ell}$ $\mu_{YY}(y, y') \in \mathbb{R}$	$\mu_{XY} \in \mathcal{H}_{k\ell}$ $\mu_{XY}(x, y) \in \mathbb{R}$	$\mu_{X\|Y=Y} \in \mathcal{H}_k$ $\mu_{X\|Y=y}(x) \in \mathbb{R}$
Encoded Expectation	$\langle \mu_Y, g \rangle_\ell$ $= \mathbb{E}[g(Y)]$	$\langle \mu_{YY}, g' \otimes g \rangle_{\ell\otimes\ell}$ $= \mathbb{E}[g'(Y)g(Y)]$	$\langle \mu_{XY}, f \otimes g \rangle_{k\otimes\ell}$ $= \mathbb{E}[f(X)g(Y)]$	$\langle \mu_{X\|Y=y}, f \rangle_k$ $= \mathbb{E}[f(X)\|Y = y]$
Operator Definition	$C_{X\|Y} C_{YY}$ $= C_{XY}$	$C_{YY} := \mu_{YY}$ $(C_{YY})^T = C_{YY}$	$C_{XY} := \mu_{XY}$ $(C_{XY})^T = C_{YX}$	$C_{X\|Y} \ell(y, \cdot) :=$ $\mu_{X\|Y=y}$
Encoded Expectation	$f^T C_{XY} =$ $g^T C_{YY}$	$\langle g', C_{YY} g \rangle_\ell$ $= \mathbb{E}[g'(Y)g(Y)]$	$\langle f, C_{XY} g \rangle_k$ $= \mathbb{E}[f(X)g(Y)]$	$(C_{X\|Y})^T f = g :=$ $\mathbb{E}[f(X)\|Y = \cdot]$

taking values in $\mathcal{H}_\ell$. With such a slight abuse of notation, we arrive at the result,

$$
\begin{aligned}
C_{YX} :=& \mathbb{E}[\psi(Y) \otimes \phi(X)] && \text{(definition 2.20)} \\
=& \mathbb{E}[\mathbb{E}[\psi(Y) \otimes \phi(X)|X]] && \text{(tower property)} \\
=& \mathbb{E}[\mathbb{E}[\psi(Y)|X] \otimes \phi(X)] && \text{(X is given)} \\
=& \mathbb{E}[\mu_{Y|X} \otimes \phi(X)] && \text{(definition 2.21)} \\
=& \mathbb{E}[(C_{Y|X}\phi(X)) \otimes \phi(X)] && \text{(definition 2.22)} \\
=& \mathbb{E}[C_{Y|X}(\phi(X) \otimes \phi(X))] && \text{(associativity)} \\
=& C_{Y|X}\mathbb{E}[\phi(X) \otimes \phi(X)] && \text{(linearity of } \mathbb{E}[\cdot]) \\
=& C_{Y|X} C_{XX}. && \text{(definition 2.20)}
\end{aligned}
$$

Therefore, if C_{XX}^{-1} exists, then (2.42) holds. □

Table 2.1 review mean embeddings and operators with their encoded expectations.

2.5 Empirical Mean Embeddings

In practice, the actual probability distributions of interest are not available in closed form. Instead, independent and identically distributed (*iid*) samples from such probability distributions are available. Suppose that marginal samples $\{x_i\}_{i=1}^n$

and $\{y_i\}_{i=1}^n$ are observed and collected in an *iid* fashion from the marginal distributions $\mathbb{P}_X$ and $\mathbb{P}_Y$. It is possible to represent marginal mean embeddings empirically such that in the limit of infinite data, the empirical representations would converge to the true representations at an appropriate rate. Most significant, however, is the fact that important results for the relationships between mean embeddings also hold for their empirical counterparts.

Definition 2.23 (Empirical Mean Embedding)**.** The empirical mean embedding of an empirical distribution $\hat{\mathbb{P}}_X$ (resp. $\hat{\mathbb{P}}_Y$) described by a set of *iid* samples $\{x_i\}_{i=1}^n$ (resp. $\{y_i\}_{i=1}^n$) from $\mathbb{P}_X$ (resp. $\mathbb{P}_Y$) under a kernel k (resp. ℓ) is defined by the empirical mean of the canonical feature in its induced RKHS $\mathcal{H}_k$ (resp. $\mathcal{H}_\ell$) with respect to said distribution,

$$\begin{aligned} \hat{\mu}_X &:= \frac{1}{n}\sum_{i=1}^n \phi(x_i), \\ \hat{\mu}_Y &:= \frac{1}{n}\sum_{i=1}^n \psi(y_i). \end{aligned} \tag{2.43}$$

Since they are simply defined to be empirical means of RKHS vectors, their convergence properties are analogous to empirical means of usual vectors.

Theorem 2.7 (Song et al. [2009])**.** *Empirical mean embeddings $\hat{\mu}_X \in \mathcal{H}_k$ (resp. $\hat{\mu}_Y \in \mathcal{H}_\ell$) converges to their true mean embeddings $\mu_X \in \mathcal{H}_k$ (resp. $\mu_Y \in \mathcal{H}_\ell$) in its respective RKHS norm at a rate of $O_p(n^{-\frac{1}{2}})$.*

Empirical estimates for function expectations can thus be expressed as inner products with empirical mean embeddings.

Theorem 2.8 (Empirical Function Expectation)**.** *The empirical mean of a function $f \in \mathcal{H}_k$ (resp. $g \in \mathcal{H}_\ell$) over samples $\{x_i\}_{i=1}^n$ (resp. $\{y_i\}_{i=1}^n$) from distribution $\mathbb{P}_X$ (resp. $\mathbb{P}_Y$) can be evaluated as the inner product between the corresponding empirical mean embedding and the function,*

$$\begin{aligned} \langle \hat{\mu}_X, f \rangle_k &= \frac{1}{n}\sum_{i=1}^n f(x_i), \\ \langle \hat{\mu}_Y, g \rangle_\ell &= \frac{1}{n}\sum_{i=1}^n g(y_i). \end{aligned} \tag{2.44}$$

Proof. Consider the inner product,

$$\begin{aligned}
\langle \hat{\mu}_X, f \rangle_k &= \langle \frac{1}{n} \sum_{i=1}^{n} \phi(x_i), f \rangle_k && \text{(definition 2.23)} \\
&= \frac{1}{n} \sum_{i=1}^{n} \langle \phi(x_i), f \rangle_k && \text{(bilinearity of } \langle \cdot, \cdot \rangle) \\
&= \frac{1}{n} \sum_{i=1}^{n} f(x_i). && \text{(definition 2.14)}
\end{aligned}$$

□

In particular, it is worthwhile to note that the result that $\frac{1}{n}\sum_{i=1}^{n} f(x_i)$ is an empirical estimate to $\mathbb{E}[f(X)]$ can be shown even without using the fact that f is not in the RKHS $\mathcal{H}_k$, meaning that the RKHS is consistent with standard results, and merely provides a way for encoding these operations or results as effectively infinite dimensional operators or embeddings.

2.6 Empirical Conditional Mean Embeddings

We now present important results for empirical CMEs. To do so, we will introduce convenient notations so that results can be expressed more succinctly.

Similar to marginal mean embeddings, CMEs can also be represented empirically through finite samples. While we can construct empirical CMEs by forming them from *iid* samples from a conditional distribution $\mathbb{P}_{X|Y=y}$ for a fixed $y \in \mathcal{Y}$, in practice it can be impractical to obtain enough samples for *each* conditional distribution corresponding to *each* different $y \in \mathcal{Y}$. In practice, a more common scenario involves collecting joint samples $\{x_i, y_i\}_{i=1}^{n}$ in an *iid* fashion from some joint distribution $\mathbb{P}_{XY}$. This means that only one sample of X, being x_i, is sampled from $\mathbb{P}_{X|Y=y_i}$. Unless $y_i = y_j$ for some pair of $i \neq j \in \mathbb{N}_n$, we do not benefit from having multiple samples of X at each Y as information regarding the conditional distribution. Importantly, we are only interested in $\mathbb{P}_{X|Y}$, and not $\mathbb{P}_Y$, even though $\mathbb{P}_{XY}$ contains information for both. The beauty of empirical CMEs is reflected in its ability to encode the empirical mean embedding of the conditional distribution $\mathbb{P}_{X|Y=y}$ from samples $\{x_i, y_i\}_{i=1}^{n}$ of the joint distribution $\mathbb{P}_{XY}$ directly, regardless of the marginal distribution $\mathbb{P}_Y$ that was involved.

Nevertheless, as we only have joint samples $\{x_i, y_i\}_{i=1}^{n}$ from the joint distribution $\mathbb{P}_{XY}$ instead of sets of samples from conditional distributions, empirical estimations for CMEs cannot be obtained from a simple empirical mean. Instead, we decompose the conditional mean operator into cross-covariance operators, which

are second-order mean embeddings, and use joint or marginal samples to empirically estimate those operators.

Definition 2.24 (Empirical Uncentered Cross-Covariance Operators)**.** The empirical uncentered cross-covariance operator of an empirical distribution $\hat{\mathbb{P}}_{XY}$ described by a set of *iid* samples $\{x_i, y_i\}_{i=1}^n$ from $\mathbb{P}_{XY}$ under the kernels k and ℓ is defined by the empirical mean of the tensor products of the canonical features in their induced RKHSs $\mathcal{H}_k$ and $\mathcal{H}_\ell$ with respect to said distribution,

$$\begin{aligned}
\hat{C}_{XX} &:= \frac{1}{n}\sum_{i=1}^{n} \phi(x_i) \otimes \phi(x_i), \\
\hat{C}_{XY} &:= \frac{1}{n}\sum_{i=1}^{n} \phi(x_i) \otimes \psi(y_i), \\
\hat{C}_{YX} &:= \frac{1}{n}\sum_{i=1}^{n} \psi(y_i) \otimes \phi(x_i), \\
\hat{C}_{YY} &:= \frac{1}{n}\sum_{i=1}^{n} \psi(y_i) \otimes \psi(y_i).
\end{aligned} \tag{2.45}$$

Since the conditional mean operator can be decomposed into cross-covariance operators as per theorem 2.6, we can define the empirical conditional mean operator by replacing the corresponding cross-covariance operators by their empirical estimates as per definition 2.24.

Definition 2.25 (Empirical Conditional Mean Operator)**.** Motivated by theorem 2.6, the empirical CMO is defined by

$$\hat{C}_{Y|X} := \hat{C}_{YX}(\hat{C}_{XX} + \lambda I)^{-1}. \tag{2.46}$$

where λ is a regularization hyperparameter and I is the identity operator.

Definition 2.26 (Empirical Conditional Mean Embedding)**.** Motivated by definition 2.22, the empirical CME conditioned on $x \in \mathcal{X}$ is defined by

$$\hat{\mu}_{Y|X=x} := \hat{C}_{Y|X}\phi(x). \tag{2.47}$$

Due to the reproducing property in $\mathcal{H}_\ell$, the evaluation of the empirical CME at $y \in \mathcal{Y}$ is thus

$$\hat{\mu}_{Y|X=x}(y) = \psi(y)^T \hat{C}_{Y|X}\phi(x). \tag{2.48}$$

The regularization hyperparameter λ is introduced to relax the operator inversion [Song et al., 2009]. Its value is to be decayed accordingly as n increases in order for the empirical CME to converge to the true CME.

Theorem 2.9. *Empirical CMEs $\hat{\mu}_{Y|X=x} \in \mathcal{H}_\ell$ converge to their true CMEs $\mu_{Y|X=x} \in \mathcal{H}_\ell$ in its respective RKHS norm at a rate $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.*

Consequently, if λ decays at $O(n^{-\frac{1}{2}})$, then the CME converges at $O_p(n^{-\frac{1}{4}})$. Importantly, like empirical marginal mean embedding, it is worthwhile to note that the convergence rate of empirical CMEs do not depend on the dimensionality of the random variables involved explicitly. This is because each sample $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ have been mapped into infinite dimensional features $k(x, \cdot) \in \mathcal{H}_k$ and $\ell(y, \cdot) \in \mathcal{H}_\ell$. Consequently, kernel mean estimation do not suffer from the curse of dimensionality as much as methods that operate directly in the usual probability space, such as density estimation.

In practice, the regularization hyperparameter avoids numerical singularity errors, as well as serves to prevent underfitting. However, the exact form of inverse regularization remains to be justified. Furthermore, the methodology for determining the appropriate value of λ remains to be established. Insights towards these issues can be gained by viewing the empirical CMO the solution to a function-valued regularized least squares regression problem in the RKHS [Grünewälder et al., 2012],

$$\hat{C}_{Y|X} = \underset{C:\mathcal{H}_k \to \mathcal{H}_\ell}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \|\ell(y_i, \cdot) - Ck(x_i, \cdot)\|^2_{\mathcal{H}_\ell} + \lambda \|C\|^2_{HS}, \tag{2.49}$$

where $\|\cdot\|_{HS}$ denotes the HS norm. This provides a regression view to empirical CMOs, and explains that the regularization hyperparameter serves to shrink the size of the empirical CMO with respect to its HS norm. However, this regression objective may not correspond to the learning or inference task in particular settings.

Note that the optimization is over the space of operators from $\mathcal{H}_k$ to $\mathcal{H}_\ell$ for fixed kernels k and ℓ and fixed regularization hyperparameter λ. We can write this result to make the dependence on the hyperparameters explicit,

$$\hat{C}^{(\alpha,\beta,\lambda)}_{Y|X} = \underset{C:\mathcal{H}_k \to \mathcal{H}_\ell}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \|\ell_\beta(y_i, \cdot) - Ck_\alpha(x_i, \cdot)\|^2_{\mathcal{H}_{\ell_\beta}} + \lambda \|C\|^2_{HS_{\alpha,\beta}}. \tag{2.50}$$

We now append the superscript (T) to emphasize that the data the solution of the empirical CMO is trained on is called the training set,

$$\hat{C}^{(\alpha,\beta,\lambda),(T)}_{Y|X} = \underset{C:\mathcal{H}_k \to \mathcal{H}_\ell}{\arg\min} \frac{1}{n^{(T)}} \sum_{i=1}^{n^{(T)}} \|\ell_\beta(y_i^{(T)}, \cdot) - Ck_\alpha(x_i^{(T)}, \cdot)\|^2_{\mathcal{H}_{\ell_\beta}} + \lambda \|C\|^2_{HS_{\alpha,\beta}}. \tag{2.51}$$

To select hyperparameters, we cannot minimize the above regularized least squares objective on the training data further over the hyperparameters. Since the empirical CMO is already the optimal solution over the space of operators, if we further minimize this objective over the hyperparameters, this can result in hyperparameters that lead to empirical CMOs that have a low value for the objective only on the training set. In other words, we run the risk of overfitting. Consequently, we

instead minimize the regularized least squares objective with the trained empirical CMO (definition 2.25) over a validation set denoted by the superscript (V) [Song et al., 2013],

$$\mathcal{L}^{(V)}(\alpha,\beta,\lambda) = \frac{1}{n^{(V)}}\sum_{j=1}^{n^{(V)}} \|\ell_\beta(y_j^{(V)},\cdot) - \hat{C}_{Y|X}^{(\alpha,\beta,\lambda),(T)} k_\alpha(x_j^{(V)},\cdot)\|_{\mathcal{H}_\ell}^2 + \lambda \|\hat{C}_{Y|X}^{(\alpha,\beta,\lambda),(T)}\|_{HS_{\alpha,\beta}}^2. \tag{2.52}$$

To compute the final cross validation objective, shuffle the folds as per standard procedure for training and validation sets and average the validation objectives. Note that the regularization term $\lambda\|\hat{C}_{Y|X}^{(\alpha,\beta,\lambda),(T)}\|_{HS}$ is still included even though it appears to trivially vanish when $\lambda = 0$ because the CMO $\hat{C}_{Y|X}^{(\alpha,\beta,\lambda),(T)}$ also depends on λ and the overall objective may not be minimized at $\lambda = 0$.

While this hyperparameter learning strategy can be applied in any general setting, it has two major problems. The first and more straightforward problem is that it can be computationally heavy to evaluate the cross validation objective and its gradients, if required for optimization, at each optimization iteration. The second and less defined problem is that such a regularized square loss in the RKHS is not necessarily always the most appropriate loss to use for hyperparameter learning in many learning and inference settings. That is, hyperparameters learned this way may not necessarily be optimal for the task. As such, hyperparameter learning of CMEs in different settings still remain as a challenging problem.

Since the empirical CMO can be expressed entirely as linear operations, with constituents formed by linear combinations of canonical features in the RKHS, they only require inner products of features rather than features themselves. Consequently, we can employ the kernel trick and write their nonparametric form. To do this, however, it is useful to introduce and borrow notations from standard matrix algebra so that derivations can be written more succinctly.

As RKHSs are vector spaces, we make use of the notion that functions can be seen as vectors. Consequently, we informally borrow the notion of dimensionality and set it as the cardinality of the its domain. For example, if the domain $\mathcal{X} = \mathbb{R}^d$ is the d dimensional euclidean space whose cardinality is uncountably infinite, then the feature function can be viewed as an uncountably infinite dimensional feature vector, indexed by the elements of $\mathcal{X} = \mathbb{R}^d$.

For the purpose of building intuition using analogy, in the following discussion we will informally refer to the cardinality of $\mathcal{X}$ as $\|\mathcal{X}\|$, and similarly the cardinality of $\mathcal{Y}$ as $\|\mathcal{Y}\|$, despite their cardinality often being uncountably infinite. With the observations $\{x_i, y_i\}_{i=1}^n$ sampled from $\mathbb{P}_{XY}$, there are n feature vectors for each RKHS, $\{\phi(x_i)\}_{i=1}^n$ for $\mathcal{H}_k$ and $\{\psi(y_i)\}_{i=1}^n$ for $\mathcal{H}_\ell$. As elements within their respective RKHS, $\phi(x_i)$ has an effective dimension of $\|\mathcal{X}\| \times 1$ and $\psi(y_i)$ has an effective dimension of $\|\mathcal{Y}\| \times 1$. Since this can be said about the feature functions,

which form the basis for the RKHS as per theorem 2.2, this also applies to general functions within the RKHS. As such, an alternative notation for the inner product between functions $f_1, f_2 \in \mathcal{H}_k$ is

$$\langle f_1, f_2 \rangle_k = f_1^T f_2. \tag{2.53}$$

It can be conceptually helpful to informally check that f_1^T is of size $1 \times \|\mathcal{X}\|$ and f_2 is of size $\|\mathcal{X}\| \times 1$ so that the result is a scalar of size 1×1.

Similarly, an alternative notation for second-order tensor products between functions $f_1, f_2 \in \mathcal{H}_k$ is

$$f_1 \otimes f_2 = f_1 f_2^T. \tag{2.54}$$

Recall that a matrix $A := \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix} \in \mathbb{R}^{d \times n}$, $\mathbf{a}_i \in \mathbb{R}^d \; \forall i \in \mathbb{N}_n$, operated on a vector $\mathbf{v} \in \mathbb{R}^d$ results in a vector that is the linear combination of the columns of A with coefficients given by the components of $\mathbf{v}$,

$$A\mathbf{v} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix} \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} = \sum_{i=1}^{n} v_i \mathbf{a}_i. \tag{2.55}$$

Similarly, a feature matrix can be defined in the same way such that empirical representations of KMEs can be reduced down to linear algebraic operations. However, instead of d, the dimensionality of each vector becomes informally the cardinality of the domain.

Definition 2.27 (Feature Matrix)**.** A feature matrix is formed by stacking the corresponding feature vectors *horizontally*, where each feature vector represents a column of that matrix,

$$\begin{aligned} \Phi &:= \begin{bmatrix} \phi(x_1) & \cdots & \phi(x_n) \end{bmatrix}, \\ \Psi &:= \begin{bmatrix} \psi(y_1) & \cdots & \psi(y_n) \end{bmatrix}. \end{aligned} \tag{2.56}$$

Informally, Φ has effective size $\|\mathcal{X}\| \times n$ and Ψ has effective size $\|\mathcal{Y}\| \times n$.

Often, kernel methods require kernel evaluations between all pairs of data samples. These values are stored in a gram matrix.

Definition 2.28 (Kernel Gram Matrix)**.** The gram matrix K (resp. L) of a kernel k (resp. ℓ) for some dataset $\{x_i\}_{i=1}^n$ (resp. $\{y_i\}_{i=1}^n$) is the matrix of all paired kernel evaluations between the samples,

$$\begin{aligned} K &:= \{k(x_i, x_j)\}_{i=1,j=1}^{n,n}, \\ L &:= \{\ell(y_i, y_j)\}_{i=1,j=1}^{n,n}. \end{aligned} \tag{2.57}$$

When a new sample is to be compared to the set of data samples, such as a new query point, we also require kernel evaluations between the new sample and data samples. These values are stored in a gram vector.

Definition 2.29 (Kernel Gram Vector)**.** The gram vector $\mathbf{k}(x)$ (resp. $\boldsymbol{\ell}(y)$) of a kernel k (resp. ℓ) evaluated on some sample $x \in \mathcal{X}$ (resp. $y \in \mathcal{Y}$) is the vector of all kernel evaluations between the sample and the data samples $\{x_i\}_{i=1}^n$ (resp. $\{y_i\}_{i=1}^n$),

$$\begin{aligned} \mathbf{k}(x) &:= \{k(x_i, x)\}_{i=1}^n, \\ \boldsymbol{\ell}(y) &:= \{\ell(y_i, y)\}_{i=1}^n. \end{aligned} \tag{2.58}$$

With the convenient notations introduced above, we can write vectorized equations that use the reproducing property as follows,

$$\begin{aligned} K &= \Phi^T \Phi, \\ L &= \Psi^T \Psi, \\ \mathbf{k}(x) &= \Phi^T \phi(x), \\ \boldsymbol{\ell}(y) &= \Psi^T \psi(y). \end{aligned} \tag{2.59}$$

We can further write empirical cross-covariance operators (2.45) more succinctly as

$$\begin{aligned} \hat{C}_{XX} &= \frac{1}{n} \Phi \Phi^T, \\ \hat{C}_{XY} &= \frac{1}{n} \Phi \Psi^T, \\ \hat{C}_{YX} &= \frac{1}{n} \Psi \Phi^T, \\ \hat{C}_{YY} &= \frac{1}{n} \Psi \Psi^T. \end{aligned} \tag{2.60}$$

In order to write the nonparametric form for empirical CMEs, we would require a special case of a standard result in linear algebra known as the Sherman–Morrison-Woodbury formula.

Theorem 2.10 (Sherman–Morrison–Woodbury)**.** *A special case of the Sherman–Morrison–Woodbury formula states that for any linear operators $B \in \mathbb{R}^{n \times m}$ and $C \in \mathbb{R}^{m \times n}$ and identity operators I_n and I_m, the following identity holds*

$$B(CB + I_m)^{-1} = (BC + I_n)^{-1} B. \tag{2.61}$$

The result holds for all operators of finite dimensionality. This results also holds for bounded linear operators [Xu, 2017]. Note that if our features $\phi(x)$ are bounded for all $x \in \mathcal{X}$, then our feature matrices as per definition 2.27 is a bounded operator since it is finite collection of such features.

Therefore, from here onwards we will require that our features are bounded, meaning that $\|\phi(x)\|_k^2 = \|k(x,\cdot)\|_k^2 = \langle k(x,\cdot), k(x,\cdot)\rangle_k = k(x,x)$ is bounded for all $x \in \mathcal{X}$. This is satisfied for many kernels we use in practice, such as the Gaussian kernel.

Theorem 2.11 (Nonparametric Conditional Mean Operator)**.** *The empirical CMO can be expressed nonparametrically as*

$$\hat{C}_{Y|X} = \Psi(K + \lambda I_n)^{-1}\Phi^T. \tag{2.62}$$

Proof. The result follows from theorem 2.10 and the vectorized notations (2.60) and (2.59),

$$\begin{aligned}
\hat{C}_{Y|X} &= \hat{C}_{YX}(\hat{C}_{XX} + \lambda I)^{-1} && \text{(definition 2.25)}\\
&= \frac{1}{n}\Psi\Phi^T(\frac{1}{n}\Phi\Phi^T + \lambda I)^{-1} && \text{(equation (2.60))}\\
&= \Psi\Phi^T(\Phi\Phi^T + n\lambda I)^{-1} && (\tfrac{1}{n} \text{ cancels})\\
&= \Psi(\Phi^T\Phi + n\lambda I)^{-1}\Phi^T && \text{(theorem 2.10)}\\
&= \Psi(K + n\lambda I)^{-1}\Phi^T. && \text{(equation (2.59))}
\end{aligned}$$

□

It can be conceptually helpful to note that Ψ has size $\|\mathcal{Y}\| \times n$, $(K + \lambda I)^{-1}$ has size $n \times n$, and Φ^T has size $n \times \|\mathcal{X}\|$, resulting in the appropriate size $\|\mathcal{Y}\| \times \|\mathcal{X}\|$ for $\hat{C}_{Y|X}$.

Theorem 2.12 (Nonparametric Conditional Mean Embedding)**.** *The empirical CME conditioned on $x \in \mathcal{X}$ can be expressed nonparametrically as*

$$\hat{\mu}_{Y|X=x} := \hat{C}_{Y|X}\phi(x) = \Psi(K + n\lambda I)^{-1}\mathbf{k}(x). \tag{2.63}$$

Proof. The result follows from theorem 2.11 and the vectorized notations (2.59),

$$\begin{aligned}
\hat{C}_{Y|X}\phi(x) &= \Psi(K + n\lambda I)^{-1}\Phi^T\phi(x), && \text{(theorem 2.11)}\\
&= \Psi(K + n\lambda I)^{-1}\mathbf{k}(x). && \text{(equation (2.59))}
\end{aligned}$$

□

Due to the reproducing property in $\mathcal{H}_\ell$, the evaluation of the empirical CME at $y \in \mathcal{Y}$ is thus

$$\hat{\mu}_{Y|X=x}(y) = \psi(y)^T\hat{C}_{Y|X}\phi(x) = \boldsymbol{\ell}(y)^T(K + n\lambda I)^{-1}\mathbf{k}(x). \tag{2.64}$$

Importantly, the empirical CME may evaluate to negative values even if its true CME never evaluates to negative values. Consider the case where the kernels k and ℓ always evaluates to nonnegative values. That is $k(x, x') \geq 0$ for all $x, x' \in \mathcal{X}$ and similarly for ℓ. This means that the evaluation of the true CME (2.36) $\mu_{Y|X=x}(y)$ for all $y \in \mathcal{Y}$ and $x \in \mathcal{X}$ must be nonnegative. However, due to the inversion of the regularized cross-covariance operator in the empirical CME (2.48) or correspondingly the inversion of the gram matrix within the nonparametric form of the empirical CME (2.64), the evaluation of empirical CMEs need not be nonnegative. When we obtain such an empirical CME this immediately implies that the empirical CME do not have a proper pre-image.

Nevertheless, while there is a long story behind the elegant theory of CMEs, their empirical forms take relatively simple forms and require only standard linear algebraic operations. In this sense, it is similar to GPRs and many other kernel models, where training and prediction is straightforward to implement. Due to the general nature of the way it is formulated, CMEs do not make strong or limiting assumptions on the distributions involved, allowing for frameworks that require such flexibility to be built upon it.

In the rest of the thesis, we will present methodologies to leverage the representational power of CMEs in three different settings – classification (chapter 3), inference (chapter 4), and regression (chapter 5). We address the problem of hyperparameter learning and probabilistic inference in all three settings, as well as further challenges that are relevant to each setting individually.

Chapter 3

Hyperparameter Learning for Conditional Kernel Mean Embeddings with Rademacher Complexity Bounds

Conditional kernel mean embeddings are nonparametric models that encode conditional expectations in a reproducing kernel Hilbert space. While they provide a flexible and powerful framework for probabilistic inference, their performance is highly dependent on the choice of kernel and regularization hyperparameters. Nevertheless, current hyperparameter tuning methods predominantly rely on expensive cross validation or heuristics that is not optimized for the inference task. For conditional kernel mean embeddings with categorical targets and arbitrary inputs, we propose a hyperparameter learning framework based on Rademacher complexity bounds to prevent overfitting by balancing data fit against model complexity. Our approach only requires batch updates, allowing scalable kernel hyperparameter tuning without invoking kernel approximations. Experiments demonstrate that our learning framework outperforms competing methods, and can be further extended to incorporate and learn deep neural network weights to improve generalization.

3.1 Introduction

Conditional mean embeddings (CMEs) are attractive because they encode conditional expectations in a RKHS, bypassing the need for a parametrized distribution [Song et al., 2013]. They are part of a broader class of techniques known as kernel mean embeddings, where nonparametric probabilistic inference can be carried out entirely within the RKHS because difficult marginalization integrals become simple linear algebra [Muandet et al., 2017]. This very general framework is core to modern kernel probabilistic methods, including kernel two-sample testing [Gretton

et al., 2007], kernel Bayesian inference [Fukumizu et al., 2013], density estimation [Kanagawa and Fukumizu, 2014, Song et al., 2008], component analysis [Muandet et al., 2013], dimensionality reduction [Fukumizu et al., 2004], feature discovery [Jitkrittum et al., 2016], and state space filtering [Kanagawa et al., 2016].

Nevertheless, like most kernel based models, their performance is highly dependent on the hyperparameters chosen. For these models, the model selection process usually begins by selecting a kernel, whose parameters become part of the model *hyperparameters*, which may further include noise or regularization hyperparameters. Given a set of hyperparameters, training is performed by solving either a convex optimization problem, such as the case in support vector machines (SVMs) [Schölkopf and Smola, 2002], or a set of linear equations, such as the case for Gaussian process regressor (GPR) [Rasmussen and Williams, 2006], regularized least squares classifiers (RLSCs) [Rifkin et al., 2003], and CMEs. Unfortunately, hyperparameter tuning is not straight forward, and often cross validation [Song et al., 2013] or median length heuristics [Muandet et al., 2017] remain as the primary approaches for this task. The former can be computationally expensive and sensitive to the selection and number of validation sets, while the latter only applies to hyperparameters with a length scale interpretation and makes no reference to the conditional inference problem involved as it does not make use of targets.

One notable success story in this domain are GPs, which employ their marginal likelihood as an objective for hyperparameter learning. The marginal likelihood arises from its Bayesian formulation, and exhibits certain desirable properties – in particular, the ability to automatically balance between data fit and model complexity. On the other hand, CMEs are not necessarily Bayesian, and hence they do not benefit from a natural marginal likelihood formulation, yet such a balance is critical when generalizing the model beyond known examples.

Can we formulate a learning objective for CMEs to balance data fit and model complexity, similar to the marginal likelihood of GPs? For CMEs with categorical targets and arbitrary input, we present such a learning objective as our main contribution. In particular, we: (1) derive a data-dependent model complexity measure $r(\theta, \lambda)$ for a CME with hyperparameters (θ, λ) based on the Rademacher complexity of a relevant class of CMEs, (2) propose a novel learning objective based on this complexity measure to control generalization risk by balancing data fit against model complexity, and (3) design a scalable hyperparameter learning algorithm under this objective using stochastic batch gradient updates. We show that this learning objective produces CMEs that generalize better than that learned from cross validation (CV), empirical risk minimization (ERM), and median length heuristic (MLH) on standard benchmarks, and apply such an algorithm to incorporate and learn neural network weights to improve generalization.

3.2 Related Work

3.2.1 Conditional Mean Embeddings

To construct a conditional mean operator $\mathcal{U}_{Y|X}$ corresponding to the distribution $\mathbb{P}_{Y|X}$, where $X : \Omega \to \mathcal{X}$ and $Y : \Omega \to \mathcal{Y}$ are measurable random variables, we first choose a kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for the input space $\mathcal{X}$ and another kernel $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ for the output space $\mathcal{Y}$. These kernels k and l each describe how similarity is measured within their respective domains $\mathcal{X}$ and $\mathcal{Y}$, and are symmetric positive definite such that they uniquely define the RKHS $\mathcal{H}_k$ and $\mathcal{H}_l$. The conditional mean operator $\mathcal{U}_{Y|X}$ is then the operator $\mathcal{U} : \mathcal{H}_k \to \mathcal{H}_l$ for which $\mu_{Y|X=x} = \mathcal{U}k(x, \cdot)$, where $\mu_{Y|X=x} := \mathbb{E}[l(Y, \cdot)|X = x]$ is the CME [Song et al., 2009]. In this sense, it sweeps out a family of CMEs $\mu_{Y|X=x}$ in $\mathcal{H}_l$, each indexed by the input variable $x \in \mathcal{X}$. We then define cross covariance operators $C_{YX} := \mathbb{E}[l(Y, \cdot) \otimes k(X, \cdot)] : \mathcal{H}_k \to \mathcal{H}_l$ and $C_{XX} := \mathbb{E}[k(X, \cdot) \otimes k(X, \cdot)] : \mathcal{H}_k \to \mathcal{H}_k$. Alternatively, they can be seen as elements within the tensor product space $C_{YX} \in \mathcal{H}_l \otimes \mathcal{H}_k$ and $C_{XX} \in \mathcal{H}_k \otimes \mathcal{H}_k$.

Under the assumption that $k(x, \cdot) \in \text{image}(C_{XX})$, it can be shown that $\mathcal{U}_{Y|X} = C_{YX}C_{XX}^{-1}$. While this assumption is satisfied for finite domains $\mathcal{X}$ with a characteristic kernel k, it does not necessarily hold when $\mathcal{X}$ is a continuous domain [Fukumizu et al., 2004], which is the case for many classification problems. In this case, $C_{YX}C_{XX}^{-1}$ becomes only an approximation to $\mathcal{U}_{Y|X}$, and we instead regularize the inversion and use $\mathcal{U}_{Y|X} = C_{YX}(C_{XX} + \lambda I)^{-1}$, which also serves to avoid overfitting [Song et al., 2013]. CMEs are useful for probabilistic inference since conditional expectations of a function $g \in \mathcal{H}_l$ can be expressed as inner products with the CME, $\mathbb{E}[g(Y)|X = x] = \langle \mu_{Y|X=x}, g \rangle$, provided that $\mathbb{E}[g(Y)|X = \cdot] \in \mathcal{H}_k$ [Song et al., 2009, Theorem 4].

Furthermore, as both C_{YX} and C_{XX} are defined via expectations, we can estimate them with their respective empirical means to derive a nonparametric estimate for $\mathcal{U}_{Y|X}$ based on observations $\{x_i, y_i\} \in \mathcal{X} \times \mathcal{Y}$, $i \in \mathbb{N}_n := \{1, \ldots, n\}$,

$$\hat{\mathcal{U}}_{Y|X} = \Psi(K + n\lambda I)^{-1}\Phi^T, \tag{3.1}$$

where $K_{ij} := k(x_i, x_j)$, $\Phi := \begin{bmatrix}\phi(x_1) & \ldots & \phi(x_n)\end{bmatrix}$, $\Psi := \begin{bmatrix}\psi(y_1) & \ldots & \psi(y_n)\end{bmatrix}$, $\phi(x) := k(x, \cdot)$, and $\psi(y) := l(y, \cdot)$ [Song et al., 2013]. The empirical CME defined by $\hat{\mu}_{Y|X=x} := \hat{\mathcal{U}}_{Y|X}k(x, \cdot)$ then stochastically converges to the CME $\mu_{Y|X=x}$ in the RKHS norm at rate of $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$, assuming that $k(x, \cdot) \in \text{image}(C_{XX})$ [Song et al., 2009, Theorem 6]. This allows us to approximate the conditional expectation with $\langle \hat{\mu}_{Y|X=x}, g \rangle$ instead, where $\mathbf{g} := \{g(y_i)\}_{i=1}^n$ and $\mathbf{k}(x) := \{k(x_i, x)\}_{i=1}^n$,

$$\mathbb{E}[g(Y)|X = x] \approx \langle \hat{\mu}_{Y|X=x}, g \rangle = \mathbf{g}^T(K + n\lambda I)^{-1}\mathbf{k}(x). \tag{3.2}$$

3.2.2 Hyperparameter Learning

Hyperparameter learning for CMEs is particularly difficult compared to marginal or joint embeddings, since the kernel $k = k_\theta$ with hyperparameters $\theta \in \Theta$ is to be learned jointly with a regularization hyperparameter $\lambda \in \Lambda = \mathbb{R}_+$. Grünewälder et al. [2012] proposed to hold out a validation set $\{k(x_{t_j}, \cdot), l(y_{t_j}, \cdot)\}_{j=1}^J$ and minimize $\frac{1}{J}\sum_{j=1}^J \left\| l(y_{t_j}, \cdot) - \hat{\mathcal{U}}_{Y|X} k(x_{t_j}, \cdot) \right\|_{\mathcal{H}_l}^2$ where $\hat{\mathcal{U}}_{Y|X}$ is estimated from the remaining training set using (3.1). This could also be repeated over multiple folds for cross validation. Song et al. [2013, p. 15] also uses this cross validation approach, but adds regularization $\lambda \|\hat{\mathcal{U}}\|_{HS}^2$ to the validation objective. Validation sets are necessary for improving generalization to unseen examples. This is because the CME is already the solution that minimizes the objective from Grünewälder et al. [2012] over the operator space, so further optimization over the hyperparmeters using the same training set would lead to overfitting. Moreover, the cross validation objective changes depending on the particular split and number of folds. Additionally, by fitting a separate model for each fold during learning, they incur a large computational cost of $O(Jn^3)$ for J folds, and become prohibitive with large datasets. This spells a need for an alternative hyperparameter learning framework using a different objective.

When cross validation is too expensive, length scales can be set by the median heuristic [Muandet et al., 2017] via $\ell = \text{median}_{i,j}(\|x_i - x_j\|_2)$ for many stationary kernels. However, they cannot be used to set hyperparameters other than length scales, such as λ. In the setting of two sample testing, Gretton et al. [2012] note that they can possibly lead to poor performance. In the context of CMEs, they are also unable to leverage supervision from labels. Flaxman et al. [2016] proposed a Bayesian learning framework for marginal mean embeddings via inducing points, although it is unclear how this can be extended to CMEs. Fukumizu et al. [2009] also investigated the choice of kernel bandwidth for stationary kernels in the setting of binary classification and two sample testing using maximum mean discrepancy (MMD), but has yet to generalize to CMEs or multiclass settings.

3.2.3 Rademacher Complexity

Rademacher complexity [Bartlett and Mendelson, 2002] measures the expressiveness of a function class F by its ability to shatter, or fit, noise. They are data-dependent measures, and are thus particularly well suited to learning tasks where generalization is vital, since complexity penalties that are not data dependent cannot be universally effective [Kearns et al., 1997]. The Rademacher complexity [Bartlett and Mendelson, 2002, Definition 2] of a function class F is defined by $\mathcal{R}_n(F) := \mathbb{E}[\sup_{f \in F} \|\frac{2}{n}\sum_{i=1}^n \sigma_i f(X_i)\|]$, where $\{\sigma_i\}_{i=1}^n$ are *iid* Rademacher random variables, taking values in $\{-1, 1\}$ with equal probability, and $\{X_i\}_{i=1}^n$ are *iid*

random variables from the same distribution $\mathbb{P}_X$. Since $\{\sigma_i\}_{i=1}^n$ are distributed independently without knowledge of f, the intuition is to interpret $\{\sigma_i\}_{i=1}^n$ as labels that are simply noise. For a given set of inputs $\{X_i\}_{i=1}^n$, the term inside the norm is high when the sign of $f(X_i)$ matches the signs of σ_i averaged across $i \in \mathbb{N}_n$, meaning that f has managed to fit the noise well. We take this as the defining feature of what it means for a model f to be complex. The supremum then finds the f within F that fits the noise the best, intuitively representing the most complex f within F. The final expectation then averages this quantity across realizations of $\{X_i\}_{i=1}^n$ from $\mathbb{P}_X$.

Rademacher complexities are usually applied in the context where classifiers are trained by minimizing some empirical loss subject to a bounded Rademacher complexity within the class of classifiers. In the context of multi-label learning, Yu et al. [2014] used trace norm regularization to bound the Rademacher complexity, achieving tight generalization bounds. Xu et al. [2016] extends the trace norm regularization approach by considering the local Rademacher complexity on a subset of the predictor class, where they instead minimize the tail sum of the predictor singular values. Local Rademacher complexity has also been employed for multiple kernel learning [Cortes et al., 2013, Kloft and Blanchard, 2011] to learn convex combinations of fixed kernels for SVMs. Similarly, Pontil and Maurer [2013] also used trace norm regularization to bound the Rademacher complexity and minimize the truncated hinge loss. Nevertheless, while Rademacher complexities have been employed to restrict the function class considered for training weight parameters, they have not been applied to learn kernel hyperparameters itself.

3.3 Multiclass Conditional Embeddings

We present a particular type of CMEs that are suitable for prediction tasks with categorical targets. We show that for CMEs with categorical targets and arbitrary inputs, we can further infer conditional probabilities directly, and not just conditional expectations. As there are no restrictions on the number of target categories, we refer to these CMEs as multiclass conditional embeddings (MCEs). Section 3.7 contains the proofs for theorems claimed in this section.

For categorical targets, the output label space is finite and discrete, taking values only in $\mathcal{Y} = \mathbb{N}_m := \{1, \ldots, m\}$. Naturally, we choose the Kronecker delta kernel $\delta : \mathbb{N}_m \times \mathbb{N}_m \to \{0, 1\}$ as the output kernel l, where labels that are the same have unit similarity and labels that are different have no similarity. That is, for all pairs of labels $y_i, y_j \in \mathcal{Y}$, $\delta(y_i, y_j) = 1$ only if $y_i = y_j$ and is 0 otherwise. As δ is an integrally strictly positive definite kernel on $\mathbb{N}_m$, it is therefore characteristic [Sriperumbudur et al., 2010b, Theorem 7]. Therefore, by definition [Fukumizu et al., 2004], δ uniquely defines a RKHS $\mathcal{H}_\delta = \overline{\text{span}\{\delta(y, \cdot) : y \in \mathcal{Y}\}}$, which is

the closure of the span of its kernel induced features [Xu and Zhang, 2009]. For $\mathcal{Y} = \mathbb{N}_m$, this means that any $g : \mathbb{N}_m \to \mathbb{R}$ that is bounded on its discrete domain $\mathbb{N}_m$ is in the RKHS of δ, because we can always write $g = \sum_{y=1}^{m} g(y)\delta(y, \cdot) \in \text{span}\{\delta(y, \cdot) : y \in \mathcal{Y}\} \subseteq \mathcal{H}_\delta$. In particular, indicator functions on $\mathbb{N}_m$ are in $\mathcal{H}_\delta$, since $\mathbb{1}_c(y) := \mathbb{1}_{\{c\}}(y) = \delta(c, y)$, so that $\mathbb{1}_c = \delta(c, \cdot)$ are simply the canonical features of $\mathcal{H}_\delta$. Such properties do not necessarily hold for continuous target domains in general. For discrete target domains, this convenient property enables consistent estimations of decision probabilities.

Let $p_c(x) := \mathbb{P}[Y = c|X = x]$ be the *decision probability function* for class $c \in \mathbb{N}_m$, which is the probability of the class label Y being c when the example X is x. Importantly, note that there are no restrictions on the input domain $\mathcal{X}$ as long as a kernel k can be defined on it. For example, $\mathcal{X}$ could be the continuous Euclidean space $\mathbb{R}^d$, the space of images, or the space of strings. We begin by writing this probability as an expectation of indicator functions,

$$p_c(x) := \mathbb{P}[Y = c|X = x] = \mathbb{E}[\mathbb{1}_c(Y)|X = x]. \tag{3.3}$$

With $\mathbb{1}_c \in \mathcal{H}_\delta$, we let $g = \mathbb{1}_c$ in (3.2) and $\mathbf{1}_c := \{\mathbb{1}_c(y_i)\}_{i=1}^n$ to estimate the right hand side of (3.3) by

$$\hat{p}_c(x) = f_c(x) := \mathbf{1}_c^T (K + n\lambda I)^{-1} \mathbf{k}(x). \tag{3.4}$$

Let $\mathbf{Y} := \begin{bmatrix} \mathbf{1}_1 & \mathbf{1}_2 & \cdots & \mathbf{1}_m \end{bmatrix} \in \{0, 1\}^{n \times m}$ be the one hot encoded labels of $\{y_i\}_{i=1}^n$. The vector of empirical decision probabilities over the classes $c \in \mathbb{N}_m$ is then

$$\hat{\mathbf{p}}(x) = \mathbf{f}(x) := \mathbf{Y}^T (K + n\lambda I)^{-1} \mathbf{k}(x) \in \mathbb{R}^m. \tag{3.5}$$

Since $\mathcal{U} = \hat{\mathcal{U}}_{Y|X}$ (3.1) is the solution to a regularized least squares problem in the RKHS from $k(x, \cdot) \in \mathcal{H}_k$ to $l(y, \cdot) \in \mathcal{H}_l$ [Grünewälder et al., 2012], CMEs are essentially kernel ridge regressions (KRRs) with targets in the RKHS. In this case, because $\mathcal{Y} = \mathbb{N}_m$ is discrete, $\mathcal{H}_\delta$ can be identified with $\mathbb{R}^m$. As a result, the rows of the MCE can also be seen as m KRRs [Friedman et al., 2001] on binary $\{0, 1\}$-targets, where they all share the same input kernel k. Because they all share the same kernel to form the MCE, we prove that the empirical decision probabilities (3.4) do converge to the population decision probability.

Theorem 3.1 (Convergence of Empirical Decision Probability Function)**.** *Assuming that $k(x, \cdot)$ is in the image of C_{XX}, the empirical decision probability function $\hat{p}_c : \mathcal{X} \to \mathbb{R}$ (3.4) converges uniformly to the true decision probability $p_c : \mathcal{X} \to [0, 1]$ (3.3) at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$.*

In particular, the assumption $k(x, \cdot) \in \text{image}(C_{XX})$ is a statement on the input kernel k, not the output kernel l, which is a Kronecker delta $l = \delta$ for MCEs. It is worthwhile to note that this assumption is common for CMEs, and is often relaxed through introducing the regularization hyperparameter λ (3.1) in practice [Muandet et al., 2017, Song et al., 2013, 2009, p.74-75, Sec. 3 and 3.1 *resp.*].

Note that for finite n the probability estimates (3.4) may not necessarily lie in the range $[0, 1]$ nor form a normalized distribution for finite n. Nonetheless, theorem 3.1 guarantees that they approach one with increasing sample size. When normalized distributions are required, clip-normalized estimates can be used,

$$\tilde{p}_c(x) := \frac{\max\{\hat{p}_c(x), 0\}}{\sum_{j=1}^{m} \max\{\hat{p}_j(x), 0\}}. \tag{3.6}$$

This does not change the resulting prediction, since $\hat{y}(x) = \arg\max_{c \in \mathbb{N}_m} \hat{p}_c(x) = \arg\max_{c \in \mathbb{N}_m} \tilde{p}_c(x)$. Theorem 3.1 also implies that eventually the effect of clip-normalization vanishes, where $\tilde{p}_c(x)$ approaches to both $\hat{p}_c(x)$ and thus $p_c(x)$ with increasing sample sizes.

Importantly, this enables MCEs to be naturally applied to perform probabilistic classification in multiclass settings with categorical targets. In contrast, in terms of probabilistic classification, support vector classifiers (SVCs) do not output probabilities and probabilistic extensions require difficult calibration, while Gaussian process classifiers (GPCs) require posterior approximations. Furthermore, in terms of the multiclass setting, multiclass extensions to SVCs and GPCs often employ the one versus all (OVA) or one versus one (OVO) scheme [Aly, 2005], resulting in multiple separately trained binary classifiers with no guarantees of coherence between their outputs. Instead, training a single MCE is sufficient for producing consistent multiclass probabilistic estimates.

Similar to RLSC, MCEs are solutions to a regularized least squares problem in a RKHS [Grünewälder et al., 2012], resulting in a similar system of linear equations. Nevertheless, RLSCs primarily differ in the way they handle the labels, in which binary labels $\{-1, 1\}$ appear directly in the squared loss instead of its kernel feature $\delta(y_i, \cdot)$ or, equivalently, its one hot encoded form $\mathbf{y}_i$. Consequently, multiclass extensions for RLSC either require using the OVA scheme [Rifkin et al., 2003] which suffers from computational and coherence issues, or alternatively require minimizing the total loss across all binarized tasks for the overall least squares problem [Pahikkala et al., 2012]. Although the latter attempts to link the classifiers together through its loss, both approaches still produce separate classifiers for each class. As a result, multiclass RLSC does not produce consistent estimates of class probabilities akin to that of theorem 3.1 for MCEs.

3.4 Information Entropy

Probabilistic classifiers such as the MCE allows us to quantify the uncertainty of its predictions for any given example $x \in \mathcal{X}$ through the information entropy. This is ideal for detecting the decision boundaries of the classifier. However, the MCEs outputs decision probability estimates which may not necessarily lie in the range $[0, 1]$ nor form a normalized distribution for finite n. Consequently, we present two main approaches for approximating the information entropy from the classifier, with the latter taking advantage of RKHS properties directly.

The information entropy of the possible labels Y for a given example $X = x$ is

$$h(x) := \mathbb{H}[Y|X = x] = -\sum_{c=1}^{m} p_c(x) \log p_c(x). \tag{3.7}$$

The first approach is straightforward, involving simply computing the information entropy with the clip normalized probabilities (3.6), at the query point $x \in \mathcal{X}$,

$$\tilde{h}(x) := -\sum_{c=1}^{m} \tilde{p}_c(x) \log \tilde{p}_c(x). \tag{3.8}$$

We call (3.8) the *clip-normalized information entropy*. Since $\tilde{p}_c(x)$ converges pointwise to $p_c(x)$ with increasing data, $\tilde{h}(x)$ also converges pointwise to $h(x)$.

Just as decision probabilities can be expressed as an expectation of indicator functions, information entropy can be expressed as expected information gain,

$$\begin{aligned} \mathbb{H}[Y|X = x] &= -\sum_{c=1}^{m} \mathbb{P}[Y = c|X = x] \log \mathbb{P}[Y = c|X = x] \\ &= \mathbb{E}[-\log \mathbb{P}[Y|X = x]|X = x] \\ &= \mathbb{E}[u_x(Y)|X = x], \end{aligned} \tag{3.9}$$

where $u_x(y) := -\log \mathbb{P}[Y = y|X = x]$ is the *information* (in nats) we would gain when we discover that example x actually has label y, under the true decision probability $\mathbb{P}[Y = y|X = x]$. Note that while $\mathbb{P}[Y = y|X = x]$ is a constant, we employ the shorthand notation $\mathbb{P}[Y|X = x]$ for the random variable $g(Y)$ where $g(y) = \mathbb{P}[Y = y|X = x]$. If $u_x : \mathbb{N}_m \to \mathbb{R}$ is in the RKHS $\mathcal{H}_\delta$, then we know that this expectation can also be approximated by $\langle \hat{\mu}_{Y|X=x}, u_x \rangle$.

This forms the basis of our second approach. Assuming that $\mathbb{P}[Y = y|X = x]$ is never exactly zero for all labels $y \in \mathcal{Y}$ and examples $x \in \mathcal{X}$, then $u_x(y)$ is bounded on its discrete domain $\mathbb{N}_m$. We can thus write $u_x = \sum_{c=1}^{m} -\log \mathbb{P}[Y = c|X = x]\delta(c, \cdot)$ which shows that u_x is in the span of the canonical kernel features and is thus in

the RKHS. Hence, similar to the case with decision probabilities, with $u_x \in \mathcal{H}_\delta$ and $\mathbf{u}_x := \{u_x(y_i)\}_{i=1}^n$ we let $g = u_x$ in (3.2) and estimate $h(x)$ by

$$\langle \hat{\mu}_{Y|X=x}, u_x \rangle = \mathbf{u}_x^T (K + n\lambda I)^{-1} \mathbf{k}(x). \tag{3.10}$$

Unfortunately, u_x is not known exactly, since $\mathbb{P}[Y = y|X = x]$ is not known exactly. Instead, since $\hat{p}_c(x)$ is a consistent estimate for $\mathbb{P}[Y = c|X = x]$ by theorem 3.7, we propose to replace $u_x(y)$ with the information gain under $\hat{p}_y(x)$ instead. However, we cannot simply take the log of this estimator, as $\hat{p}_y(x)$ may produce non-positive estimates to the prediction probabilities. The straight forward way to mitigate this problem is to clip $\hat{p}_y(x)$ from the bottom by a very small number, before taking the log. However, experiments show that this produces non-smooth estimates over $\mathcal{X}$ and the degree of smoothness varies drastically between different choices of that small number. Instead, in virtue of the fact that $\lim_{p\to 0} -p \log p = 0$ even though $\lim_{p\to 0} -\log p = \infty$, we simply define the information gain estimate $\hat{u}_x(y)$ as zero if the empirical decision probability is non-positive,

$$\hat{u}_x(y) := \begin{cases} -\log \hat{p}_y(x) & \text{if} \quad \hat{p}_y(x) > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.11}$$

It remains to show that $\hat{u}_x \in \mathcal{H}_\delta$. Indeed, the information gain estimate can be written as $\hat{u}_x = \sum_{c=1}^m \hat{u}_x(c)\delta(c, \cdot)$ and thus $\hat{u}_x$ is in the span of the kernel canonical features. We then arrive at the following estimate for $h(x)$,

$$\hat{h}(x) := \langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle = \hat{\mathbf{u}}_x^T (K + n\lambda I)^{-1} \mathbf{k}(x), \tag{3.12}$$

where $\hat{\mathbf{u}}_x := \{\hat{u}_x(y_i)\}_{i=1}^n$. Similar to the case with decision probabilities (3.3), the information entropy estimate (3.12) is not guaranteed to be non-negative. However, in practice these negative values are close to zero. Furthermore, negative estimated information entropy implies that the model is very confident about its prediction, and it suffices to simply clip the entropy at zero if strict information entropy is required. Since this estimator is now based on the inner product between the empirical CME and another empirically estimate function, instead of between the empirical CME and a known function like the decision probability estimate, it is not immediately clear that such an estimator converges. Nevertheless, intuition tells us that the inner product between two converging quantities should converge. We proceed to show that this intuition is correct. The proof is provided in section 3.7.

Theorem 3.2 (Convergence of Empirical Information Entropy Function)**.** *Assuming that $k(x, \cdot)$ is in the image of C_{XX}, the empirical information entropy function $\hat{h} : \mathcal{X} \to \mathbb{R}$ (3.12) converges pointwise to the true information entropy function $h : \mathcal{X} \to [0, \infty)$ at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.*

3.5 Hyperparameter Learning with Rademacher Complexity Bounds

We now derive learning theoretic bounds that motivate and serve as the foundations to our proposed hyperparameter learning algorithm. From here onwards, we denote θ as the kernel hyperparameters of the kernel $k = k_\theta$. The hyperparameters we are interested in learning are θ and λ.

We begin by defining a loss function as a measure for performance. For decision functions of the form $\mathbf{f} : \mathcal{X} \to \mathcal{A} = \mathbb{R}^m$ whose entries are probability estimates, we employ a modified cross entropy loss,

$$\mathcal{L}_\epsilon(y, \mathbf{f}(x)) := -\log\left[\mathbf{y}^T \mathbf{f}(x)\right]_\epsilon^1 = -\log\left[f_y(x)\right]_\epsilon^1, \tag{3.13}$$

to express risk, where we use the notation $[\,\cdot\,]_\epsilon^1 := \min\{\max\{\,\cdot\,, \epsilon\}, 1\}$ for $\epsilon \in (0, 1)$. It is worthwhile to point out that this choice only makes sense due to theorem 3.1, as it allows us to interpret the outputs of the CME as asymptotic probability estimates. Note that we employ the loss on the original probability estimates (3.5), not the clip-normalized version (3.6). We employ this loss in virtue of theorem 3.1, where we expect $\mathbf{f}(x)$ (3.5) to be approximations to the population decision probabilities. In contrast, direct outputs from SVCs, GPCs, or RLSCs are not consistent probability estimates and cannot take advantage of (3.13) easily.

However, simply minimizing the empirical loss $\frac{1}{n}\sum_{i=1}^n \mathcal{L}_\epsilon(y_i, \mathbf{f}_{\theta,\lambda}(x_i))$ over the hyperparameters (θ, λ) could lead to an overfitted model. We therefore employ Rademacher complexity bounds to control the model complexity of MCEs.

Let Θ and Λ be a space of kernel and regularization hyperparameters respectively. We define the class of MCEs over these hyperparameter spaces by

$$F_n(\Theta, \Lambda) := \{\mathbf{f}_{\theta,\lambda}(x) : \theta \in \Theta, \lambda \in \Lambda\}. \tag{3.14}$$

We denote $W_{\theta,\lambda}^T \equiv \hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)}$ so that $\|W_{\theta,\lambda}\|_{\text{tr}} = \|\hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)}\|_{HS}$ to reflect the dependence on (θ, λ) and also to emphasize the role it plays as the weights of the decision function. We first restrict the space of hyperparameters by the norms of $W_{\theta,\lambda}$ and $k_\theta(x, x)$ to obtain an upper bound to the Rademacher complexity of $F_n(\Theta, \Lambda)$.

Theorem 3.3 (MCE Rademacher Complexity Bound)**.** *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map is bounded in RKHS norm $\|\phi_\theta(x)\|_{\mathcal{H}_{k_\theta}}^2 = k_\theta(x, x) \leq \alpha^2$, $\alpha > 0$, for all $x \in \mathcal{X}, \theta \in \Theta$. For any set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the class of MCEs $F_n(\Theta, \Lambda)$ (3.14) is bounded by*

$$\mathcal{R}_n(F_n(\Theta, \Lambda)) \leq 2\alpha\rho. \tag{3.15}$$

Bartlett and Mendelson [2002] showed that the expected risk can be bounded with high probability using the empirical risk and the Rademacher complexity of the loss composed with the function class. For a Lipchitz loss, Ledoux and Talagrand [2013] further showed that the latter quantity can be bounded using the Rademacher complexity of the function class itself. We use these two results to arrive at the following probabilistic upper bound to our expected loss.

Theorem 3.4 (MCE ϵ-Specific Expected Risk Bound)**.** *Assume the same assumptions as theorem 3.3. For any integer $n \in \mathbb{N}_+$, any $\epsilon \in (0, e^{-1})$, and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability of at least $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $f \in F_n(\Theta, \Lambda)$ satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, f(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_\epsilon(Y_i, f(X_i)) + 4e\ \alpha\rho + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}. \tag{3.16}$$

However, for hyperparameter learning, we would require a risk bound for specific choice of hyperparameters, not just for a set of hyperparameters. For some $\tilde{\theta} \in \Theta$ and $\tilde{\lambda} \in \Lambda$, we construct a subset of hyperparameters $\Xi(\tilde{\theta}, \tilde{\lambda}) \subseteq \Theta \times \Lambda$ defined by $\Xi(\tilde{\theta}, \tilde{\lambda}) := \{(\theta, \lambda) \in \Theta \times \Lambda : \|W_{\theta,\lambda}\|_{\text{tr}} \leq \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}},\ \sup_{x\in\mathcal{X}} k_\theta(x,x) \leq \alpha^2(\tilde{\theta}) := \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)\}$. Clearly, this subset is non-empty, since $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ is itself an element of this subset. Thus, we can assert that $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}}$ is bounded for all $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, and that $\|\phi_\theta(x)\|^2_{\mathcal{H}_{k_\theta}} = k_\theta(x,x) \leq \alpha^2 = \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)$ is bounded for all $x \in \mathcal{X}, (\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$.

We can now choose some arbitrary $\tilde{\theta} \in \Theta$, $\tilde{\lambda} \in \Lambda$ and apply theorem 3.4 with $\rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\text{tr}}$ and $\alpha^2 = \sup_{x\in\mathcal{X}} k_{\tilde{\theta}}(x,x)$ and by considering only the hyperparameters $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$. The probabilistic statement (3.16) then only holds for $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$. In particular, since $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, it holds for $(\theta, \lambda) = (\tilde{\theta}, \tilde{\lambda})$. Applying this choice, the only hyperparameters that remain in the statement are $(\tilde{\theta}, \tilde{\lambda})$. We then replace these symbols with (θ, λ) again to avoid cluttered notation. Since they were chosen arbitrarily from $\Theta \times \Lambda$, we arrive at our final result.

Theorem 3.5 (MCE Expected Risk Bound for Hyperparameters)**.** *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$ used to define $\mathbf{f}_{\theta,\lambda}$ (3.5), with probability $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $\theta \in \Theta$, $\lambda \in \Lambda$, and $\epsilon \in (0, e^{-1})$ satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, \mathbf{f}_{\theta,\lambda}(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_\epsilon(Y_i, \mathbf{f}_{\theta,\lambda}(X_i)) + 4e\ r(\theta,\lambda) + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}, \tag{3.17}$$

where $r(\theta,\lambda) := \sqrt{\text{trace}(V_{\theta,\lambda}^T K_\theta V_{\theta,\lambda}) \sup_{x\in\mathcal{X}} k_\theta(x,x)}$ *and* $V_{\theta,\lambda} := (K_\theta + n\lambda I)^{-1}\mathbf{Y}$.

In particular, $r(\theta, \lambda)$ is an upper bound to the Rademacher complexity of a relevant class of MCEs based on the hyperparameters $\Xi(\theta, \lambda)$. We call $r(\theta, \lambda)$ the

Algorithm 1 MCE Hyperparameter Learning with Stochastic Gradient Updates

Input: kernel family $k_\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, dataset $\{x_i, y_i\}_{i=1}^n$, initial kernel hyperparameters θ_0, initial regularization hyperparameters λ_0, learning rate η, cross entropy loss threshold ϵ, batch size n_b
$\theta \leftarrow \theta_0$, $\lambda \leftarrow \lambda_0$
repeat
  Sample the next batch $\mathcal{I}_b \subseteq \mathbb{N}_n$ s.t. $|\mathcal{I}_b| = n_b$
  $Y \leftarrow \{\delta(y_i, c) : i \in \mathcal{I}_b, c \in \mathbb{N}_m\}$ $\in \{0,1\}^{n_b \times m}$
  $K_\theta \leftarrow \{k_\theta(x_i, x_j) : i \in \mathcal{I}_b, j \in \mathcal{I}_b\}$ $\in \mathbb{R}^{n_b \times n_b}$
  $L_{\theta,\lambda} \leftarrow \text{cholesky}(K_\theta + n_b \lambda I_{n_b})$ $\in \mathbb{R}^{n_b \times n_b}$
  $V_{\theta,\lambda} \leftarrow L_{\theta,\lambda}^T \backslash (L_{\theta,\lambda} \backslash Y)$ $\in \mathbb{R}^{n_b \times m}$
  $P_{\theta,\lambda} \leftarrow K_\theta V_{\theta,\lambda}$ $\in \mathbb{R}^{n_b \times m}$
  $r(\theta, \lambda) \leftarrow \alpha(\theta) \sqrt{\text{trace}(V_{\theta,\lambda}^T K_\theta V_{\theta,\lambda})}$
  $q(\theta, \lambda) \leftarrow \frac{1}{n_b} \sum_{i=1}^{n_b} \mathcal{L}_\epsilon((Y)_i, (P_{\theta,\lambda})_i) + 4e\, r(\theta, \lambda)$
  $(\theta, \lambda) \leftarrow \text{GradientBasedUpdate}(q, \theta, \lambda; \eta)$
until maximum iterations reached
Output: kernel hyperparameters θ, regularization λ

Rademacher complexity bound (RCB) and use it to measure the model complexity of a MCE with hyperparameters (θ, λ). Since the training set itself is a sample of length n drawn from $\mathbb{P}_{XY}$, the inequality (3.17) holds with probability $1 - \beta$ when the random variables (X_i, Y_i) are realized as the training observations (x_i, y_i). Motivated by this, we employ this upper bound as the learning objective for hyperparameter learning,

$$q(\theta, \lambda) := \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_\epsilon(y_i, \mathbf{f}_{\theta,\lambda}(x_i)) + 4e\, r(\theta, \lambda). \tag{3.18}$$

Importantly, the first term is an empirical risk that measures data fit, and the second term is the RCB that measures model complexity. Together, this learning objective achieves a balance between data fit and model complexity, similar to the corresponding property of a negative log marginal likelihood learning objective.

3.6 Scalable Hyperparameter Learning

In the big data domain where it becomes prohibitively expensive to compute the full hyperparameter learning objective (3.18), we scale our approach by using only a batch subset of the data to construct hyperparameter learning objective and its gradients. This is possible due to interesting consequences of the statements proved in theorem 3.5.

3.6.1 Batch Stochastic Gradient Update

Since theorem 3.5 holds for any $n \in \mathbb{N}_+$ and any set of data $\{x_i, y_i\}_{i=1}^n$ from $\mathbb{P}_{XY}$, the bound (3.17) also holds with high probability for a batch subset of the training data. We therefore propose to use only a random batch subset of the data to perform each gradient update. This enables scalable hyperparameter learning through batch stochastic gradient updates, where each gradient update stochastically improves a different probabilistic upper bound of the generalization risk. Note that without theorem 3.5, it is not straightforward to simply apply stochastic gradient updates to optimize q, since r depends on the dataset but is not written in terms of a summation over the data. Furthermore, the batch size cannot be too small, in order to keep the constant $\sqrt{8 \log(2/\beta)/n}$ relatively small. We present this scalable hyperparameter learning approach via batch stochastic gradient updates in algorithm 1, reducing the time complexity from $O(n^3)$ to $O(n_b^3)$, where n_b is the batch size. The Cholesky decomposition for the full training set requires $O(n^3)$ time and is necessary only for inference, instead of once every learning iteration. It can be further avoided by using random Fourier features [Rahimi and Recht, 2008] or kernel herding [Chen et al., 2010] to approximate the already learned MCE. All further inference takes $O(n^2)$ time, or potentially less with approximation, using back substitution.

3.6.2 Batch Validation

While we simply instantiated (X_i, Y_i) to be the training observations in theorem 3.5 to obtain (3.18), this does not have to be the case for batch updates. Instead, in each learning iteration, we could further split the batch into two sub-batches – one for training and one for validation. The training batch is used to form the MCE $\mathbf{f}_{\theta,\lambda}$ and RCB $r(\theta, \lambda)$, while we evaluate the empirical risk on the validation batch,

$$q^{(V)}(\theta, \lambda) := \frac{1}{n_{(V)}} \sum_{i=1}^{n_{(V)}} \mathcal{L}_\epsilon(y_i^{(V)}, \mathbf{f}_{\theta,\lambda}^{(T)}(x_i^{(V)})) + \tau \, r^{(T)}(\theta, \lambda), \tag{3.19}$$

where (T) and (V) denotes training and validation. Importantly, in contrast to standard cross validation, not all data is required for each update due to the presence of the RCB. Furthermore, although the multiplier on the RCB is $4e$, experiments show that generalization performance can improve if we use a smaller multiplier $\tau < 4e$, suggesting an upper bound tighter than (3.17) may exist. In practice, these two extensions work well together. Intuitively, by introducing a validation batch to measure empirical data fit, a smaller weight on the complexity penalty is required.

3.7 Convergence Theorems and Proofs

In this section we provide theorems and derivations that establish convergence properties of MCEs. Most of the convergence results hold due to MCEs being special cases of CMEs, whose empirical estimates are known to converge. This section contains the proofs for theorems claimed in section 3.3 and section 3.4.

Suppose $\{X_i, Y_i\} \sim \mathbb{P}_{XY}$ are *iid* for all $i \in \mathbb{N}_n$, with $X_i : \Omega \to \mathcal{X}$ and $Y_i : \Omega \to \mathcal{Y}$. We wish to estimate some target function $f : \mathcal{X} \to \mathbb{R}$ by $\hat{f} : \mathcal{X} \to \mathbb{R}$ empirically with a dataset $\{X_i, Y_i\}_{i=1}^n$ of size $n \in \mathbb{N}_+$. Since $\hat{f}$ is empirically estimated, it is a random function over the possible data observation events $\omega \in \Omega$. The aim is to provide a sense of the stochastic convergence of $\hat{f}$ to f by providing an upper bound of their absolute pointwise difference $|\hat{f}(x) - f(x)|$, and show that such an upper bound converges to zero at some stochastic rate. Such an upper bound is provided by the convergence properties of CMEs. In particular, the empirical CME stochastically converges to the CME at rate $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$, under the assumption that $k(x, \cdot) \in \text{image}(C_{XX})$ [Song et al., 2009, Theorem 6]. That is,

$$\begin{aligned} &\forall x \in \mathcal{X},\ \forall \epsilon > 0,\ \exists M_\epsilon > 0 \quad s.t. \\ &\mathbb{P}\Big[\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l} > M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon. \end{aligned} \tag{3.20}$$

In practice, the assumption that $k(x, \cdot) \in \text{image}(C_{XX})$ can be relaxed by replacing $\mathcal{U}_{Y|X} = C_{YX}C_{XX}^{-1}$ with $\mathcal{U}_{Y|X} = C_{YX}(C_{XX} + \lambda I)^{-1}$ [Song et al., 2013]. This will apply to all subsequent theorems in this section.

Theorem 3.6 (Pointwise and Uniform Convergence of Conditional Mean Embedding Estimators)**.** *Suppose that $k(x, \cdot)$ is in the image of C_{XX} and that there exists $0 \leq \gamma(x) < \infty$ such that for some estimator function $\hat{f} : \mathcal{X} \to \mathbb{R}$ and target function $f : \mathcal{X} \to \mathbb{R}$,*

$$|\hat{f}(x) - f(x)| \leq \gamma(x)\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_l}, \forall x \in \mathcal{X}, \tag{3.21}$$

then the estimator $\hat{f}$ converges pointwise to the target f at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Further, if $\gamma(x) = \gamma$ is independent of $x \in \mathcal{X}$, then this convergence is uniform.

Proof. Suppose that there exists $0 \leq \gamma(x) < \infty$ such that (3.21) is satisfied. That is, the inequality (3.21) holds for all possible data observations $\{X_i, Y_i\}_{i=1}^n$ where $X_i : \Omega \to \mathcal{X}$, $Y_i : \Omega \to \mathcal{Y}$ for all $i \in \mathbb{N}_n$. For any constant C, the implication statement $\big\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\big\|_{\mathcal{H}_\delta} \leq C \implies |\hat{f}(x) - f(x)| \leq C\gamma(x)$ holds for all possible observation events $\omega \in \Omega$. Writing this explicitly in event space translates

this to a probability statement,

$$\begin{aligned}&\{\omega \in \Omega : \left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_l} \leq C\} \subseteq \{\omega \in \Omega : |\hat{f}(x) - f(x)| \leq C\gamma(x)\} \\ &\implies \mathbb{P}\Big[\left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_l} \leq C\Big] \leq \mathbb{P}\Big[|\hat{f}(x) - f(x)| \leq C\gamma(x)\Big].\end{aligned} \tag{3.22}$$

Since we assume that $k(x, \cdot) \in \text{image}(C_{XX})$, statement (3.20) is valid. By letting $C = M_\epsilon((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ in (3.22), we immediately have that the probability inequality in statement (3.20) is also true if we replace $\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|$ with $|\hat{f}(x) - f(x)|$ and M_ϵ with $\gamma(x)M_\epsilon$,

$$\begin{aligned}&\mathbb{P}\Big[\left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_l} > M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\ \implies &1 - \mathbb{P}\Big[\left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_l} \leq M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\ \implies &\mathbb{P}\Big[\left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_l} \leq M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] > 1 - \epsilon \\ \implies &\mathbb{P}\Big[|\hat{f}(x) - f(x)| \leq \gamma(x)M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] > 1 - \epsilon \\ \implies &1 - \mathbb{P}\Big[|\hat{f}(x) - f(x)| \leq \gamma(x)M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\ \implies &\mathbb{P}\Big[|\hat{f}(x) - f(x)| > \gamma(x)M_\epsilon\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon,\end{aligned} \tag{3.23}$$

where we employed statement (3.22) between the third and fourth line for $C = M_\epsilon((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Therefore, since M_ϵ is arbitrary, define $\tilde{M}_\epsilon(x) := \gamma(x)M_\epsilon$ so that, with the above result, the statement (3.20) implies the following,

$$\forall x \in \mathcal{X},\ \epsilon > 0,\ \exists \tilde{M}_\epsilon(x) > 0 \quad s.t. \quad \mathbb{P}\Big[\big|\hat{f}(x) - f(x)\big| > \tilde{M}_\epsilon(x)\Big((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon. \tag{3.24}$$

In other words, the function $\hat{f}$ stochastically converges pointwise to f with a rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. The convergence is pointwise as the constant $\tilde{M}_\epsilon(x)$ may be different for each point $x \in \mathcal{X}$. If $\gamma(x) = \gamma$ such that $\tilde{M}_\epsilon(x) = \tilde{M}_\epsilon$ does not depend on $x \in \mathcal{X}$, then this stochastic convergence is uniform in its domain $\mathcal{X}$. □

With theorem 3.6, we can now show the convergence of various estimators based on the conditional mean embedding, as long as we can show that their estimator error is upper bounded by a multiple of the conditional mean embedding error in the RKHS norm. As such, we turn to the convergence of the empirical decision probability function (3.4) below.

Theorem 3.7 (Convergence of Empirical Decision Probability Function)**.** *Assuming that $k(x, \cdot)$ is in the image of C_{XX}, the empirical decision probability function $\hat{p}_c : \mathcal{X} \to \mathbb{R}$ (3.4) converges uniformly to the true decision probability $p_c : \mathcal{X} \to [0, 1]$ (3.3) at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$.*

Proof. Consider the pointwise absolute difference between the decision probability and its empirical estimate,

$$\begin{aligned} |\hat{p}_c(x) - p_c(x)| &= |\langle \hat{\mu}_{Y|X=x}, \mathbb{1}_c \rangle - \langle \mu_{Y|X=x}, \mathbb{1}_c \rangle| \\ &= |\langle \hat{\mu}_{Y|X=x} - \mu_{Y|X=x}, \mathbb{1}_c \rangle| \\ &\leq \left\| \hat{\mu}_{Y|X=x} - \mu_{Y|X=x} \right\|_{\mathcal{H}_\delta} \left\| \mathbb{1}_c \right\|_{\mathcal{H}_\delta}, \end{aligned} \tag{3.25}$$

where the last inequality follows from the Cauchy Schwarz inequality in a Hilbert space.

Since $\mathbb{1}_c = \delta(c, \cdot)$ and using the fact that δ is a reproducing kernel, we have that for all $c \in \mathcal{Y} = \mathbb{N}_m$.

$$\left\| \mathbb{1}_c \right\|_{\mathcal{H}_\delta}^2 = \langle \mathbb{1}_c, \mathbb{1}_c \rangle = \langle \delta(c, \cdot), \delta(c, \cdot) \rangle = \delta(c, c) = 1. \tag{3.26}$$

Therefore, by theorem 3.6 with $\gamma(x) = 1$ independent of $x \in \mathcal{X}$, $\hat{p}_c$ converges uniformly to p_c at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$. □

The above proof is for uniform convergence over all $x \in \mathcal{X}$ at the stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Intuitively, however, for stationary zero-centered kernels like the Gaussian kernel, the convergence rate may be higher at regions of high data density, since the kernel effects, being centered around the training data, are stronger at these regions. The worse case convergence rate described here in the theorem would be a tight lower bound for regions in $\mathcal{X}$ with lower data density, where the kernel effects have decayed and most empirical probabilities are smaller and further from summing up to one.

Because the label space $\mathcal{Y} = \mathbb{N}_m$ is discrete and finite, *bounded* functions $g \in \mathcal{H}_\delta$ in the RKHS are equivalent to their vector representations $\mathbf{g} := \{g(c)\}_{c=1}^m$, because one can always write $g = \sum_{c=1}^m g(c)\delta(c, \cdot)$. In other words, there is an isomorphism between $\mathbb{H}_\delta$ and $\mathbb{R}^m$. A convenient consequence is that inner products in the RKHS are simply the usual dot products in a Euclidean space, since

$$\begin{aligned} \langle g_1, g_2 \rangle_{\mathcal{H}_\delta} &= \left\langle \sum_{c=1}^m g_1(c)\delta(c, \cdot), \sum_{c'=1}^m g_2(c')\delta(c', \cdot) \right\rangle_{\mathcal{H}_\delta} \\ &= \sum_{c=1}^m \sum_{c'=1}^m g_1(c) g_2(c') \langle \delta(c, \cdot), \delta(c', \cdot) \rangle_{\mathcal{H}_\delta} \\ &= \sum_{c=1}^m g_1(c) g_2(c) \\ &= \mathbf{g}_1 \cdot \mathbf{g}_2. \end{aligned} \tag{3.27}$$

Consequently, the RKHS norm for bounded functions $g \in \mathcal{H}_\delta$ is simply the ℓ_2-norm of its vector representation $\mathbf{g}$,

$$\|g\|_{\mathcal{H}_\delta} = \|\mathbf{g}\|_{\ell_2}. \tag{3.28}$$

A special and convenient result that arises due to this discrete and finite label space is that the decision probabilities and its empirical estimate are simply the conditional mean embeddings and its empirical estimate.

Lemma 3.1 (Decision Probabilities are Conditional Mean Embeddings)**.** *The decision probability for class $c \in \mathbb{N}_m$ given an example $x \in \mathcal{X}$ is the conditional mean embedding with $l = \delta$ conditioned at example x evaluated at label c,*

$$p_c(x) := \mathbb{P}[Y = c|X = x] = \mu_{Y|X=x}(c). \tag{3.29}$$

Therefore, $\mathbf{p}(x) \equiv \mu_{Y|X=x}$.

Proof. Since indicator functions are the canonical features of the label RKHS $\mathcal{H}_\delta$, we employ the fact that expectations of indicator functions are probabilities to prove this claim,

$$\begin{aligned} \mu_{Y|X=x}(c) :=& \mathbb{E}[l(Y,c)|X=x] = \mathbb{E}[\delta(Y,c)|X=x] \\ =& \mathbb{E}[\mathbb{1}_c(Y)|X=x] = \mathbb{P}[Y \in \{c\}|X=x] \\ =& \mathbb{P}[Y=c|X=x] =: p_c(x). \end{aligned} \tag{3.30}$$

□

Lemma 3.2 (Empirical Decision Probabilities are Empirical Conditional Mean Embeddings)**.** *The empirical decision probability (3.4) for class $c \in \mathbb{N}_m$ given an example $x \in \mathcal{X}$ is the empirical conditional mean embedding with $l = \delta$ conditioned at example x evaluated at label c,*

$$\hat{p}_c(x) = \hat{\mu}_{Y|X=x}(c). \tag{3.31}$$

Therefore, $\hat{\mathbf{p}}(x) \equiv \hat{\mu}_{Y|X=x}$.

Proof. Let the canonical feature maps of $\mathcal{X}$ and $\mathcal{Y}$ be $\phi(x) = k(x,\cdot)$ and $\psi(y) = l(y,\cdot) = \delta(y,\cdot)$, then the empirical conditional mean embedding is defined by

$$\hat{\mu}_{Y|X=x} := \hat{\mathcal{U}}_{Y|X}\phi(x). \tag{3.32}$$

By the reproducing property, the evaluation of $\hat{\mu}_{Y|X=x} \in \mathcal{H}_l$ is given by a dot product,

$$\begin{aligned}\hat{\mu}_{Y|X=x}(c) &= \langle l(c,\cdot), \hat{\mu}_{Y|X=x}\rangle \\ &= \langle \psi(c), \hat{\mu}_{Y|X=x}\rangle \\ &= \psi(c)^T \hat{\mu}_{Y|X=x} \\ &= \psi(c)^T \hat{\mathcal{U}}_{Y|X}\phi(x) \\ &= \psi(c)^T \Psi(K+n\lambda I)^{-1}\Phi^T\phi(x) \\ &= \mathbf{l}_c^T(K+n\lambda I)^{-1}\mathbf{k}(x),\end{aligned} \tag{3.33}$$

where $\mathbf{l}_c := \{l(y_i,c)\}_{i=1}^n$ and $\mathbf{k}_x := \{k(x_i,x)\}_{i=1}^n$. While the notation $\mathbf{l}_c$ is usually avoided due do its similarity to $\mathbf{1}_c$, in this context they happen to represent equal quantities,

$$\mathbf{l}_c := \{l(y_i,c)\}_{i=1}^n = \{\delta(y_i,c)\}_{i=1}^n = \{\mathbb{1}_c(y_i)\}_{i=1}^n =: \mathbf{1}_c. \tag{3.34}$$

The claim then immediately follows by the definition of our decision probability estimator,

$$\hat{\mu}_{Y|X=x}(c) = \mathbf{1}_c^T(K+n\lambda I)^{-1}\mathbf{k}(x) =: \hat{p}_c(x). \tag{3.35}$$

□

Lemma 3.2 shows that the decision function $\mathbf{f}(x)$ (3.5) of a MCE is no more than the empirical conditional mean embedding estimated from the data.

Since we have identified the equivalence of decision probabilities and the conditional mean embedding, we can now also show that the empirical decision probability vector also converges to the true decision probability vector.

Lemma 3.3 (Uniform Convergence of Empirical Decision Probability Vector Function in ℓ_1 and ℓ_2)**.** *Assuming that $k(x,\cdot)$ is in the image of C_{XX}, the empirical decision probability vector function $\hat{\mathbf{p}} : \mathcal{X} \to \mathbb{R}^m$ (3.5) converges uniformly to the true decision probability vector function $\mathbf{p} : \mathcal{X} \to [0,1]^m$ in the ℓ_1-norm and ℓ_2-norm, where $\mathbf{p}(x) := \{p_c(x)\}_{c=1}^m$, at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $c \in \mathcal{Y} = \mathbb{N}_m$.*

Proof. For convergence in ℓ_1, we simply extend theorem 3.7, which proved that each entry of $\hat{\mathbf{p}}(x)$ converges pointwise uniformly at a rate of $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ to the corresponding entry of $\mathbf{p}(x)$. Since each entry converges stochastically at a rate of $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$, then so does the entire vector. More formally, from

(3.25) and (3.26), the ℓ_1-norm of the difference can be bounded,

$$\begin{aligned}\|\hat{\mathbf{p}}(x) - \mathbf{p}(x)\|_{\ell_1} :=& \sum_{c=1}^{m} |\hat{p}_c(x) - p_c(x)| \\ \leq& \sum_{c=1}^{m} \left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_\delta} \\ =& m\left\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\right\|_{\mathcal{H}_\delta}.\end{aligned} \tag{3.36}$$

Therefore, by theorem 3.6 with $\gamma(x) = m$ independent of $x \in \mathcal{X}$, we have uniform convergence in ℓ_1 where we replace all instances of $|\hat{f}(x) - f(x)|$ in the proof of theorem 3.6 with $\|\hat{\mathbf{p}}(x) - \mathbf{p}(x)\|_{\ell_1}$.

For convergence in ℓ_2, we show that the ℓ_2-norm of the difference between the true and empirical decision probability vector functions is the same as the RKHS norm of the difference between the true and empirical conditional mean embedding, which converges to zero at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $x \in \mathcal{X}$ and $c \in \mathcal{Y} = \mathbb{N}_m$ by (3.20). To this end, we use lemma 3.1 and lemma 3.2 and write

$$\begin{aligned}\|\hat{\mathbf{p}}(x) - \mathbf{p}(x)\|_{\ell_2} &= \|\{\hat{p}_c(x)\}_{c=1}^m - \{p_c(x)\}_{c=1}^m\|_{\ell_2} \\ &= \|\{\hat{p}_c(x) - p_c(x)\}_{c=1}^m\|_{\ell_2} \\ &= \|\{\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)\}_{c=1}^m\|_{\ell_2} \\ &= \|\hat{\boldsymbol{\mu}}_{Y|X=x} - \boldsymbol{\mu}_{Y|X=x}\|_{\ell_2} \\ &= \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta},\end{aligned} \tag{3.37}$$

where the last equality comes from (3.28) and the fact that the empirical and true conditional mean embeddings are bounded functions in the RKHS. Again, by theorem 3.6 with $\gamma(x) = 1$ independent of $x \in \mathcal{X}$, we have uniform convergence in ℓ_2. □

Finally, since the estimated decision probabilities converge, the estimated information entropy also converges.

Theorem 3.8 (Convergence of Empirical Information Entropy Function)**.** *Assuming that $k(x, \cdot)$ is in the image of C_{XX}, the empirical information entropy function $\hat{h} : \mathcal{X} \to \mathbb{R}$ (3.12) converges pointwise to the true information entropy function $h : \mathcal{X} \to [0, \infty)$ at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.*

Proof. Since we are interested in the asymptotic properties of our estimators when $n \to \infty$, and we have proved that the empirical decision probabilities converges to the true probabilities (theorem 3.7), the condition $\hat{p}_c(x) > 0$ holds for large n such that we simply have $\hat{u}_x(c) = -\log \hat{p}_c(x)$. That is, the effects of clipping for the information estimate (3.11) vanishes.

Consider the pointwise absolute difference between the empirical and true information entropy,

$$\begin{aligned}
|\hat{h}(x) - h(x)| &= |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle_{\mathcal{H}_\delta} - \langle \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&= |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle_{\mathcal{H}_\delta} - \langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta} \\
&\quad + \langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta} - \langle \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&\leq |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x \rangle_{\mathcal{H}_\delta} - \langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&\quad + |\langle \hat{\mu}_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta} - \langle \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&= |\langle \hat{\mu}_{Y|X=x}, \hat{u}_x - u_x \rangle_{\mathcal{H}_\delta}| + |\langle \hat{\mu}_{Y|X=x} - \mu_{Y|X=x}, u_x \rangle_{\mathcal{H}_\delta}| \\
&\leq \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} \|\hat{u}_x - u_x\|_{\mathcal{H}_\delta} + \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \|u_x\|_{\mathcal{H}_\delta},
\end{aligned} \tag{3.38}$$

where to obtain the inequalities we used the triangle inequality and Cauchy Schwarz inequality in a Hilbert space respectively. Since the kernel $l = \delta$ is bounded, so is $\hat{\mu}_{Y|X=x}(c) = \sum_{i=1}^{n} w_i \delta(y_i, c)$ for some embedding weights w_i and all $c \in \mathbb{N}_m$, and thus its RKHS norm is finite for all $n \in \mathbb{N}_n$. Similarly, assuming that $p_c(x)$ is never exactly zero, $u_x(c)$ is also finite for all $c \in \mathbb{N}_m$ and thus so is its RKHS norm. We already know that $\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}$ stochastically converges to zero at the rate $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ (3.20). Thus, it remains to bound $\|\hat{u}_x - u_x\|_{\mathcal{H}_\delta}$ by a multiple of $\|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}$.

To this end, we first use lemma 3.1 and lemma 3.2 and to express the theoretical and empirical information as the negative log of the embedding, so that it is explicitly written as a function of $c \in \mathcal{Y}$ in $\mathcal{H}_\delta$ indexed by $x \in \mathcal{X}$,

$$\begin{aligned}
u_x(c) &= -\log p_c(x) = -\log \mu_{Y|X=x}(c), \\
\hat{u}_x(c) &= -\log \hat{p}_c(x) = -\log \hat{\mu}_{Y|X=x}(c).
\end{aligned} \tag{3.39}$$

Since log is a concave function, we have the property that $\log a - \log b \leq \frac{1}{b}(a - b)$. This allows us to bound $|\hat{u}_x(c) - u_x(c)|$ by $|\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|$ for all $c \in \mathbb{N}_m$,

$$\begin{aligned}
|\hat{u}_x(c) - u_x(c)| &= |\log \hat{\mu}_{Y|X=x}(c) - \log \mu_{Y|X=x}(c)| \\
&\leq \frac{1}{|\mu_{Y|X=x}(c)|} |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)| \\
&\leq \alpha_x |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|,
\end{aligned} \tag{3.40}$$

where we define $\alpha_x := \max_{c \in \mathbb{N}_m} \frac{1}{|\mu_{Y|X=x}(c)|}$, which is well defined as the conditional mean embedding is bounded. Since the RKHS norm of bounded functions in $\mathcal{H}_\delta$

is simply the ℓ_2-norm of their vector representations (3.28), we have

$$\begin{aligned}
\|\hat{u}_x - u_x\|_{\mathcal{H}_\delta}^2 &= \|\hat{\mathbf{u}}_x - \mathbf{u}_x\|_{\ell_2}^2 \\
&= \sum_{c=1}^{m} |\hat{u}_x(c) - u_x(c)|^2 \\
&\leq \sum_{c=1}^{m} \alpha_x^2 |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|^2 \\
&\leq \alpha_x^2 \sum_{c=1}^{m} |\hat{\mu}_{Y|X=x}(c) - \mu_{Y|X=x}(c)|^2 \\
&\leq \alpha_x^2 \|\hat{\boldsymbol{\mu}}_{Y|X=x} - \boldsymbol{\mu}_{Y|X=x}\|_{\ell_2}^2 \\
&\leq \alpha_x^2 \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}^2.
\end{aligned} \tag{3.41}$$

Therefore, $\|\hat{u}_x - u_x\|_{\mathcal{H}_\delta} \leq \alpha_x \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}$, and (3.38) becomes

$$\begin{aligned}
|\hat{h}(x) - h(x)| &\leq \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} \|\hat{u}_x - u_x\|_{\mathcal{H}_\delta} + \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \|u_x\|_{\mathcal{H}_\delta} \\
&= \alpha_x \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \\
&\quad + \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta} \|u_x\|_{\mathcal{H}_\delta} \\
&= (\alpha_x \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} + \|u_x\|_{\mathcal{H}_\delta}) \|\hat{\mu}_{Y|X=x} - \mu_{Y|X=x}\|_{\mathcal{H}_\delta}.
\end{aligned} \tag{3.42}$$

Hence, with $\gamma(x) = \alpha_x \|\hat{\mu}_{Y|X=x}\|_{\mathcal{H}_\delta} + \|u_x\|_{\mathcal{H}_\delta}$, theorem 3.6 implies that $\hat{h}$ converges pointwise to h at a stochastic rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. □

3.8 Learning Theoretic Bounds and Proofs

In this section we derive RCBs for MCEs, and show that it can be used in conjunction with cross entropy loss to bound the expected risk with high probability. This section contains the proofs for theorems claimed in section 3.5, as well as detailed discussions on how the theorems and proofs were motivated and constructed.

3.8.1 Rademacher Complexity Bounds

Suppose a set of training data $\{x_i, y_i\}_{i=1}^n$ is drawn from $\mathbb{P}_{XY}$ in an *iid* fashion. We denote the one hot encoded target labels of $\{y_i\}_{i=1}^n$ by $\mathbf{y}_i := \{\mathbb{1}_c(y_i)\}_{c=1}^m \in \{0,1\}^m$ and $\mathbf{Y} := \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_n\end{bmatrix}^T \in \{0,1\}^{n\times m}$. Similarly, let $\mathbf{y} \in \{0,1\}^m$ denote the one hot encoded target labels for a generic label $y \in \mathcal{Y}$. Let $k_\theta : \mathcal{X} \times \mathcal{X} \to [0, \infty)$ be a family of positive definite kernels indexed by $\theta \in \Theta$. As before, we define the shorthand notation for the gram matrices $K_\theta := \{k_\theta(x_i, x_j) : i \in \mathbb{N}_n, j \in \mathbb{N}_n\}$ and

$\mathbf{k}_\theta(x) := \{k_\theta(x_i, x) : i \in \mathbb{N}_n\}$, and λ denotes the regularization hyperparameter of the conditional mean embedding (3.1). The MCE has a predictor form $\hat{\mathbf{p}}(x) = \mathbf{f}_{\theta,\lambda}(x)$ (3.5) defined by

$$\mathbf{f}_{\theta,\lambda}(x) := \mathbf{Y}^T(K_\theta + n\lambda I)^{-1}\mathbf{k}_\theta(x), \tag{3.43}$$

where each entry of the predictor $\mathbf{f}_{\theta,\lambda}(x)$ is the decision probability estimate for $p_c(x)$. This defines the function class of the predictor over the kernel family and a set of regularization hyperparameters for any set of training observations $\{x_i, y_i\}_{i=1}^n$,

$$F_n(\Theta, \Lambda) := \{\mathbf{f}_{\theta,\lambda}(x) : \theta \in \Theta, \lambda \in \Lambda\}. \tag{3.44}$$

The predictor form (3.43) is linear in the reproducing kernel Hilbert space $\mathcal{H}_{k_\theta}$ induced by k_θ in the sense that

$$\begin{aligned} \mathbf{f}_{\theta,\lambda}(x) &:= W_{\theta,\lambda}^T \phi_\theta(x), \\ W_{\theta,\lambda} &:= \Phi_\theta(K_\theta + n\lambda I)^{-1}\mathbf{Y}, \end{aligned} \tag{3.45}$$

where we decompose $\mathbf{k}_\theta(x) = \Phi_\theta^T \phi_\theta(x)$ by the reproducing property. By lemma 3.2, $\mathbf{f}_{\theta,\lambda}(x) = \hat{\mathbf{p}}_{\theta,\lambda}(x) = \hat{\mu}_{Y|X=x}^{(\theta,\lambda)} = \hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)} \phi_\theta(x)$. Therefore, we have that $\hat{\mathcal{U}}_{Y|X}^{(\theta,\lambda)} \equiv W_{\theta,\lambda}^T$. Throughout this paper, inner products are defined in the Hilbert-Schmidt sense, which induces the Hilbert-Schmidt norm $\|\cdot\|_{HS}$ and generalises the Frobenius inner product with induced norm $\|\cdot\|_{\text{tr}}$ for finite dimensional operators. Nevertheless, while they refer to the same quantity, we will use the standard notations $\|\hat{\mathcal{U}}_{Y|X}^{\theta,\lambda}\|_{HS}$ as per the literature in Hilbert space embeddings and $\|W_{\theta,\lambda}\|_{\text{tr}}$ as per the literature for linear classifiers.

Theorem 3.9 (MCE Rademacher Complexity Bound)**.** *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map is bounded in RKHS norm $\|\phi_\theta(x)\|_{\mathcal{H}_{k_\theta}}^2 = k_\theta(x, x) \leq \alpha^2$, $\alpha > 0$, for all $x \in \mathcal{X}, \theta \in \Theta$. For any set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the class of MCEs $F_n(\Theta, \Lambda)$ (3.44) defined over $\theta \in \Theta, \lambda \in \Lambda$ is bounded by*

$$\mathcal{R}_n(F_n(\Theta, \Lambda)) \leq 2\alpha\rho. \tag{3.46}$$

Proof. The Rademacher complexity [Bartlett and Mendelson, 2002, Definition 2] of the function class $F_n(\Theta, \Lambda)$ is

$$\begin{aligned} \mathcal{R}_n(F_n(\Theta, \Lambda)) :=& \mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda} \left\| \frac{2}{n} \sum_{i=1}^n \sigma_i \mathbf{f}_{\theta,\lambda}(X_i) \right\|\right] \\ =& \frac{2}{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda} \left\| \sum_{i=1}^n \sigma_i \mathbf{f}_{\theta,\lambda}(X_i) \right\|\right], \end{aligned} \tag{3.47}$$

where σ_i are *iid* Rademacher random variables, taking values in $\{-1, 1\}$ with equal probability, and X_i are *iid* random variables from the same distribution $\mathbb{P}_X$ as our training data. We further define $\boldsymbol{\sigma} := \{\sigma_i\}_{i=1}^n$.

We first bound the term inside the supremum using the Cauchy Schwarz inequality,

$$\begin{aligned}
\left\| \sum_{i=1}^n \sigma_i \mathbf{f}_{\theta,\lambda}(X_i) \right\| &= \left\| \sum_{i=1}^n \sigma_i W_{\theta,\lambda}^T \phi_\theta(X_i) \right\| \\
&= \left\| W_{\theta,\lambda}^T \boldsymbol{\Phi}_\theta \boldsymbol{\sigma} \right\| \\
&\leq \|W_{\theta,\lambda}\|_{\text{tr}} \| \|\boldsymbol{\Phi}_\theta \boldsymbol{\sigma}\| \\
&\leq \|W_{\theta,\lambda}\|_{\text{tr}} \| \|\boldsymbol{\Phi}_\theta^T\|_{\text{tr}} \|\boldsymbol{\sigma}\| \\
&= \|W_{\theta,\lambda}\|_{\text{tr}} \| \|\boldsymbol{\Phi}_\theta\|_{\text{tr}} \|\boldsymbol{\sigma}\|,
\end{aligned} \tag{3.48}$$

where we define the random operator $\boldsymbol{\Phi}_\theta := \begin{bmatrix} \phi(X_1) & \phi(X_2) & \cdots & \phi(X_n) \end{bmatrix}$. Note that this is distinct from Φ_θ, whose columns are the canonical RKHS features at the training observations and is not random. Now, random or not, entries of $\boldsymbol{\sigma} := \{\sigma_i\}_{i=1}^n$ are either -1 or 1, so its norm is simply $\|\boldsymbol{\sigma}\| = \sqrt{n}$. We can then also compute the trace norm of the other random component $\boldsymbol{\Phi}_\theta$,

$$\begin{aligned}
\|\boldsymbol{\Phi}_\theta\|_{\text{tr}} :=& \sqrt{\text{trace}(\boldsymbol{\Phi}_\theta^T \boldsymbol{\Phi}_\theta)} \\
=& \sqrt{\text{trace}(\mathbf{K}_\theta)} \\
=& \sqrt{\sum_{i=1}^n k_\theta(X_i, X_i)} \\
=& \sqrt{\sum_{i=1}^n \alpha^2} \\
=& \sqrt{n\alpha^2} \\
=& \sqrt{n}\alpha,
\end{aligned} \tag{3.49}$$

where the inequality comes from the assertion that $k_\theta(x, x) \leq \alpha^2$ for all $x \in \mathcal{X}, \theta \in \Theta$. This bounds all the random components in the expectation by a constant, so that later the expectation can vanish.

Using the assertion that $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ for all $\theta \in \Theta, \lambda \in \Lambda$, we can now bound the Rademacher complexity,

$$\begin{aligned}
\mathcal{R}_n(F_n(\Theta,\Lambda)) &= \frac{2}{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\left\|\sum_{i=1}^{n}\sigma_i \mathbf{f}_{\theta,\lambda}(X_i)\right\|\right] \\
&\leq \frac{2}{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\|\mathbf{\Phi}_\theta\|_{\text{tr}}\|\boldsymbol{\sigma}\|\right] \\
&= \frac{2}{n}\sqrt{n}\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\|\mathbf{\Phi}_\theta\|_{\text{tr}}\right] \\
&\leq \frac{2}{n}\sqrt{n}\sqrt{n}\alpha\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\right] \\
&\leq 2\alpha\mathbb{E}\left[\sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}}\right] \\
&= 2\alpha \sup_{\theta\in\Theta,\lambda\in\Lambda}\|W_{\theta,\lambda}\|_{\text{tr}} \\
&\leq 2\alpha\rho.
\end{aligned} \tag{3.50}$$

□

Theorem 3.9 provides a generic Rademacher complexity bound for any type of MCE with a bounded positive definite kernel and bounded trace norm. One of the most widely used kernels in practice are the family of stationary kernels. We provide a more specific bound for the case of stationary kernels below.

Corollary 3.1 (Rademacher Complexity Bound for Stationary Kernels)**.** *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Suppose that k_θ is a family of positive definite stationary kernels. That is, $k_\theta(x,x') = \tilde{k}_\theta(\|x-x'\|)$ for some real-valued function $\tilde{k} : [0,\infty) \to [0,\infty)$. Select $\tilde{\theta} \in \Theta$ and define $\Theta(\tilde{\theta})$ such that $k_\theta(0,0) \leq k_{\tilde{\theta}}(0,0)$ for all $\theta \in \Theta(\tilde{\theta})$. For any $\tilde{\theta} \in \Theta$ and set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the resulting class of MCEs $F_n(\Theta(\tilde{\theta}),\Lambda)$ defined over $\theta \in \Theta(\tilde{\theta}), \lambda \in \Lambda$ is bounded by*

$$\mathcal{R}_n(F_n(\Theta(\tilde{\theta}),\Lambda)) \leq 2\rho\sqrt{k_{\tilde{\theta}}(0,0)}. \tag{3.51}$$

Proof. Observe that $k_{\tilde{\theta}}(0,0)$ is an upper bound for $k_\theta(x,x)$ for all $x \in \mathcal{X}$ and $\theta \in \Theta$,

$$k_\theta(x,x) = \tilde{k}_\theta(\|x-x\|) = \tilde{k}_\theta(\|0\|) = k_\theta(0,0) \leq k_{\tilde{\theta}}(0,0). \tag{3.52}$$

We simply choose $\alpha^2 = k_{\tilde{\theta}}(0,0)$ in theorem 3.9. □

Corollary 3.1 motivates the choice $\alpha^2(\theta) = k_\theta(0,0) = \sigma_f^2$ for stationary radial basis type kernels such as the Gaussian or Matérn kernels, where σ_f is the sensitivity

[Rasmussen and Williams, 2006] of the stationary kernel, which we employ in our learning algorithm when the kernel is stationary.

3.8.2 Expected Risk Bounds

In order to quantify the performance of the MCE, we specify a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{A} \to [0, \infty)$, where $\mathcal{L}(y, f(x))$ measures the loss of a decision function $f : \mathcal{X} \to \mathcal{A}$ on a paired example $x \in \mathcal{X}$ and label $y \in \mathcal{Y}$. In the MCE context, the decision function is $\mathbf{f}_{\theta,\lambda} : \mathcal{X} \to \mathbb{R}^m$, with $\mathcal{A} = \mathbb{R}^m$ and $\mathcal{Y} = \mathbb{N}_m$. The loss function is to capture the desire for $\mathbf{y}^T \mathbf{f}_{\theta,\lambda}(x) = f_y^{(\theta,\lambda)}(x)$ to be high for all likely test points $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

A suitable choice of the loss function in the probabilistic multiclass classification context is the cross entropy loss,

$$\mathcal{L}(y, \mathbf{f}(x)) := -\log \mathbf{y}^T \mathbf{f}(x) = -\log f_y(x), \tag{3.53}$$

where $\mathbf{f}(x)$ are the inferred decision probability estimates of each class for the example $x \in \mathcal{X}$. Since logarithms explode at zero, in practice the probability estimate is often clipped from below at a predetermined threshold $\epsilon \in (0, 1)$. Furthermore, it is also convenient to clip the probability estimate from above at one to avoid negative losses. Consequently, with the notation $[\,\cdot\,]_\epsilon^1 := \min\{\max\{\,\cdot\,, \epsilon\}, 1\}$, we define the effective cross entropy loss as

$$\mathcal{L}_\epsilon(y, \mathbf{f}(x)) := -\log [\mathbf{y}^T \mathbf{f}(x)]_\epsilon^1 = -\log [f_y(x)]_\epsilon^1. \tag{3.54}$$

In this way, our cross entropy loss (3.54) is both bounded and positive. In our subsequent analysis, we require that our loss function has an image in $[0, 1]$. To do this, we simply rescale the loss function by dividing it by its largest value,

$$\begin{aligned} \bar{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) &:= \frac{1}{M_\epsilon} \mathcal{L}_\epsilon(y, \mathbf{f}(x)) = -\frac{1}{M_\epsilon} \log [f_y(x)]_\epsilon^1, \\ M_\epsilon &:= -\log \epsilon. \end{aligned} \tag{3.55}$$

We will refer to (3.55) as the normalized cross entropy loss. We then further define the centered normalized cross entropy loss,

$$\tilde{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) := \bar{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) - \bar{\mathcal{L}}_\epsilon(y, \mathbf{0}) = -\frac{1}{M_\epsilon} \log [f_y(x)]_\epsilon^1 - 1. \tag{3.56}$$

With the normalized cross entropy loss (3.55) as our loss function, we now employ Theorem 8 of Bartlett and Mendelson [2002] for this loss and provide a bound for the expected normalized cross entropy loss for an unseen test example.

Lemma 3.4 (Expected Risk Bound)**.** *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $f \in F_n(\Theta, \Lambda)$ satisfies*

$$\frac{1}{M_\epsilon}\mathbb{E}[\mathcal{L}_\epsilon(Y, f(X))] \leq \frac{1}{nM_\epsilon}\sum_{i=1}^{n}\mathcal{L}_\epsilon(Y_i, f(X_i)) + \mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}. \tag{3.57}$$

Proof. Since $\bar{\mathcal{L}}_\epsilon : \mathcal{Y}\times\mathcal{A} \to [0,1]$ has a unit range and dominates itself, $\bar{\mathcal{L}}_\epsilon(y, f(x)) \leq \bar{\mathcal{L}}_\epsilon(y, f(x))$, the result follows directly from Theorem 8 of Bartlett and Mendelson [2002]. We then use the definition (3.55) for the normalized cross entropy loss. □

Equivalently, by definition (3.44), this result holds for $f = \mathbf{f}_{\theta,\lambda}(x)$ for every $\theta \in \Theta, \lambda \in \Lambda$. The bound (3.57) involves the Rademacher complexity $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda))$ of the centered normalized cross entropy loss applied onto the class of functions $F_n(\Theta, \Lambda)$, and not just the Rademacher complexity $\mathcal{R}_n(F_n(\Theta, \Lambda))$ of the class of functions $F_n(\Theta, \Lambda)$ itself. In theorem 3.9, we have bounded the latter. We now proceed to bound the former with the latter (3.46), so that the upper bound in lemma 3.4 can be written in terms of the latter.

Lemma 3.5 (Rademacher Complexity Bound with Cross Entropy Loss)**.** *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, the Rademacher complexity of the class of cross entropy loss applied onto the MCE is bounded by*

$$\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 2\frac{1}{\epsilon \log\frac{1}{\epsilon}}\mathcal{R}_n(F_n(\Theta, \Lambda)), \tag{3.58}$$

where $\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda) := \{(x, y) \mapsto \tilde{\mathcal{L}}_\epsilon(y, \mathbf{f}_{\theta,\lambda}(x)) : \theta \in \Theta, \lambda \in \Lambda\}$.

Proof. Let $\tilde{\psi}(z) := -\frac{1}{M_\epsilon}\log[z]_\epsilon^1 - 1$ so that $\tilde{\psi} : \mathbb{R} \to \mathbb{R}$ satisfies $\tilde{\psi}(0) = 0$. Then, the centered normalized cross entropy loss can be written as $\tilde{\mathcal{L}}_\epsilon(y, \mathbf{f}(x)) = \tilde{\psi}(f_y(x))$. In particular, $\tilde{\psi}(z)$ is piecewise differentiable. We proceed to show that $\tilde{\psi}$ is Lipschitz by showing that the supremum of its absolute derivative over all piecewise regions is finite, and thus infer its Lipschitz constant.

The real-valued function $\tilde{\psi}$ can be split into three piecewise regions over the real domain,

$$\tilde{\psi}(z) = \begin{cases} 0, & z \in (-\infty, \epsilon], \\ -\frac{1}{M_\epsilon}\log z - 1, & z \in (\epsilon, 1), \\ -1, & z \in [1, \infty). \end{cases} \tag{3.59}$$

The derivative over the regions $z \in (-\infty, \epsilon]$ and $z \in [1, \infty)$ is thus 0 and the local Lipschitz constant over that region is thus 0. We then focus on the other region,

$$\sup_{z\in(\epsilon,1)} |\tilde{\psi}'(z)| = \sup_{z\in(\epsilon,1)} \left| -\frac{1}{zM_\epsilon} \right| = \sup_{z\in(\epsilon,1)} \frac{1}{zM_\epsilon} = \frac{1}{\epsilon M_\epsilon} = \frac{1}{\epsilon \log \frac{1}{\epsilon}}. \tag{3.60}$$

Thus, $\tilde{\psi}$ is Lipschitz with a Lipschitz constant of $L_{\tilde{\psi}} = \frac{1}{\epsilon \log \frac{1}{\epsilon}}$.

For a given general loss function $\mathcal{L}$, Ledoux and Talagrand [2013, Corollary 3.17] proved that if there exists a Lipschitz real-valued function $\psi : \mathbb{R} \to \mathbb{R}$, $\psi(0) = 0$, with constant L_ψ such that $\mathcal{L}(y, f(x)) = \psi(f_y(x))$, then $\mathcal{R}_n(\mathcal{L} \circ F) \leq 2L_\psi \mathcal{R}_n(F)$ for any class of functions F. This result is also described in Bartlett and Mendelson [2002, Theorem 12.4].

Applying this result to our loss function with $\mathcal{L} = \tilde{\mathcal{L}}_\epsilon$ with $\psi = \tilde{\psi}$ and $F = F_n(\Theta, \Lambda)$, we have $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 2L_{\tilde{\psi}} \mathcal{R}_n(F_n(\Theta, \Lambda))$, which proves the claim. □

The bound (3.58) in lemma 3.5 will be the bridge that relates the expected cross entropy loss over our function class to the Rademacher complexity of our function class. We now proceed to state the main theorem which forms the backbone of our learning algorithm for the MCE.

Lemma 3.6 (MCE ϵ-General Expected Risk Bound)**.** *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map $\|\phi_\theta(x)\|^2_{\mathcal{H}_{k_\theta}} = k_\theta(x, x) \leq \alpha^2$, $\alpha > 0$, is bounded in RKHS norm for all $x \in \mathcal{X}, \theta \in \Theta$. For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability of at least $1 - \beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $f \in F_n(\Theta, \Lambda)$ satisfies*

$$\frac{1}{M_\epsilon} \mathbb{E}[\mathcal{L}_\epsilon(Y, f(X))] \leq \frac{1}{nM_\epsilon} \sum_{i=1}^n \mathcal{L}_\epsilon(Y_i, f(X_i)) + 4\frac{1}{\epsilon \log \frac{1}{\epsilon}} \alpha\rho + \sqrt{\frac{8}{n} \log \frac{2}{\beta}}, \tag{3.61}$$

for any $\epsilon \in (0, 1)$. Equivalently, the bound (3.61) holds for $f = \mathbf{f}_{\theta,\lambda}(x)$ for every $\theta \in \Theta, \lambda \in \Lambda$.

Proof. From theorem 3.9, we have $\mathcal{R}_n(F_n(\Theta, \Lambda)) \leq 2\alpha\rho$. Further, from lemma 3.5, we have $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 2\frac{1}{\epsilon \log \frac{1}{\epsilon}} \mathcal{R}_n(F_n(\Theta, \Lambda))$. These are both deterministic inequalities, leading to $\mathcal{R}_n(\tilde{\mathcal{L}}_\epsilon \circ F_n(\Theta, \Lambda)) \leq 4\frac{1}{\epsilon \log \frac{1}{\epsilon}} \alpha\rho$. We then apply this inequality to lemma 3.4, which proves the claim. □

Similar to many learning theoretic bounds, the expected risk bound (3.61) is composed of three qualitatively different terms. The first term is a training loss or

data fit term, which is a measure of how poorly the decision function f is performing on a given training dataset. The second term is a model complexity or regularization term, which measures how complicated the model is. In this case, the model complexity is measured by the Rademacher complexity, which captures the expressiveness of the function class by quantifying how well the function class is able to shatter noise. The third term is a statistical constant which plays no specific role to the function class.

We will eventually be minimizing the first two terms over some class of functions $f \in F_n(\Theta, \Lambda)$ with some approach, as a proxy to minimizing the actual expected risk. It would be fruitful to develop an intuition for the tightness of the bound from the contributions of the training loss term and the model complexity term. Since, like the expected loss, the training loss term is always in the unit range $[0, 1]$, we focus on understanding the tightness of the bound contributed from the complexity term.

Consider a clipped cross entropy loss with either a very small clipping factor $\epsilon \approx 0$, or a very large clipping factor $\epsilon \approx 1$. In these scenarios, $\epsilon \log \frac{1}{\epsilon}$ would be very small, so that the coefficient on the complexity term would then be very large, regardless of what the complexity bound factors α and ρ are. As a result, intuitively, this bound is unlikely to be tight due to the large coefficient on the complexity term.

Consequently, it would then be natural to consider a middle-ground choice of the cross entropy loss where this bound is the most tight by varying $\epsilon \in (0, 1)$. Since $\epsilon \log \frac{1}{\epsilon}$ is maximized at $\epsilon = \frac{1}{e}$ for a maximal value of $\frac{1}{e}$, such a choice in the clipping factor would indeed yield the tightest bound for the complexity bound in terms of the bounding slack of the result stated in lemma 3.5.

This is great news for the complexity term. What about the training loss term? Intuition tells us that, with a clipping factor of $\epsilon = e^{-1}$ that is slightly more than a third of the way into the interval $(0, 1)$ from zero, the classifier is not being penalised as strongly for assigning probabilities smaller than e^{-1} to observed classes as compared to very small values of ϵ. Furthermore, beyond the clipping point, assigning even lower probabilities to the observations does not result in a higher loss. In practice, the cross entropy loss is renowned for its rapidly growing penalty as the probability assignment gets lower, which is advantageous when using a gradient based optimization scheme. In this case, the gradients are large in magnitude and the classifier can adjust and fix these assignment errors relatively quickly. In other words, by using a slightly larger clipping factor than usual, we have seemingly lost the faster convergence properties from using a cross entropy loss.

Nevertheless, observe that for such a clipping factor $\epsilon = e^{-1}$, the normalization constant becomes $M_{e^{-1}} = -\log \frac{1}{e} = 1$, so that it is effectively removed. Furthermore, we also have the following simple upper bound for the cross entropy loss

clipped at $\epsilon = e^{-1}$,

$$\bar{\mathcal{L}}_{e^{-1}}(y, f(x)) = \mathcal{L}_{e^{-1}}(y, f(x)) \leq \mathcal{L}_{\epsilon}(y, f(x)) \quad \forall \epsilon \in (0, e^{-1}), x \in \mathcal{X}, y \in \mathcal{Y}. \tag{3.62}$$

To see why inequality (3.62) holds, note that $[f_y(x)]_{\epsilon}^1 \leq [f_y(x)]_{e^{-1}}^1$ holds for all $\epsilon \in (0, e^{-1}), x \in \mathcal{X}, y \in \mathcal{Y}$. Applying negative log to both sides yields the inequality from definition (3.54).

Therefore, we propose to choose $\epsilon = e^{-1}$, and then replace replace $\mathcal{L}_{e^{-1}}$ with $\mathcal{L}_{\epsilon}$ for some new generic $\epsilon \in (0, e^{-1})$ much smaller than e^{-1} on the training loss terms. In this way, we still maintain an upper bound for the training loss term. While this bound would not necessarily be tight for high training losses, the gradients from the high training loss would drive the system to a lower training loss, where the bound would become tight again as equality holds in (3.62) whenever $f_y(x) \geq e^{-1}$.

The above intuition motivates the result in the following theorem.

Theorem 3.10 (MCE ϵ-Specific Expected Risk Bound)**.** *Suppose that the trace norm $\|W_{\theta,\lambda}\|_{\text{tr}} \leq \rho$ is bounded for all $\theta \in \Theta, \lambda \in \Lambda$. Further suppose that the canonical feature map $\|\phi_\theta(x)\|_{\mathcal{H}_{k_\theta}}^2 = k_\theta(x,x) \leq \alpha^2$, $\alpha > 0$, is bounded in RKHS norm for all $x \in \mathcal{X}, \theta \in \Theta$. For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability of at least $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $f \in F_n(\Theta, \Lambda)$ satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, f(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{\epsilon}(Y_i, f(X_i)) + 4e\,\alpha\rho + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}, \tag{3.63}$$

for any $\epsilon \in (0, e^{-1})$. Equivalently, the bound (3.63) holds for $f = \mathbf{f}_{\theta,\lambda}(x)$ for every $\theta \in \Theta, \lambda \in \Lambda$.

Proof. We first apply lemma 3.6 with $\epsilon = e^{-1}$,

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, f(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{e^{-1}}(Y_i, f(X_i)) + 4e\,\alpha\rho + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}. \tag{3.64}$$

For any $\epsilon \in (0, e^{-1})$, the inequality $\mathcal{L}_{e^{-1}}(Y_i, f(X_i)) \leq \mathcal{L}_{\epsilon}(Y_i, f(X_i))$ holds almost surely (a.s.) due to the deterministic inequality (3.62). These sets of inequalities together proves the claim. □

3.8.3 Hyperparameter Learning

We are now ready to use the result of theorem 3.10 to derive a specific expected risk bound for a given choice of hyperparameters $\theta \in \Theta$ and $\lambda \in \Lambda$ of the MCE,

and not just for a general set of hyperparameters. We focus on kernels k_θ that are bounded over the domain $\mathcal{X}$ in the sense that for each $\theta \in \Theta$, $k_\theta(x,x) < \infty$ for all $x \in \mathcal{X}$.

For some kernel hyperparameters $\tilde{\theta} \in \Theta$ and regularization hyperparameter $\tilde{\lambda} \in \Lambda$, we construct a subset of hyperparameters (kernel hyperparameters and regularization hyperparameters) $\Xi(\tilde{\theta}, \tilde{\lambda}) \subseteq \Theta \times \Lambda$ such that

$$\begin{aligned}\Xi(\tilde{\theta}, \tilde{\lambda}) := \{(\theta, \lambda) \in \Theta \times \Lambda : \|W_{\theta,\lambda}\|_{\mathrm{tr}} \leq \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\mathrm{tr}}, \\ \sup_{x \in \mathcal{X}} k_\theta(x,x) \leq \alpha^2(\tilde{\theta}) := \sup_{x \in \mathcal{X}} k_{\tilde{\theta}}(x,x)\}.\end{aligned} \tag{3.65}$$

Clearly, this subset is non-empty, since $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ is itself an element of this subset. Note that $\alpha : \Theta \to \mathbb{R}_+$ must necessarily exist as the kernel family k_θ is assumed to be bounded over the domain $\mathcal{X}$. The class of MCEs over this subset of hyperparameters is

$$F_n(\Xi(\tilde{\theta}, \tilde{\lambda})) := \{\mathbf{f}_{\theta,\lambda}(x) : (\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})\}. \tag{3.66}$$

Thus, we can assert that the trace norm $\|W_{\theta,\lambda}\|_{\mathrm{tr}} \leq \rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\mathrm{tr}}$ is bounded for all $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, and that the canonical feature map $\|\phi_\theta(x)\|^2_{\mathcal{H}_{k_\theta}} = k_\theta(x,x) \leq \alpha^2 = \sup_{x \in \mathcal{X}} k_{\tilde{\theta}}(x,x)$ is bounded in RKHS norm for all $x \in \mathcal{X}, (\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$. By theorem 3.10, we can now claim the following.

Lemma 3.7 (MCE Expected Risk Bound for Hyperparameter Sets)**.** *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability $1 - \beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ satisfies*

$$\begin{aligned}\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, \mathbf{f}_{\theta,\lambda}(X))] \leq \frac{1}{n}\sum_{i=1}^n \mathcal{L}_\epsilon(Y_i, \mathbf{f}_{\theta,\lambda}(X_i)) \\ + 4e\sqrt{\sup_{x \in \mathcal{X}} k_{\tilde{\theta}}(x,x)}\|W_{\tilde{\theta},\tilde{\lambda}}\|_{\mathrm{tr}} + \sqrt{\frac{8}{n}\log\frac{2}{\beta}},\end{aligned} \tag{3.67}$$

for every $\epsilon \in (0, e^{-1})$, where

$$\begin{aligned}\mathbf{f}_{\theta,\lambda}(x) :=& \mathbf{Y}^T(K_\theta + n\lambda I)^{-1}\mathbf{k}_\theta(x), \\ \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\mathrm{tr}} =& \sqrt{\mathrm{trace}\left(\mathbf{Y}^T(K_{\tilde{\theta}} + n\tilde{\lambda}I)^{-1}K_{\tilde{\theta}}(K_{\tilde{\theta}} + n\tilde{\lambda}I)^{-1}\mathbf{Y}\right)}.\end{aligned} \tag{3.68}$$

Proof. We first apply theorem 3.10 with the choice of $\rho = \|W_{\tilde{\theta},\tilde{\lambda}}\|_{\mathrm{tr}}$ and $\alpha^2 = \sup_{x \in \mathcal{X}} k_{\tilde{\theta}}(x,x)$. The inequality (3.63) then only holds for a subset of kernel hyperparameters and regularizations $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ as defined by (3.65). □

Since inequality (3.67) holds for any $(\theta, \lambda) \in \Xi(\tilde{\theta}, \tilde{\lambda})$ and we know that $(\tilde{\theta}, \tilde{\lambda}) \in \Xi(\tilde{\theta}, \tilde{\lambda})$, we choose $\theta = \tilde{\theta}$ and $\lambda = \tilde{\lambda}$. We now arrive at our final result from which we can bound the expected risk for a specific choice of hyperparameters $\theta \in \Theta$ and $\lambda \in \Lambda$.

Theorem 3.11 (MCE Expected Risk Bound for Hyperparameters)**.** *For any integer $n \in \mathbb{N}_+$ and any set of training observations $\{x_i, y_i\}_{i=1}^n$, with probability $1-\beta$ over iid samples $\{X_i, Y_i\}_{i=1}^n$ of length n from $\mathbb{P}_{XY}$, every $\theta \in \Theta$ and $\lambda \in \Lambda$ satisfies*

$$\mathbb{E}[\mathcal{L}_{e^{-1}}(Y, \mathbf{f}_{\theta,\lambda}(X))] \leq \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}_{\epsilon}(Y_i, \mathbf{f}_{\theta,\lambda}(X_i)) + 4e\, r(\theta, \lambda) + \sqrt{\frac{8}{n}\log\frac{2}{\beta}}, \tag{3.69}$$

for every $\epsilon \in (0, e^{-1})$, where

$$\begin{aligned} \mathbf{f}_{\theta,\lambda}(x) &:= \mathbf{Y}^T(K_\theta + n\lambda I)^{-1}\mathbf{k}_\theta(x), \\ r(\theta, \lambda) &:= \sqrt{\operatorname{trace}\left(\mathbf{Y}^T(K_\theta + n\lambda I)^{-1}K_\theta(K_\theta + n\lambda I)^{-1}\mathbf{Y}\right)\sup_{x\in\mathcal{X}} k_\theta(x, x)}. \end{aligned} \tag{3.70}$$

Proof. We first apply lemma 3.7 with the choice of $\theta = \tilde{\theta}$ and $\lambda = \tilde{\lambda}$. We then replace the notation $\tilde{\theta} \to \theta$ and $\tilde{\lambda} \to \lambda$ back to avoid cluttered notation. Note that this should not be confused with the general θ and λ from earlier theorems. □

3.9 Model Architectures

For MCEs, the modeling lies in the choice of the kernel family $k_\theta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ over the input space $\mathcal{X}$. The only requirement for the kernel k is that it is symmetric and positive definite, and thus we may construct richer and more expressive kernel families in any way subject to such requirements. Once such a kernel family is constructed, the kernel hyperparameters θ, as well as the regularization hyperparameter λ, can be learned effectively using algorithm 1. Our learning algorithm does not restrict the way the kernel k_θ is constructed from its hyperparameters $\theta \in \Theta$.

One way to construct richer and more expressive kernels is to compose them from simpler kernels. For example, we can construct new kernels through convex combinations or products of multiple simpler kernels [Genton, 2001]. Any new parameters, such as coefficients for linear combinations of simpler kernels, can be included into the kernel hyperparameters θ and learned in the same way as before. Alternatively, there may be domain specific structures or representations within the data that can be exploited. We can then construct the kernel family by incorporating such structural representations into the kernel. Even better, we

Table 3.1: Properties of MCE model architectures

MCE Variant	Width	Depth	Scalability	Flexibility	Typical Datasets
Implicit MCE	Wide	Shallow	Low	High	High or Low d, Low n
Explicit MCE	Narrow	Shallow	High	Low	Low d, High n
Implicit CEN	Wide	Deep	Low	High	Structured d, Low n
Explicit CEN	Narrow	Deep	High	High	Structured d, High n

can construct the kernel family so that it is capable or learning such structural representations by itself, by parameterizing such representations into the kernel.

In this section, we focus on special cases of the MCE where the kernel family is constructed through explicit feature maps. This construction allows the incorporation of trainable domain specific structures and enables scalability to larger datasets. We first begin by introducing the explicit MCE in section 3.9.1, where explicit feature maps can be learned while enabling scalability to larger datasets. We then construct the conditional embedding network (CEN) in section 3.9.2, where the kernel family is formed from multiple layers of learned representations before a simpler kernel encodes their similarity for inference. Finally, we marry both constructions into the explicit CEN in section 3.9.3, which provides a scalable and more applicable version of the deep CEN by placing a linear kernel on the network features.

In essence, we can categorize MCE model architectures using two properties: the model width and the model depth. The model width represents the dimensionality of the feature space used to construct the linear decision boundaries. The model depth represents the number of transformations used to map examples from the input space to the feature space. By implicitly defining a high dimensional feature space through simple transformations, typical nonlinear kernels produce classifiers that have a shallow but wide architecture. In contrast, the three MCE variants to be introduced in this section form other combinations of model architecture in both depth and width. Of course, as a direct consequence of the kernel trick, this characterization of architecture is not mutually exclusive. For example, a polynomial kernel can be seen as a nonlinear kernel where higher order polynomial features are implicitly defined, or as a linear kernel on explicit polynomial features. We summarize those architectures in table 3.1.

3.9.1 Explicit Multiclass Conditional Embedding

The advantage of using a kernel-based classifier is that the kernel k allows us to express nonlinearities in a simple way. It does this by implicitly mapping the input space $\mathcal{X}$ to a high dimensional feature space $\mathcal{H}_k$ of non-linear basis functions such that decision boundaries become linear in that space. For many

kernels, such as the Gaussian kernel defined over the Euclidean space, the feature space $\mathcal{H}_k$ has dimensionality that is uncountably infinite. Nevertheless, by virtue of the Representer Theorem [Kimeldorf and Wahba, 1971], the resulting decision functions can be represented by a finite linear combination of kernels centered at the training data, and the MCE is no exception. This elegant and convenient result enables exact inference to be performed while only requiring a finite kernel gram matrix of the size of the dataset ($n \times n$) to be computed. In this way, the capacity of the model grows with the size of the dataset, which makes kernel methods nonparametric and very flexible, as it can adapt to the complexity of a dataset even with relatively simple kernels.

However, this elegant property is also the very reason that prevents kernel-based methods from scaling to larger datasets, as the size of such a gram matrix grows very quickly by $O(n^2)$. Many kernel-based methods also require the inversion of a regularized gram matrix, which has a time complexity of $O(n^3)$, and cannot be easily parallelized like standard matrix multiplications. As such, inference on datasets beyond tens of thousands of observations quickly becomes impractical to perform with kernel-based techniques.

In order to scale to big datasets, instead of placing a kernel over the input space directly and let it implicitly define the feature space, we explicitly define a finite dimensional feature space $\mathcal{Z} \subseteq \mathbb{R}^p$ of lower dimension p, where $p < n$, and place a linear kernel over it. That is, we specify a family of explicit features maps $\varphi_\theta : \mathcal{X} \to \mathcal{Z}$, and place a linear kernel on top of these explicit features,

$$k_\theta(x, x') = \varphi_\theta(x)^T \varphi_\theta(x'). \tag{3.71}$$

By explicitly defining a finite dimensional feature space, the matrix to be inverted during both learning and inference in the MCE can be reduced from size $n \times n$ to size $p \times p$ by using the Woodbury matrix inversion identity [Higham, 2002], reducing the time complexity to $O(p^3 + np^2)$. This allows scalable learning for $n >> p$ even without using batch gradient updates. For inference, standard map reduce methods can be used. We use this identity to modify algorithm 1 to algorithm 2 to exploit this computational speed up.

However, with a fixed and finite amount of feature basis, the model becomes parametric and its flexibility is compromised. In other words, the model is narrow in the number of feature representations. We therefore turn to multi-layered feature compositions, where the flexibility of a model comes from the deep architecture instead of implicit high dimensional features.

Algorithm 2 MCE Hyperparameter Learning with Stochastic Gradient Updates: Parametric Model Architecture with Explicit Features

Input: feature family $\varphi_\theta : \mathcal{X} \to \mathcal{Z} \subseteq \mathbb{R}^p$, data $\{x_i, y_i\}_{i=1}^n$, feature parameters θ_0, regularization hyperparameter λ_0, learning rate η, batch size n_b
$\theta \leftarrow \theta_0$, $\lambda \leftarrow \lambda_0$
repeat
  Sample the next batch $\mathcal{I}_b \subseteq \mathbb{N}_n$, s.t. $|\mathcal{I}_b| = n_b$
  $Y \leftarrow \{\delta(y_i, c) : i \in \mathcal{I}_b, c \in \mathbb{N}_m\}$ $\in \{0,1\}^{n_b \times m}$
  $Z_\theta \leftarrow \{\varphi_\theta(x_i) : i \in \mathcal{I}_b\}$ $\in \mathbb{R}^{n_b \times p}$
  $L_{\theta,\lambda} \leftarrow \text{cholesky}(Z_\theta^T Z_\theta + n_b \lambda I_p)$ $\in \mathbb{R}^{p \times p}$
  $W_{\theta,\lambda} \leftarrow L_{\theta,\lambda}^T \backslash (L_{\theta,\lambda} \backslash Z_\theta^T Y)$ $\in \mathbb{R}^{p \times m}$
  $P_{\theta,\lambda} \leftarrow Z_\theta W_{\theta,\lambda}$ $\in \mathbb{R}^{n_b \times m}$
  $r(\theta, \lambda) = \alpha(\theta) \sqrt{\sum_{c=1}^m \sum_{j=1}^p (W_{\theta,\lambda})_{j,c}^2}$
  $q(\theta, \lambda) \leftarrow \frac{1}{n_b} \sum_{i=1}^{n_b} \mathcal{L}_\epsilon((Y)_i, (P_{\theta,\lambda})_i) + 4e\, r(\theta, \lambda)$
  $(\theta, \lambda) \leftarrow \text{GradientBasedUpdate}(q, \theta, \lambda; \eta)$
until maximum iterations reached
Output: kernel hyperparameters θ, regularization hyperparameter λ

3.9.2 Conditional Embedding Network

For many application domains, there are natural structures in the data. For example, in image recognition, pixel dimensions are spatially correlated: nearby pixels are more related, and ordering between the pixel dimensions matter. One would expect convolutional features [LeCun et al., 1998] to be natural in this domain, and provide a performance boost to our classifier should it be included. In this way, we can often benefit by including domain specific structures and features into our model.

In this section, we focus on constructing kernels for which inputs $x, x' \in \mathcal{X}$ is to undergo various stages of feature transformations before such it is passed into a simpler kernel κ that captures the similarity between the representations. Specifically, we pay particular attention to feature transformations in the form of a perceptron, so that the cumulative stages of feature transformation become the (feed-forward) multi-layer perceptron that is familiar within the neural network literature.

Formally, let $\mathcal{F}_0 := \mathcal{X}$ be the original input space. The j^{th} layer of the network $\varphi_{\theta_j}^{(j)} : \mathcal{F}_{j-1} \to \mathcal{F}_j, j = 1, 2, \ldots, L$ is to transform features from the previous layer to features in the current layer, where L is the total number of such feature transformation layers, and $\theta_j \in \Theta_j$ parametrizes each of those transformations.

For example, in a typical multi-layer perceptron context, each layer can be written as $\varphi_{\theta_j}^{(j)}(x) = \sigma(W_j x + b_j)$, where W_j and b_j are the weight and bias parameters of the layer, and σ is an element-wise activation function, typically the rectified

linear unit (ReLU) or the sigmoid. In this case, the layer is parametrized by $\theta_j = \{W_j, b_j\}$.

Let $\kappa_{\theta_0} : \mathcal{F}_p \times \mathcal{F}_p \to \mathbb{R}$ be parametrized by $\theta_0 \in \Theta_0$. We will construct our kernel network k by

$$k_\theta(x, x') := \kappa_{\theta_0}\Bigg(\varphi_{\theta_L}^{(L)}\Big(\varphi_{\theta_{L-1}}^{(L-1)}\Big(\dots\varphi_{\theta_2}^{(2)}\big(\varphi_{\theta_1}^{(1)}(x)\big)\Big)\Big), \varphi_{\theta_L}^{(L)}\Big(\varphi_{\theta_{L-1}}^{(L-1)}\Big(\dots\varphi_{\theta_2}^{(2)}\big(\varphi_{\theta_1}^{(1)}(x')\big)\Big)\Big)\Bigg), \tag{3.72}$$

where $\theta = (\theta_1, \theta_2, \dots, \theta_{L-1}, \theta_L, \theta_0) \in \Theta = \Theta_1 \otimes \Theta_2 \otimes \cdots \otimes \Theta_{L-1} \otimes \Theta_L \otimes \Theta_0$ are the collection of all parameters of each layer and the kernel κ.

In order to train the multi-layered representations in an end-to-end fashion, we employ algorithm 1. With a deep architecture, the feature representations the CEN can learn are very flexible, and can work very well for structured data by employing suitable network architectures.

If we choose to employ nonlinear kernels κ, the model architecture is also wide in that an even higher dimensional feature space is implicitly defined on top of the feature space of the last network layer. Despite its supreme flexibility, this again prevents the model from being scalable. We therefore turn to the specific case where we employ a linear kernel κ on top of the multi-layered features.

3.9.3 Explicit Conditional Embedding Network

The explicit CEN is simply a special case at the intersection of the explicit MCE and the CEN. From the explicit MCE perspective, we simply choose the feature map $\varphi_\theta(x) = \varphi_{\theta_L}^{(L)}\big(\varphi_{\theta_{L-1}}^{(L-1)}(\dots\varphi_{\theta_2}^{(2)}(\varphi_{\theta_1}^{(1)}(x)))\big)$. From the CEN perspective, we simply choose $\kappa(z, z') = z^T z'$ to be a linear kernel.

This model architecture is a very practical and powerful form of the MCE. By having a deep architecture, the classifier is still capable of learning flexible representations on structured data, while being able to scale to larger datasets due to the linear kernel at the output layer, provided that the dimensionality of the last layer is relatively small compared to the size of the dataset.

An extremely useful example is when the features are formed from a deep convolutional neural network (CNN) for image datasets, where the CEN could leverage from the inductive bias the convolutional features provides while benefiting from improved tractability.

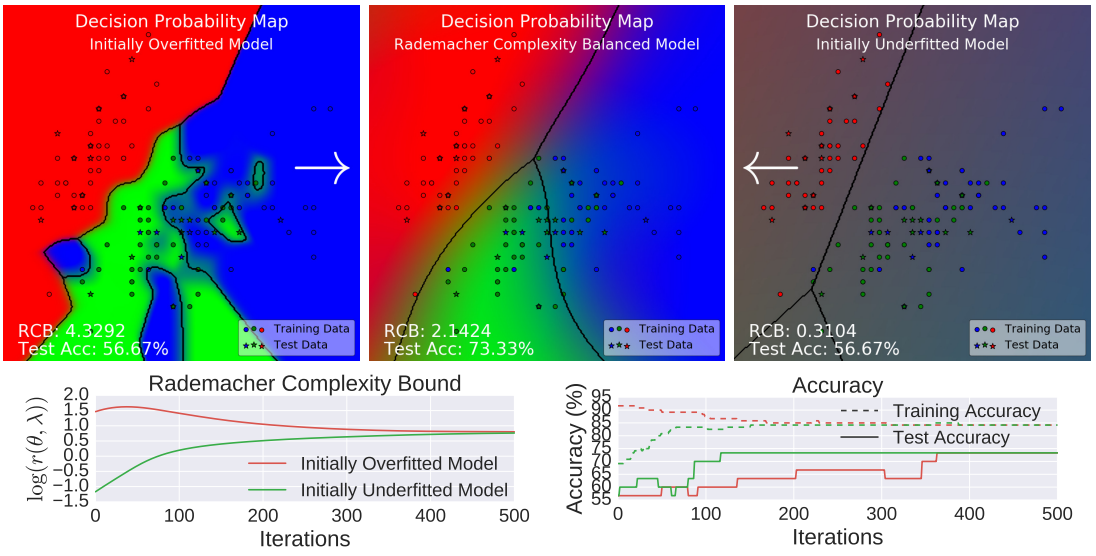

FIGURE 3.1: Rademacher complexity balanced learning of hyperparameters for an isotropic Gaussian MCE using the first two attributes of the iris dataset

As a subclass of explicit MCE, we can employ algorithm 2 to learn the multi-layered features effectively. In fact, by not mapping the multi-layered features into a nonlinear kernel, the gradients for each network weight and bias are usually more pronounced, and learning is usually faster in comparison. This approach was used to train the neural network features in our experiments.

3.10 Experiments

3.10.1 Toy Example

The first two of four total attributes of the iris dataset [Fisher, 1936] are known to have class labels that are non-separable by any means, in that the same example $x \in \mathbb{R}^2$ may be assigned different output labels $y \in \mathbb{N}_3 := \{1, 2, 3\}$. In these difficult scenarios, the notion of model complexity is extremely important, and the success of a learning algorithm greatly depends on how it balances training performance and model complexity to avoid both underfitting and overfitting.

Figure 3.1 demonstrates algorithm 1 with full gradient updates ($n_b = n$) to learn hyperparameters of the MCE on the two attribute iris dataset. The kernel used is isotropic Gaussian with diagonal length scales $\Sigma = \ell^2 I_2$ and sensitivity $\alpha = \sigma_f$, so that the hyperparameters are $\theta = (\alpha, \ell)$ and λ. We evaluate the performance of the learning algorithm on a withheld test set using 20% of the available 150 data samples. Attributes are scaled into the unit range $[0, 1]$ and decision probability maps are plotted for the region $[-0.05, 1.05]^2$, where the red, green, and blue

TABLE 3.2: Test accuracy (%) on UCI datasets

Method	banknote	ecoli	robot	segment	wine	yeast
GMCE	$\mathbf{99.9 \pm 0.2}$	$\mathbf{87.5 \pm 4.4}$	$\mathbf{96.7 \pm 0.9}$	$\mathbf{98.4 \pm 0.8}$	$\mathbf{97.2 \pm 3.7}$	52.5 ± 2.1
GMCE-SGD	98.8 ± 0.9	84.5 ± 5.0	95.5 ± 0.9	96.1 ± 1.5	93.3 ± 6.0	$\mathbf{60.3 \pm 4.4}$
CEN-1	99.5 ± 1.0	$\mathbf{87.5 \pm 3.2}$	82.3 ± 7.1	94.6 ± 1.6	96.1 ± 5.0	55.8 ± 5.0
CEN-2	99.4 ± 0.9	86.3 ± 6.0	94.5 ± 0.8	96.7 ± 1.1	97.2 ± 5.1	59.6 ± 4.0
ERM	$\mathbf{99.9 \pm 0.2}$	72.1 ± 20.5	91.0 ± 3.7	98.1 ± 1.1	93.9 ± 5.2	45.9 ± 6.4
CV	$\mathbf{99.9 \pm 0.2}$	73.8 ± 23.8	90.9 ± 3.4	98.3 ± 1.3	93.3 ± 7.4	58.0 ± 5.8
MLH	92.0 ± 4.3	42.1 ± 47.7	81.1 ± 6.2	27.3 ± 26.4	93.3 ± 7.8	31.2 ± 14.1
Others	99.78[a]	81.1[b]	97.59[c]	96.83[d]	100[e]	55.0[b]

color channels represent the clip-normalized decision probability (3.6) for classes $c = 1, 2, 3$. We begin from two initial sets of hyperparameters, one originally overfitting and another underfitting the training data. Initially, both models perform sub-optimally with a test accuracy of 56.67%. We see that the RCB $r(\theta, \lambda)$ appropriately measures the amount of overfitting with high (resp. low) values for the overfitted (resp. underfitted) model. We then learn hyperparameters with algorithm 1 for 500 iterations from both initializations at rate $\eta = 0.01$, where both models converges to a balanced model with a moderate RCB and an improved test accuracy of 73.33%. In particular, the initially overfitted model learns a simpler model at the expense of lower training performance, emphasizing the benefits of complexity based regularization, without which the learning would only maximize training performance at the cost of further overfitting. Meanwhile, the initially underfitted model learns to increase complexity to improve the sub-optimal performance on the training set.

3.10.2 UCI Datasets

We demonstrate the average performance of learning anisotropic Gaussian kernels and kernels constructed from neural networks on standard UCI datasets [Bache and Lichman, 2013], summarized in table 3.2. The former has a shallow but wide model architecture, while the latter has a deeper but narrower model architecture. The Gaussian kernel is learned with both full and batch stochastic gradient updates, referred as Gaussian multiclass conditional embedding (GMCE) and GMCE-stochastic gradient descent (SGD) respectively, using a tenth ($n_b \approx \frac{n}{10}$) of the training set each training iteration, with sensitivity and length scales initialized to 1. For CENs, we randomly select two simple fully connected architectures with 16-32-8 (CEN-1) and 96-32 (CEN-2) hidden units respectively, and learn the conditional mean embedding without dropout under ReLU activation. Biases and standard deviations of zero mean truncated normal distributed weights are initialized to 0.1, and are to be learned with full gradient updates. For all experiments,

λ is initialized to 1 and is learned jointly with the kernel. Optimization is performed with the Adam optimizer [Kingma and Ba, 2016] in TensorFlow [Abadi et al., 2016] with a rate of $\eta = 0.1$ and $\epsilon = 10^{-15}$ under the learning objective $q(\theta, \lambda)$ (3.18). Learning is run for 1000 epochs to allow direct comparison. All attributes are scaled to the unit range. Each model is trained on 9 out of 10 folds and tested on the remaining fold, which are shuffled over all 10 combinations to obtain the test accuracy average and deviation. We compare our results to MCEs whose hyperparameters are tuned by ERM (without the RCB term in (3.18)), CV, and the MLH, as well as to other approaches using neural networks [Freire et al., 2009, Kaya et al., 2016, a; c], probabilistic binary trees [Horton and Nakai, 1996, b], decision trees [Zhou et al., 2004, d], and regularized discriminant analysis [Aeberhard et al., 1992, e].

Table 3.2 shows that our learning algorithm outperforms other hyperparameter tuning algorithms, and performs similarly to competing methods. Our method achieves this without any case specific tuning or heuristics, but by simply placing a conditional mean embedding on training data and applying a complexity bound based learning algorithm. The stochastic gradient approach for Gaussian kernels performs similarly to the full gradient approach, supporting the claim of theorem 3.5 for $n = n_b$. For CENs, we did not attempt to choose an optimal architecture for each dataset. The learning algorithm is tasked to train the same simple network for different datasets using 1000 epochs to achieve comparable performance.

3.10.3 Learning pixel relevance

We apply algorithm 1 to learn length scales of anisotropic Gaussian, or ARD, kernels on pixels of the MNIST digits dataset [LeCun et al., 1998]. In the top left plot of figure 3.2, we train on datasets of varying sizes, from 50 to 5000 images, and show the accuracy on the standard test set of 10000 images. All hyperparameters are initialized to 1 before learning. We train both SVCs and GPCs under the OVA scheme, and use a Laplace approximation for the GPC posterior. In all cases MCEs outperform SVCs as it cannot learn hyperparameters without expensive cross validation. MCEs also outperform GPCs as more data becomes available. Under the OVA scheme, the GPC approach learns a set of kernel hyperparameters for each class, while our approach learns a consistent set of hyperparameters for all classes. Consequently, for 5000 data points, the computational time required for hyperparameter learning of GPCs is on the order of days even for isotropic Gaussian kernels, while algorithm 1 is on the order of hours for anistropic Gaussian kernels even without batch updates. We also compare hyperparameter learning with and without the RCB. For small n below 750 samples, the latter outperforms the former (e.g. 86.69% and 86.96% for $n = 500$), while for large n the former

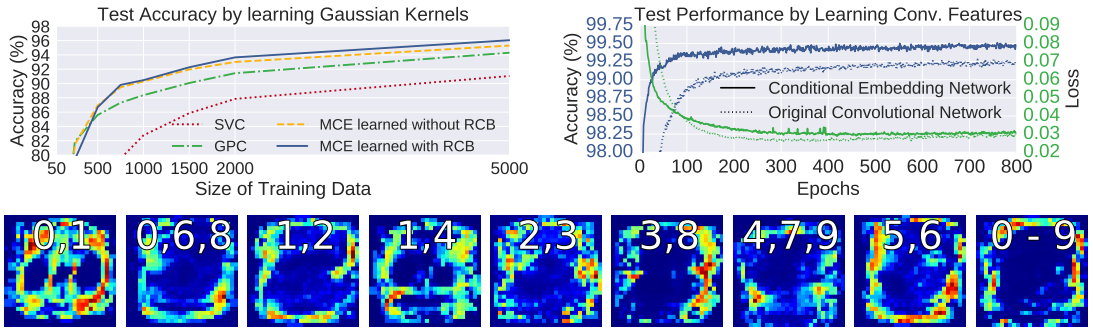

FIGURE 3.2: Top: Test accuracy by learning Gaussian kernels (left) and deep convolutional features (right); Bottom: Learned pixel length scales with ARD

outperforms the latter (e.g. 96.05% and 95.3% for $n = 5000$). This verifies that complexity based regularization becomes especially important as data size grows, when overfitting starts to decrease generalization performance. The images at the bottom of figure 3.2 show the pixel length scales learned through batch stochastic gradient updates ($n_b = 1200$) over all available training images the groups of digits shown, demonstrating the most discriminative regions.

3.10.4 Learning convolutional layers

We now apply algorithm 1 to train a CEN with convolutional layers on MNIST. We employ an example architecture from the TensorFlow tutorial on deep MNIST classification [Abadi et al., 2016]. This ReLU activated CNN uses two convolutional layers, each with max pooling, followed by a fully connected layer with a drop out probability of 0.5. The original CNN then employs a final softmax regressor on the last hidden layer for classification. The CEN instead employs a linear kernel on the last hidden layer to construct the conditional mean embedding. We then train both networks from the same initialization using batch updates of $n_b = 6000$ images for 800 epochs, with learning rate $\eta = 0.01$. All biases and weight standard deviations are initialized to 0.1. The network weights and biases of the CEN are learned jointly with the regularization hyperparameter, initialized to $\lambda = 10$, under our learning objective (3.18), while the original CNN is trained under its usual cross entropy loss. The fully connected layer is trained with a drop out probability of 0.5 for both cases to allow direct comparison. The top right plot in figure 3.2 shows that CENs learn at a much faster rate, maintaining a higher test accuracy at all epochs. After 800 epochs, CEN reaches a test accuracy of 99.48%, compared to 99.26% from the original CNN. This demonstrates that our learning algorithm can perform end-to-end learning with convolutional layers from scratch, by simply replacing the softmax layer with a MCE. The resulting CEN can outperform the original CNN in both convergence rate and accuracy.

3.11 Summary and Future Work

In this chapter we propose a hyperparameter learning framework for CMEs when the target is discrete. This naturally results in a nonparametric probabilistic multiclass classifier whose convergence properties can be guaranteed, which we call the MCE. The MCE is inherently highly general and flexible in architecture. To prevent overfitting, we develop a scalable hyperparameter learning framework for CMEs based on learning theoretic bounds, by justifying the use of stochastic batch gradient updates. These bounds reveal the RCB as a novel data-dependent quantity that reflect the model complexity. This measure automatically appears in the learning objective and acts as a regularization term in addition to the empirical loss for hyperparameter learning. We show that the MCE can be constructed from general model architectures such as neural networks and trained under our learning algorithm, and demonstrate that it outperforms the original model architecture.

In a similar light to the case with regularized least squares, it remains to be established what type of prior, if any, could correspond to the RCB. This would lead to a fully Bayesian interpretation of our framework. We also envision that such a quantity could potentially be generalized to CMEs with arbitrary targets, which would enable hyperparameter learning for general CMEs in a way that is optimized for any general prediction task. Finally, our framework and algorithm is simple, and developments in establishing deeper connections and relationships with other kernel based models can be fruitful.

Chapter 4

Bayesian Learning of Conditional Kernel Mean Embeddings for Automatic Likelihood-Free Inference

In likelihood-free settings where likelihood evaluations are intractable, approximate Bayesian computation (ABC) addresses the formidable inference task to discover plausible parameters of simulation programs that explain the observations. However, they demand large quantities of simulation calls. Critically, hyperparameters that determine measures of simulation discrepancy crucially balance inference accuracy and sample efficiency, yet are difficult to tune. In this paper, we present *kernel embedding likelihood-free inference* (KELFI), a holistic framework that automatically learns model hyperparameters to improve inference accuracy given limited simulation budget. By leveraging likelihood smoothness with conditional mean embeddings, we nonparametrically approximate likelihoods and posteriors as surrogate densities and sample from closed-form posterior mean embeddings, whose hyperparameters are learned under its approximate marginal likelihood. Our modular framework demonstrates improved accuracy and efficiency on challenging inference problems in ecology.

4.1 Introduction

Scientific understanding of complex phenomena are deeply reliant on the study of probabilistic generative models and their match with real world data. Often, latent and convoluted interactions result in intractable likelihood evaluations, making the setting *likelihood-free*. Instead, generative models are expressed as a stochastic forward model simulator. Inference on latent variables in this setting is particularly challenging.

Approximate Bayesian computation (ABC) methods are the state-of-the-art in simulation-based Bayesian inference with intractable likelihoods [Marin et al., 2012]. They infer posterior distributions of simulator parameters that aim to explain observed data. The posterior is of interest in its own right for understanding the complex phenomena, and also useful in forming predictions of future observations. They are popular due to their simplicity and applicability, and have been used extensively in the biological sciences [Beaumont, 2010, Toni et al., 2009]. Nevertheless, complex models are often prohibitively expensive to simulate. Evolutionary processes of ecological systems, vibrational modes of a mechanical structure, and fluid flow across surfaces are all examples that result in formidable inference problems with demanding forward simulations. It is thus imperative for inference algorithms to perform under the constraint of limited simulation calls, posing an exceptionally challenging task.

Often, ABC methods rely on discrepancy measures between simulations and observations that are parametrized by hyperparameters such as ϵ. The resulting posterior approximation is highly sensitive to the choice of hyperparameters, yet appropriate hyperparameter tuning strategies remain to be established.

To address these issues, we present KELFI, a holistic framework consisting of (1) the kernel means likelihood (KML) as a consistent surrogate likelihood *model* that modularizes approximate likelihood evaluations from simulation calls, (2) the marginal kernel means likelihood (MKML) as a Bayesian *learning* objective for hyperparameters that improves inference accuracy, (3) the kernel means posterior (KMP) as a posterior surrogate density for approximate Bayesian *inference* on simulator parameters, and (4) the kernel means posterior embedding (KMPE) leading to a super-*sampling* algorithm for fast-converging posterior samples of simulator parameters. We further present the spatio-temporal kernel means likelihood (ST-KML) and independent and identically distributed kernel means likelihood (iid-KML) to alleviate the requirement for designing summary statistics for spatio-temporal data and *iid* data respectively. Finally, we address computational or analytical challenges for complex or intractable priors.

KELFI is based on approximating likelihoods with simulation samples using CMEs. CMEs encode conditional expectations empirically by leveraging smoothness within a RKHS with only a small number of examples. This modularizes inference away from simulation calls. Consequently, scientists can proceed with posterior analysis after any number of simulations. Furthermore, KELFI infers both approximate posterior densities and samples. Critically, our learning algorithm tunes hyperparameters directly for the inference problem, including adapting ϵ to the number of simulations used. This removes the need for practitioners to arduously select hyperparameters. Finally, it can be extended to automatically learn the relevance and usefulness of each summary statistic.

4.2 Likelihood-Free Inference

We begin by setting up notations for the likelihood-free setting. Let $\boldsymbol{\theta} \in \vartheta$ denote a realization of the latent variable or parameter $\boldsymbol{\Theta}$, where we use upper cases to denote random variables. Let $\mathbf{x} \in \mathcal{X}$ and $\mathbf{y} \in \mathcal{Y}$ where $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{D}$ denote realizations of simulation output $\mathbf{X}$ and observations $\mathbf{Y}$ respectively.

In the likelihood-free setting, we begin with a stochastic forward model simulator which synthesizes simulations $\mathbf{x}$ given a parameter setting $\boldsymbol{\theta}$. The simulator is often a computer program which takes a parameter setting as input, makes internal calls to a pseudo-random number generator, and outputs a a simulation result $\mathbf{x}$ in ideally the same structure as the observations $\mathbf{y}$. In this sense, while the simulation result conditioned on the parameter setting and the internal pseudo-random numbers is deterministically generated, the simulation result conditioned on the parameter setting only is considered stochastically generated. We represent the stochastic simulator as $p_{\mathbf{X}|\boldsymbol{\Theta}}$ from which we can only sample $\mathbf{x}_j \sim p_{\mathbf{X}|\boldsymbol{\Theta}}(\cdot|\boldsymbol{\theta}_j)$, but not query its density $p_{\mathbf{X}|\boldsymbol{\Theta}}(\mathbf{x}|\boldsymbol{\theta})$ at particular simulation result $\mathbf{x} \in \mathcal{X}$ and parameter $\boldsymbol{\theta} \in \vartheta$, making likelihood evaluations intractable, thus *likelihood-free*.

To begin inference we posit a prior density $p_{\boldsymbol{\Theta}}$ that encodes general prior knowledge about plausible parameter settings to guide the inference, which are usually informed by the problem setting itself. We assume that evaluations of the prior density $p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$ at any particular parameter $\boldsymbol{\theta} \in \vartheta$ is tractable. However, we will also address the setting when only samples from the prior is available.

When writing density evaluations and the random variable the density is describing is clear from the context, we will drop the subscript. For example, $p(\boldsymbol{\theta})$ is shorthand for $p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$ and $p(\mathbf{x}|\boldsymbol{\theta})$ is shorthand for $p_{\mathbf{X}|\boldsymbol{\Theta}}(\mathbf{x}|\boldsymbol{\theta})$.

The goal of the inference problem is to find plausible parameter settings that could have generated the observations $\mathbf{y}$, or the posterior of parameter settings given observations $\mathbf{y}$. The underlying assumption is that $\mathbf{y}$ was generated from $p_{\mathbf{X}|\boldsymbol{\Theta}}$ for some parameter setting. This however presents a trade-off. A rich and complex simulator may be able to replicate the process from which observations $\mathbf{y}$ was generated for a particular parameter setting. However, since each parameter setting would produce a different rich and complex process, inference can be extremely challenging. On the other hand, simple simulators may not be able to replicate the true generation process for any parameter setting. Therefore, LFI seeks to relax the inference problem it asks, and seeks to infer a posterior distribution on the parameters $\boldsymbol{\theta}$ that could generate simulations $\mathbf{x}$ that are only *similar* to our observations $\mathbf{y}$ by some comparison measure, instead of exactly the same. We refer to this as the soft posterior.

To do this, we introduce a discrepancy measure called the ϵ-kernel or ABC kernel,

$$\kappa_\epsilon(\mathbf{y}, \mathbf{x}) := p_\epsilon(\mathbf{y}|\mathbf{x}). \tag{4.1}$$

A convenient and common choice is the Gaussian density $p_\epsilon(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mathbf{x}, \epsilon^2 I)$ [Moreno et al., 2016, Price et al., 2017]. It is worthwhile to note that κ_ϵ is a conditional probability density function and does not need to be a positive definite kernel. For instance, it does not need to be symmetric, although it happens to be symmetric when we use a Gaussian density, since $\mathcal{N}(\mathbf{y}; \mathbf{x}, \epsilon^2 I) = \mathcal{N}(\mathbf{x}; \mathbf{y}, \epsilon^2 I)$. However, the latter expression is not semantically relevant in this context.

Furthermore, the observation $\mathbf{y}$ is often fixed within a given LFI problem, while the simulation $\mathbf{x}$ varies throughout the LFI procedure. Consequently, the ϵ-kernel is often tasked with comparing the same $\mathbf{y}$ against different $\mathbf{x}$, so that we are more concerned with properties of $\kappa_\epsilon(\mathbf{y}, \cdot)$, the ϵ-kernel as a function of $\mathbf{x}$ for a fixed $\mathbf{y}$.

The ϵ-kernel $\kappa_\epsilon(\mathbf{y}, \mathbf{x})$ captures the likelihood distribution of the observation $\mathbf{y}$ given a simulation $\mathbf{x}$. It posits that discrepancies of observations and simulations are only due to noise which the ϵ-kernel quantifies, since we assume the simulator is reasonably capable of reproducing the true data generation process. The LFI setup then assumes that further knowledge of the parameter setting on top of the simulation results are not informative, such that $p_\epsilon(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) = p_\epsilon(\mathbf{y}|\mathbf{x})$.

Based on this formulation, the true full likelihood of our model can be written as

$$p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) = \int_{\mathcal{X}} p_\epsilon(\mathbf{y}|\mathbf{x}) p(\mathbf{x}|\boldsymbol{\theta}) d\mathbf{x} = \mathbb{E}[\kappa_\epsilon(\mathbf{y}, \mathbf{X})|\boldsymbol{\Theta} = \boldsymbol{\theta}]. \tag{4.2}$$

The corresponding posterior of interest is $p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) = p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) / p_\epsilon(\mathbf{y})$ where $p_\epsilon(\mathbf{y}) = \int_\vartheta p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}$. Due to the presence of a non-zero ϵ, even a perfect approximation to the soft posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ will not be the exact posterior $p_{\epsilon=0}(\boldsymbol{\theta}|\mathbf{y})$ unless ϵ is annealed to zero. This is the necessary trade-off we make with limited simulations, where a non-zero ϵ is essential for tractable inference because no simulations will match the observations exactly in practice. If $\mathbf{y}$ is only available as a summary statistic, then this soft posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ that we are targeting is only an approximation to the posterior given the full data even with $\epsilon = 0$.

So far, $\mathbf{x}$ and $\mathbf{y}$ denote either the full dataset or their summary statistics. This is because summary operations can be appended to the simulator program to output statistics directly. In either case, we let the target posterior be $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$, so the inference problem remains structurally identical. For simplicity however, from here on $\mathbf{x}$ and $\mathbf{y}$ will denote summary statistics unless stated otherwise.

4.3 Kernel Embedding Likelihood-Free Inference

We present KELFI in three stages. In the model stage, we build a surrogate likelihood model by leveraging smoothness properties of CMEs. In the learning stage, we derive a differentiable marginal surrogate likelihood to drive hyperparameters learning. In the inference stage, we propose an algorithm to sample from the resulting mean embedding of the surrogate posterior.

When the prior is an anisotropic Gaussian $p(\boldsymbol{\theta}) = \prod_{d=1}^{D} \mathcal{N}(\theta_d; \mu_d, \sigma_d^2)$, closed form solutions for KELFI exists. We will focus on this setting without loss of generality. This is because, for many common continuous priors, the LFI problem can be transformed into an equivalent problem that involves a Gaussian prior. For instance, this can be achieved if an invertible transformation T exists such that $T(\mathbf{Z}) \sim p(\boldsymbol{\theta})$ for $\mathbf{Z} \sim p(\mathbf{z})$ where $p(\mathbf{z})$ is a Gaussian. We discuss this in more detail in section 4.7. When this is not possible or preferred, we present empirical forms that can be approximate KELFI solutions arbitrarily well.

4.3.1 Conditional Mean Embeddings

We begin with an overview of CMEs in the context of KELFI. KMEs are an arsenal of techniques used to represent distributions in a RKHS [Muandet et al., 2017]. The key object is the mean embedding of a distribution $X \sim \mathbb{P}$ under a positive definite kernel k via $\mu_X := \int_{\mathcal{X}} k(x, \cdot) d\mathbb{P}(x) = \int_{\mathcal{X}} k(x, \cdot) p(x) dx \in \mathcal{H}_k$, where the last equality assumes a density p for $\mathbb{P}$ exists and $\mathcal{H}_k$ denotes the RKHS of k. They encode distributions in the sense that function expectations can be written as $\mathbb{E}[f(X)] = \langle \mu_X, f \rangle_{\mathcal{H}_k}$ if $f \in \mathcal{H}_k$. When μ_X can only be estimated empirically in some form denoted as $\hat{\mu}_X$, the expectation can be approximated by $\mathbb{E}[f(X)] \approx \langle \hat{\mu}_X, f \rangle_{\mathcal{H}_k}$.

CMEs are KMEs that encode conditional distributions (see 2.4). We specifically focus on their empirical estimates as we assume we only have the resource to obtain m sets of simulation data due to budget constraints. This results in joint samples $\{\boldsymbol{\theta}_j, \mathbf{x}_j\}_{j=1}^{m}$ from $p(\mathbf{x}|\boldsymbol{\theta})\pi(\boldsymbol{\theta})$ by sampling from a proposal prior π for $\boldsymbol{\theta}_j \sim \pi(\boldsymbol{\theta})$ and simulating $\mathbf{x}_j \sim p(\mathbf{x}|\boldsymbol{\theta}_j)$ at each $\boldsymbol{\theta}_j$. Note these samples are not necessarily from the original joint distribution $p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ if $\pi \neq p_{\boldsymbol{\Theta}}$.

We define positive definite and characteristic kernels [Sriperumbudur et al., 2010a] $k : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ and $\ell : \vartheta \times \vartheta \to \mathbb{R}$. When relevant, we denote the hyperparameters of k and ℓ with α and β, and refer to them as $k_\alpha = k(\cdot, \cdot; \alpha)$ and $\ell_\beta = \ell(\cdot, \cdot; \beta)$. A useful example of such a kernel is an anisotropic Gaussian kernel $\ell(\boldsymbol{\theta}, \boldsymbol{\theta}'; \boldsymbol{\beta}) = \exp\big(-\frac{1}{2}\sum_{d=1}^{D}(\theta_d - \theta'_d)^2/\beta_d^2\big)$ whose hyperparameters are length scales $\boldsymbol{\beta} = \{\beta_d\}_{d=1}^{D}$ for each dimension $d \in [D] := \{1, \ldots, D\}$, and similarly for k.

For any function $f \in \mathcal{H}_k$, we construct an approximation to $\mathbb{E}[f(\mathbf{X})|\boldsymbol{\Theta} = \boldsymbol{\theta}]$ by the inner product $\langle f, \hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} \rangle_{\mathcal{H}_k}$ with an empirical CME $\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}$. Importantly, $\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}$ is estimated from the *joint* samples $\{\boldsymbol{\theta}_j, \mathbf{x}_j\}_{j=1}^m$, even though it is encoding the corresponding conditional distribution $p(\mathbf{x}|\boldsymbol{\theta})$. This approximation admits the following form [Song et al., 2009],

$$\mathbb{E}[f(\mathbf{X})|\boldsymbol{\Theta} = \boldsymbol{\theta}] \approx \mathbf{f}^T (L + m\lambda I)^{-1} \boldsymbol{\ell}(\boldsymbol{\theta}), \tag{4.3}$$

where $\mathbf{f} := \{f(\mathbf{x}_j)\}_{j=1}^m$, $L := \{\ell(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)\}_{i,j=1}^m$, $\boldsymbol{\ell}(\boldsymbol{\theta}) := \{\ell(\boldsymbol{\theta}_j, \boldsymbol{\theta})\}_{j=1}^m$, and $\lambda \geq 0$ is a regularization parameter. This approximation is known to converge at $O_p(m^{-\frac{1}{4}})$ if λ is chosen to decay at $O_p(m^{-\frac{1}{2}})$ or better under appropriate assumptions on $p(\mathbf{x}|\boldsymbol{\theta})$ [Song et al., 2013].

4.3.2 Model: Kernel Means Likelihood

We begin by presenting our surrogate likelihood model. Since the likelihood (4.2) is an expectation under $p(\mathbf{x}|\boldsymbol{\theta})$, we propose to approximate it via an inner product with the CME of $p(\mathbf{x}|\boldsymbol{\theta})$. Specifically, if we choose k such that $\kappa_\epsilon(\mathbf{y}, \cdot) \in \mathcal{H}_k$, then $p_\epsilon(\mathbf{y}|\boldsymbol{\theta})$ can be approximated by $q(\mathbf{y}|\boldsymbol{\theta}) := \langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} \rangle_{\mathcal{H}_k}$. We refer to $q(\mathbf{y}|\boldsymbol{\theta})$ as the kernel means likelihood (KML). While the KML provides an asymptotically correct likelihood surrogate, for finitely many simulations it is not necessarily positive nor normalized. By using $f = \kappa_\epsilon(\mathbf{y}, \cdot)$ in (4.3) where $\boldsymbol{\kappa}_\epsilon(\mathbf{y}) := \{\kappa_\epsilon(\mathbf{y}, \mathbf{x}_j)\}_{j=1}^m$ and $\mathbf{v}(\mathbf{y}) := (L + m\lambda I)^{-1} \boldsymbol{\kappa}_\epsilon(\mathbf{y})$, the KML becomes

$$q(\mathbf{y}|\boldsymbol{\theta}) = \sum_{j=1}^m v_j(\mathbf{y}) \ell(\boldsymbol{\theta}_j, \boldsymbol{\theta}) = \boldsymbol{\kappa}_\epsilon(\mathbf{y})^T (L + m\lambda I)^{-1} \boldsymbol{\ell}(\boldsymbol{\theta}). \tag{4.4}$$

The KML converges at the same rate as the CME. See theorem 4.1 for proof. It is worthwhile to note that the assumption $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$ is common for CMEs, and is not as restrictive as it may first appear, as it can be relaxed through introducing the regularization hyperparameter λ [Song et al., 2013].

Theorem 4.1. *Assume $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$. The kernel means likelihood (KML) $q(\mathbf{y}|\boldsymbol{\theta})$ converges to the likelihood $p_\epsilon(\mathbf{y}|\boldsymbol{\theta})$ uniformly at rate $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ as a function of $\boldsymbol{\theta} \in \vartheta$ and $\mathbf{y} \in \mathcal{Y}$.*

To satisfy $\kappa_\epsilon(\mathbf{y}, \cdot) \in \mathcal{H}_k$, we choose the standard Gaussian ϵ-kernel $\kappa_\epsilon(\mathbf{y}, \mathbf{x}) = \mathcal{N}(\mathbf{y}; \mathbf{x}, \epsilon^2 I)$ and let $k_\alpha = k_\epsilon$ be a Gaussian kernel with length scale $\alpha = \epsilon$. Since $\kappa_\epsilon(\mathbf{y}, \mathbf{x})$ and $k_\epsilon(\mathbf{y}, \mathbf{x})$ are scalar multiples of each other, we have that $\kappa_\epsilon(\mathbf{y}, \cdot) \in \mathcal{H}_k$. In fact, any positive definite kernel κ_ϵ can be used, since we can simply choose k_α to be its scalar multiple to form the RKHS.

Importantly, instead of modeling the posterior mean embedding directly in a fashion similar to kernel ABC (K-ABC), kernel recursive ABC (KR-ABC), and KBR,

KELFI begins by using CMEs to approximate the full likelihood (4.2) first as a surrogate likelihood, the KML. While the KML provides an asymptotically correct surrogate for the likelihood, for finitely many simulations the KML is not necessarily positive nor normalized. This makes the KML incompatible with to standard approximate inference methods such as Markov chain Monte Carlo (MCMC) and variational inference (VI), which require likelihoods or likelihood approximations to be positive normalized densities. If one were to attempt to make the KML compatible with MCMC or VI methods, we would require further amendments to the KML, ranging from simple clipping $[q(\mathbf{y}|\boldsymbol{\theta})]^+$ or even a positivity constraint in the empirical least-squares problem for the CME weights. The latter stems from the fact that CMEs can be seen as the solution to a vector valued regression problem in the RKHS [Grünewälder et al., 2012]. However, these amendments would however introduce further bias to the already biased likelihood approximation. Even though these biases vanishes asymptotically as the KML approaches a valid density due to theorem 4.1, the asymptotic behavior is nevertheless rarely reached under limited simulations, which is the scenario of interest. Consequently, we propose to use the KML as is, without further unnecessary modifications that would introduce further bias, and design learning and inference algorithms using the KML directly.

4.3.3 Learning: Marginal Kernel Means Likelihood

We now propose a hyperparameter learning algorithm for our surrogate likelihood model. The main advantage of using an approximate surrogate likelihood surrogate model is that it readily provides a marginal surrogate likelihood quantity that lends itself to a hyperparameter learning algorithm. We define the MKML as follows,

$$q(\mathbf{y}) := \int_{\vartheta} q(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta} = \sum_{j=1}^{m} v_j(\mathbf{y})\mu_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_j) = \boldsymbol{\kappa}_{\epsilon}(\mathbf{y})^T(L + m\lambda I)^{-1}\boldsymbol{\mu}_{\boldsymbol{\Theta}}, \tag{4.5}$$

where $\mu_{\boldsymbol{\Theta}} := \int_{\vartheta} \ell(\boldsymbol{\theta}, \cdot)p(\boldsymbol{\theta})d\boldsymbol{\theta}$ is the mean embedding of $p_{\boldsymbol{\Theta}}$, also called the prior mean embedding, and $\boldsymbol{\mu}_{\boldsymbol{\Theta}} := \{\mu_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_j)\}_{j=1}^m$ are the evaluations of the prior mean embedding. Note that we notate the subscripts of the random variable $\boldsymbol{\Theta}$ since the shorthand $p(\boldsymbol{\theta})$ of $p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$ loses meaning if we want to refer to the density without evaluating at a particular parameter $\boldsymbol{\theta}$, which would resulting in writing just p.

If we choose ℓ to be an anisotropic Gaussian kernel with length scales $\boldsymbol{\beta} = \{\beta_d\}_{d=1}^D$, then $\mu_{\boldsymbol{\Theta}}$ is closed-form for anisotropic Gaussian priors $p(\boldsymbol{\theta}) = \prod_{d=1}^D \mathcal{N}(\theta_d; \mu_d, \sigma_d^2)$.

Let $\nu_d^2 := \beta_d^2 + \sigma_d^2$, then we have

$$\mu_{\boldsymbol{\Theta}}(\boldsymbol{\theta}) = \ell_{\boldsymbol{\nu}}(\boldsymbol{\theta}, \boldsymbol{\mu}) \prod_{d=1}^{D} \frac{\beta_d}{\nu_d}. \tag{4.6}$$

Similar to the KML, the MKML converges at the same rate as the CME.

Theorem 4.2. *Assume* $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$. *The marginal kernel means likelihood (MKML)* $q(\mathbf{y})$ *converges to marginal likelihood* $p_\epsilon(\mathbf{y})$ *uniformly at rate* $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ *as a function of* $\mathbf{y} \in \mathcal{Y}$.

Consequently, the MKML $q(\mathbf{y}) = q(\mathbf{y}; \boldsymbol{\epsilon}, \boldsymbol{\beta}, \lambda)$ approximates the true marginal likelihood $p_\epsilon(\mathbf{y})$ of the inference problem defined by our likelihood-prior pair. It is a function of the hyperparameters $(\boldsymbol{\epsilon}, \boldsymbol{\beta}, \lambda)$ of the ϵ-kernel and KML model. As $p_\epsilon(\mathbf{y})$ is unavailable, we instead maximize the MKML for hyperparameter learning. Furthermore, prior hyperparameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ can also be included and learned jointly. Since the map $(\boldsymbol{\epsilon}, \boldsymbol{\beta}, \lambda) \mapsto q(\mathbf{y}; \boldsymbol{\epsilon}, \boldsymbol{\beta}, \lambda)$ is differentiable, optimization can be done in an auto-differentiation environment. The learning objective to be optimized is computed in line 3 of algorithm 3. Each automatic gradient update has complexity dominated by $O(m^3)$ due to the Cholesky decomposition in line 2. However, since we are addressing scenarios where simulations are limited so that m is small, this optimization is relatively fast.

Importantly, if we use an anisotropic Gaussian density for the ϵ-kernel $\kappa_{\boldsymbol{\epsilon}}$ where $\boldsymbol{\epsilon} = \{\epsilon_i\}_{i=1}^n$ are the length scales corresponding to each summary statistic $\mathbf{y} = \{y_i\}_{i=1}^n$, we can perform ARD to learn the relevance and usefulness of each summary statistic, where a small length scale indicate high relevance for that statistic. This is because $\boldsymbol{\epsilon}$ are also the length scales of the kernel k which defines the RKHS $\mathcal{H}_k$. Since the anistropic Gaussian kernel is learned, we also refer to it as an ARD kernel. We can also learn the length scales $\boldsymbol{\beta} = \{\beta_d\}_{d=1}^D$ for the kernel $\ell_{\boldsymbol{\beta}}$ on $\boldsymbol{\theta}$, although we found that it is more useful to let $\boldsymbol{\beta} = \beta_0 \boldsymbol{\sigma}$ where $\boldsymbol{\sigma} = \{\sigma_d\}_{d=1}^D$ are the standard deviations of the Gaussian prior. By doing this, we make better use of the scale differences within $\boldsymbol{\theta}$ from the prior, and let β_0 learn the overall scale that is most useful for the KML.

For general non-Gaussian kernels and priors, $\mu_{\boldsymbol{\Theta}}$ in (4.5) can be approximated using T independent prior samples $\tilde{\boldsymbol{\theta}}_t \sim p(\boldsymbol{\theta})$, $t \in [T]$, as

$$\tilde{\mu}_{\boldsymbol{\Theta}} := \frac{1}{T} \sum_{t=1}^{T} \ell(\tilde{\boldsymbol{\theta}}_t, \cdot) = \frac{1}{T} \tilde{L} \mathbf{1}_T. \tag{4.7}$$

In this case, the MKML can be approximated as

$$\tilde{q}(\mathbf{y}) := \frac{1}{T} \sum_{t=1}^{T} q(\mathbf{y}|\boldsymbol{\theta}_t) = \sum_{j=1}^{m} v_j(\mathbf{y}) \tilde{\mu}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_j) = \boldsymbol{\kappa}_\epsilon(\mathbf{y})^T (L + m\lambda I)^{-1} \tilde{L} \mathbf{1}_T. \tag{4.8}$$

Algorithm 3 KELFI: Kernel Embedding Likelihood-Free Inference

Input: Data $\mathbf{y}$, simulations $\{\boldsymbol{\theta}_j, \mathbf{x}_j\}_{j=1}^m \sim p(\mathbf{x}|\boldsymbol{\theta})\pi(\boldsymbol{\theta})$, query parameters $\{\boldsymbol{\theta}_r^\star\}_{r=1}^R$, KML hyperparameters $(\boldsymbol{\epsilon}, \boldsymbol{\beta}, \lambda)$, prior hyperparameters $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ or samples $\{\tilde{\boldsymbol{\theta}}_t\}_{t=1}^T$, number of samples S, kernel ℓ and ϵ-kernel κ
Compute $L \leftarrow \{\ell_{\boldsymbol{\beta}}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)\}_{i,j=1}^m$ and $\boldsymbol{\kappa}_{\boldsymbol{\epsilon}}(\mathbf{y}) \leftarrow \{\kappa_{\boldsymbol{\epsilon}}(\mathbf{y}, \mathbf{x}_j)\}_{j=1}^m$
Compute KML weights $\mathbf{v} \leftarrow (L + m\lambda I)^{-1}\boldsymbol{\kappa}_{\boldsymbol{\epsilon}}(\mathbf{y})$
Compute $\tilde{L} \leftarrow \{\ell_{\boldsymbol{\beta}}(\boldsymbol{\theta}_j, \tilde{\boldsymbol{\theta}}_t)\}_{j,t=1}^{m,T}$ and $\tilde{L}^\star \leftarrow \{\ell_{\boldsymbol{\beta}}(\tilde{\boldsymbol{\theta}}_t, \boldsymbol{\theta}_r^\star)\}_{t,r=1}^{T,R}$
Compute $\boldsymbol{\mu}_{\boldsymbol{\Theta}} \leftarrow \{\mu_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_j)\}_{j=1}^m$ using (4.6) or $\boldsymbol{\mu}_{\boldsymbol{\Theta}} \leftarrow \frac{1}{T}\tilde{L}\mathbf{1}_T$ using (4.7)
Compute MKML $q(\mathbf{y}) \leftarrow \mathbf{v}^T\boldsymbol{\mu}_{\boldsymbol{\Theta}}$
Compute $H \leftarrow \{h(\boldsymbol{\theta}_j, \boldsymbol{\theta}_r^\star)\}_{j=1,r=1}^{m,R}$ using (4.10) or $H \leftarrow \frac{1}{T}\tilde{L}\tilde{L}^\star$ using (4.11)
Compute KMPE $\boldsymbol{\mu} \leftarrow H^T\mathbf{v}/q(\mathbf{y}) \in \mathbb{R}^R$ and initialize $\mathbf{a} \leftarrow \mathbf{0} \in \mathbb{R}^R$
for $s \in \{1, \ldots, S\}$ **do**
  Obtain super-sample $\hat{\boldsymbol{\theta}}_s \leftarrow \boldsymbol{\theta}_{r^\star}^\star$ where $r^\star \leftarrow \arg\max_{r\in\{1,\ldots,R\}} \mu_r - (a_r/s)$
  Update kernel sum $\mathbf{a} \leftarrow \mathbf{a} + \{\ell_{\boldsymbol{\beta}}(\boldsymbol{\theta}_r^\star, \hat{\boldsymbol{\theta}}_s)\}_{r=1}^R$
end for
Output: Posterior super-samples $\{\hat{\boldsymbol{\theta}}_s\}_{s=1}^S$

By formulating a learning objective directly for the inference problem, KELFI provides a way to automatically tune ϵ and its own model hyperparameters.

While our hyperparameter learning algorithm stemmed from the LFI setting, it can be applied to learn hyperparmeters for general CMEs built from joint samples by using the marginal distribution of the conditioned variable as the prior.

4.3.4 Inference: Kernel Means Posterior and Sampling

We finally present an approach for posterior inference by super-sampling directly from the equivalent posterior mean embedding defined by the KML model and the prior. Our approach begins by defining a surrogate density to approximate the posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ in analogy to the Bayes' rule, $q(\boldsymbol{\theta}|\mathbf{y}) := q(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})/q(\mathbf{y})$. We refer to $q(\boldsymbol{\theta}|\mathbf{y})$ as the kernel means posterior (KMP). Importantly, $q(\boldsymbol{\theta}|\mathbf{y})$ is unaffected even if κ_ϵ is unnormalized, so that ϵ-kernels on distributions can be readily used. While both $q(\boldsymbol{\theta}|\mathbf{y})$ and $q(\mathbf{y})$ are surrogate densities, by construction the KMP$q(\boldsymbol{\theta}|\mathbf{y})$ approaches the true posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ whenever the KML $q(\mathbf{y}|\boldsymbol{\theta})$ approaches the true likelihood $p_\epsilon(\mathbf{y}|\boldsymbol{\theta})$. Consequently, the KMP has the following convergence properties.

Theorem 4.3. *Assume $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$ and that there exists $\delta > 0$ such that $q(\mathbf{y}) \geq \delta$ for all $m \geq M$ where $M \in \mathbb{N}_+$. The kernel means posterior (KMP) $q(\boldsymbol{\theta}|\mathbf{y})$ converges pointwise to the posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ at rate $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ as a function of $\boldsymbol{\theta} \in \vartheta$ and $\mathbf{y} \in \mathcal{Y}$. If $\sup_{\boldsymbol{\theta}\in\vartheta} p(\boldsymbol{\theta}) < \infty$ and $\sup_{\boldsymbol{\theta}\in\vartheta} p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) < \infty$, then the convergence is uniform in $\boldsymbol{\theta} \in \vartheta$. If $\sup_{\mathbf{y}\in\mathcal{Y}} p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) < \infty$, then the convergence is uniform in $\mathbf{y} \in \mathcal{Y}$.*

Importantly, the requirement for a $\delta > 0$ such that $q(\mathbf{y}) \geq \delta$ for all $m \geq M$ where $M \in \mathbb{N}_+$ provides an intuition for why high MKML values are favorable for learning a good approximate posterior. This requirement is a reflection on the capability of the simulator to recreate the observations $\mathbf{y}$ relative to the scale ϵ. Intuitively, the more capable the simulator $p(\mathbf{x}|\boldsymbol{\theta})$ is at generating simulations $\mathbf{x}$ that is close to $\mathbf{y}$ with respect to ϵ, the higher $p_\epsilon(\mathbf{y}) > 0$ will be relatively. Since theorem 4.2 guarantees that, for large $m > M$, $q(\mathbf{y})$ will be close to $p_\epsilon(\mathbf{y})$, we have that $q(\mathbf{y}) > 0$ for all large $m > M$ with increasing probability. In this situation, theorem 4.3 guarantees that the KMP will converge to the posterior of interest. However, consider the case when the simulator is ill-designed to recreate $\mathbf{y}$ such that the true marginal likelihood $p_\epsilon(\mathbf{y}) \approx 0$ is small. As $q(\mathbf{y})$ tends to $p_\epsilon(\mathbf{y}) \approx 0$ due to theorem 4.2, it may struggle to always stay strictly positive even for large $m > M$ since it is stochastically converging to approximately zero. In this case, convergence is difficult since the simulator was ill-designed. However, by learning ϵ through maximizing $q(\mathbf{y})$, we adapt the threshold ϵ to make $p_\epsilon(\mathbf{y})$ as high as possible, leading to a more stable posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ for the KMP to converge to.

Constructed from the KML, the KMP is also a surrogate density, although it is normalized. While the KMP is useful for finding maximum a posteriori (MAP) solutions and visualizing posterior uncertainties, we cannot directly sample from a surrogate density that is possibly non-positive. To address this, we leverage the fact that conditional distributions encoded by general CMEs can still be super-sampled with kernel herding [Chen et al., 2010]. Although marginal mean embeddings are strictly positive for strictly positive kernels, when they are estimated from empirical CMEs, the resulting mean embedding may not be strictly positive [Song et al., 2009]. Despite this, kernel herding can still obtain super-samples from CME estimates which effectively minimizes the MMD discrepancy between the original CME estimate and the new embedding formed from super-samples. This idea has been used to sample from conditional distributions through its empirical CME representation in kernel Monte Carlo filter (KMCF) [Kanagawa et al., 2016] and KR-ABC [Kajihara et al., 2018]. Furthermore, super-samples are more informative than random samples, in the sense that empirical expectations under super-samples can potentially converge faster at $O(S^{-1})$ for S samples instead of $O(S^{-\frac{1}{2}})$ for random samples.

Motivated by this, we now define an analogous form of mean embeddings for surrogate densities. Specifically, we define the kernel means posterior embedding (KMPE), the mean embedding of the KMP, as $\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}(\boldsymbol{\theta}^\star) := \int_\vartheta \ell(\boldsymbol{\theta}, \boldsymbol{\theta}^\star) q(\boldsymbol{\theta}|\mathbf{y}) d\boldsymbol{\theta}$. This becomes

$$\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}(\boldsymbol{\theta}^\star) = \frac{1}{q(\mathbf{y})} \sum_{j=1}^{m} \upsilon_j(\mathbf{y}) h(\boldsymbol{\theta}_j, \boldsymbol{\theta}^\star), \tag{4.9}$$

where $h(\boldsymbol{\theta}, \boldsymbol{\theta}^\star) := \int_\vartheta \ell(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}) \ell(\tilde{\boldsymbol{\theta}}, \boldsymbol{\theta}^\star) p(\tilde{\boldsymbol{\theta}}) d\tilde{\boldsymbol{\theta}}$. Importantly, as KMPE is constructed from the CME used to form KML, it converges in RKHS norm at the same rate.

Theorem 4.4. *Assume* $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$ *and that there exists* $\delta > 0$ *such that* $q(\mathbf{y}) \geq \delta$ *for all* $m \geq M$ *where* $M \in \mathbb{N}_+$. *The kernel means posterior embedding (KMPE)* $\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}$ *converges in RKHS norm to the posterior mean embedding* $\mu_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}$ *at rate* $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.

Note that we accent the KMPE with a bar to emphasize and distinguish it from the true posterior mean embedding $\mu_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}(\boldsymbol{\theta}^\star) := \int_\vartheta \ell(\boldsymbol{\theta}, \boldsymbol{\theta}^\star) p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) d\boldsymbol{\theta}$.

If we choose ℓ to be an anisotropic Gaussian kernel with length scales $\boldsymbol{\beta} = \{\beta_d\}_{d=1}^D$, h exhibits the following closed-form under anisotropic Gaussian priors,

$$h(\boldsymbol{\theta}, \boldsymbol{\theta}^\star) = \prod_{d=1}^{D} \frac{s_d}{\sigma_d} \exp\left[-\frac{1}{2s_d^2}\left(a_d - b_d^2\right)\right], \tag{4.10}$$

where $a_d := (\theta_d^2 + \theta_d^{\star 2} + \gamma_d^2 \mu_d^2)/(2+\gamma_d^2)$, $b_d := (\theta_d + \theta_d^\star + \gamma_d^2 \mu_d)/(2+\gamma_d^2)$, $\gamma_d^2 := \beta_d^2/\sigma_d^2$ and $s_d^{-2} := 2\beta_d^{-2} + \sigma_d^{-2}$. For general non-Gaussian kernels and priors, h can be approximated as

$$\tilde{h}(\boldsymbol{\theta}, \boldsymbol{\theta}^\star) = \frac{1}{T} \sum_{t=1}^{T} \ell(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}_t) \ell(\tilde{\boldsymbol{\theta}}_t, \boldsymbol{\theta}^\star) = \frac{1}{T} \tilde{L} \tilde{L}^\star. \tag{4.11}$$

In this case, the KMPE can be approximated by

$$\tilde{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}(\boldsymbol{\theta}^\star) = \frac{1}{\tilde{q}(\mathbf{y})} \sum_{j=1}^{m} v_j(\mathbf{y}) \tilde{h}(\boldsymbol{\theta}_j, \boldsymbol{\theta}^\star) = \frac{\mathbf{v}(\mathbf{y})^T \mathbf{h}(\boldsymbol{\theta}^\star)}{\mathbf{v}(\mathbf{y})^T \tilde{\mu}_{\boldsymbol{\Theta}}} = \frac{\mathbf{v}(\mathbf{y})^T L \boldsymbol{\ell}(\boldsymbol{\theta}^\star)}{\mathbf{v}(\mathbf{y})^T \tilde{L} \mathbf{1}_T}, \tag{4.12}$$

where $\mathbf{h}(\boldsymbol{\theta}^\star) := \{\tilde{h}(\boldsymbol{\theta}_j, \boldsymbol{\theta}^\star)\}_{j=1}^m$ and $\boldsymbol{\ell}(\boldsymbol{\theta}^\star) := \{\ell(\boldsymbol{\theta}_j, \boldsymbol{\theta}^\star)\}_{j=1}^m$.

The KMP $q(\cdot|\mathbf{y})$ is bounded and normalized but potentially non-positive. Consequently, it can be seen as a surrogate density corresponding to a signed measure. This suggests that the map $q(\cdot|\mathbf{y}) \mapsto \bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}$ is injective for characteristic kernels ℓ, analogous to mean embeddings [Sriperumbudur et al., 2011]. Furthermore, as the integral (4.9) is a linear operator on $\ell(\boldsymbol{\theta}^\star, \cdot)$, the surrogate posterior mean embedding $\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}} \in \mathcal{H}_\ell$ is in the RKHS of ℓ. With a surrogate embedding that is injective to our surrogate posterior and in the RKHS, we can apply kernel herding [Chen et al., 2010] on $\tilde{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}$ (4.9) using kernel ℓ to obtain S super-samples $\{\hat{\boldsymbol{\theta}}_s\}_{s=1}^S$ from the surrogate density $q(\boldsymbol{\theta}|\mathbf{y})$. That is, for each $s \in [S]$, the samples are obtained by

$$\hat{\boldsymbol{\theta}}_s = \arg\max_{\boldsymbol{\theta} \in \vartheta} \tilde{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}(\boldsymbol{\theta}) - \frac{1}{s} \sum_{s'=1}^{s-1} \ell(\hat{\boldsymbol{\theta}}_{s'}, \boldsymbol{\theta}). \tag{4.13}$$

The inference algorithm is presented in algorithm 3.

4.4 Theoretical Guarantees on Convergence

We provide theoretical guarantees that establish convergence properties of the kernel embedding likelihood-free inference (KELFI) framework. Section 4.4.1 begins by summarizing the properties of kernels used in KELFI and introducing relevant quantities. Sections 4.4.2 and 4.4.3 provide an overview of conditional mean embeddings (CMEs) and their empirical estimates respectively in the context of KELFI. Section 4.4.4 establishes general convergence theorems for estimators based on the CME. Finally, using these results, we prove convergence guarantees for the kernel means likelihood (KML), marginal kernel means likelihood (MKML), kernel means posterior (KMP), and kernel means posterior embedding (KMPE) in sections 4.4.5, 4.4.6, 4.4.7 and 4.4.8 respectively.

4.4.1 Kernel Properties

The KELFI framework uses a data kernel $k : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ where $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{D}$. We do not assume that $\mathcal{X}$ and $\mathcal{Y}$ are necessarily the same. For example, it is possible to record an observation $\mathbf{y}$ in which the simulator $p(\mathbf{x}|\boldsymbol{\theta})$ can never generate or fully recover, such as when $\mathcal{X} \subset \mathcal{Y}$. Conversely, it is also possible that the simulator $p(\mathbf{x}|\boldsymbol{\theta})$ can generate a larger variety of simulations $\mathbf{x}$ than that is possible to observe from the actual data generation process, such as when $\mathcal{Y} \subset \mathcal{X}$. It can also be neither of such cases such as when $\mathcal{X}$ an $\mathcal{Y}$ only have some overlap. However, since we assume $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{D}$, the kernel k is able to measure the similarity between simulated data $\mathbf{x} \in \mathcal{X} \subseteq \mathcal{D}$ and observed data $\mathbf{y} \in \mathcal{Y} \subseteq \mathcal{D}$.

The KELFI framework employs bounded symmetric positive definite kernels ℓ and k. Because they are bounded, we can explicitly denote the following upper bounds to their RKHS norm,

$$\bar{\ell} := \sup_{\boldsymbol{\theta} \in \vartheta} \|\ell(\boldsymbol{\theta}, \cdot)\|_{\mathcal{H}_\ell} = \sup_{\boldsymbol{\theta} \in \vartheta} \sqrt{\ell(\boldsymbol{\theta}, \boldsymbol{\theta})}, \tag{4.14}$$

$$\bar{k} := \sup_{\mathbf{d} \in \mathcal{D}} \|k(\mathbf{d}, \cdot)\|_{\mathcal{H}_k} = \sup_{\mathbf{d} \in \mathcal{D}} \sqrt{k(\mathbf{d}, \mathbf{d})}. \tag{4.15}$$

When ℓ and k are stationary, we have $\bar{\ell} = \sqrt{\ell(\mathbf{0}, \mathbf{0})}$ and $\bar{k} = \sqrt{k(\mathbf{0}, \mathbf{0})}$.

In the KELFI framework, we first select the ϵ-kernel κ_ϵ. Based on this the choice of the ϵ-kernel, we then select the kernel k to satisfy

$$\kappa_\epsilon(\mathbf{y}, \mathbf{x}) = c_\epsilon k(\mathbf{y}, \mathbf{x}), \tag{4.16}$$

where $c_\epsilon > 0$ is a scaling constant to ensure that $\kappa_\epsilon(\mathbf{y}, \mathbf{x}) = p_\epsilon(\mathbf{y}|\mathbf{x})$ is a normalized density on $\mathcal{Y}$. In contrast, the kernel k has no such restriction. Since it is a scaled

version of k, κ_ϵ is also bounded symmetric positive definite as a function of $\mathbf{x}$ and $\mathbf{y}$. In this way, $\kappa_\epsilon(\mathbf{d}, \cdot) \in \mathcal{H}_k$ is always in the RKHS $\mathcal{H}_k$ characterized by k for all $\mathbf{d} \in \mathcal{D}$. As a consequence, ϵ is also a hyperparameter of k, although this is not explicitly notated for brevity.

Since $\mathbf{y} \in \mathcal{Y} \subseteq \mathcal{D}$, we have $\kappa_\epsilon(\mathbf{y}, \cdot) \in \mathcal{H}_k$. We can then find its RKHS norm,

$$\|\kappa_\epsilon(\mathbf{y}, \cdot)\|_{\mathcal{H}_k} = c_\epsilon \|k(\mathbf{y}, \cdot)\|_{\mathcal{H}_k} = c_\epsilon \sqrt{k(\mathbf{y}, \mathbf{y})} = \sqrt{c_\epsilon} \sqrt{c_\epsilon k(\mathbf{y}, \mathbf{y})} = \sqrt{c_\epsilon} \sqrt{\kappa_\epsilon(\mathbf{y}, \mathbf{y})}, \tag{4.17}$$

which is different to $\|\kappa_\epsilon(\mathbf{y}, \cdot)\|_{\mathcal{H}_{\kappa_\epsilon}} = \sqrt{\kappa_\epsilon(\mathbf{y}, \mathbf{y})}$. Therefore, while the KELFI algorithm only requires κ_ϵ to be specified and k is not explicitly used, this subtle difference is a reminder that k is the underlying kernel that defines the RKHS, not κ_ϵ. As a consequence, we have that the upper bound to the RKHS norm of κ_ϵ satisfies

$$\bar{\kappa}_\epsilon := \sup_{\mathbf{d} \in \mathcal{D}} \|\kappa_\epsilon(\mathbf{d}, \cdot)\|_{\mathcal{H}_k} = \sqrt{c_\epsilon} \sup_{\mathbf{d} \in \mathcal{D}} \sqrt{\kappa_\epsilon(\mathbf{d}, \mathbf{d})}. \tag{4.18}$$

Furthermore, if κ_ϵ is stationary, then $\kappa_\epsilon(\mathbf{d}, \mathbf{d}) = \kappa_\epsilon(\mathbf{0}, \mathbf{0})$ for all $\mathbf{d} \in \mathcal{D}$. A typical example is the Gaussian density $\kappa_\epsilon(\mathbf{y}, \mathbf{x}) = \mathcal{N}(\mathbf{y}; \mathbf{x}, \epsilon^2 I)$. In this case, $c_\epsilon = 1/(\sqrt{2\pi}\epsilon)^n$ and $\kappa_\epsilon(\mathbf{y}, \mathbf{y}) = 1/(\sqrt{2\pi}\epsilon)^n$ are the same, and thus $\|\kappa_\epsilon(\mathbf{y}, \cdot)\|_{\mathcal{H}_k} = 1/(\sqrt{2\pi}\epsilon)^n = c_\epsilon$. The corresponding kernel k is the isotropic Gaussian kernel.

When $\mathcal{D} = \mathbb{R}^n$, the most commonly used kernel for the KELFI framework is the anisotropic Gaussian kernel where each dimension uses a potentially different length scale σ_i. When its length scales are learned via some hyperparameter learning algorithm, it is also referred to as the ARD kernel. This kernel has the following form,

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2} \sum_{i=1}^{n} \left(\frac{x_i - x_i'}{\sigma_i} \right)^2 \right). \tag{4.19}$$

Since $\kappa_\epsilon(\mathbf{y}, \mathbf{x}) = c_\epsilon k(\mathbf{y}, \mathbf{x})$, this means that the length scales are simply the ABC tolerance $\sigma_i = \epsilon_i$ for $i \in [n]$, and that there can be a separate tolerance for each dimension of the data or summary statistic. Similarly, when $\vartheta = \mathbb{R}^D$, we also often employ the ARD kernel for ℓ, but we use β_d, $d \in [D]$, to denote the length scales.

4.4.2 Conditional Mean Embedding

To construct a conditional mean operator $\mathcal{U}_{\mathbf{X}|\mathbf{\Theta}}$ corresponding to the distribution $p(\mathbf{x}|\boldsymbol{\theta})$, we first choose a kernel $\ell : \vartheta \times \vartheta \to \mathbb{R}$ for domain ϑ and another kernel $k : \mathcal{D} \times \mathcal{D} \to \mathbb{R}$ for domain $\mathcal{D}$. These kernels ℓ and k each describe how similarity is measured within their respective domains, and are bounded symmetric positive definite and characteristic such that they uniquely define the RKHS $\mathcal{H}_\ell$ and $\mathcal{H}_k$.

The conditional mean operator $\mathcal{U}_{\mathbf{X}|\mathbf{\Theta}} : \mathcal{H}_\ell \to \mathcal{H}_k$ is defined by the equation $\mu_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}} = \mathcal{U}_{\mathbf{X}|\mathbf{\Theta}}\ell(\boldsymbol{\theta}, \cdot)$, where $\mu_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}}$ is the CME defined by

$$\mu_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}} := \mathbb{E}[k(\mathbf{X}, \cdot)|\mathbf{\Theta} = \boldsymbol{\theta}]. \tag{4.20}$$

In this sense, $\mathcal{U}_{\mathbf{X}|\mathbf{\Theta}}$ sweeps out a family of CMEs $\mu_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}} \in \mathcal{H}_k$, each indexed by $\boldsymbol{\theta} \in \vartheta$.

We then define cross covariance operators $C_{\mathbf{X}\mathbf{\Theta}} := \mathbb{E}[k(\mathbf{X}, \cdot) \otimes \ell(\mathbf{\Theta}, \cdot)] : \mathcal{H}_\ell \to \mathcal{H}_k$ and $C_{\mathbf{\Theta}\mathbf{\Theta}} := \mathbb{E}[\ell(\mathbf{\Theta}, \cdot) \otimes \ell(\mathbf{\Theta}, \cdot)] : \mathcal{H}_\ell \to \mathcal{H}_\ell$. Alternatively, they can be seen as elements within the tensor product space $C_{\mathbf{X}\mathbf{\Theta}} \in \mathcal{H}_k \otimes \mathcal{H}_\ell$ and $C_{\mathbf{\Theta}\mathbf{\Theta}} \in \mathcal{H}_\ell \otimes \mathcal{H}_\ell$. That is, they are second order mean embeddings.

Under the assumption that $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\mathbf{\Theta}\mathbf{\Theta}})$, it can be shown that $\mathcal{U}_{\mathbf{X}|\mathbf{\Theta}} = C_{\mathbf{X}\mathbf{\Theta}}(C_{\mathbf{\Theta}\mathbf{\Theta}})^{-1}$. While this assumption is satisfied for finite domains ϑ with a characteristic kernel ℓ, it does not necessarily hold when ϑ is a continuous domain [Fukumizu et al., 2004]. Instead, in this case $C_{\mathbf{X}\mathbf{\Theta}}(C_{\mathbf{\Theta}\mathbf{\Theta}})^{-1}$ becomes only an approximation to $\mathcal{U}_{\mathbf{X}|\mathbf{\Theta}}$, and we instead regularize the inversion with a regularization hyperparameter $\lambda \geq 0$ and use $\mathcal{U}_{\mathbf{X}|\mathbf{\Theta}} = C_{\mathbf{X}\mathbf{\Theta}}(C_{\mathbf{\Theta}\mathbf{\Theta}} + \lambda I)^{-1}$, which also serves to avoid overfitting [Song et al., 2013]. This relaxation can be applied to all subsequent results and theorems.

For any function $f \in \mathcal{H}_k$, the conditional expectation of f under $p(\mathbf{x}|\boldsymbol{\theta})$, or $g(\boldsymbol{\theta}) := \mathbb{E}[f(\mathbf{X})|\mathbf{\Theta} = \boldsymbol{\theta}]$, can be expressed by the inner product $g(\boldsymbol{\theta}) := \langle f, \mu_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}} \rangle_{\mathcal{H}_k}$ by using the CME $\mu_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}}$.

4.4.3 Empirical Conditional Mean Embedding

Suppose $\{\boldsymbol{\theta}_j, \mathbf{x}_j\} \sim p(\mathbf{x}|\boldsymbol{\theta})\pi(\boldsymbol{\theta})$ are *iid* across $j \in [m]$. The conditional mean operator $\mathcal{U}_{\mathbf{X}|\mathbf{\Theta}}$ is estimated by

$$\hat{\mathcal{U}}_{\mathbf{X}|\mathbf{\Theta}} = \Phi(L + m\lambda I)^{-1}\Psi^T, \tag{4.21}$$

where $\Phi := \begin{bmatrix} k(\mathbf{x}_1, \cdot) & \cdots & k(\mathbf{x}_m, \cdot) \end{bmatrix}$, $\Psi := \begin{bmatrix} \ell(\boldsymbol{\theta}_1, \cdot) & \cdots & \ell(\boldsymbol{\theta}_m, \cdot) \end{bmatrix}$, and $L := \{\ell(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)\}_{i,j=1}^m$. The CME can then be estimated by

$$\hat{\mu}_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}} = \hat{\mathcal{U}}_{\mathbf{X}|\mathbf{\Theta}}\ell(\boldsymbol{\theta}, \cdot) = \Phi(L + m\lambda I)^{-1}\boldsymbol{\ell}(\boldsymbol{\theta}) \tag{4.22}$$

where $\boldsymbol{\ell}(\boldsymbol{\theta}) := \{\ell(\boldsymbol{\theta}_j, \boldsymbol{\theta})\}_{j=1}^m$ [Song et al., 2009].

For any function $f \in \mathcal{H}_k$, the conditional expectation of f under $p(\mathbf{x}|\boldsymbol{\theta})$, or $g(\boldsymbol{\theta}) := \mathbb{E}[f(\mathbf{X})|\mathbf{\Theta} = \boldsymbol{\theta}]$, can be approximated by the inner product $\hat{g}(\boldsymbol{\theta}) := \langle f, \hat{\mu}_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}} \rangle_{\mathcal{H}_k}$ by using the empirical CME $\hat{\mu}_{\mathbf{X}|\mathbf{\Theta}=\boldsymbol{\theta}}$. Letting $\mathbf{f} := \{f(\mathbf{x}_j)\}_{j=1}^m$, this approximation

admits the following form,

$$\hat{g}(\boldsymbol{\theta}) = \mathbf{f}^T(L + m\lambda I)^{-1}\boldsymbol{\ell}(\boldsymbol{\theta}). \tag{4.23}$$

Importantly, $\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}$ is estimated from *joint* samples $\{\boldsymbol{\theta}_j, \mathbf{x}_j\}_{j=1}^m$, even though it is encoding the corresponding conditional distribution $p(\mathbf{x}|\boldsymbol{\theta})$. It is this fact that allows for an arbitrary choice $\pi(\boldsymbol{\theta})$ on the marginal distribution of $\boldsymbol{\Theta}$, which does not necessarily need to be the same as $p(\boldsymbol{\theta})$.

Under the assumption that $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$, the empirical CME $\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}$ converges to the true CME $\mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}$ in RKHS norm at rate $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ [Song et al., 2009, Theorem 6]. That is,

$$\begin{aligned} &\forall \boldsymbol{\theta} \in \vartheta,\ \forall \epsilon > 0,\ \exists M_\epsilon > 0 \quad s.t. \\ &\mathbb{P}\Big[\big\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} - \mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\big\|_{\mathcal{H}_k} > M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon. \end{aligned} \tag{4.24}$$

Consequently, the empirical CME converges at rate $O_p(m^{-\frac{1}{4}})$ if λ is chosen to decay at rate $O_p(m^{-\frac{1}{2}})$, and often better convergence rates can be achieved under appropriate assumptions on $p(\mathbf{x}|\boldsymbol{\theta})$ [Song et al., 2013]. Again, the regularization hyperparameter λ relaxes the assumption that $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$.

Finally, since $\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} = \hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}\ell(\boldsymbol{\theta}, \cdot)$ convergences to $\mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} = \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\ell(\boldsymbol{\theta}, \cdot)$ in RKHS norm at rate $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ for all $\boldsymbol{\theta} \in \vartheta$ and $\ell(\boldsymbol{\theta}, \cdot)$ does not depend on m, we also have that $\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}$ converges to $\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}$ in HS norm at the same rate. That is,

$$\begin{aligned} &\forall \epsilon > 0,\ \exists M_\epsilon > 0 \quad s.t. \\ &\mathbb{P}\Big[\big\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\big\|_{HS} > M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon. \end{aligned} \tag{4.25}$$

4.4.4 General Convergence Theorems

We now establish some general convergence theorems for estimators based on inner products with the CME. The aim is to provide a sense of the stochastic convergence of any estimator $\hat{a}$ to its true quantity a with respect to some metric $d(\hat{a}, a)$. We do this by showing that either $\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} - \mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\|_{\mathcal{H}_k}$ or $\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}$ is an upper bound of $d(\hat{a}, a)$ up to a scaling constant.

Lemma 4.1. *Suppose that $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$ and that there exists $0 \leq \gamma < \infty$ such that for some estimator $\hat{a}$, target a, and metric $d(\hat{a}, a)$,*

$$d(\hat{a}, a) \leq \gamma\big\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\big\|_{HS}, \tag{4.26}$$

then the estimator $\hat{a}$ converges to the target a with respect to the metric d at rate $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.

Proof. Suppose that there exists $0 \leq \gamma < \infty$ such that (4.26) is satisfied. That is, the inequality (4.26) holds for all possible data observations $\{\boldsymbol{\theta}_j, \mathbf{x}_j\}_{j=1}^m$. For any constant C, the implication statement $\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} \leq C \implies d(\hat{a}, a) \leq C\gamma$ holds for all possible observation events $\omega \in \Omega$. Writing this explicitly in event space translates this to a statement of probability inequality,

$$\begin{aligned} &\{\omega \in \Omega : \left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} \leq C\} \subseteq \{\omega \in \Omega : d(\hat{a}, a) \leq C\gamma\} \\ &\implies \mathbb{P}\Big[\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} \leq C\Big] \leq \mathbb{P}\Big[d(\hat{a}, a) \leq C\gamma\Big]. \end{aligned} \tag{4.27}$$

Since we assume that $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$, statement (4.24) is valid. By letting $C = M_\epsilon((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ in (4.27), we immediately have that the probability inequality in statement (4.25) is also true if we replace $\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}$ with $d(\hat{a}, a)$ and M_ϵ with γM_ϵ,

$$\begin{aligned} &\mathbb{P}\Big[\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} > M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\ \implies &1 - \mathbb{P}\Big[\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} \leq M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\ \implies &\mathbb{P}\Big[\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} \leq M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] > 1 - \epsilon \\ \implies &\mathbb{P}\Big[d(\hat{a}, a) \leq \gamma M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] > 1 - \epsilon \\ \implies &1 - \mathbb{P}\Big[d(\hat{a}, a) \leq \gamma M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon \\ \implies &\mathbb{P}\Big[d(\hat{a}, a) > \gamma M_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon, \end{aligned} \tag{4.28}$$

where we employed statement (4.27) between the third and fourth line for $C = M_\epsilon((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Therefore, since M_ϵ is arbitrary, define $\tilde{M}_\epsilon := \gamma M_\epsilon$ so that the following statement holds,

$$\forall \epsilon > 0,\ \exists \tilde{M}_\epsilon > 0 \quad s.t. \quad \mathbb{P}\Big[d(\hat{a}, a) > \tilde{M}_\epsilon\Big((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}}\Big)\Big] < \epsilon. \tag{4.29}$$

In other words, the estimator $\hat{a}$ stochastically converges to a at a rate of at least $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$ with respect to the metric d. □

Lemma 4.2. *Suppose that $\ell(\boldsymbol{\theta}, \cdot) \in \text{image}(C_{\boldsymbol{\Theta}\boldsymbol{\Theta}})$ and that there exists $0 \leq \gamma < \infty$ such that for some estimator $\hat{a}$, target a, and metric $d(\hat{a}, a)$,*

$$d(\hat{a}, a) \leq \gamma \left\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} - \mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\right\|_{\mathcal{H}_k}, \tag{4.30}$$

then the estimator $\hat{a}$ converges to the target a with respect to the metric d at rate $O_p((m\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.

Proof. The proof is identical to the proof for lemma 4.1, where $\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}$ is replaced with $\left\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} - \mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\right\|_{\mathcal{H}_k}$ throughout. Alternatively, since $\left\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}} - \mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\right\|_{\mathcal{H}_k} = \left\|(\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}})\ell(\boldsymbol{\theta}, \cdot)\right\|_{\mathcal{H}_k} \leq \left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}\left\|\ell(\boldsymbol{\theta}, \cdot)\right\|_{\mathcal{H}_\ell} = \left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} -\right.$

$\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}\sqrt{\ell(\boldsymbol{\theta},\boldsymbol{\theta})}$, $\forall\boldsymbol{\theta}\in\vartheta$, we have that $d(\hat{a},a)\leq\gamma\ell(\boldsymbol{\theta},\boldsymbol{\theta})\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}-\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}\leq\gamma(\sup_{\boldsymbol{\theta}\in\vartheta}\sqrt{\ell(\boldsymbol{\theta},\boldsymbol{\theta})})\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}-\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}=\gamma\bar{\ell}\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}-\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}$, $\forall\boldsymbol{\theta}\in\vartheta$. Since $\gamma\bar{\ell}$ is finite and does not depend on m, we apply lemma 4.1 to arrive at lemma 4.2. □

With lemmas 4.1 and 4.2, we are now equipped to prove the convergence of various estimators based on CMEs.

In all subsequent proofs, recall that the approximate surrogate densities q depend on m and ϵ, as well as other kernel and regularization hyperparameters, even though this is not explicitly notated.

4.4.5 Convergence of Kernel Means Likelihood

Proof of Theorem 4.1. Consider the absolute difference between the KML $q(\mathbf{y}|\boldsymbol{\theta})$ and the likelihood $p_\epsilon(\mathbf{y}|\boldsymbol{\theta})$,

$$\begin{aligned}
|q(\mathbf{y}|\boldsymbol{\theta})-p_\epsilon(\mathbf{y}|\boldsymbol{\theta}))| =&|\langle\kappa_\epsilon(\mathbf{y},\cdot),\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\rangle_{\mathcal{H}_k}-\langle\kappa_\epsilon(\mathbf{y},\cdot),\mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\rangle_{\mathcal{H}_k}|\\
=&|\langle\kappa_\epsilon(\mathbf{y},\cdot),\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}-\mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\rangle_{\mathcal{H}_k}|\\
\leq&\|\kappa_\epsilon(\mathbf{y},\cdot)\|_{\mathcal{H}_k}\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}-\mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\|_{\mathcal{H}_k}\\
\leq&\bar{\kappa}_\epsilon\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}-\mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\theta}}\|_{\mathcal{H}_k}\\
=&\bar{\kappa}_\epsilon\|(\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}-\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}})\ell(\boldsymbol{\theta},\cdot)\|_{\mathcal{H}_k}\\
\leq&\bar{\kappa}_\epsilon\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}-\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}\|\ell(\boldsymbol{\theta},\cdot)\|_{\mathcal{H}_\ell}\\
=&\bar{\kappa}_\epsilon\sqrt{\ell(\boldsymbol{\theta},\boldsymbol{\theta})}\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}-\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}\\
\leq&\bar{\kappa}_\epsilon\bar{\ell}\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}}-\mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}.
\end{aligned}\tag{4.31}$$

□

Since $\gamma=\bar{\kappa}_\epsilon\bar{\ell}$ is independent of m, we apply lemma 4.1 to establish the convergence. Since this upper bound does not depend on $\boldsymbol{\theta}\in\vartheta$ or $\mathbf{y}\in\mathcal{Y}$ and the metric is the absolute difference, this convergence is uniform as a function of both $\boldsymbol{\theta}\in\vartheta$ and $\mathbf{y}\in\mathcal{Y}$.

Alternatively, convergence guarantees for the KML can be established by its connection to the form of a Gaussian process regressor (GPR), leveraging frameworks and properties from a regression perspective.

4.4.6 Convergence of Marginal Kernel Means Likelihood

Proof of Theorem 4.2. We begin by writing the marginalization as an expectation over $p(\boldsymbol{\theta})$. This gives us $q(\mathbf{y}):=\int_\vartheta q(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}=\mathbb{E}[q(\mathbf{y}|\boldsymbol{\Theta})]$ and $p_\epsilon(\mathbf{y}):=$

$\int_\vartheta p_\epsilon(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta} = \mathbb{E}[p_\epsilon(\mathbf{y}|\boldsymbol{\Theta})]$. Consider the absolute difference between the MKML $q(\mathbf{y})$ and the marginal likelihood $p_\epsilon(\mathbf{y})$,

$$\begin{aligned}
|q(\mathbf{y}) - p_\epsilon(\mathbf{y})| &= |\mathbb{E}[q(\mathbf{y}|\boldsymbol{\Theta}) - p(\mathbf{y}|\boldsymbol{\Theta})]| \\
&\leq \mathbb{E}[|q(\mathbf{y}|\boldsymbol{\Theta}) - p(\mathbf{y}|\boldsymbol{\Theta})|] \\
&\leq \bar{\kappa}_\epsilon \mathbb{E}[\|\hat{\mu}_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\Theta}} - \mu_{\mathbf{X}|\boldsymbol{\Theta}=\boldsymbol{\Theta}}\|_{\mathcal{H}_k}] \\
&= \bar{\kappa}_\epsilon \mathbb{E}[\|(\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}})\ell(\boldsymbol{\Theta}, \cdot)\|_{\mathcal{H}_k}] \\
&\leq \bar{\kappa}_\epsilon \mathbb{E}[\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}\|\ell(\boldsymbol{\Theta}, \cdot)\|_{\mathcal{H}_\ell}] \\
&= \bar{\kappa}_\epsilon \mathbb{E}[\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}\sqrt{\ell(\boldsymbol{\Theta}, \boldsymbol{\Theta})}] \\
&= \bar{\kappa}_\epsilon \mathbb{E}[\sqrt{\ell(\boldsymbol{\Theta}, \boldsymbol{\Theta})}]\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS} \\
&\leq \bar{\kappa}_\epsilon \mathbb{E}[\bar{\ell}]\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS} \\
&= \bar{\kappa}_\epsilon \bar{\ell}\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\|_{HS}
\end{aligned} \tag{4.32}$$

Since $\gamma = \bar{\kappa}_\epsilon \bar{\ell}$ is independent of m, we apply lemma 4.1 to establish the convergence. Since this upper bound does not depend on $\mathbf{y} \in \mathcal{Y}$ and the metric is the absolute difference, this convergence is uniform as a function of $\mathbf{y} \in \mathcal{Y}$. □

4.4.7 Convergence of Kernel Means Posterior

Proof of Theorem 4.3. Consider the density ratio between the approximate and true densities for the likelihood and its absolute difference to unity,

$$\begin{aligned}
\left|\frac{q(\mathbf{y}|\boldsymbol{\theta})}{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})} - 1\right| &\leq \frac{1}{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})}\left|q(\mathbf{y}|\boldsymbol{\theta}) - p_\epsilon(\mathbf{y}|\boldsymbol{\theta})\right| \\
&\leq \frac{\bar{\kappa}_\epsilon \bar{\ell}}{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}.
\end{aligned} \tag{4.33}$$

Similarly, consider the density ratio between the approximate and true densities for the marginal likelihood and its absolute difference to unity,

$$\begin{aligned}
\left|\frac{q(\mathbf{y})}{p_\epsilon(\mathbf{y})} - 1\right| &\leq \frac{1}{p_\epsilon(\mathbf{y})}\left|q(\mathbf{y}) - p_\epsilon(\mathbf{y})\right| \\
&\leq \frac{\bar{\kappa}_\epsilon \bar{\ell}}{p_\epsilon(\mathbf{y})}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}.
\end{aligned} \tag{4.34}$$

Now, consider the absolute difference between the KMP $q(\boldsymbol{\theta}|\mathbf{y})$ and the posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ for all $m > M$.

$$
\begin{aligned}
&\Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big| \\
=&\left|\frac{q(\mathbf{y}|\boldsymbol{\theta})}{q(\mathbf{y})} - \frac{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})}{p_\epsilon(\mathbf{y})}\right| p(\boldsymbol{\theta}) \\
=&\left|\frac{q(\mathbf{y}|\boldsymbol{\theta})}{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})} - \frac{q(\mathbf{y})}{p_\epsilon(\mathbf{y})}\right| \frac{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{|q(\mathbf{y})|} \\
=&\left|\Big(\frac{q(\mathbf{y}|\boldsymbol{\theta})}{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})} - 1\Big) - \Big(\frac{q(\mathbf{y})}{p_\epsilon(\mathbf{y})} - 1\Big)\right| \frac{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{|q(\mathbf{y})|} \\
\leq&\left(\left|\frac{q(\mathbf{y}|\boldsymbol{\theta})}{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})} - 1\right| + \left|\frac{q(\mathbf{y})}{p_\epsilon(\mathbf{y})} - 1\right|\right) \frac{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{|q(\mathbf{y})|} \\
\leq&\left(\frac{\bar{\kappa}_\epsilon \bar{\ell}}{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} + \frac{\bar{\kappa}_\epsilon \bar{\ell}}{p_\epsilon(\mathbf{y})}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}\right) \frac{p_\epsilon(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{|q(\mathbf{y})|} \\
\leq&\left(\bar{\kappa}_\epsilon \bar{\ell} p(\boldsymbol{\theta})\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} + \bar{\kappa}_\epsilon \bar{\ell} p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}\right) \frac{1}{|q(\mathbf{y})|} \\
\leq&\bar{\kappa}_\epsilon \bar{\ell}\big(p(\boldsymbol{\theta}) + p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\big)\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} \frac{1}{|q(\mathbf{y})|} \\
\leq&\frac{\bar{\kappa}_\epsilon \bar{\ell}\big(p(\boldsymbol{\theta}) + p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\big)}{\delta}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}.
\end{aligned}
\tag{4.35}
$$

Since $\gamma = \frac{\bar{\kappa}_\epsilon \bar{\ell}}{\delta}\big(p(\boldsymbol{\theta}) + p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\big)$ is independent of m and the upper bound holds for all $m > M$, we apply lemma 4.1 to establish the convergence. Since this upper bound does depend on $\boldsymbol{\theta} \in \vartheta$ and $\mathbf{y} \in \mathcal{Y}$ and the metric is the absolute difference, this convergence is pointwise as a function of $\boldsymbol{\theta} \in \vartheta$ and $\mathbf{y} \in \mathcal{Y}$.

Furthermore, if $\bar{p}_{\boldsymbol{\Theta}} := \sup_{\boldsymbol{\theta}\in\vartheta} p(\boldsymbol{\theta}) < \infty$ and $\bar{p}_{\mathbf{Y}|\boldsymbol{\Theta}} := \sup_{\boldsymbol{\theta}\in\vartheta} p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) < \infty$, then

$$
\begin{aligned}
p(\boldsymbol{\theta}) + p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) &\leq \sup_{\boldsymbol{\theta}\in\vartheta}\big(p(\boldsymbol{\theta}) + p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\big) \leq \sup_{\boldsymbol{\theta}\in\vartheta} p(\boldsymbol{\theta}) + \sup_{\boldsymbol{\theta}\in\vartheta} p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) \\
&\leq \sup_{\boldsymbol{\theta}\in\vartheta} p(\boldsymbol{\theta}) + \frac{\sup_{\boldsymbol{\theta}\in\vartheta} p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) \sup_{\boldsymbol{\theta}\in\vartheta} p(\boldsymbol{\theta})}{p_\epsilon(\mathbf{y})} \\
&= \bar{p}_{\boldsymbol{\Theta}} + \frac{\bar{p}_{\mathbf{Y}|\boldsymbol{\Theta}}\bar{p}_{\boldsymbol{\Theta}}}{p_\epsilon(\mathbf{y})}.
\end{aligned}
\tag{4.36}
$$

So, $\Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big| \leq \frac{\bar{\kappa}_\epsilon \bar{\ell}}{\delta}\big(\bar{p}_{\boldsymbol{\Theta}} + \frac{\bar{p}_{\mathbf{Y}|\boldsymbol{\Theta}}\bar{p}_{\boldsymbol{\Theta}}}{p_\epsilon(\mathbf{y})}\big)\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}$. Since the upper bound does not depend on $\boldsymbol{\theta} \in \vartheta$, the convergence is uniform as a function of $\boldsymbol{\theta} \in \vartheta$.

Similarly, if $\bar{p}_{\boldsymbol{\Theta}|\mathbf{Y}} := \sup_{\mathbf{y}\in\mathcal{Y}} p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) < \infty$, then $\Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big| \leq \frac{\bar{\kappa}_\epsilon \bar{\ell}}{\delta}\big(p(\boldsymbol{\theta}) + \bar{p}_{\boldsymbol{\Theta}|\mathbf{Y}}\big)\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}$. Since the upper bound does not depend on $\mathbf{y} \in \mathcal{Y}$, the convergence is uniform as a function of $\mathbf{y} \in \mathcal{Y}$. □

4.4.8 Convergence of Kernel Means Posterior Embedding

Proof of Theorem 4.4. Since ℓ is a bounded kernel, let $\bar{\bar{\ell}} := \sup_{\boldsymbol{\theta}\in\vartheta}\sup_{\boldsymbol{\theta}'\in\vartheta}\ell(\boldsymbol{\theta},\boldsymbol{\theta}') > 0$. Note that this is not necessarily the same as $\bar{\ell} := \sup_{\boldsymbol{\theta}\in\vartheta}\ell(\boldsymbol{\theta},\boldsymbol{\theta})$. Consider the difference between KMPE $\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}$ and $\mu_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}$ for all $m > M$ in RKHS norm,

$$
\begin{aligned}
&\left\|\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}} - \mu_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}\right\|^2_{\mathcal{H}_\ell} \\
&= \left\| \int_\vartheta \ell(\boldsymbol{\theta},\cdot)q(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta} - \int_\vartheta \ell(\boldsymbol{\theta},\cdot)p_\epsilon(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}\right\|^2_{\mathcal{H}_\ell} \\
&= \left\| \int_\vartheta \ell(\boldsymbol{\theta},\cdot)\Big(q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big)d\boldsymbol{\theta}\right\|^2_{\mathcal{H}_\ell} \\
&= \left\langle \int_\vartheta \ell(\boldsymbol{\theta},\cdot)\Big(q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big)d\boldsymbol{\theta}, \int_\vartheta \ell(\boldsymbol{\theta}',\cdot)\Big(q(\boldsymbol{\theta}'|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}'|\mathbf{y})\Big)d\boldsymbol{\theta}'\right\rangle_{\mathcal{H}_\ell} \\
&= \int_\vartheta\int_\vartheta \langle \ell(\boldsymbol{\theta},\cdot),\ell(\boldsymbol{\theta}',\cdot)\rangle_{\mathcal{H}_\ell}\Big(q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big)\Big(q(\boldsymbol{\theta}'|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}'|\mathbf{y})\Big)d\boldsymbol{\theta}d\boldsymbol{\theta}' \\
&= \int_\vartheta\int_\vartheta \ell(\boldsymbol{\theta},\boldsymbol{\theta}')\Big(q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big)\Big(q(\boldsymbol{\theta}'|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}'|\mathbf{y})\Big)d\boldsymbol{\theta}d\boldsymbol{\theta}' \\
&= \left|\int_\vartheta\int_\vartheta \ell(\boldsymbol{\theta},\boldsymbol{\theta}')\Big(q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big)\Big(q(\boldsymbol{\theta}'|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}'|\mathbf{y})\Big)d\boldsymbol{\theta}d\boldsymbol{\theta}'\right| \\
&\leq \int_\vartheta\int_\vartheta \Big|\ell(\boldsymbol{\theta},\boldsymbol{\theta}')\Big|\Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big|\Big|q(\boldsymbol{\theta}'|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}'|\mathbf{y})\Big|d\boldsymbol{\theta}d\boldsymbol{\theta}' \\
&\leq \int_\vartheta\int_\vartheta \bar{\bar{\ell}}^2\Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big|\Big|q(\boldsymbol{\theta}'|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}'|\mathbf{y})\Big|d\boldsymbol{\theta}d\boldsymbol{\theta}' \\
&= \bar{\bar{\ell}}^2\int_\vartheta \Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big|d\boldsymbol{\theta}\int_\vartheta \Big|q(\boldsymbol{\theta}'|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}'|\mathbf{y})\Big|d\boldsymbol{\theta}' \\
&= \bar{\bar{\ell}}^2\left(\int_\vartheta \Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big|d\boldsymbol{\theta}\right)^2.
\end{aligned}
\tag{4.37}
$$

We now employ inequality (4.35) that was derived within the proof of theorem 4.3,

$$
\begin{aligned}
\left\|\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}} - \mu_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}}\right\|_{\mathcal{H}_\ell} &\leq \bar{\bar{\ell}}\int_\vartheta \Big|q(\boldsymbol{\theta}|\mathbf{y}) - p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\Big|d\boldsymbol{\theta} \\
&\leq \bar{\bar{\ell}}\int_\vartheta \frac{\bar{\kappa}_\epsilon\bar{\ell}\big(p(\boldsymbol{\theta}) + p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\big)}{\delta}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}d\boldsymbol{\theta} \\
&= \bar{\bar{\ell}}\left(\int_\vartheta \big(p(\boldsymbol{\theta}) + p_\epsilon(\boldsymbol{\theta}|\mathbf{y})\big)d\boldsymbol{\theta}\right)\frac{\bar{\kappa}_\epsilon\bar{\ell}}{\delta}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS} \\
&= \frac{2\bar{\kappa}_\epsilon\bar{\ell}\bar{\bar{\ell}}}{\delta}\left\|\hat{\mathcal{U}}_{\mathbf{X}|\boldsymbol{\Theta}} - \mathcal{U}_{\mathbf{X}|\boldsymbol{\Theta}}\right\|_{HS}.
\end{aligned}
\tag{4.38}
$$

Since $\gamma = \frac{2\bar{\kappa}_\epsilon\bar{\ell}\bar{\bar{\ell}}}{\delta}$ is independent of m and the upper bound holds for all $m > M$, we apply lemma 4.1 to establish the convergence under the RKHS norm. □

4.5 Spatio-Temporal Kernel Means Likelihood

We specifically address the scenario for performing LFI on spatio-temporal data using KELFI. A subset of this scenario is the case where the data comes in the form of a time series. While this is a common setting for LFI problems, the sequential or spatially ordered nature of spatio-temporal data is rarely leveraged in standard LFI methods. This is because spatio-temporal data are often still summarized into summary statistics before the LFI algorithm is applied, meaning that the raw sequential and spatial data points were not fully utilized. The advantage of using summary statistics is that it reduces the problem down to the standard case. However, these summary statistics are rarely sufficient, as sufficient statistics can be extremely difficult to design or learn even from expert knowledge due to the very lack of knowledge of its generating distribution. Thanks to the flexible and generalizable framework KELFI provides, in this section we design suitable forms of the KML for applying KELFI in the spatio-temporal domain.

In general, the approach we will present applies to spatio-temporal data formed from observing any stochastic random field with multivariate inputs. However, for the purpose of simplicity and clarity, we will notate inputs with names that resonate with time, so that the resulting data can be intuitively understood as analogous to a time series.

Suppose our observed spatio-temporal data comes in the form of $\{t_i, y_i\}_{i=1}^n$ and a simulation of such data comes in the form of $\{s_i, x_i\}_{i=1}^{n'}$. We will assume that $x_i, y_i \in \mathbb{R}$ for simplicity, although this framework can be extended for multivariate outputs. In this case, t_i and s_i can be time stamps, in which they are scalar values, or spatio-temporal locations, in which they are multivariate inputs. In either case, we will denote the space they lie in as $\mathcal{T} = \mathcal{S}$. We will denote $\mathbf{y} = \{y_i\}_{i=1}^n$, $\mathbf{x} = \{x_i\}_{i=1}^{n'}$, $\mathbf{t} = \{t_i\}_{i=1}^n$, and $\mathbf{s} = \{s_i\}_{i=1}^{n'}$ (not to be confused with summary statistics, which is not required here).

We first define a positive definite kernel $h : \mathcal{T} \times \mathcal{T} \to \mathbb{R}$ to measure the similarity between the spatio-temporal inputs. Then, we construct an ϵ-kernel by using the full predictive distribution of a GPR,

$$\begin{aligned}\kappa_\epsilon((\mathbf{t}, \mathbf{y}), (\mathbf{s}, \mathbf{x})) &= p(\mathbf{y}|\mathbf{x}, \mathbf{t}, \mathbf{s}) \\ &= \mathcal{N}(\mathbf{y}; H_{\mathbf{st}}^T(H_{\mathbf{ss}} + \epsilon^2 I_{n'})^{-1}\mathbf{x}, H_{\mathbf{tt}} + \epsilon^2 I_n - H_{\mathbf{st}}^T(H_{\mathbf{ss}} + \epsilon^2 I_{n'})^{-1} H_{\mathbf{st}}).\end{aligned} \tag{4.39}$$

where $H_{\mathbf{s},\mathbf{t}} = \{H(s_i, t_j)\}_{i=1,j=1}^{n',n}$, $H_{\mathbf{s},\mathbf{s}} = \{H(s_i, s_j)\}_{i=1,j=1}^{n',n'}$, $H_{\mathbf{t},\mathbf{t}} = \{H(t_i, t_j)\}_{i=1,j=1}^{n,n}$.

That is, $p(\mathbf{y}|\mathbf{x}, \mathbf{t}, \mathbf{s})$ is the full predictive distribution of a GPR trained on $\{s_i, x_i\}_{i=1}^{n'}$ and evaluated at the query points $\{t_i, y_i\}_{i=1}^n$. The derivation of this full predictive distribution is given in Rasmussen and Williams [2006].

This was a modeling choice to leverage the spatio-temporal relationship between each data point. So, $\{s_i, x_i\}_{i=1}^{n'}$ and $\{t_i, y_i\}_{i=1}^{n}$ are modeled as noisy realizations, with noise level ϵ, from a GP. The ϵ-kernel then measures the likelihood of observing our observed spatio-temporal data $\{t_i, y_i\}_{i=1}^{n}$ given simulation data $\{s_i, x_i\}_{i=1}^{n'}$.

With the above ϵ-kernel, the new KML surrogate is

$$q(\mathbf{y}|\mathbf{t}, \boldsymbol{\theta}) = \boldsymbol{\kappa}_\epsilon(\mathbf{t}, \mathbf{y})^T (L + m\lambda I)^{-1} \boldsymbol{\ell}(\boldsymbol{\theta}), \tag{4.40}$$

where $\boldsymbol{\kappa}_\epsilon(\mathbf{t}, \mathbf{y}) := \{\kappa_\epsilon((\mathbf{t}, \mathbf{y}), (\mathbf{s}_j, \mathbf{x}_j))\}_{j=1}^{m}$. We call this the spatio-temporal kernel means likelihood (ST-KML).

Notice that we use GPs in a very different way to other ABC approaches that use GPs. For example, Gaussian process surrogate ABC (GPS-ABC) [Meeds and Welling, 2014] models the generation process of summary statistics from the simulator and summary operation as a GPR. The input to the GPR in GPS-ABC is a parameter $\boldsymbol{\theta} \in \vartheta$, whereas the input to the GPR in ST-KML is a spatio-temporal coordinate $t \in \mathcal{T}$ or $s \in \mathcal{S}$. In GPS-ABC, the GPR is trained on simulated pairs of $\{\boldsymbol{\theta}_j, \mathbf{x}_j\}_{j=1}^{m}$, and the predictive distribution is evaluated at $\{\boldsymbol{\theta}_i, \mathbf{y}\}_{i=1}^{n_q}$ with any query parameters $\{\boldsymbol{\theta}_i\}_{i=1}^{n_q}$ at the same observed data $\mathbf{y}$ to evaluate the approximate likelihood. Furthermore, each index correspond to the summary statistic of a whole dataset. This is in contrast to the GPR in ST-KML. In the ϵ-kernel of the ST-KML, the GPR is trained on $\{s_i, x_i\}_{i=1}^{n'}$ and the predictive distribution is evaluated at the observed spatio-temporal data $\{t_i, y_i\}_{i=1}^{n}$. Here, each index correspond to a datapoint. The relationship across datasets is then handled by the CME in the overall KML.

On the other hand, the GPR in Gaussian process accelerated ABC (GPA-ABC) [Wilkinson, 2014] treats the log likelihood density directly as targets, instead of summary statistics. Again, the input to their GPR is a parameter $\boldsymbol{\theta} \in \vartheta$.

Furthermore, unlike GPS-ABC where the overall likelihood of summary statistics given parameters is assumed to be Gaussian, in ST-KML the Gaussian assumption from the GPR is only on the *residual* process from the latent process to simulated processes $x(s)$ or observed processes $y(t)$, and particular forms for the overall likelihood is not assumed.

This is the first work in our knowledge that specifically address spatio-temporal data by defining a GP-based ϵ-kernel to capture spatio-temporal. Specifically, we leverage the smoothness properties in such a data, where y_i and y_j would be more related if t_i and t_j are close.

It is worthwhile to point out again that the GPR for the ϵ-kernel and the CME for the overall KML are modeling entirely different relationships between different pairs of spaces. The former models relationships from $\mathcal{T} = \mathcal{S}$ to $\mathcal{Y} = \mathcal{X}$. The latter models the relationship from ϑ to $\mathcal{X}$.

4.6 *iid* Kernel Means Likelihood

We specifically address the scenario for performing LFI on *iid* data using KELFI. Similar to the motivation behind the construction of ST-KML, ϵ-kernels that operate on the raw data directly instead of on the summary statistics can leverage more information for the KML approximation. In the setting where simulation data is limited, this extra information can make noticeable improvements to the KML approximation such that more accurate posterior inference can be achieved.

Importantly, it is worthwhile to point out that even under the scenario where a sufficient statistic is available for *iid* data, the KML still benefits from an ϵ-kernel that operate on the raw data directly. Recall that a statistic is sufficient with respect to a statistical model and its associated unknown parameter if "no other statistic that can be calculated from the same sample provides any additional information as to the value of the parameter" [Fisher, 1922]. In our setting, the statistical model is the data likelihood, which in the LFI setting is assumed to be unavailable or intractable. While the statistic is sufficient with respect to the true likelihood, it may not be necessarily sufficient to likelihood approximations, including the KML. Since we are performing inference using the KML, this means that we may still gain extra information if we use the original raw data.

In order to make use of the full raw *iid* dataset, we can construct ϵ-kernels through the MMD between empirical distributions described by the datasets. One common example is the unnormalized ϵ-kernel $\kappa_{\epsilon,\alpha}(\mathbf{y},\mathbf{x}) \propto k_{\epsilon,\alpha}(\mathbf{y},\mathbf{x}) = \exp\big(-\frac{1}{2\epsilon^2}\|\hat{\mu}_{\mathbf{Y}} - \hat{\mu}_{\mathbf{X}}\|^2_{\mathcal{H}_k}\big)$, where $\hat{\mu}_{\mathbf{Y}} = \frac{1}{n}\sum_{i=1}^n \bar{k}_\alpha(\mathbf{y}_i,\cdot)$, $\hat{\mu}_{\mathbf{X}} = \frac{1}{n}\sum_{i=1}^n \bar{k}_\alpha(\mathbf{x}_i,\cdot)$ are empirical mean embeddings of the observed and simulated raw data. Here $\bar{k}$ is another kernel with hyperparameters α, which can be learned via the MKML. This was also used in double kernel ABC (K2-ABC) [Park et al., 2016] and distribution regression ABC (DR-ABC) [Mitrovic et al., 2016] to remove the requirement of summary statistics. Note that the normalization constant for the density $\kappa_{\epsilon,\alpha}$ is not required as it will be canceled out in the surrogate density for the posterior.

In contrast to the MMD-based ϵ-kernel, a more direct approach would be to leverage the conditional independence of observations y_i given simulations x_i, which gives $\kappa_\epsilon(\mathbf{y},\mathbf{x}) = p_\epsilon(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^n p_\epsilon(y_i|x_i) = \prod_{i=1}^n \mathcal{N}(y_i; x_i, \epsilon^2)$. On top of the raw data itself, this has the added advantage of leveraging another crucial information, which is simply the size of the dataset. With this ϵ-kernel, the new KML surrogate is $q(\mathbf{y}|\boldsymbol{\theta}) = \boldsymbol{\kappa}_\epsilon(\mathbf{y})^T(L + m\lambda I)^{-1}\boldsymbol{\ell}(\boldsymbol{\theta})$, where $\boldsymbol{\kappa}_\epsilon(\mathbf{y}) := \{\kappa_\epsilon(\mathbf{y},\mathbf{x}_j)\}_{j=1}^m$. We call this the independent and identically distributed kernel means likelihood (iid-KML).

This can be extended to multivariate data by replacing $p_\epsilon(y_i|x_i) = \mathcal{N}(y_i; x_i, \epsilon^2)$ with $p_\epsilon(\mathbf{y}_i|\mathbf{x}_i) = \mathcal{N}(\mathbf{y}_i; \mathbf{x}_i; E)$ so that $\kappa_\epsilon(\mathbf{Y},\mathbf{X}) = \prod_{i=1}^n p_\epsilon(\mathbf{y}_i|\mathbf{x}_i)$, where E is a noise covariance matrix that can be learned also under the MKML.

4.7 Normalizing Priors

Under certain conditions, we can always transform a particular LFI problem into another LFI problem that involves a Gaussian or normal prior without loss of generality. These assumptions are that $p_{\boldsymbol{\Theta}}(\boldsymbol{\theta}) = \prod_{d=1}^{D} p_{\Theta_d}(\theta_d)$ is a continuous PDF whose entries are independent, and that its inverse marginal CDFs $P_{\Theta_d}^{-1}$ exists and is tractable.

In terms of notation, we denote the parameters as $\boldsymbol{\theta} = \{\theta_d\}_{d=1}^{D} \in \vartheta$ for D parameters. For this section only, multiple *iid* copies will be indexed by a superscript $\boldsymbol{\theta}^{(j)}$ for $j \in [m]$. Hence, the d-th parameter of the j-th parameter values is $\theta_d^{(j)}$. For densities, we use the corresponding random variable as the subscript to denote which distribution we are referring to. For example, we used $p(\boldsymbol{\theta})$ as the shorthand for the more formal notation of $p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$ in the rest of the paper, but here we will keep the subscript to make this explicit.

Suppose the original prior $p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$ is not necessarily Gaussian, but satisfies the aforementioned assumptions. Let $\mathbf{Z}$ be a random variable of the same dimensionality as $\boldsymbol{\Theta}$ with realization $\mathbf{z} \in \mathcal{Z}$. Let $p_{\mathbf{Z}}(\mathbf{z}) = \prod_{d=1}^{D} p_{Z_d}(z_d)$, where $p_{Z_d}(z_d) = \mathcal{N}(z_d; \mu_d, \sigma_d^2)$ so that its density is a multivariate anisotropic Gaussian. Convenient choices that simplify transformations are $\mu_d = 0$ and $\sigma_d = \sigma$ for all $d \in [D]$, although the general methodology remains.

Below we outline the general procedure for transforming a LFI problem into another LFI problem that involves a Gaussian prior.

1. Generate Gaussian samples $\mathbf{z}^{(j)} \sim p_{\mathbf{Z}}(\mathbf{z})$ for $j \in [m]$.
2. Convert Gaussian samples $\mathbf{z}$ into uniform samples $\mathbf{u}$ through $u_d^{(j)} = P_{Z_d}(z_d^{(j)})$ for $j \in [m]$ and $d \in [D]$. That is, $\mathbf{u}^{(j)} \sim U(0,1)^D$ for $j \in [m]$.
3. Convert uniform samples $\mathbf{u}$ into prior samples through $\theta_d^{(j)} = P_{\Theta_d}^{-1}(u_d^{(j)})$ for $j \in [m]$ and $d \in [D]$. The overall forward transformation is $\mathbf{T}(\mathbf{z}) := \{T_d(z_d)\}_{d=1}^{D}$ where $T_d(z_d) = P_{\Theta_d}^{-1}(P_{Z_d}(z_d))$. Since $P_{Z_d}^{-1}$ exists, the inverse transformation is $\mathbf{T}^{-1}(\boldsymbol{\theta}) = \{T_d^{-1}(\theta_d)\}_{d=1}^{D}$ where $T_d^{-1}(\theta_d) = P_{Z_d}^{-1}(P_{\Theta_d}(\theta_d))$. Hence, we have $\boldsymbol{\theta}^{(j)} = \mathbf{T}(\mathbf{z}^{(j)})$ for $j \in [m]$.
4. Run simulator at parameter samples $\mathbf{x}^{(j)} \sim p_{\mathbf{X}|\boldsymbol{\Theta}}(\cdot|\boldsymbol{\theta}^{(j)}) = p_{\mathbf{X}|\boldsymbol{\Theta}}(\cdot|T(\mathbf{z}^{(j)})) = p_{\mathbf{X}|\mathbf{Z}}(\cdot|\mathbf{z}^{(j)})$. We now have joint samples $\{\mathbf{z}^{(j)}, \mathbf{x}^{(j)}\}_{j=1}^{m}$.
5. Use the KELFI framework to approximate the posterior $p_{\mathbf{Z}|\mathbf{Y}}(\mathbf{z}|\mathbf{y})$ using the simulation pairs $\{\mathbf{z}^{(j)}, \mathbf{x}^{(j)}\}_{j=1}^{m}$. Either we obtain the KMP $q_{\mathbf{Z}|\mathbf{Y}}(\mathbf{z}|\mathbf{y})$, or we obtain KMPE super-samples $\{\hat{\mathbf{z}}_s\}_{s=1}^{S}$.

6. If we have samples $\{\hat{\mathbf{z}}_s\}_{s=1}^S$, then to obtain the corresponding samples for $q_{\boldsymbol{\Theta}|\mathbf{Y}}(\boldsymbol{\theta}|\mathbf{y})$, we simply pass the samples $\{\hat{\mathbf{z}}_s\}_{s=1}^S$ through the transformation $\mathbf{T}$ so that $\hat{\boldsymbol{\theta}}_s = \mathbf{T}(\hat{\mathbf{z}}_s)$ for $s \in [S]$.

7. If we have the KMP, then to obtain the corresponding posterior density we use the standard change of variable transformation which would yield $q_{\boldsymbol{\Theta}|\mathbf{Y}}(\boldsymbol{\theta}|\mathbf{y}) = q_{\mathbf{Z}|\mathbf{Y}}(\mathbf{T}^{-1}(\boldsymbol{\theta})|\mathbf{y})|\det J_{\mathbf{T}^{-1}}(\boldsymbol{\theta})|$. The Jacobian of $\mathbf{T}^{-1}$ is a $D \times D$ matrix whose (i,j)-th entry is $(J_{\mathbf{T}^{-1}}(\boldsymbol{\theta}))_{ij} := \frac{\partial T_i^{-1}}{\partial \theta_j}(\boldsymbol{\theta})$. Since the transformations of each parameter is done independently from each other, T_i^{-1} does not depend on θ_j if $i \neq j$. Consequently, the Jacobian is diagonal. The diagonal entries are $\frac{\partial T_i^{-1}}{\partial \theta_i}(\theta_i) = \frac{\partial}{\partial \theta_i} P_{Z_i}^{-1}(P_{\Theta_i}(\theta_i)) = (P_{Z_i}^{-1})'(P_{\Theta_i}(\theta_i)) p_{\Theta_i}(\theta_i) = [p_{Z_i}(P_{Z_i}^{-1}(P_{\Theta_i}(\theta_i)))]^{-1} p_{\Theta_i}(\theta_i) = [p_{Z_i}(T_i^{-1}(\theta_i))]^{-1} p_{\Theta_i}(\theta_i)$. In the second last equality we made use of the fact that the computation of the derivative of the quantile function requires only the knowledge of the density and the quantile function itself, since $(P^{-1})'(u) = (P'(P^{-1}(u)))^{-1}$. Thus, the determinant of the Jacobian is $\det J_{\mathbf{T}^{-1}}(\boldsymbol{\theta}) = \prod_{i=1}^d [p_{Z_i}(T_i^{-1}(\theta_i))]^{-1} p_{\Theta_i}(\theta_i) = p_{\boldsymbol{\Theta}}(\boldsymbol{\theta}) \big[\prod_{i=1}^d p_{Z_i}(T_i^{-1}(\theta_i))\big]^{-1} = p_{\boldsymbol{\Theta}}(\boldsymbol{\theta}) \big[p_{\mathbf{Z}}(\mathbf{T}^{-1}(\boldsymbol{\theta}))\big]^{-1}$. The change of variable transformation becomes

$$q_{\boldsymbol{\Theta}|\mathbf{Y}}(\boldsymbol{\theta}|\mathbf{y}) = q_{\mathbf{Z}|\mathbf{Y}}(\mathbf{T}^{-1}(\boldsymbol{\theta})|\mathbf{y}) \frac{p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})}{p_{\mathbf{Z}}(\mathbf{T}^{-1}(\boldsymbol{\theta}))}. \tag{4.41}$$

Finally, the form simplifies when the form of the KMP $q_{\mathbf{Z}|\mathbf{Y}}(\mathbf{T}^{-1}(\boldsymbol{\theta})|\mathbf{y})$ is substituted back in,

$$\begin{aligned} q_{\boldsymbol{\Theta}|\mathbf{Y}}(\boldsymbol{\theta}|\mathbf{y}) &= \frac{q_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y}|\mathbf{T}^{-1}(\boldsymbol{\theta})) p_{\mathbf{Z}}(\mathbf{T}^{-1}(\boldsymbol{\theta}))}{q_{\mathbf{Y}}(\mathbf{y})} \frac{p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})}{p_{\mathbf{Z}}(\mathbf{T}^{-1}(\boldsymbol{\theta}))} \\ &= \frac{q_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y}|\mathbf{T}^{-1}(\boldsymbol{\theta})) p_{\boldsymbol{\Theta}}(\boldsymbol{\theta})}{q_{\mathbf{Y}}(\mathbf{y})}. \end{aligned} \tag{4.42}$$

Note that the MKML $q_{\mathbf{Y}}(\mathbf{y})$ is still marginalized over the simpler Gaussian distribution,

$$q_{\mathbf{Y}}(\mathbf{y}) = \int_{\mathcal{Z}} q_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y}|\mathbf{z}) p_{\mathbf{Z}}(\mathbf{z}) d\mathbf{z}. \tag{4.43}$$

In this way, we simplify the LFI problem into another LFI problem which involves a Gaussian prior such that KELFI solutions are closed-form under Gaussian kernels. Once KELFI solutions have been computed in the new parameter space $\mathcal{Z}$, the solutions can be easily transformed back into the original parameter space ϑ as above.

This process is possible since the likelihood is intractable already. Hence, transformations $\mathbf{T}$ of variables $\mathbf{z}$ into simulator parameters $\boldsymbol{\theta}$ can be included as part of the simulator without changing the nature of the problem.

If simulation pairs $\{\boldsymbol{\theta}^{(j)}, \mathbf{x}^{(j)}\}_{j=1}^{m}$ in the original space are already provided, parameters $\boldsymbol{\theta}^{(j)}$ can be converted into Gaussian variables via $\mathbf{z}^{(j)} = \mathbf{T}^{-1}(\boldsymbol{\theta}^{(j)})$ for $j \in [m]$ so that the pairs $\{\mathbf{z}^{(j)}, \mathbf{x}^{(j)}\}_{j=1}^{m}$ can be used to proceed.

As an extension, instead of transforming the LFI problem with a general continuous prior into one with a Gaussian prior, if the prior is fundamentally multimodal, we can also transform it into one with a Gaussian mixture model as the prior. Since the prior density is a linear combination of Gaussians, all derivations remain closed-form from a linear combination of the results with each Gaussian component.

Finally, it is important to recognize that while there is no loss of generality to the inference problem when performing this prior transform, the transformation do change the interpretation of the hyperparameters learned with the MKML. Since the kernel ℓ is now placed in the $\mathcal{Z}$ space, the hyperparameters of ℓ cannot be interpreted directly for the original parameter space ϑ unless the transformation between $\mathcal{Z}$ and ϑ is simple enough to translate the interpretation. Nevertheless, hyperparameters can still be learned by optimizing the MKML.

4.8 Related Work

The simplest ABC algorithm is arguably the rejection ABC (REJ-ABC) sampler [Pritchard et al., 1999]. It posits a set of prior parameters and rejects those whose simulations do not match the observations within a fixed threshold $\epsilon > 0$ under a distance measure. This can be extremely expensive even for cheap simulators. Critically, the hyperparameter ϵ crucially balances accuracy and computational efficiency and its learning remains to be addressed.

Instead of sampling from the prior, MCMC-ABC and sequential Monte Carlo ABC (SMC-ABC) sample from proposal distributions iteratively and carefully accepts or discards each proposal stochastically based on approximate likelihood ratios [Marjoram et al., 2003, Sisson et al., 2007]. They can however suffer from slow mixing, where it is difficult to escape a lucky sample with a high likelihood. They also do not leverage likelihood smoothness and thus require multiple new simulations every iteration, which are then discarded and may still not result in an accepted sample. This is because approximations to the likelihood $p_\epsilon(\mathbf{y}|\boldsymbol{\theta})$ (4.2) rely on empirical means over S simulations for that particular parameter setting $\boldsymbol{\theta}$, $p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) \approx \frac{1}{S}\sum_{s=1}^{S} p_\epsilon(\mathbf{y}|\mathbf{x}^{(s)}) = \frac{1}{S}\sum_{s=1}^{S} \kappa_\epsilon(\mathbf{y}, \mathbf{x}^{(s)})$ [Andrieu et al., 2009].

Synthetic likelihood ABC (SL-ABC) and adaptive synthetic likelihood ABC (ASL-ABC) alternatively use Gaussian approximations $p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) \approx \mathcal{N}(\mathbf{y}; \boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}} + \epsilon^2 I)$ and estimate the mean and covariance from simulations [Wood, 2010]. Nevertheless,

these approaches also require generating S new simulations $\{\mathbf{x}^{(s)}\}_{s=1}^{S}$ corresponding to the particular parameter setting $\boldsymbol{\theta}$ where the likelihood is queried. Not only are synthetic likelihoods parametric Gaussian approximations, they also approximate separately at each $\boldsymbol{\theta}$. In contrast, surrogate likelihood approaches like KELFI use consistent nonparametric approximations so that (1) only one new simulation is required at each new parameter $\boldsymbol{\theta}$ and (2) likelihood queries do not need to be at parameters where simulations are available.

Another branch of study include stochastic variational inference (SVI) approaches to ABC, which treats the likelihood approximation as another source of stochasticity in the stochastic gradient. This includes AV-ABC [Moreno et al., 2016], VBIL [Tran et al., 2017b], and VBSL [Ong et al., 2018]. In contrast, likelihood-free variational inference (LFVI) [Tran et al., 2017a] uses density ratio estimation to approximate the variational objective, emphasizing inference on local latent variables. Nevertheless, SVI approaches posit parametric approximations that may have asymptotic bias, and are also harder to implement in practice. In contrast, nonparametric surrogate approaches like KELFI have asymptotic convergence and implementations boil down to simple linear algebra.

Kernel-based approaches that leverage likelihood smoothness have been studied recently to reduce simulation requirements. The philosophy is that simulations of close-by parameters are informative, thus past results should not be discarded but remembered, even if this introduces model bias. K-ABC [Nakagome et al., 2013], KR-ABC [Kajihara et al., 2018], and KBR [Fukumizu et al., 2013] also employ CMEs to reduce simulation requirements. They differ to KELFI in the three aspects of model, learning, and inference. (Model) While they build posterior mean embeddings directly, KELFI builds likelihood surrogates first and make use of the full prior density to further leverage prior information before building posterior surrogates, which are then embedded into closed-form posterior mean embeddings. In contrast, the prior only appears as samples from $p(\boldsymbol{\theta})$ in K-ABC, KR-ABC, and KBR. This both limits the prior knowledge leveraged and prohibit the use of proposal prior samples. (Learning) KELFI crucially addresses hyperparameter learning in reference to the inference problem directly which was not straightforward previously. (Inference) K-ABC and KBR primarily infer posterior expectations, while KR-ABC produce point estimates. Instead, we design a posterior sampling algorithm, which subsumes inferring posterior expectation. We further provide approximate posterior density KMP, which can both produce point estimates and quantify uncertainty.

As a consequence of theorem 4.4, the KMPE converges at rate $O_p(m^{-\frac{1}{4}})$ in RKHS norm if the regularization hyperparameter λ is chosen to decay at rate $O_p(m^{-\frac{1}{2}})$. Notably, this is faster than the convergence rate of KBR at $O_p(m^{-\frac{8}{27}\alpha})$ where $0 < \alpha \leq \frac{1}{2}$, which also requires further assumptions on cross-covariance operators and appropriate decays for regularization hyperparameters [Fukumizu et al., 2013].

Finally, we highlight that hyperparameter learning is a crucial aspect and differentiator of KELFI. This is especially true for learning ϵ, which tunes the critical balance between an accurate posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) \approx p_0(\boldsymbol{\theta}|\mathbf{y})$ with small ϵ requiring high numbers of simulation calls, or a less accurate posterior with large ϵ relaxing the number of simulations required. This has been a challenging issue to address in the ABC literature in reference to the inference problem, even though its selection is often pivotal to the performance of the algorithm.

In the GP literature, hyperparameter learning through maximum marginal likelihood plays an important role in the success of a GPR. GPS-ABC [Meeds and Welling, 2014] and GPA-ABC [Wilkinson, 2014] model the summary statistics surface and log likelihood surface respectively via a GP surrogate. In contrast, the KML model is equivalent to placing a GP surrogate on the likelihood surface itself. This removes the assumption that summary statistics are independent and Gaussian distributed as in GPS-ABC. It further avoids the need for a GP prior with non-zero mean as in GPA-ABC, since likelihoods should revert to zero in the absence of simulations. In exchange, we lose non-negativity guarantees in the likelihood and posterior. We address this by proposing to sample from the equivalent posterior mean embedding instead.

Importantly, while GPS-ABC and GPA-ABC apply the GP marginal likelihood to learn their surrogate hyperparameters, it cannot learn ϵ or other hyperparameters since they are not part of the surrogate. This is because both approaches maximize the marginal likelihood for the GPR problem on the their respective target surfaces, but not the marginal likelihood for the overall inference problem, thus excluding other hyperparameters in the process.

4.9 Experiments

The goal of the experiments is to demonstrate the inference accuracy of KELFI under limited simulation budget and the effectiveness of MKML hyperparameter learning. We begin with isotropic ϵ and anisotropic $\boldsymbol{\beta} = \beta_0\boldsymbol{\sigma}$, and learn (ϵ, β_0) by maximizing the MKML (4.5) while keeping $\lambda = 10^{-3}\beta_0$ fixed for simplicity. We use Halton sequences [Halton, 1960] to sample from the proposal prior, which is set to the original prior $\pi(\boldsymbol{\theta}) = p(\boldsymbol{\theta})$ for simplicity. In all experiments we found that we did not need to clip the KML or KMP even though they are not guaranteed a-priori to be strictly positive. This is because we used an universal kernel such as a Gaussian kernel on both ϑ and $\mathcal{D}$ so that their RKHS is dense in their respective L^2 spaces [Carmeli et al., 2010]. Because densities and likelihoods are often square-integrable, accurate estimations can be achieved. Finally, since we use kernel herding to super-sample the KMPE, the KMPE is not required to be positive to begin with.

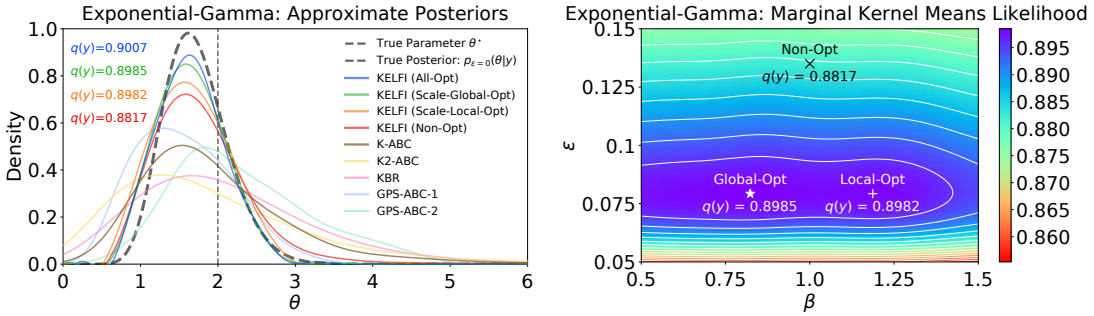

FIGURE 4.1: **(Left)** Comparison of approximate posteriors obtained from surrogate methods on the toy exponential-gamma problem. **(Right)** The corresponding MKML surface $q(\mathbf{y})$ as a function of (ϵ, β) with $\lambda = 10^{-3}\beta$.

4.9.1 Toy Problem: Exponential-Gamma

The toy exponential-gamma problem is a standard benchmark for likelihood-free inference, since the true posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ is known and tractable even for $\epsilon = 0$.

To stress-test each method, we compare inference accuracy under very limited simulations of $m = 100$. We focus on comparing surrogate approaches, since other methods such as REJ-ABC, MCMC-ABC, SL-ABC, and ASL-ABC have reported simulation requirements several orders higher than $m = 100$ on this problem [Meeds and Welling, 2014]. We use datasets of $n = 15$ for both observations and simulations, with their sample means as the summary statistic.

For GPS-ABC only we set a simulation budget of $m \leq 200$ and run it until 10000 posterior samples are generated. The hyperparameters of the GP surrogate itself are learned by maximizing the marginal likelihood of the GPR [Rasmussen and Williams, 2006]. For the remaining hyperparameters that are not part of the surrogate, several configurations are compared and the results of the best two are shown, which used $m = 130$ and $m = 197$ simulations. For K-ABC, K2-ABC, and KBR, we use the median heuristic to set their length scale hyperparameters and manually search for the most appropriate regularization hyperparameters. We use kernel density estimation (KDE) to visualize the posterior density from the unweighted samples of GPS-ABC and normalized weighted samples of K-ABC, K2-ABC, and KBR in figure 4.1 (left).

For KELFI, we show the KMPs directly in figure 4.1 (left). We first demonstrate the case when all hyperparameters $(\epsilon, \beta, \lambda)$ are learned (All-Opt). To enable visualization in 2D, we also present the case when the regularization hyperparameter λ is set to $10^{-3}\beta$ and only length scale hyperparameters (ϵ, β) are learned. In this case, we show KMPs under globally optimal (Scale-Global-Opt), locally optimal (Scale-Local-Opt), and arbitrarily chosen hyperparameters (Non-Opt). The corresponding MKML surface is shown in figure 4.1 (right).

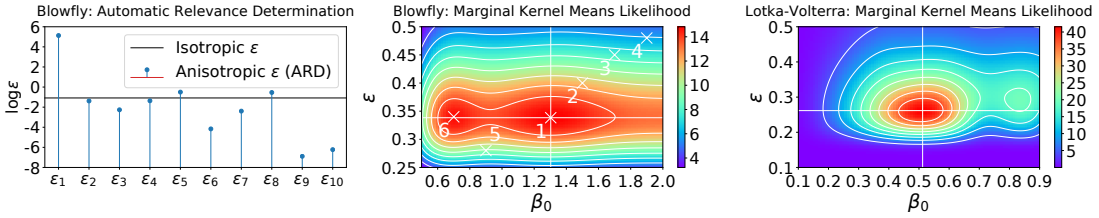

FIGURE 4.2: **(Left)** Blowfly: ARD on $\boldsymbol{\epsilon}$ for 10 summary statistics. **(Mid. & Right)** The MKML surface ($\times 10^5$) as a function of (ϵ, β_0) for fixed $\lambda = 10^{-3}\beta_0$ where $\boldsymbol{\beta} = \beta_0\boldsymbol{\sigma}$. White intersection indicate optimum. For Blowfly, the NMSE (in %) for the indicated hyperparameter choices are: $(1)0.72 \pm 0.02, (2)1.10 \pm 0.01, (3)2.07 \pm 0.01, (4)2.15 \pm 0.02, (5)1.11 \pm 0.02, (6)1.11 \pm 0.03$. At $(\epsilon, \beta_0) = (10, 10)$ (outside the plot) the NMSE (in %) is 6.28 ± 0.03.

In figure 4.1 we compare approximate posteriors from each algorithm against the true posterior $p_{\epsilon=0}(\boldsymbol{\theta}|\mathbf{y})$. While $\epsilon = 0$ for $p_{\epsilon=0}(\boldsymbol{\theta}|\mathbf{y})$, with only 100 simulations $\epsilon > 0$ is required for most LFI methods. Furthermore, except for K2-ABC, they only make use of summary statistics without further knowledge of the dataset size n. Consequently, most LFI methods produce approximations wider than $p_{\epsilon=0}(\boldsymbol{\theta}|\mathbf{y})$. Intuitively, there is not enough simulations and thus information to justify a more confident and peaked posterior. Nevertheless, by learning hyperparameters under the MKML, KELFI determines an appropriate scale ϵ for 100 simulations. As a result, KMPs are the closest to the true posterior $p_{\epsilon=0}(\boldsymbol{\theta}|\mathbf{y})$, with higher MKML $q(\mathbf{y})$ leading to more accurate KMPs $q(\boldsymbol{\theta}|\mathbf{y})$. This demonstrates the effectiveness of MKML as a hyperparameter learning objective for improving inference accuracy. In contrast, the two instances of GPS-ABC reveals that varying hyperparameters lead to significant changes in the resulting approximate posterior, yet without a similar objective like MKML it is unclear which one to use without ground truth. This is further emphasized by the wider posterior approximations obtained from K-ABC, K2-ABC, and KBR, which use the median heuristic to set hyperparameters. This is often sub-optimal as it makes no reference to the inference problem.

4.9.2 Chaotic Ecological Systems: Blowfly

The Blowfly simulator describes the complex population dynamics and evolution of adult blowflies. Across a range of parameters it exhibits chaotic behavior that have distinct discrepancies from real observations, resulting in a challenging inference problem. Our experimental setup follows that of Wood [2010]. There are 6 model parameters from which the simulator generates a time series of 180 data points that is then summarized into 10 statistics as described in Meeds and Welling [2014], Moreno et al. [2016], and Park et al. [2016]. The 10 summary statistics are the log of the mean of each quartile of $\{N_t/1000\}_{t=1}^T$ (4 statistics), the mean of each quartile of first-order differences of $\{N_t/1000\}_{t=1}^T$ (4 statistics), and the maximal peaks of smoothed $\{N_t\}_{t=1}^T$ with two different thresholds (2 statistics). We also

use a diagonal Gaussian prior on $\log \boldsymbol{\theta}$ with means $[2, -1.5, 6, -1, -1, \log(15)]$ and standard deviations $[2, 0.5, 0.5, 1, 1, \log(5)]$. Notice that we have slightly modified the standard deviation to be broader to make the problem more challenging.

The standard Blowfly problem has no ground truth parameters, only a set of observations. We therefore measure inference accuracy by considering the normalized mean squared error (NMSE), which is computed in the following manner.

Firstly, before the experiments, we sample 10000 parameters from the prior and simulate a set of summary statistic from each of them. We then compute mean squared errors (MSEs) of each simulated summary statistic against the observed summary statistic, and average them cross the 10000 MSEs. This leaves a vector of 10 numbers, consisting of average MSEs under the prior for each summary statistic. We use 10000 samples as MSE estimates has stabilized with little variance.

Secondly, during each experiment, we run 1000 simulations under the posterior mean or mode obtained from the algorithm, and compute MSEs of each simulated summary statistic against the observed summary statistic, and average them across the 1000 MSEs. This also produces a vector of 10 numbers, consisting of average MSEs under the posterior mean or mode for each summary statistic. We then divide the MSEs under the posterior mean or mode by the MSEs under the prior computed earlier. This results in a vector of 10 numbers which is now the NMSE for the 10 summary statistics. Since now all 10 error measures are normalized with respect to the prior, we average them for a final single NMSE score.

In this way, each statistic is normalized in the final average and a NMSE of 100% correspond to the performance of the prior. Hence, the NMSE measures the error as a percentage of the error achieved by the prior.

Finally, note that this is the NMSE score for a particular experiment. For each algorithm, we further repeat the experiment and thus this calculation process 10 times and show the average and the deviations in figure 4.3.

As simulations are expensive, in figure 4.3 (left) we record average NMSE against the number of simulations used to understand inference efficiency.

As new simulations become available, we relearn and update the hyperparameters for KELFI by maximizing the MKML. Figure 4.2 (center) shows an instance of the MKML surface used to learn the hyperparameters for KELFI when using $m = 280$ simulations. For KBR and K-ABC we update hyperparameters by the median length heuristic. For K-ABC we also report the case where the heurstic is scaled by an arbitrarily chosen constant denoted with (S), which achieved significantly better accuracy and confirms that the heuristic is often sub-optimal.

For all algorithms except KBR, we evaluate their performance by simulating from their posterior mean. For KBR only, we simulate from its posterior mode. This is

because KBR posterior mode consistently outperformed KBR posterior mean for the Blowfly problem. Using the posterior mode will present KBR in its best light.

We now detail the hyperparameter choices for each algorithm other than KELFI, since most algorithms do not have a hyperparameter learning algorithm for the inference problem. Refer to their respective papers for a description of the meaning of each hyperparameter. For algorithms that use a MCMC proposal distribution, we choose a Gaussian proposal distribution with a proposal standard deviations that are 10% of the prior standard deviations. For MCMC-ABC, we used $\epsilon = 5$. For SL-ABC, we used $\epsilon = 0.5$ and $S = 10$. For ASL-ABC, we used $S_0 = 10$, $\epsilon = 0.5$, $\xi = 0.3$, $m = 10$, and $\Delta S = 10$. For GPS-ABC, we used $S_0 = 20$ samples from ASL-ABC to initialize the GP surrogate, and choose $\epsilon = 2$, $\xi = 0.05$, $m = 10$, and $\Delta S = 5$. For K-ABC and KBR, we used median length heuristic to set length scale hyperparameters, and choose $\lambda = 10^{-4}$. Note that KBR uses two kernels on both the parameter and the summary statistics and have two regularization hyperparameters.

Overall, the top three performers are KELFI, KBR, and GPS-ABC. Across a range of simulation calls, KELFI achieves the lowest error. It is also the only method that achieved less than 1% average NMSE within 1000 simulations and achieves this as early as 300 simulations. The most competitive methods to KELFI are KBR and GPS-ABC. For these three methods, we also show their variability from best to worst case NMSEs out of the 10 repeats to visualize their sensitivity to the stochasticity in randomized simulations. This reveals that KELFI is a stable outperformer with comparatively less variability across randomized runs.

We proceed to demonstrate and emphasize the effectiveness and suitability of MKML as a hyperparameter learning objective, using the case with 280 simulations as an example. Figure 4.2 (center) illustrates that hyperparameters with a higher MKML (4.5) result in lower NMSE consistently. Notably, even with sub-optimal hyperparameter choices, KELFI still achieves competitive average NMSE scores of less than 2.2%. At 280 simulations, the next best average NMSE score is almost 3% by MCMC-ABC as shown in figure 4.3 (left).

Figure 4.3 (center) suggests that learning the scale ϵ under MKML reveals an automatic decay schedule which does not have to be set a-priori. As ϵ controls the scale within which discrepancies between simulations and observations are measured, it is expected that this scale decays as more simulation data is available. Without the MKML, both the initialization of ϵ and its decay schedule are not straight forward to determine.

In figure 4.2 (left), we show that we can perform ARD on the ABC ϵ-kernel $\kappa_{\boldsymbol{\epsilon}}$, and hence the kernel $k_{\boldsymbol{\epsilon}}$, by using a different ϵ_i for each of the 10 statistics. We do this by initializing each ϵ_i to the isotropic solution in figure 4.2 (center) and further optimize the MKML to learn all ϵ_i jointly. In particular, the first summary statistic

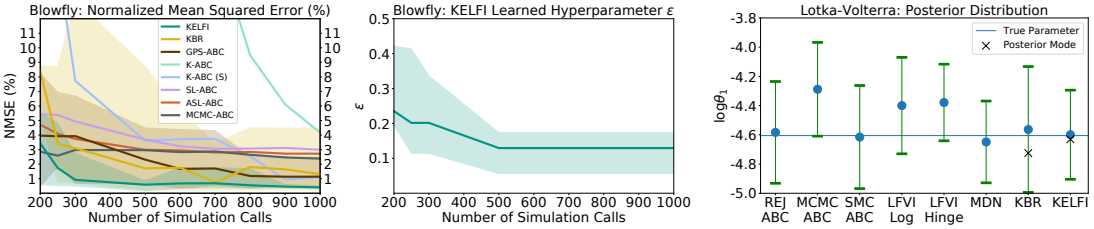

FIGURE 4.3: **(Left)** Blowfly: Average NMSE (in %) under posteriors against simulation calls. Shaded regions show NMSE variability for KELFI, KBR, and GPS-ABC. **(Mid.)** Blowfly: Learned ϵ value under maximum MKML. **(Right)** Lotka-Volterra: The middle 95% credible interval of the marginal posterior distribution of $\log \theta_1$.

describes the average log population numbers nears its troughs (first quartile), and is determined to be comparatively irrelevant (high ϵ_i). Meanwhile, the last two statistics describe the number of peaks at two thresholds, and are determined to be comparatively relevant (low ϵ_i). This agrees with the intuition that Blowfly population dynamics are highly characterized by its peaks, instead of i its troughs [Wood, 2010].

4.9.3 Predator-Prey Dynamics: Lotka-Volterra

The Lotka-Volterra simulator describes the time evolution of the populations within a predator-prey system. Only for a small set of parameters does the model simulate a realistic scenario with oscillatory behavior, making the inference task formidably challenging.

We reproduce the setup as described in Papamakarios and Murray [2016]. There are 4 parameters and 9 normalized summary statistics. We simulate data and hence summary statistics using the ground truth parameters and treat this as the observational data, and use it across all experiments and algorithms.

We place the same uniform prior on the log parameters. In particular, the problem places a uniform prior over $\log \boldsymbol{\theta}$. Since the parameters are independent from each other in the prior, transforming the LFI task into one with a Gaussian prior is straight forward by doing it separately for each parameter as described in section 4.7. To convert from $\log \boldsymbol{\theta}$ to $\mathbf{z}$, denoting a realization of a Gaussian random variable, we first offset and scale it to a uniform in $[0, 1]$ then apply the standard normal quantile function. To convert it back, which is required before we pass our parameter query to the simulator or to present our results, we apply the standard normal cumulative distribution function and scale and offset the uniform back to its original ranges. Similar to the other experiments, we do not learn the prior hyperparameters in this paper to enable benchmarking against other methods with the same prior, so the transformed prior stay as a standard normal.

To apply the closed-form solutions for KELFI, we transform the prior samples into a standard Gaussian distributed samples, apply KELFI, and transform the posterior samples back to the original space for $\log \boldsymbol{\theta}$.

After performing inference on all four parameters, we show in figure 4.3 (right) the marginal posterior distribution for $\log \theta_1$. With a uniform prior and a complex intractable likelihood, the posterior is unlikely to be a Gaussian. KELFI does not assume that the posterior is a Gaussian and thus can provide more flexible and accurate posteriors. After learning appropriate hyperparameters for KELFI under MKML, we draw 10000 super-samples from the KMPE to compute the posterior mean, and maximize the KMP to compute the posterior mode. Finally, to compute the 95% credible interval as shown in figure 4.3 (right), we compute the empirical 2.5% and 97.5% quantile using the 10000 super-samples.

KELFI achieves competitive performance using only 2500 simulations, with both posterior mean and mode close to the true value. The MKML for hyperparameter learning is shown in figure 4.2 (right). Posterior mode is obtained by maximizing the KMP. Meanwhile, the three ABC methods used up to 100000 simulations. While confident, LFVI [Tran et al., 2017a] tends to have a biased posterior mean. For direct comparison, both KELFI and mixture density network (MDN) [Papamakarios and Murray, 2016] use the original prior as the proposal prior. KELFI achieves slightly higher accuracy than MDN which used 10000 simulations, 4 times that used for KELFI. Finally, we also similarly use 2500 simulations for KBR. With the same number of simulations, KELFI achieves higher accuracy in both mean and mode with higher confidence.

4.10 Summary and Future Work

KELFI provides a holistic framework for automatic likelihood-free inference. It is a stable outperformer compared to state-of-the-art methods, while producing interpretable automatic relevance determination of summary statistics and automatic decay schedules for ϵ. By optimizing an approximate Bayesian marginal likelihood, it automatically learns and adapts hyperparameters including the ϵ-kernel to improve inference accuracy when limited simulations are available.

The framework is general and flexible, and can be extended in multiple directions. Since the samples $\boldsymbol{\theta}_j \sim \pi(\boldsymbol{\theta})$ do not have to be from the prior $p(\boldsymbol{\theta})$, to further reduce simulation requirements it is possible to choose or adapt π during the simulation process in a way that focuses on high likelihood regions.

The KML enables approximate likelihood queries at any $\boldsymbol{\theta} \in \vartheta$, even if simulation data is not available at the corresponding $\boldsymbol{\theta}$. By using the KML as a surrogate model for the true likelihood and accepting some modeling bias, we avoid

requiring multiple expensive simulations at each query $\boldsymbol{\theta}$ that is used by many MCMC-based ABC approaches. In fact, as a function of $\boldsymbol{\theta}$ the KML $q(\mathbf{y}|\cdot)$ is the predictive mean of a GPR [Rasmussen and Williams, 2006] trained on observations $\{\boldsymbol{\theta}_j, \kappa_\epsilon(\mathbf{y}, \mathbf{x}_j)\}_{j=1}^m$ with a GP prior $\mathcal{GP}(0, \ell)$ and Gaussian likelihood $\mathcal{N}(\mathbf{0}, m\lambda I)$, since they admit the same resulting form. This connection could provide uncertainty estimates in the KML approximation of the likelihood via the GP predictive variance. It is possible to then use Bayesian optimization (BO) [Snoek et al., 2012] or active learning methods to guide the proposal prior π in a sequential learning fashion that will result in the more accurate KML approximations for a fixed number m of simulations.

While our posterior mean embedding (4.9) is closed-form and thus exact for the surrogate density $q(\boldsymbol{\theta}|\mathbf{y})$, it is an approximation to the mean embedding $\mu_{\boldsymbol{\Theta}|\mathbf{Y}=\mathbf{y}} := \int_\vartheta \ell(\boldsymbol{\theta}, \cdot) p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) d\boldsymbol{\theta}$ of the true soft posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y}) \equiv p^{(\epsilon)}_{\boldsymbol{\Theta}|\mathbf{Y}}(\boldsymbol{\theta}|\mathbf{y})$, and converges in RKHS norm at the same rate as the KML. This is different in a subtle way to the CME of the posterior used by K-ABC, KR-ABC, and KBR, which in fact is an approximation to $\mu_{\boldsymbol{\Theta}|\mathbf{X}=\mathbf{y}} := \int_\vartheta \ell(\boldsymbol{\theta}, \cdot) p_{\boldsymbol{\Theta}|\mathbf{X}}(\boldsymbol{\theta}|\mathbf{y}) d\boldsymbol{\theta}$, the mean embedding of $p_{\boldsymbol{\Theta}|\mathbf{X}}(\boldsymbol{\theta}|\mathbf{y})$, which avoids using the ϵ-kernel. A key difference is that there is no known associated marginal likelihood or approximations thereof for the direct posterior mean embedding, so cross validation is required for selecting the remaining kernel hyperparameters in K-ABC, KR-ABC, and KBR. K-ABC also do not address sampling, although kernel herding can be readily applied in the same way. Kernel herding is applied to KBR in KMCF [Kanagawa et al., 2016] for resampling distributions represented as a CME. We believe it would be an interesting direction to investigate the relationships between the original empirical posterior mean embedding and the surrogate posterior mean embedding.

With regards to hyperparameter learning, in the KME literature, Bayesian learning of hyperparameters in marginal mean embeddings have been addressed through a different marginal likelihood approach by placing a GP prior on the embedding [Flaxman et al., 2016]. However, a general approach for learning CME hyperparameters in a Bayesian framework remains an open question. Our simple surrogate density approach can be an alternative solution to the CME Bayesian hyperparameter learning problem, and may lead to interesting connections.

With regards to sampling, by super-sampling the surrogate posterior mean embedding, the number of posterior samples is decoupled from the number of simulations. This is unlike likelihood-free MCMC methods for which the algorithm guides the simulator queries at parameter values that is not necessarily drawn from the prior, but rather from proposals of a Markov chain. This avoids the problem of slow mixing that is inherent in MCMC methods, and make KELFI more suitable for multi-modal posteriors, which remains to be experimented upon.

Chapter 5

Bayesian Deconditional Kernel Mean Embeddings

Conditional kernel mean embeddings form an attractive nonparametric framework for representing conditional means of functions, describing the observation processes for many complex models. However, the recovery of the original underlying function of interest whose conditional mean was observed is a challenging inference task. We formalize *deconditional kernel mean embeddings* as a solution to this inverse problem, and show that it can be naturally viewed as a nonparametric Bayes' rule. Critically, we introduce the notion of *task transformed Gaussian processes* and establish deconditional kernel means as their posterior predictive mean. This connection provides Bayesian interpretations and uncertainty estimates for deconditional kernel mean embeddings, explains their regularization hyperparameters, and reveals a marginal likelihood for kernel hyperparameter learning. These revelations further enable practical applications such as likelihood-free inference and learning sparse representations for big data.

5.1 Introduction

Observations of complex phenomena often lead to likelihoods that are described by a conditional mean. A widely applicable setting where this occurs is collecting observations under uncertain inputs, where the task is to learn a function $f : \mathcal{X} \to \mathbb{R}$ to model a real-valued response z as a function of inputs $x \in \mathcal{X}$ without being able to query or measure x directly to observe this phenomenon. Instead, another measured input $y \in \mathcal{Y}$ relates to x through $p(x|y)$. Consequently, given y, the response Z has mean $g(y) := \mathbb{E}[f(X)|Y = y]$, where g is called the conditional mean of f. Furthermore, $p(x|y)$ is often only available as sample pairs $\{x_i, y_i\}_{i=1}^n$, from simulations, algorithms, or separate experiments, making recovery of latent functions f from conditional means g a challenging inference task.

Our first contribution begins with formulating deconditional mean embeddings (DMEs) as solutions to this inference problem by building upon the framework of conditional mean embeddings (CMEs) [Song et al., 2013]. We show that the DME can be established as a nonparametric Bayes' rule in the RKHS and used for likelihood-free Bayesian inference. In contrast to kernel Bayes' rules (KBRs) [Fukumizu et al., 2013] which uses third order tensors that can result in vanishing priors, DMEs use second order tensors and avoids this problem.

Together with CMEs and KBR, DMEs form a critical part of the KME [Muandet et al., 2017] framework, where probabilistic rules can be represented nonparametrically as operators that are linear in the RKHS. This greatly simplifies probabilistic inference without requiring parametrized distributions and compromising flexibility.

Despite this connection, there are elements unique to the KME framework that cannot be interpreted or solved via the parallel between probability rules and RKHS mean operations. Similar to empirical forms for KBR and CMEs, empirical DMEs are obtained by replacing expectations in its constituent operators with their empirical means, and introduce regularization for operator inverses to relax RKHS assumptions, instead of as the optimal solution to a particular loss. Setting regularization hyperparameters is difficult in practice without an appropriate loss for the inference task. Furthermore, similar to KBR, the nonparametric Bayes' rule provided by DMEs is a statement between observed (or simulated) variables and not on latent functions or quantities. Consequently, uncertainty estimation in inference of latent functions f still require a separate Bayesian formulation.

Our second contribution establishes a Bayesian regression view of DMEs as posterior predictive means of the task transformed Gaussian process (TTGP), a novel nonparametric Bayesian model that recover latent relationships between variables without observing them jointly. TTGPs are so named because we show that they are a type of transformed Gaussian process [Murray-Smith and Pearlmutter, 2005] where the transformations and noise covariances are learned, by transforming one Gaussian process (GP) task to another, rather than designed from expert knowledge. We use this connection to derive posterior and predictive uncertainty estimates for DMEs and explain their regularization hyperparameters as a function of noise variance. Finally, we derive marginal likelihoods and their scalable computational forms to learn DME hyperparameters, which can also be applied to learn inducing points for sparse representations as a special case.

5.2 Kernel Mean Embeddings

We begin with an overview of the KME framework from which DMEs are built upon. KMEs are an arsenal of techniques concerned with representations and transformations of function expectations under highly flexible distributions. They consider functions that lie within RKHSs $\mathcal{H}_k$ and $\mathcal{H}_\ell$, formed by positive definite kernels $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. The RKHSs $\mathcal{H}_k$ and $\mathcal{H}_\ell$ are the closure span of the features $\phi(x) = k(x, \cdot)$ and $\psi(y) = \ell(y, \cdot)$ across $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ respectively, endowed with the inner products $\langle \cdot, \cdot \rangle_k \equiv \langle \cdot, \cdot \rangle_{\mathcal{H}_k}$ and $\langle \cdot, \cdot \rangle_\ell \equiv \langle \cdot, \cdot \rangle_{\mathcal{H}_\ell}$.

The key object is the mean embedding of a distribution $\mu_X := \mathbb{E}[k(X, \cdot)] \in \mathcal{H}_k$. They encode function expectations in the sense that $\mathbb{E}[f(X)] = \langle \mu_X, f \rangle_k$, due to the reproducing property that $\langle k(x, \cdot), f \rangle_k = f(x)$ for all $f \in \mathcal{H}_k$.

Higher ordered mean embeddings are vital components of the framework. Specifically, second order mean embeddings such as $C_{YY} := \mathbb{E}[\ell(Y, \cdot) \otimes \ell(Y, \cdot)] \in \mathcal{H}_\ell \otimes \mathcal{H}_\ell$ and $C_{XY} := \mathbb{E}[k(X, \cdot) \otimes \ell(Y, \cdot)] \in \mathcal{H}_k \otimes \mathcal{H}_\ell$ can be identified as cross-covariance operators $C_{YY} : \mathcal{H}_\ell \to \mathcal{H}_\ell$ and $C_{XY} : \mathcal{H}_\ell \to \mathcal{H}_k$ that serve as building blocks of CMEs and DMEs.

In practical scenarios where only *iid* samples $\{x_i, y_i\}_{i=1}^n$ that are realizations of $(X_i, Y_i) \sim \mathbb{P}_{XY}$ for $i \in \{1, \ldots, n\}$ are available, the KME framework becomes attractive for nonparametric inference because core objects only require expectations under distributions. Consequently, they can be estimated via empirical means as $\hat{\mu}_X := \frac{1}{n} \sum_{i=1}^n k(x_i, \cdot)$, $\hat{C}_{YY} := \frac{1}{n} \sum_{i=1}^n \ell(y_i, \cdot) \otimes \ell(y_i, \cdot)$, and $\hat{C}_{XY} := \frac{1}{n} \sum_{i=1}^n k(x_i, \cdot) \otimes \ell(y_i, \cdot)$ [Muandet et al., 2017].

For feature matrices, we stack features by columns $\Phi := \begin{bmatrix} \phi(x_1) & \cdots & \phi(x_n) \end{bmatrix}$ and $\Psi := \begin{bmatrix} \psi(y_1) & \cdots & \psi(y_n) \end{bmatrix}$. We write gram matrices as $K := \Phi^T \Phi$ and $L := \Psi^T \Psi$, where the (i, j)-th element of $A^T B$ is the inner product of the i-th column of A with the j-th column of B. That is, $K_{ij} = \phi(x_i)^T \phi(x_j)$ and $L_{ij} = \psi(y_i)^T \psi(y_j)$. When columns are elements of RKHSs such as when $\phi(x) = k(x, \cdot)$ in Φ and $\psi(y) = \ell(y, \cdot)$ in Ψ, the notation $(\cdot)^T(\cdot)$ is a shorthand for the corresponding RKHS inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ when it is clear from context what $\mathcal{H}$ is. For example, $f^T h$ is shorthand for $\langle f, h \rangle_k$ if $f, h \in \mathcal{H}_k$. Another common usage is $\Phi^T f = \{\phi(x_i)^T f\}_{i=1}^n = \{k(x_i, \cdot)^T f\}_{i=1}^n = \{\langle k(x_i, \cdot), f \rangle_k\}_{i=1}^n = \{f(x_i)\}_{i=1}^n =: \mathbf{f}$. For summing outer products, we write $\hat{C}_{YY} = \frac{1}{n} \Psi \Psi^T$ and $\hat{C}_{XY} = \frac{1}{n} \Phi \Psi^T$. Note that we use non-bold letters for single points x and y, even though they are often multivariate in practice.

5.3 Conditional Kernel Mean Embeddings

We now present CMEs in a fashion that focuses on their operator properties. By reviewing CMEs this way, parallels and contrast with DMEs in the subsequent section 5.4 become more apparent. Importantly, instead of defining CMEs via an explicit form, we begin by forming problem statements.

Definition 5.1 (Conditional Mean Problem Statement)**.** Given a function $f : \mathcal{X} \to \mathbb{R}$, infer the function $g : \mathcal{Y} \to \mathbb{R}$ such that $g(y) = \mathbb{E}[f(X)|Y = y] \equiv \mathbb{E}_{X|Y}[f](y)$. We call g the *conditional mean* of f with respect to $\mathbb{P}_{X|Y}$ and write the shorthand $g = \mathbb{E}_{X|Y}[f] = \mathbb{E}[f(X)|Y = \cdot]$.

This naturally leads to the notion of operators that map functions f to their conditional means $g = \mathbb{E}[f(X)|Y = \cdot]$.

Definition 5.2 (Conditional Mean Operators)**.** The CMO $C_{X|Y} : \mathcal{H}_\ell \to \mathcal{H}_k$ corresponding to $\mathbb{P}_{X|Y}$ is the operator that satisfies

$$(C_{X|Y})^T f = \mathbb{E}[f(X)|Y = \cdot], \quad \forall f \in \mathcal{H}_k, \tag{5.1}$$

where $(C_{X|Y})^T : \mathcal{H}_k \to \mathcal{H}_\ell$ denotes the adjoint of $C_{X|Y}$.

Depending on the nature of ℓ, unique solutions exist.

Theorem 5.1 (Fukumizu et al. [2004])**.** *Assume that $\ell(y, \cdot) \in \text{image}(C_{YY})$ for all $y \in \mathcal{Y}$. The conditional mean operator (CMO) $C_{X|Y}$ is unique and given by*

$$C_{X|Y} = C_{XY} C_{YY}^{-1}. \tag{5.2}$$

The assumption that $\ell(y, \cdot) \in \text{image}(C_{YY})$ for all $y \in \mathcal{Y}$ is commonly relaxed by introducing a regularization hyperparameter $\lambda > 0$ to the inverse, so that the CMO is replaced with $C_{XY}(C_{YY} + \lambda I)^{-1}$ [Song et al., 2013].

Contrary to definition 5.2, it is more common in the literature to define the CMO as the operator $C_{X|Y}$ that satisfies

$$C_{X|Y}\ell(y, \cdot) = \mathbb{E}[k(X, \cdot)|Y = y], \quad \forall y \in \mathcal{Y}, \tag{5.3}$$

while (5.1) is taken as an immediate property of CMOs [Fukumizu et al., 2004]. However, due to lemma 5.1, we instead take definition 5.2 as the definition of CMOs, emphasizing CMOs as solutions to the conditional mean problem, and treat (5.3) as an immediate property.

Lemma 5.1. *Statements (5.1) and (5.3) are equivalent.*

The CME of $\mathbb{P}_{X|Y=y}$ is $\mu_{X|Y=y} := C_{X|Y}\ell(y,\cdot) \in \mathcal{H}_k$, equivalent to querying the CMO at a particular input y. Consequently,

$$\begin{aligned}\langle \mu_{X|Y=y}, f\rangle_k &= \langle C_{X|Y}\ell(y,\cdot), f\rangle_k \\ &= \langle \ell(y,\cdot), (C_{X|Y})^T f\rangle_\ell \\ &= \langle \ell(y,\cdot), g\rangle_\ell \\ &= g(y).\end{aligned} \tag{5.4}$$

Motivated by theorem 5.1, empirical CMOs and CMEs are defined by estimating their constituents by empirical means.

Definition 5.3 (Empirical Conditional Mean Operator)**.** The empirical CMO is $\hat{C}_{X|Y} := \hat{C}_{XY}(\hat{C}_{YY} + \lambda I)^{-1}$, $\lambda > 0$.

Theorem 5.2 (Song et al. [2009])**.** *The nonparametric form for* $\hat{C}_{X|Y}$ *is*

$$\hat{C}_{X|Y} = \Phi(L + n\lambda I)^{-1}\Psi^T. \tag{5.5}$$

The empirical CME is then $\hat{\mu}_{X|Y=y} := \hat{C}_{X|Y}\ell(y,\cdot)$.

Consequently, with $\boldsymbol{\ell}(y) := \{\ell(y_i, y)\}_{i=1}^n$, an estimate for $\mathbb{E}_{X|Y}[f](y)$ is $\langle f, \hat{\mu}_{X|Y=y}\rangle_k = \langle f, \hat{C}_{X|Y}\ell(y,\cdot)\rangle_k = f^T\Phi(L + n\lambda I)^{-1}\Psi^T\ell(y,\cdot) = \mathbf{f}^T(L + n\lambda I)^{-1}\boldsymbol{\ell}(y)$.

Critically, while empirical CMOs (5.5) are estimated from joint samples from the joint distribution $\mathbb{P}_{XY}$, they only encode the conditional distribution $\mathbb{P}_{X|Y}$. This means that the empirical CMOs will encode the same conditional distribution even if the joint distribution $\mathbb{P}_{XY}$ changes but the conditional distribution $\mathbb{P}_{X|Y}$ stays the same. That is, the empirical CMO built from joint samples of $p(x,y) = p(x|y)p(y)$ and the empirical CMO built from joint samples of $q(x,y) := p(x|y)q(y)$ will encode the same conditional distribution $p(x|y)$ and converge to the same CMO.

5.4 Deconditional Kernel Mean Embeddings

We now present a novel class of KMEs referred to as deconditional mean embeddings (DMEs). They are natural counterparts to CMEs. The presentation of definitions and theorems in this section is mainly parallel to section 5.3. We define the *deconditional mean* problem as the task of recovering latent functions from their conditional means.

Definition 5.4 (Deconditional Mean Problem Statement)**.** Given a function $g : \mathcal{Y} \to \mathbb{R}$, infer a function $f : \mathcal{X} \to \mathbb{R}$ such that $g(y) = \mathbb{E}[f(X)|Y = y]$. We call f a *deconditional mean* of g with respect to $\mathbb{P}_{X|Y}$ and write the shorthand $f = \mathbb{E}^\dagger_{X|Y}[g]$.

The deconditional mean of a function g infers the function f whose conditional mean would be g with respect to $\mathbb{P}_{X|Y}$. The corresponding operator that encodes this transformation is the DMO.

Definition 5.5 (Deconditional Mean Operators)**.** The deconditional mean operator (DMO) $C'_{X|Y} : \mathcal{H}_k \to \mathcal{H}_\ell$ corresponding to $\mathbb{P}_{X|Y}$ is the operator that satisfies

$$(C'_{X|Y})^T \mathbb{E}[f(X)|Y = \cdot] = f, \quad \forall f \in \mathcal{H}_k. \tag{5.6}$$

Depending on the nature of ℓ and k, unique solutions exist.

Theorem 5.3. *Assume that $\ell(y, \cdot) \in \text{image}(C_{YY})$ for all $y \in \mathcal{Y}$ and $k(x, \cdot) \in \text{image}(C_{X|Y} C_{YY} (C_{X|Y})^T)$ for all $x \in \mathcal{X}$. The deconditional mean operator (DMO) $C'_{X|Y}$ is unique and given by*

$$C'_{X|Y} = (C_{X|Y} C_{YY})^T (C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1}. \tag{5.7}$$

Similar to the case with CMOs [Song et al., 2013], the assumption that $k(x, \cdot) \in \text{image}(C_{X|Y} C_{YY} (C_{X|Y})^T)$ for all $x \in \mathcal{X}$ can be relaxed by introducing a regularization hyperparameter $\epsilon > 0$ to the inverse, so that the DMO is replaced with $(C_{X|Y} C_{YY})^T (C_{X|Y} C_{YY} (C_{X|Y})^T + \epsilon I)^{-1}$.

Since DMOs invert the results of CMOs, they can also be understood as pseudo-inverses of CMOs.

Theorem 5.4. *If the symmetric inverse $((C_{X|Y})^T C_{X|Y})^{-1}$ exists such that the pseudo-inverse $C^\dagger_{X|Y} := ((C_{X|Y})^T C_{X|Y})^{-1} (C_{X|Y})^T$ of the CMO is well defined, and further the assumptions in theorem 5.3 hold, then DMOs are pseudo-inverses of CMOs $C'_{X|Y} = C^\dagger_{X|Y}$.*

The DME of $\mathbb{P}_{X=x|Y}$ is $\mu'_{X=x|Y} := C'_{X|Y} k(x, \cdot) \in \mathcal{H}_\ell$, equivalent to querying the DMO at a particular input x. Consequently,

$$\begin{aligned} \langle \mu'_{X=x|Y}, g \rangle_\ell &= \langle C'_{X|Y} k(x, \cdot), g \rangle_\ell \\ &= \langle k(x, \cdot), (C'_{X|Y})^T g \rangle_k \\ &= \langle k(x, \cdot), f \rangle_k \\ &= f(x). \end{aligned} \tag{5.8}$$

The form in (5.7) makes it evident that a DMO can be fully specified once $C_{X|Y}$ and C_{YY}, encoding the measures $\mathbb{P}_{X|Y}$ and $\mathbb{P}_Y$ respectively, are known. If densities exist, we write them as $p_{X|Y} \equiv p_{X|Y}(\cdot|\cdot)$ and $p_Y \equiv p_Y(\cdot)$, and drop the subscripts in density evaluations as $p(x|y)$ and $p(y)$ whenever the context is clear. Note that $\mathbb{P}_{X=x|Y}$ corresponds to $p_{X|Y}(x|\cdot)$ which is evaluated at x and now a function of y.

This is in contrast with $\mathbb{P}_{X|Y=y}$ corresponding to $p_{X|Y}(\cdot|y)$ evaluated at y and now a function of x.

Consider the case where X and Y play the roles of observed and unobserved (latent) variables respectively. The DMO considers the conditional $p_{X|Y}$ and the marginal p_Y encoded as $C_{X|Y}$ and C_{YY} (theorem 5.3), and inverts the CMO $C_{X|Y}$ (theorem 5.4) with the help of the encoded marginal C_{YY}. This is analogous to the Bayes' rule, where the posterior $p_{Y|X}(\cdot|x) = \frac{p_{X|Y}(x|\cdot)p_Y(\cdot)}{\int_{\mathcal{Y}} p_{X|Y}(x|y)p_Y(y)dy}$ is fully specified by the likelihood $p_{X|Y}$ and prior p_Y. We can then interpret DMEs as querying the rule at the observed quantity x while leaving the rule as a function of y for inference. Consequently, we also refer to $C_{X|Y}$ and C_{YY} as the likelihood operator and the prior operator respectively.

The difference between the DMO (5.7) and CMO (5.2) equations is akin to writing $p_{Y|X}(\cdot|x)$ using Bayes' rule against using the conditional density rule. Compare the DMO decomposition (5.7) with the CMO decomposition $C_{Y|X} = C_{YX}C_{XX}^{-1} = (C_{XY})^T C_{XX}^{-1}$ in the other direction by reversing the roles of X and Y in (5.2), which would correspond to the posterior $\mathbb{P}_{Y|X}$. The CMO is composed of a *joint* operator C_{XY} and an *evidence* operator C_{XX} corresponding to the joint $\mathbb{P}_{XY}$ and evidence $\mathbb{P}_X$ distributions. Similarly, the DMO is also composed of a *joint* operator $C_{XY} = C_{X|Y}C_{YY} : \mathcal{H}_\ell \to \mathcal{H}_k$ and an *evidence* operator $C'_{XX} := C_{X|Y}C_{YY}(C_{X|Y})^T : \mathcal{H}_k \to \mathcal{H}_k$, but both specified from the likelihood and prior operators.

Motivated by this, we propose to estimate the likelihood and prior operators using separate and independently drawn samples. The likelihood operator $C_{X|Y}$ is estimated as $\hat{C}_{X|Y}$ (definition 5.3) using *iid* samples $\{x_i, y_i\}_{i=1}^n$, also denoted as $\mathbf{x} := \{x_i\}_{i=1}^n$ and $\mathbf{y} := \{y_i\}_{i=1}^n$. Note that as the likelihood operator is a CMO, these joint samples can be from any joint distribution $\mathbb{Q}_{XY} \neq \mathbb{P}_{XY}$ as long as its conditional distribution is also $\mathbb{P}_{X|Y}$. The prior operator C_{YY} is estimated as $\tilde{C}_{YY} := \frac{1}{m}\sum_{j=1}^m \ell(\tilde{y}_j, \cdot) \otimes \ell(\tilde{y}_j, \cdot)$ using another set of *iid* samples $\tilde{\mathbf{y}} := \{\tilde{y}_j\}_{j=1}^m$ from $\mathbb{P}_Y$.

Definition 5.6 (Empirical Deconditional Mean Operator)**.** Let $\epsilon > 0$ be a regularization hyperparameter and define $\hat{C}_{X|Y}$ and $\tilde{C}_{YY}$ as above. The empirical DMO is

$$\bar{C}'_{X|Y} := (\hat{C}_{X|Y}\tilde{C}_{YY})^T(\hat{C}_{X|Y}\tilde{C}_{YY}(\hat{C}_{X|Y})^T + \epsilon I)^{-1}. \tag{5.9}$$

The accents notate the set of samples used for estimation. When both sets are used such as in the estimation of the DMO $C'_{X|Y}$, we denote it with a bar such as $\bar{C}'_{X|Y}$.

Theorem 5.5. *The nonparametric form for* $\bar{C}'_{X|Y}$ *is*

$$\bar{C}'_{X|Y} = \tilde{\Psi}\left[A^T K A + m\epsilon I\right]^{-1} A^T \Phi^T, \tag{5.10}$$

where $A := (L + n\lambda I)^{-1}\tilde{L}$, $\tilde{L} := \Psi^T\tilde{\Psi}$, *and* $\tilde{\Psi} := \begin{bmatrix}\psi(\tilde{y}_1) & \cdots & \psi(\tilde{y}_m)\end{bmatrix}$.

The empirical DME is then $\bar{\mu}'_{X=x|Y} := \bar{C}'_{X|Y}k(x,\cdot)$.

Consequently, with $\boldsymbol{k}(x) := \{k(x_i,x)\}_{i=1}^n$ and $\tilde{\mathbf{g}} := \{g(\tilde{y}_j)\}_{j=1}^m$, an estimate for $\mathbb{E}^{\dagger}_{X|Y}[g](x)$ is $\langle g, \bar{\mu}'_{X=x|Y}\rangle_\ell = \tilde{\mathbf{g}}^T\left[A^TKA + m\epsilon I\right]^{-1}A^T\boldsymbol{k}(x)$. This motivates the following definitions, where the notation $\tilde{\mathbf{g}}$ is replaced with $\tilde{\mathbf{z}}$, to be interpreted as target observations of g at $\tilde{\mathbf{y}}$.

Definition 5.7 (Nonparametric DME Estimator)**.** The nonparametric DME estimator, also called the kernel DME estimator or the DME estimator in function space view, is $\bar{f}(x) = \bar{\boldsymbol{\alpha}}^T\boldsymbol{k}(x) = \sum_{i=1}^n \bar{\alpha}_i k(x_i,x)$, where $\bar{\boldsymbol{\alpha}} := A\left[A^TKA + m\epsilon I\right]^{-1}\tilde{\mathbf{z}}$ and $A := (L + n\lambda I)^{-1}\tilde{L}$. Equivalently, $\bar{f}(x) = \tilde{\mathbf{z}}^T\left[A^TKA + m\epsilon I\right]^{-1}A^T\boldsymbol{k}(x)$. An alternative form is $\bar{f}(x) = \tilde{\mathbf{z}}^TA^T\left[KAA^T + m\epsilon I\right]^{-1}\boldsymbol{k}(x)$.

When features $\phi(x) \in \mathbb{R}^p$ and $\psi(y) \in \mathbb{R}^q$ are finite dimensional, we define the parametric DME estimator as follows by rewriting definition 5.7 using the Woodbury identity

Definition 5.8 (Parametric DME Estimator)**.** The parametric DME estimator, also called the feature DME estimator or the DME estimator in weight space view, is $\bar{f}(x) = \bar{\mathbf{w}}^T\phi(x)$, where $\bar{\mathbf{w}} = [\Phi AA^T\Phi^T + m\epsilon I]^{-1}\Phi A\tilde{\mathbf{z}}$ and $A := \Psi^T(\Psi\Psi^T + n\lambda I)^{-1}\tilde{\Psi}$. Equivalently, $\bar{f}(x) = \tilde{\mathbf{z}}^TA^T\Phi^T[\Phi AA^T\Phi^T + m\epsilon I]^{-1}\phi(x)$.

In definition 5.7 (resp. 5.8), computational complexity is dominated by inversions for $L + n\lambda I$ and $A^TKA + m\epsilon I$ (resp. $\Psi\Psi^T + n\lambda I$ and $\Phi AA^T\Phi^T + m\epsilon I$) at $O(n^3)$ and $O(m^3)$ (resp. $O(q^3)$ and $O(p^3)$). For the alternative form in definition 5.7, both inversions are $O(n^3)$, allowing for larger m at $O(m)$ without compromising tractability.

5.5 Task Transformed Gaussian Processes

DMEs are constructed as solutions to the task of inferring deconditional means, which are often real-valued functions. Regression problems also address inference of real-valued functions from data. This raises curiosity towards whether DMEs can be formulated as solutions to a regression-like problem, and what insights this connection would provide.

In this section, we formulate the task transformed regression problem to provide regression views of DMEs. To do this, we first briefly review transformed regression in section 5.5.1 before we present our contributions in section 5.5.2.

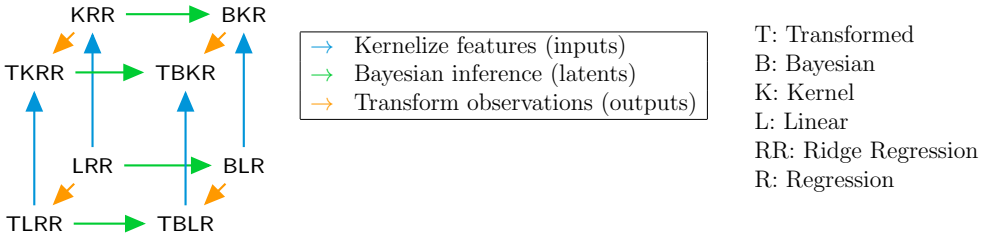

FIGURE 5.1: Three dimensions of model extensions. Kernel extensions (blue) specify feature spaces implicitly through a kernel. Bayesian extensions (green) introduce notions of uncertainty on latent quantities (weights or functions). Finally, transformed extensions (orange) capture indirect function observations.

5.5.1 Transformed Regression

Standard regression models often assume a Gaussian full data likelihood $p(\mathbf{z}|\mathbf{f}) = \mathcal{N}(\mathbf{z}; \mathbf{f}, \sigma^2 I)$ with targets $\mathbf{z} := \{z_i\}_{i=1}^n \in \mathbb{R}^n$. In the generalized setting when observations of $\mathbf{f}$ at $\mathbf{x}$ are not available but observations of linear combinations thereof are, we can use $p(\tilde{\mathbf{z}}|\mathbf{f}) = \mathcal{N}(\tilde{\mathbf{z}}; M^T\mathbf{f}, \Sigma)$ for some transformation $M \in \mathbb{R}^{n\times m}$ and noise covariance Σ, where $\tilde{\mathbf{z}} := \{\tilde{z}_j\}_{j=1}^m \in \mathbb{R}^m$ are the available observations.

We refer to this setting as *transformed regression*. They can be seen as another dimension of modeling with linear ridge regression (LRR) as the base model (figure 5.1). KRR is obtained from LRR via the kernel trick and Woodbury identity, and they are MAP solutions or predictive means of GPR and Bayesian linear regression (BLR) respectively [Rasmussen and Williams, 2006]. Consequently, we also refer to GPR as Bayesian kernel regression (BKR). Analogous relationships hold between transformed models.

5.5.2 Task Transformed Regression

We define *task transformed regression (TTR)* as the problem of learning to predict a target variable Z from features X when no direct sample pairs of X and Z are available but instead indirect samples $\{x_i, y_i\}_{i=1}^n$ and $\{\tilde{y}_j, \tilde{z}_j\}_{j=1}^m$ with a mediating variable Y are available. The name illustrates the idea of transforming the task of regressing Z on Y to learn $g : \mathcal{Y} \to \mathbb{R}$, using the *task* or *original* dataset $\{\tilde{y}_j, \tilde{z}_j\}_{j=1}^m$, to the task of regressing Z on X to learn $f : \mathcal{X} \to \mathbb{R}$, by mediating the *task* dataset through the *transformation* dataset $\{x_i, y_i\}_{i=1}^n$. As the mediating variable Y links the two sets together, $\mathbf{y}$ and $\tilde{\mathbf{y}}$ control the task transformation.

5.5.2.1 DME as solution to chained loss

We formulate losses for TTR, and establish DMEs as solutions. We begin with the parametric case with $f(x) = \mathbf{w}^T\phi(x)$ and $g(y) = \mathbf{v}^T\psi(y)$.

Theorem 5.6 (Task transformed linear ridge regression (TTLRR))**.** *The weights of the parametric DME estimator $\bar{f}(x) = \bar{\mathbf{w}}^T\phi(x)$ (definition 5.8) solve chained regularized least square losses,*

$$\begin{aligned} \hat{\mathbf{v}}[\mathbf{w}] &:= \operatorname*{arg\,min}_{\mathbf{v}\in\mathbb{R}^q} \frac{1}{n}\sum_{i=1}^{n}(\mathbf{w}^T\phi(x_i) - \mathbf{v}^T\psi(y_i)))^2 + \lambda\|\mathbf{v}\|^2, \\ \bar{\mathbf{w}} &:= \operatorname*{arg\,min}_{\mathbf{w}\in\mathbb{R}^p} \frac{1}{m}\sum_{j=1}^{m}(\tilde{z}_j - \hat{\mathbf{v}}[\mathbf{w}]^T\psi(\tilde{y}_j))^2 + \epsilon\|\mathbf{w}\|^2. \end{aligned} \tag{5.11}$$

The notation $\hat{\mathbf{v}}[\mathbf{w}]$ explicitly denotes that $\hat{\mathbf{v}}$ depends on $\mathbf{w}$. Conceptually, in function space view the first optimization finds g so that f at $\mathbf{x}$ best matches with g at $\mathbf{y}$, leading to a solution $\hat{g}[f]$ that is dependent on f. The second finds f so that $\hat{g}[f]$ at $\tilde{\mathbf{y}}$ best matches targets $\tilde{\mathbf{z}}$. Using the kernel trick $k(x, x') = \phi(x)^T\phi(x')$, we obtain the nonparametric case.

Lemma 5.2 (Task transformed kernel ridge regression (TTKRR))**.** *The weights of the nonparametric DME estimator $\bar{f}(x) = \bar{\boldsymbol{\alpha}}^T\boldsymbol{k}(x)$ (definition 5.7) satisfies $\bar{\mathbf{w}} = \Phi\bar{\boldsymbol{\alpha}}$ (the kernel trick).*

5.5.2.2 DME as posterior predictive mean of TTGP

We extend TTLRR and TTKRR to the Bayesian case. This connection reveals that TTR models are transformed regression models with transformations and noise covariances that are *learned*.

In the Bayesian parametric case, we have task transformed Bayesian linear regression (TTBLR). We place separate independent Gaussian priors $p(\mathbf{v}) = \mathcal{N}(\mathbf{v}; \mathbf{0}, \beta^2 I)$ and $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \gamma^2 I)$ for g and f respectively. As z is not observed directly from f but only for g, we include noise only for observing g to arrive at likelihoods $p(z|\mathbf{v}) = \mathcal{N}(z; \mathbf{v}^T\psi(y), \sigma^2)$ and $p(z|\mathbf{w}) = \mathcal{N}(z; \mathbf{w}^T\phi(x), 0)$ for g and f respectively.

In the Bayesian nonparametric case, we have task transformed Bayesian kernel regression (TTBKR). We place GP priors $g \sim \mathcal{GP}(0, \ell)$ and $f \sim \mathcal{GP}(0, k)$ on the functions directly. Consequently, TTBKR is also referred to as task transformed Gaussian process regression (TTGPR). Similar to TTBLR, the likelihoods are $p(z|g) = \mathcal{N}(z; g(y), \sigma^2)$ and $p(z|f) = \mathcal{N}(z; f(x), 0)$.

The graphical model for TTGPR is shown in figure 5.2. The two GPs for g and f are linked by constraining their targets to be the same at $\mathbf{y}$ and $\mathbf{x}$ respectively. The

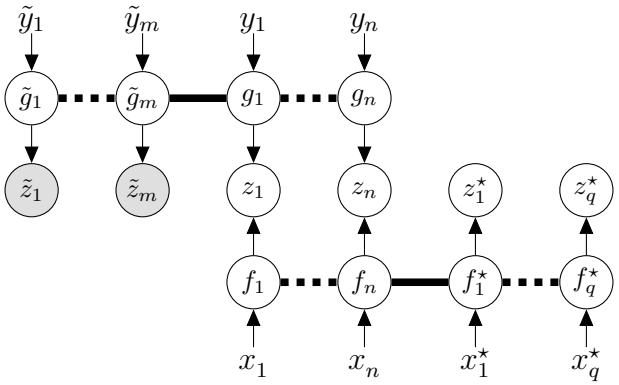

FIGURE 5.2: Graphical model (chain graph) for task transformed Gaussian process regression (TTGPR). Circles are random variables. Shaded circles are observed random variables. Undirected edges indicate the GP field, where all the random variables on the field are fully connected to each other [Rasmussen and Williams, 2006]. The goal is to infer $\mathbf{f}^\star$ to predict $\mathbf{z}^\star$ at $\mathbf{x}^\star$, using only a *task* or *original* dataset $\{\tilde{y}_j, \tilde{z}_j\}_{j=1}^m$ and a *transformation* dataset $\{x_i, y_i\}_{i=1}^n$. To connect the two GPs, we posit that the unobserved targets $\mathbf{z}$ at $\mathbf{x}$ and at $\mathbf{y}$ would have been the same if they were observed. Note that like regular GPs, to TTGPs the inputs x and y are not modeled as random variables but treated as index variables instead.

GP for g is used to infer the predictive distribution $p(\tilde{\mathbf{z}}|\mathbf{z})$, which in turn specifies the overall likelihood $p(\tilde{\mathbf{z}}|\mathbf{f})$ used to infer f. Detailed derivations are provided in the proof of theorem 5.7.

Theorem 5.7 (Task transformed Bayesian linear regression (TTBLR) and Task transformed Bayesian kernel regression (TTBKR))**.** ***(1)*** *The TTBLR is a TBLR with* $M = \Psi^T(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I)^{-1}\tilde{\Psi}$ *and* $\Sigma = \sigma^2\tilde{\Psi}^T(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I)^{-1}\tilde{\Psi} + \sigma^2 I$ *as the transformation and noise covariance.* ***(2)*** *The TTBKR is a TBKR with transformation* $M = (L + \sigma^2 I)^{-1}\tilde{L}$ *and noise covariance* $\Sigma = \tilde{\tilde{L}} + \sigma^2 I - \tilde{L}^T(L + \sigma^2 I)^{-1}\tilde{L}$*.* ***(3)*** *The TTBLR and TTBKR marginal likelihoods are* $p(\tilde{\mathbf{z}}) = \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, [\Sigma^{-1} - \Sigma^{-1}A^T\Phi^T C\Phi A\Sigma^{-1}]^{-1})$ *with shorthand* $C = [\Phi A\Sigma^{-1}A^T\Phi^T + \frac{1}{\gamma^2}I]^{-1}$ *and* $p(\tilde{\mathbf{z}}) = \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, A^T K A + \Sigma)$ *respectively.* ***(4)*** *For both models, when the posterior for* g *is approximated via MAP, the covariance becomes* $\Sigma = \sigma^2 I$*. In this case, the parametric (resp. nonparametric) DME estimator (definitions 5.7 and 5.8) is the predictive mean of a TTBLR (resp. TTBKR) with* $\lambda = \frac{\sigma^2}{n\beta^2}$ *and* $\epsilon = \frac{\sigma^2}{m\gamma^2}$ *(resp.* $\lambda = \frac{\sigma^2}{n}$ *and* $\epsilon = \frac{\sigma^2}{m}$*). An alternative TTBKR marginal likelihood is* $p(\tilde{\mathbf{z}}) = \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, \sigma^2[I - A^T(KAA^T + \sigma^2 I)^{-1}KA]^{-1})$*.* ***(5)*** *When both posteriors for* g *and* f *are approximated via MAP, TTBLR and TTBKR becomes TTLRR and TTKRR respectively with* λ *and* ϵ *from (4).*

Importantly, our end goal is to infer f. While this involves inferring g, g is not of direct interest. A simpler alternative is to only perform Bayesian inference on f and approximate g with its MAP solution, simplifying the noise covariance via (4) of theorem 5.7. This establishes a Bayesian interpretation for DMEs as MAP

estimates of TTGPs. Critically, by maximizing the TTGP marginal likelihood, we can learn DME hyperparameters of kernels k and ℓ, and also λ and ϵ. Furthermore, the computational complexity for alternative marginal likelihood is dominated by inversions that are $O(n^3)$ only, again allowing for larger m at $O(m)$.

In summary, we first establish the DME as a nonparametric solution to the TTR problem under chained regularized least squares losses that make learning f dependent on learning g. While λ and ϵ are previously seen as numerical adjustments to relax RKHS assumptions and stabilize matrix inversions in the KME framework, they can now be seen as controlling the amount of function regularization under this loss. Secondly, we present TTGPs as nonparametric Bayesian solutions to this regression problem and show that DMEs are their posterior predictive means. Again, inference of f is dependent on the inference of g, allowing GP uncertainties to propagate through. This connection provides Bayesian interpretations of DMEs and enable uncertainty estimation in inferring deconditional means. Critically, we use this to derive marginal likelihoods for hyperparameter learning.

5.6 Nonparametric Bayes' Rule

While DMOs were constructed as solutions to the deconditional mean problem, they also resemble Bayes' rule when we focus solely on considering the encoded relationship between X and Y. This was motivated by theorem 5.3, which revealed that the DMO can be fully specified by the CMO $C_{X|Y}$ and the second order mean embedding C_{YY} that encoded the likelihood $\mathbb{P}_{Y|X}$ and prior $\mathbb{P}_Y$ respectively. To establish this view, we investigate the conditions for which the DMO $C'_{X|Y}$ coincide with the CMO $C_{Y|X}$ that encodes the posterior $\mathbb{P}_{Y|X}$, leading to a nonparametric Bayes' rule.

While first class citizens of probability rules are density evaluations, first class citizens of the KME framework are expectations. Consequently, instead of relating density evaluations, rules under the KME framework relate mean embeddings of distributions at various orders. Importantly, while a distribution $Y \sim \mathbb{P}_Y$ has one simple density evaluation $p_Y(y)$, it can have different RKHS representations at different orders such as μ_Y and C_{YY} or higher.

A nonparametric Bayes' rule is a rule which translates Bayes' rule into the RKHS, where distributions are represented as RKHS operators, alleviating limitations from parametric assumptions or approximations such as Gaussian likelihoods or posteriors. It computes a posterior operator $C_{Y|X}$ when given only likelihood operators (e.g. $C_{X|Y}$) and prior operators (e.g. C_{YY}). The DMO is appealing as all operators involved are of second order and the same second order likelihood and prior operators are used for both the joint and evidence operator.

However, because C'_{XX} is not necessarily the same as C_{XX}, the DMO $C'_{X|Y} = (C_{XY})^T(C'_{XX})^{-1}$ is not necessarily the posterior operator $C_{Y|X} = (C_{XY})^T(C_{XX})^{-1}$. Nevertheless, under certain conditions they coincide with each other.

Theorem 5.8. *If $C_{X|Y}C_{Y|X}C_{XX} = C_{XX}$, then $C'_{XX} = C_{XX}$ and $C'_{X|Y} = C_{Y|X}$.*

A special instance where the assumptions are met is when $X = r(Y)$ where r is not necessarily invertible. Importantly, for empirical DMOs, having $x_i = x_j$ for any $y_i = y_j$ suffices, which can be achieved if all y_i are unique. Furthermore, empirical DMOs $\bar{C}'_{X|Y}$ can be seen as generalizations of empirical CMOs $\hat{C}_{Y|X}$ in the other direction.

Theorem 5.9. *If $m = n$ and $\tilde{y}_i = y_i$ for all $i \in \{1, \ldots, n\}$, then the empirical DMO corresponding to $\mathbb{P}_{X|Y}$ becomes the empirical CMO corresponding to $\mathbb{P}_{Y|X}$ for $\lambda \to 0^+$,*

$$\lim_{\lambda \to 0^+} \bar{C}'_{X|Y} = \Psi \big[K + n\epsilon I\big]^{-1} \Phi^T = \hat{C}_{Y|X}. \tag{5.12}$$

Intuitively, suppose $\{x_i, y_i\}_{i=1}^n$ are from $p(x, y) := p(x|y)p(y)$ and $\{\tilde{y}_j\}_{j=1}^m = \{y_i\}_{i=1}^n$ is from $p(y)$, then the DMO of $p(x|y)$ is equivalent to the CMO of $p(y|x) = p(x|y)p(y)/\int_{\mathcal{Y}} p(x|y)p(y)dy$ in the other direction, as per theorem 5.9. In general, however, if $\{x_i, y_i\}_{i=1}^n$ are from $q(x, y) := p(x|y)q(y)$ and $\{\tilde{y}_j\}_{j=1}^m$ is from $p(y)$, then using the joint samples from $q(x, y)$ only to build the CMO will yield the CMO of $q(y|x) = p(x|y)q(y)/\int_{\mathcal{Y}} p(x|y)q(y)dy$, while using both the joint samples from $q(x, y)$ and marginal samples from $p(y)$ to build the DMO will yield the CMO corresponding to $p(y|x) = p(x|y)p(y)/\int_{\mathcal{Y}} p(x|y)p(y)dy$.

Both the usual and nonparametric Bayes' rule are derived to reverse the relationship specified by the likelihood (density or operator, resp.) by matching the joint. In both cases, the prior (density or operator, resp.) is inevitably required to perform this computation.

Consider the derivation for Bayes' rule. When given a *forward* density $p(x|y)$ and a marginal density on its conditioned variable $p(y)$ which specifies a joint $p(x, y) = p(x|y)p(y)$, we seek a *backward* density $q(y|x)$ and a marginal density $q(x)$ that would yield the same joint $q(y|x)q(x) = p(x, y) = p(x|y)p(y)$. It is only when applying $\int_{\mathcal{Y}} \cdot dy$ on both sides, requiring that $q(y|x)$ is a density, that we have $q(x) = \int_{\mathcal{Y}} p(x|y)p(y)dy$ and thus Bayes' rule.

Similarly, when given a *forward* CMO $C_{X|Y} : \mathcal{H}_\ell \to \mathcal{H}_k$ and a symmetric operator $C_{YY} : \mathcal{H}_\ell \to \mathcal{H}_\ell$ on its conditioned variable which specifies a joint $C_{XY} = C_{X|Y}C_{YY}$, we seek a *backward* operator $D_{Y|X} : \mathcal{H}_k \to \mathcal{H}_\ell$ and a symmetric operator $D_{XX} : \mathcal{H}_k \to \mathcal{H}_k$ that would yield the same joint $D_{Y|X}D_{XX} = C_{YX} = (C_{XY})^T = (C_{X|Y}C_{YY})^T$. Without further requirement we see that $D_{Y|X} = C'_{X|Y}$ (5.7) and $D_{XX} = C'_{XX}$ is one solution. It is only when applying $C_{X|Y}$ on both sides, requiring the assumption of theorem 5.8, that we have $D_{Y|X} = C_{Y|X}$ and $D_{XX} = C_{XX}$ and thus a nonparametric Bayes' rule.

TABLE 5.1: Empirical estimators for DMO and KBR. We use the shorthand $A := (L + n\lambda I)^{-1}\tilde{L}$ and $D := \mathrm{diag}(A\mathbf{1})$.

Method	Joint Operator $\bar{C}_{XY}$	Evidence Operator $\bar{C}_{XX}$ or $\bar{C}'_{XX}$	Posterior Operator $\bar{C}_{Y\|X}$ or $\bar{C}'_{X\|Y}$	Computational Form $\bar{C}_{Y\|X}$ or $\bar{C}'_{X\|Y}$
DMO	$\hat{C}_{X\|Y}\tilde{C}_{YY}$	$\hat{C}_{X\|Y}\tilde{C}_{YY}(\hat{C}_{X\|Y})^T$	$(\bar{C}_{XY})^T(\bar{C}'_{XX} + \epsilon I)^{-1}$	$\tilde{\Psi}\left[A^T K A + m\epsilon I\right]^{-1} A^T \Phi^T$
DMO(W)	$\hat{C}_{X\|Y}\tilde{C}_{YY}$	$\hat{C}_{X\|Y}\tilde{C}_{YY}(\hat{C}_{X\|Y})^T$	$(\bar{C}_{XY})^T(\bar{C}'_{XX} + \epsilon I)^{-1}$	$\tilde{\Psi}A^T\left[KAA^T + m\epsilon I\right]^{-1}\Phi^T$
KBR(a)-I	$\hat{C}_{X\|Y}\tilde{C}_{YY}$	$\hat{C}_{XX\|Y}\tilde{\mu}_Y$	$(\bar{C}_{XY})^T(\bar{C}_{XX} + \epsilon I)^{-1}$	$\tilde{\Psi}A^T\left[KD + m\epsilon I\right]^{-1}\Phi^T$
KBR(a)-II	$\hat{C}_{X\|Y}\tilde{C}_{YY}$	$\hat{C}_{XX\|Y}\tilde{\mu}_Y$	$(\bar{C}_{XY})^T(\bar{C}^2_{XX} + \epsilon I)^{-1}\bar{C}_{XX}$	$\tilde{\Psi}A^T\left[(KD)^2 + m^2\epsilon I\right]^{-1} KD\Phi^T$
KBR(b)-I	$\hat{C}_{XY\|Y}\tilde{\mu}_Y$	$\hat{C}_{XX\|Y}\tilde{\mu}_Y$	$(\bar{C}_{XY})^T(\bar{C}_{XX} + \epsilon I)^{-1}$	$\Psi D\left[KD + m\epsilon I\right]^{-1}\Phi^T$
KBR(b)-II	$\hat{C}_{XY\|Y}\tilde{\mu}_Y$	$\hat{C}_{XX\|Y}\tilde{\mu}_Y$	$(\bar{C}_{XY})^T(\bar{C}^2_{XX} + \epsilon I)^{-1}\bar{C}_{XX}$	$\Psi D\left[(KD)^2 + m^2\epsilon I\right]^{-1} KD\Phi^T$

5.7 Connections to Kernel Bayes' Rule

Bayesian inference often requires computation of the posterior $\mathbb{P}_{Y|X}$ when given the likelihood $\mathbb{P}_{X|Y}$ and the prior $\mathbb{P}_Y$. When density evaluations exist, the Bayes' rule provides their relationship as $p_{Y|X}(\cdot|x) = \frac{p_{X|Y}(x|\cdot)p_Y(\cdot)}{\int_{\mathcal{Y}} p_{X|Y}(x|y)p_Y(y)dy}$.

Nevertheless, several levels of intractability may arise. The first is when both likelihood and prior density evaluations are tractable but the evidence integral $\int_{\mathcal{Y}} p_{X|Y}(x|y)p_Y(y)dy$ is intractable, leading to literatures such as VI [Blei et al., 2017] and MCMC [Hastings, 1970]. The next is when only likelihood evaluations are intractable but sampling is possible, leading literatures such as LFI and ABC [Marin et al., 2012]. More rarely, only prior evaluations are intractable but available via sampling, leading to literatures in implicit priors. The last is when both the likelihood and prior evaluations are intractable but available via sampling, leading to newer literatures such as implicit generative models.

While there are many approaches that addresses posterior approximations in each of these scenarios, the underlying limitation that is shared across all these settings is that Bayes' rule requires density evaluations that are difficult to approximate in high dimensions from samples. Instead, if relationships between the posterior, likelihood, and prior can be captured without using density evaluations, but directly by using samples, this issue could be more naturally sidestepped. Both DMOs and KBR provide such a nonparametric Bayes' rule. This is especially useful for performing inference on implicit models such as those expressed as simulators, since implicit distributions generate samples directly.

DMOs have strong connections to KBR. Both provide a nonparametric Bayes' rule under the KME framework. In contrast to DMOs where both likelihood and prior operators are of second order, both KBR(a) and KBR(b) [Fukumizu et al., 2013, Song et al., 2013] use a third order likelihood mean operator $C_{XX|Y}$ and a first order prior mean embedding μ_Y for the evidence mean operator $C_{XX} = C_{XX|Y}\mu_Y$. KBR(b) further uses a different third order likelihood mean operator $C_{XY|Y}$ for the joint mean operator $C_{XY} = C_{XY|Y}\mu_Y$. Consequently, KBR becomes sensitive

TABLE 5.2: Degenerate case for empirical estimators when $\epsilon = 0$

DMO(W)	KBR(a)	KBR(b)
$\tilde{\Psi}A^T\left[AA^T\right]^{-1}K^{-1}\Phi^T$	$\tilde{\Psi}A^TD^{-1}K^{-1}\Phi^T$	$\Psi K^{-1}\Phi^T$

to inverse regularizations and effects of prior samples $\tilde{\mathbf{y}}$ can vanish. For instance, when $\epsilon \to 0^+$, KBR(b) degenerate to $\bar{C}_{Y|X} = \Psi K^{-1}\Phi^T$, which is a CMO that no longer depend on $\tilde{\mathbf{y}}$. Instead, DMOs degenerate to $\bar{C}'_{X|Y} = \tilde{\Psi}A^T\left[AA^T\right]^{-1}K^{-1}\Phi^T$, retaining their original structure.

Importantly, it is only when DMOs and KBR are viewed as a statement of the relationship between X and Y that they are seen as nonparametric versions of the Bayes' rule. However, DMOs and KBR are not Bayesian models with respect to the task of inferring deconditional mean or conditional means. This is because both models only infer point estimates for the deconditional or conditional mean, and no measure of uncertainty in the inferred function is provided.

Table 5.1 compares all four forms of KBR [Song et al., 2013] with DMO. This table illustrates the different ways each method estimates the joint and evidence operators from likelihood and prior operators, the type of regularization used for inverting the evidence operator, and the final computational form. For KBR, (a) and (b) differ in the joint operator, and I and II differ in the type of regularization used for inverting the evidence operator. Via the Woodbury identity, for DMO we also show an alternative computational form DMO(W) that better illustrate its contrast with KBR(a)-I and KBR(b)-I. Note that unlike the four types of KBR, DMO(W) is the same model as DMO, just with a different computational form.

In particular, the diagonal matrix $D := \text{diag}(A\mathbf{1})$ arises from the use of third order operators. This can make estimators sensitive to regularizations on inverse operators. This is best seen in the degenerate case of $\epsilon \to 0^+$, shown in table 5.2, where for KBR(b) the effect of $\tilde{\mathbf{y}}$ vanishes, even though ϵ does not correspond to regularizations from the prior.

Furthermore, the original computational form of DMOs involves the inverse of a positive definite matrix. This however is not true for KBR(a) and KBR(b) since KD is not symmetric and thus the resulting matrix to be inverted cannot be positive definite. For KBR(b), by using $D = D^{\frac{1}{2}}D^{\frac{1}{2}}$ and the Woodbury identity, KBR(b)-I and KBR(b)-II can be written in forms with symmetric matrix inverses as $\bar{C}_{Y|X} = \Psi D^{\frac{1}{2}}\left[D^{\frac{1}{2}}KD^{\frac{1}{2}} + m\epsilon I\right]^{-1}D^{\frac{1}{2}}\Phi^T$ and $\bar{C}_{Y|X} = \Psi D^{\frac{1}{2}}\left[D^{\frac{1}{2}}KDKD^{\frac{1}{2}} + m^2\epsilon I\right]^{-1}D^{\frac{1}{2}}KD\Phi^T$ respectively. However, it is difficult to interpret this form.

Finally, similar to theorem 5.9, for the other degenerate case where $m = n$, $\tilde{\mathbf{y}} = \mathbf{y}$, and $\lambda \to 0^+$, all estimators revert to a CME $\hat{C}_{Y|X} = \Psi(K + n\epsilon I)^{-1}\Phi^T$.

5.8 Further Connections

Viewing KMEs as regressors provides valuable insights and interpretations to the framework. CMOs can be established as regressors where the vector-valued targets are also kernel induced features [Grünewälder et al., 2012]. In contrast, we establish DMOs as solutions to task transformed regressors which recover latent functions that, together with a likelihood, governs interactions between three variables.

The TTR problem describes the setting of learning from conditional distributions in the extreme case where only one sample of x_i is available for each y_i to describe $p(x|y)$. Dual KMEs [Dai et al., 2017] formulate this setting as a saddle point problem, and employ stochastic approximations to efficiently optimize over the function space. However, without connections to Bayesian models such as TTGPs that admit a marginal likelihood, hyperparameter selection often require inefficient grid search.

Hyperparameter learning of marginal embeddings have been investigated by placing GP priors on the embedding itself to yield a marginal likelihood objective [Flaxman et al., 2016]. However, it is unclear how this can be extended to CMEs. Our marginal likelihoods (theorem 5.7 and theorem 5.10) provide such objective for DMEs and, due to theorem 5.9, it can also be applied to CMEs as a special case.

5.9 Applications and Experiments

While DMEs are developed to complement the theoretical framework of KMEs, in this section we describe and demonstrate some of their practical applications with experiments.

5.9.1 Hyperparameter Learning for TTR

We first illustrate in figure 5.3 the TTR problem, the primary application of TTGPs and DMEs. While X and Y are multivariate in general, we use 1D examples to enable visualizations. Although this is a 1D problem, the simulation process $p(x|y)$ and observation process $p(z|y)$ are governed by non-trivial relationships where successful recovery of f requires dealing with difficult multimodalities in $p(y|x)$. To generate the data, we choose non-trivial functions r and f and generate $X_i = r(Y_i) + \eta_i$ and $\tilde{Z}_j = f(r(\tilde{Y}_j) + \tilde{\eta}_j) + \tilde{\xi}_j$, where $Y_i, \tilde{Y}_j \sim U(-6, 6)$ and $\eta_i, \tilde{\eta}_j, \tilde{\xi}_j \sim \mathcal{N}(0, 0.25^2)$ for all $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. In this

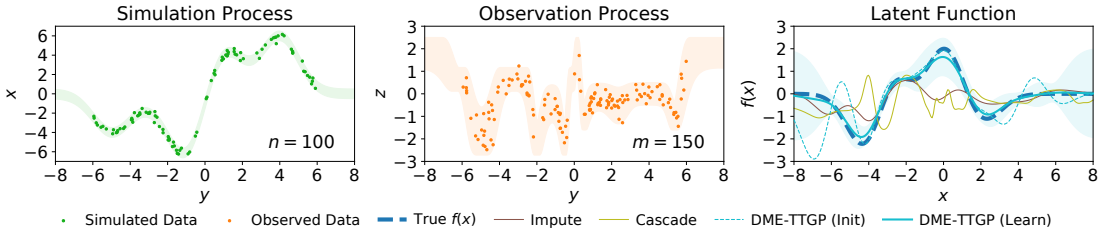

FIGURE 5.3: Illustration of latent function recovery with a task transformed Gaussian process (TTGP) on non-trivial simulation and observation processes $p(x|y)$ and $p(z|y)$. (Left) The simulation process $p(x|y)$. (Center) The observation process $p(z|y)$. (Right) The true latent function f, the naive solutions using cascaded regressors and imputed data, and the mean and uncertainty bounds of the TTGP, also the Bayesian DME, with initial and learned hyperparameters. All bounds are 2 standard deviations from the mean.

way, $p(x|y) = \mathcal{N}(x; r(y), 0.25^2)$, $p(z|x) = \mathcal{N}(z; f(x), 0.25^2)$, and $\mathbb{E}[Z|Y = y] = \mathbb{E}[f(r(y) + \tilde{\eta}) + \tilde{\xi}] = \mathbb{E}[f(X)|Y = y]$.

By optimizing the marginal likelihood in theorem 5.7 (3), we see that the DME is able to adapt from its initial hyperparameters to learn the latent function accurately.

We compare this to two naive solutions that one may propose when faced with a TTR problem. The *cascade* method trains separate regressors from X to Y, with the transformation set, and from Y to Z, with the task set. They use the former to predict $y^\star$ from $x^\star$ and the latter to predict $z^\star$ from $y^\star$. The *impute method* trains a regressor from Y to Z with the task set and predicts $\mathbf{z}_{\text{fake}}$ at locations $\mathbf{y}$, and trains a regressor on the dataset $(\mathbf{x}, \mathbf{z}_{\text{fake}})$ to predict $z^\star$ from a new $x^\star$. We use GPR means (KRR) for all such regressors. Both methods suffer because uncertainty propagation is lost by training regressors separately. The cascade method suffers further because $p(y|x)$ is usually highly multi-modal such as in this example, so unimodal regressors like GPR from X to Y are unsuitable. This also highlights that while DMEs provides unimodal Gaussian uncertainty on function evaluations and thus Z, they capture multi-modality as a nonparametric Bayes' rule between X and Y.

5.9.2 Sparse Representation Learning with TTGP

A special case of the TTR problem is to learn sparse representations for big data with trainable inducing points. Continuing with the notations used so far, we are given a large original dataset $(\tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ of size m, with inputs Y and target Z. We let the transformation dataset be a set of n inducing points $\mathbf{x} = \mathbf{y} = \mathbf{u}$ for Y where $n << m$. That is, we degenerate to $X = Y$ and $\mathcal{X} = \mathcal{Y}$. We maximize the alternative marginal likelihood in theorem 5.7 (4) with respect to the inducing

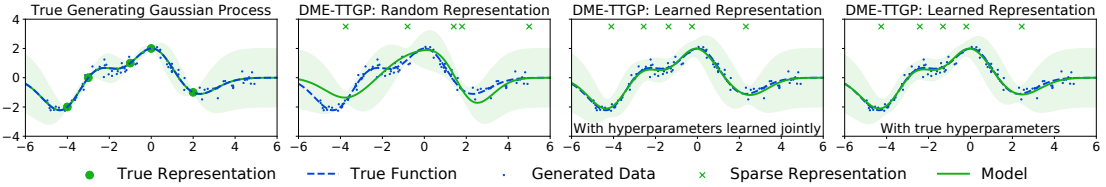

FIGURE 5.4: Sparse representation learning on a dataset of 100 points generated by using the toy process from Rasmussen and Williams [2006] as ground truth. (Left) The true function, represented exactly by a GP mean using 5 points. (Left Center) DME using 5 random points. (Right Center) DME with the 5 points and all hyperparameters learned jointly via its marginal likelihood. (Right) DME with the 5 points learned via its marginal likelihood under true hyperparameters. The vertical position of sparse representations has no meaning.

points and learn the TTGP hyperparameters jointly. For predictive mean we use the alternative computational form in definition 5.7. Similar form exists for the covariance. These alternative forms are suitable for this application because n is small for its $O(n^3)$ inversions and dependence on m is only $O(m)$. We illustrate this process in figure 5.4.

5.9.3 Automatic Likelihood-Free Inference with DME

As a nonparametric Bayes' rule, DMEs can be used for LFI [Marin et al., 2012] where likelihood evaluations are intractable but sampling from a simulator $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$ is possible. The simulator takes parameters $\boldsymbol{\theta}$ and stochastically generates simulated data that are often summarized into statistics $\mathbf{x}$. Observed data are also summarized into statistics $\mathbf{y}$, and discrepancies with $\mathbf{x}$ are often measured by an ϵ-kernel $\kappa_\epsilon(\mathbf{y}, \mathbf{x}) = p_\epsilon(\mathbf{y}|\mathbf{x})$ such that $p_\epsilon(\mathbf{y}|\boldsymbol{\theta}) = \int p_\epsilon(\mathbf{y}|\mathbf{x})p(\mathbf{x}|\boldsymbol{\theta})d\boldsymbol{\theta}$. This ϵ is not to be confused with the regularization used for C'_{XX}, which we denote as δ for this section only. After selecting a prior $p(\boldsymbol{\theta})$, the goal is to approximate the posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$.

Translating notations into the LFI setting, we have $x_i \to \mathbf{x}_i$, $y_i \to \boldsymbol{\theta}_i$, $\tilde{y}_j \to \tilde{\boldsymbol{\theta}}_j$, and $x \to \mathbf{y}$. We first simulate $\mathbf{x}_i \sim p(\mathbf{x}|\boldsymbol{\theta}_i)$ on parameters $\{\boldsymbol{\theta}_i\}_{i=1}^n \sim \pi(\boldsymbol{\theta})$ not necessarily from the prior to get $\{\boldsymbol{\theta}_i, \mathbf{x}_i\}_{i=1}^n$ for the likelihood, and sample $\{\tilde{\boldsymbol{\theta}}_j\}_{j=1}^m \sim p(\boldsymbol{\theta})$ for the prior. We then build the DME $\bar{\mu}_{\boldsymbol{\Theta}|\mathbf{X}=\mathbf{y}}$ and sample it with kernel herding [Chen et al., 2010] for posterior super-samples. This is described in algorithm 4.

Crucially, leveraging the insights gained from the KELFI framework, we can approximate the MKML empirically using prior samples and use it as a hyperparameter learning objective for DMEs in the context of LFI. Recall that the MKML approximates the marginal likelihood of the overall inference problem. We provide

Algorithm 4 Deconditional mean embeddings for likelihood-free inference

1: **Input:** Data $\mathbf{y}$, simulations $\{\boldsymbol{\theta}_i, \mathbf{x}_i\}_{i=1}^n \sim p(\mathbf{x}|\boldsymbol{\theta})\pi(\boldsymbol{\theta})$, prior samples $\{\tilde{\boldsymbol{\theta}}_j\}_{j=1}^m \sim p(\boldsymbol{\theta})$, query points $\{\boldsymbol{\theta}_r^\star\}_{r=1}^R$, kernels k, κ_ϵ, ℓ, and ℓ', regularization λ and δ
2: $L \leftarrow \{\ell(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)\}_{i,j=1}^{n,n}$, $\tilde{L} \leftarrow \{\ell\boldsymbol{\theta}_i, \tilde{\boldsymbol{\theta}}_j)\}_{i,j=1}^{n,m}$
3: $A \leftarrow (L + n\lambda I)^{-1}\tilde{L}$, $\tilde{L}^\star \leftarrow \{\ell(\tilde{\boldsymbol{\theta}}_j, \boldsymbol{\theta}_r^\star)\}_{j,r=1}^{m,R}$,
4: $K \leftarrow \{k(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1}^{n,n}$, $\boldsymbol{k}(\mathbf{y}) \leftarrow \{k(\mathbf{x}_i, \mathbf{y})\}_{i=1}^n$
5: DME: $\boldsymbol{\mu} \leftarrow (\tilde{L}^\star)^T A^T \left[KAA^T + m\delta I\right]^{-1} \boldsymbol{k}(\mathbf{y}) \in \mathbb{R}^R$
6: **for** $s \in \{1, \ldots, S\}$ with $\mathbf{a} \leftarrow \mathbf{0} \in \mathbb{R}^R$ initialized **do**
7: $\quad \hat{\boldsymbol{\theta}}_s \leftarrow \boldsymbol{\theta}_{r^\star}^\star$ where $r^\star \leftarrow \arg\max_r \mu_r - (a_r/s)$
8: $\quad \mathbf{a} \leftarrow \mathbf{a} + \{\ell'(\boldsymbol{\theta}_r^\star, \hat{\boldsymbol{\theta}}_s)\}_{r=1}^R$
9: **end for**
10: **Output:** Posterior super-samples $\{\hat{\boldsymbol{\theta}}_s\}_{s=1}^S$
11: **Learning:** $\bar{q} \leftarrow \text{mean}(A^T\boldsymbol{\kappa}_\epsilon)$, $\boldsymbol{\kappa}_{\boldsymbol{\epsilon}} \leftarrow \{\kappa_\epsilon(\mathbf{y}, \mathbf{x}_i)\}_{i=1}^n$

such an approximate marginal likelihood objective $\bar{q}$ to maximize for hyperparameter learning of the DME in line 11.

Theorem 5.10 (Approximate Marginal Likelihood for LFI)**.** *Assume $\kappa_\epsilon(\mathbf{y}, \cdot) \in \mathcal{H}_k$ and that $\hat{C}_{\mathbf{X}|\boldsymbol{\Theta}}$ is a bounded operator for all n. Denote $\boldsymbol{\kappa}_\epsilon(\mathbf{y}) = \{\kappa_\epsilon(\mathbf{y}, \mathbf{x}_i)\}_{i=1}^n$ and $\mathbf{1}_m = \{1\}_{j=1}^m$, then $\bar{q}(\mathbf{y}) := \langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\boldsymbol{\Theta}}\tilde{\mu}_{\boldsymbol{\Theta}}\rangle_{\mathcal{H}_k} = \frac{1}{m}\boldsymbol{\kappa}_\epsilon^T A\mathbf{1}_m$ is an estimator to the marginal likelihood $p_\epsilon(\mathbf{y})$ and converge at $O_p(m^{-\frac{1}{2}} + (n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$.*

In a similar fashion to KELFI, to satisfy $\kappa_\epsilon(\mathbf{y}, \cdot) \in \mathcal{H}_k$, we use $\kappa_\epsilon(\mathbf{y}, \mathbf{x}) = p_\epsilon(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}; \mathbf{x}, \epsilon^2 I)$ and Gaussian kernel for k with length scale ϵ, so that κ_ϵ is just the normalized version of the reproducing kernel k.

Importantly, while the approximate marginal likelihood $\bar{q}(\mathbf{y})$ depends on the hyperparameters of the kernels k and ℓ and the regularization λ, it does not depend on ϵ. At first, it seems that this objective cannot help us learn ϵ. Fortunately, due to points (4) and (5) of theorem 5.7, we have that a good proxy for setting ϵ once λ is learned is $\epsilon = \frac{n}{m}\lambda$. Nevertheless, for simplicity in our experiments we optimize all kernel hyperparameters and keep the regularization hyperparameters fixed, which has already achieved sufficiently accurate results.

Figure 5.5 demonstrates algorithm 4 two standard benchmarks, exponential-gamma problem and the Lotka-Volterra problem, adding onto the experimental results of section 4.9.

5.9.3.1 Exponential-Gamma

The toy exponential-gamma problem is a standard benchmark for likelihood-free inference, where the true posterior $p_\epsilon(\boldsymbol{\theta}|\mathbf{y})$ is known and tractable even for $\epsilon = 0$. We follow the experimental setup described in section 4.9.1.

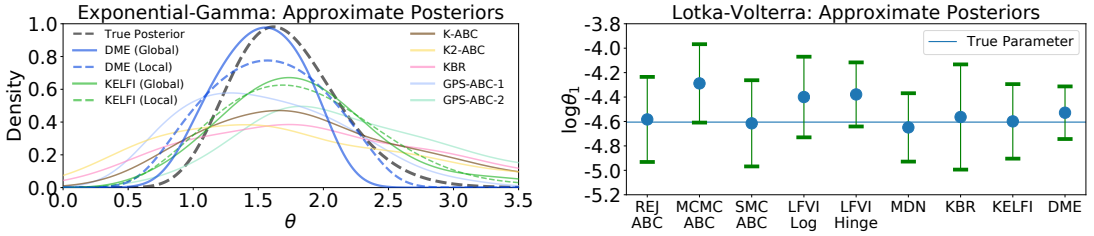

FIGURE 5.5: Application to LFI. (Left) Approximate posteriors under kernel based LFI methods for the toy exponential-gamma problem using 100 simulations. 'Global' and 'Local' refer to the optimality of model hyperparameters with respect to their respective approximate marginal likelihoods. (Right) Approximate posteriors of the first (log) parameter. Error bars represent the middle 95% credible interval.

For the exponential-gamma problem, we compare directly with other kernel approaches. Of the kernel based methods that we have benchmarked against, K-ABC [Nakagome et al., 2013], K2-ABC [Park et al., 2016], KBR [Fukumizu et al., 2013], and KELFI are also LFI methods based on the KME framework. Consequently, they are very suitable for comparisons towards DME. For all these methods, we apply kernel herding on their posterior embeddings to get posterior samples, and plot the approximate posterior density in figure 5.5 (left) using KDE on the posterior samples. In contrast, GPS-ABC [Meeds and Welling, 2014] has its own adaptive MCMC based sampling algorithm. We set a simulation budget of 200 simulations and run it until either 10000 posterior samples are generated or the simulation budget is reached.

For hyperparameters, we used standard median heuristic for K-ABC, K2-ABC, and KBR. In contrast, DME and KELFI have their own marginal likelihoods for hyperparameter learning. For both cases, we find global and local optimums of the marginal likelihood for the hyperparameters and show their results, emphasizing that maximizing the marginal likelihood objective produces better inference results. The hyperparameters of the GP surrogate itself used in GPS-ABC are learned by maximizing the marginal likelihood of the GPR [Rasmussen and Williams, 2006]. However, for hyperparameters of GPS-ABC that are not part of the surrogate, we select them based on the original paper [Meeds and Welling, 2014]. We then report its best two results.

As simulations are usually very expensive, we show the case with very limited simulations ($n = 100$), leading to most methods producing posteriors wider than the ground truth. Nevertheless, by optimizing $\bar{q}$ in line 11, DMEs can adapt their kernel length scales accordingly.

5.9.3.2 Lotka-Volterra

The Lotka-Volterra simulator describes the population dynamics of a well known predator-prey system. For most parameters, the simulation produces chaotic behavior. Realistic scenarios with oscillatory behavior appears only for a small set of parameters. Consequently, inference on this simulator is extremely challenging.

We follow the setup described in section 4.9.3. There are 4 parameters and 9 normalized summary statistics. We place the same uniform prior on the log parameters and use the same ground truth parameters. After performing inference on all four parameters, we similarly show in figure 5.5 (right) the marginal posterior distribution for $\log \theta_1$ in the same format as Papamakarios and Murray [2016] and Tran et al. [2017a].

For KBR [Fukumizu et al., 2013], KELFI, and DME, we again sample their posterior mean embeddings with kernel herding to get 10000 posterior samples. Finally, to compute the 95% interval, we compute the empirical 2.5% quantile and 97.5% quantiles on marginal samples of $\log \theta_1$ from the 10000 posterior samples. For MDN [Papamakarios and Murray, 2016] and the two LFVI methods [Tran et al., 2017a], we report the results from the original source, as well as their results for REJ-ABC, Markov chain Monte carlo ABC (MCMC-ABC), and SMC-ABC.

For Lotka-Volterra, the ABC methods used more than 100000 simulations, while MDN used 10000 simulations. To achieve competitive accuracy, kernel approaches such as DMEs, KELFI, and KBR used 2000, 2500, and 2500 simulations.

5.10 Summary and Future Work

The connections of DMEs with CMEs and GPs produce useful insights towards the KME framework, and are important steps towards establishing Bayesian views of KMEs. DMEs are novel solutions to a class of nonparametric Bayesian regression problems and enable applications such as sparse representation learning and LFI.

There are multiple possible dimensions for future work. From section 5.6, relaxing assumptions required for DMOs as a nonparametric Bayes' rule can have fruitful theoretical and practical implications. From section 5.5, the formulation of the TTR problem is general and open doors to more development. In this work we posited linear models or RKHS solutions to the TTR problem to obtain variants of the DME. Nevertheless, there is potential for using the TTR losses to train other model architectures, such as neural networks. Furthermore, just as sparse representation learning is a special case of the TTR problem, supervised dimensionality reduction could possibly be achieved in the special case when X has much lower dimensionality than Y, and is worthwhile to investigate.

5.11 Supporting Proofs for Section 5.3

Proof of Theorem 5.1. Let $f \in \mathcal{H}_k$ and $g(y) := \mathbb{E}[f(X)|Y = y]$. Assuming $g \in \mathcal{H}_\ell$, then $C_{YY}g = C_{YX}f$ [Fukumizu et al., 2004], so that

$$\begin{aligned} g &= C_{YY}^{-1}C_{YX}f \\ &= ((C_{YX})^T C_{YY}^{-1})^T f \\ &= (C_{XY}C_{YY}^{-1})^T f, \end{aligned} \tag{5.13}$$

where the inverse C_{YY}^{-1} exists because $\ell(y,\cdot)$ is assumed to be in the image of C_{YY} so that any $g \in \mathcal{H}_\ell$ is also in the image. Hence, $C_{XY}C_{YY}^{-1}$ satisfies the definition of a CMO. □

Proof of Lemma 5.1. Each of the following statements are equivalent to each other.

$$\begin{aligned} & (C_{X|Y})^T f = \mathbb{E}[f(X)|Y=\cdot], \quad \forall f \in \mathcal{H}_k \\ \iff & \langle \ell(y,\cdot), (C_{X|Y})^T f\rangle_{\mathcal{H}_\ell} = \langle \ell(y,\cdot), \mathbb{E}[f(X)|Y=\cdot]\rangle_{\mathcal{H}_\ell}, \quad \forall f \in \mathcal{H}_k, \quad \forall y \in \mathcal{Y} \\ \iff & \langle C_{X|Y}\ell(y,\cdot), f\rangle_{\mathcal{H}_k} = \mathbb{E}[f(X)|Y=y], \quad \forall f \in \mathcal{H}_k, \quad \forall y \in \mathcal{Y} \\ \iff & \langle C_{X|Y}\ell(y,\cdot), f\rangle_{\mathcal{H}_k} = \langle \mathbb{E}[k(X,\cdot)|Y=y], f\rangle_{\mathcal{H}_k}, \quad \forall f \in \mathcal{H}_k, \quad \forall y \in \mathcal{Y} \\ \iff & C_{X|Y}\ell(y,\cdot) = \mathbb{E}[k(X,\cdot)|Y=y], \quad \forall y \in \mathcal{Y}. \end{aligned} \tag{5.14}$$

Consequently, the first and last statements are equivalent. □

Proof of Theorem 5.2. We show that the empirical CMO can be written as (5.5). We use a special case of the Woodbury identity [Higham, 2002], $B(CB + \lambda I)^{-1} = (BC + \lambda I)^{-1}B$, where B and C are appropriately defined operators, such matrices with the correct shapes. Using the empirical forms for the cross-covariance operators, we have

$$\begin{aligned} \hat{C}_{X|Y} &:= \hat{C}_{XY}(\hat{C}_{YY} + \lambda I)^{-1} \\ &= \frac{1}{n}\Phi\Psi^T(\frac{1}{n}\Psi\Psi^T + \lambda I)^{-1} \\ &= \Phi\Psi^T(\Psi\Psi^T + n\lambda I)^{-1} \\ &= \Phi(\Psi^T\Psi + n\lambda I)^{-1}\Psi^T \\ &= \Phi(L + n\lambda I)^{-1}\Psi^T. \end{aligned} \tag{5.15}$$

□

5.12 Supporting Proofs for Section 5.4

Proof of Theorem 5.3. Let $f \in \mathcal{H}_k$ and $g(y) := \mathbb{E}[f(X)|Y = y]$, then from definition 5.2 we have

$$\begin{aligned} g &= (C_{X|Y})^T f \\ C_{XY} g &= C_{XY}(C_{X|Y})^T f \\ C_{X|Y} C_{YY} g &= C_{X|Y} C_{YY} (C_{X|Y})^T f \\ (C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1} C_{X|Y} C_{YY} g &= f \\ ((C_{X|Y} C_{YY})^T (C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1})^T g &= f, \end{aligned} \tag{5.16}$$

where the inverse $(C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1}$ exists because $C_{XY} g \in \mathcal{H}_k$ and $k(x, \cdot) \in \text{image}(C_{X|Y} C_{YY} (C_{X|Y})^T)$ so that $C_{XY} g$ for any $g \in \mathcal{H}_\ell$ is also in the image. In the last line we also used the fact that $(C_{X|Y} C_{YY} (C_{X|Y})^T)^T = C_{X|Y} C_{YY} (C_{X|Y})^T$ is symmetric since $(C_{YY})^T = C_{YY}$. Hence, $(C_{X|Y} C_{YY})^T (C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1}$ satisfies the definition of a DMO. The assumption $\ell(y, \cdot) \in \text{image}(C_{YY})$ is required so that the original CMO exists and is unique. □

Proof of Theorem 5.4. Since $\ell(y, \cdot) \in \text{image}(C_{YY})$ for all $y \in \mathcal{Y}$ and $k(x, \cdot) \in \text{image}(C_{X|Y} C_{YY} (C_{X|Y})^T)$ for all $x \in \mathcal{X}$, we have that C_{YY}^{-1} exists so that $C_{X|Y}$ is unique and $(C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1}$ exists so that $C'_{X|Y}$ is unique. Due to theorem 5.3 we have $C'_{X|Y} = (C_{X|Y} C_{YY})^T (C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1}$. Since $C_{X|Y} C_{YY} (C_{X|Y})^T$ is at least positive semi-definite and invertible we can write $(C_{X|Y} C_{YY} (C_{X|Y})^T)^{-1} = \lim_{\epsilon \to 0^+} (C_{X|Y} C_{YY} (C_{X|Y})^T + \epsilon I)^{-1}$,

$$\begin{aligned} C'_{X|Y} &= \lim_{\epsilon \to 0^+} (C_{X|Y} C_{YY})^T (C_{X|Y} C_{YY} (C_{X|Y})^T + \epsilon I)^{-1} \\ &= \lim_{\epsilon \to 0^+} (C_{X|Y} C_{YY})^T (C_{X|Y} (C_{X|Y} C_{YY})^T + \epsilon I)^{-1} \\ &= \lim_{\epsilon \to 0^+} (C_{XY})^T (C_{X|Y} (C_{XY})^T + \epsilon I)^{-1} \\ &= \lim_{\epsilon \to 0^+} C_{YX} (C_{X|Y} C_{YX} + \epsilon I)^{-1} \\ &= \lim_{\epsilon \to 0^+} (C_{YX} C_{X|Y} + \epsilon I)^{-1} C_{YX} \\ &= \lim_{\epsilon \to 0^+} (C_{YY} C_{YY}^{-1} C_{YX} C_{X|Y} + \epsilon C_{YY} C_{YY}^{-1})^{-1} C_{YX} \\ &= \lim_{\epsilon \to 0^+} (C_{YY}^{-1} C_{YX} C_{X|Y} + \epsilon C_{YY}^{-1})^{-1} C_{YY}^{-1} C_{YX} \\ &= \lim_{\epsilon \to 0^+} (C_{YY}^{-1} C_{YX} C_{X|Y} + \epsilon C_{YY}^{-1})^{-1} C_{YY}^{-1} C_{YX} \\ &= \lim_{\epsilon \to 0^+} ((C_{XY} C_{YY}^{-1})^T C_{X|Y} + \epsilon C_{YY}^{-1})^{-1} (C_{XY} C_{YY}^{-1})^T \\ &= \lim_{\epsilon \to 0^+} ((C_{X|Y})^T C_{X|Y} + \epsilon C_{YY}^{-1})^{-1} (C_{X|Y})^T \\ &= ((C_{X|Y})^T C_{X|Y})^{-1} (C_{X|Y})^T =: C_{X|Y}^\dagger. \end{aligned} \tag{5.17}$$

In line 6 we used the Woodbury identity [Higham, 2002]. In the last line, the limit exists as $((C_{X|Y})^T C_{X|Y})^{-1}$ exists. □

Proof of Theorem 5.5. We show that the empirical DMO can be written as (5.10). From definition 5.3 and theorem 5.2, the likelihood operator is estimated from $\{x_i, y_i\}_{i=1}^n$ as

$$\hat{C}_{X|Y} := \hat{C}_{XY}(\hat{C}_{XX} + \lambda I)^{-1} = \Phi(L + n\lambda I)^{-1}\Psi^T. \tag{5.18}$$

The prior operator corresponding to the marginal $\mathbb{P}_Y$ is estimated from $\{\tilde{y}_j\}_{j=1}^m$ as

$$\tilde{C}_{YY} = \frac{1}{m}\tilde{\Psi}\tilde{\Psi}^T. \tag{5.19}$$

Let $A := (L + n\lambda I)^{-1}\tilde{L}$, the joint operator is estimated as

$$\hat{C}_{X|Y}\tilde{C}_{YY} = \frac{1}{m}\Phi(L + n\lambda I)^{-1}\Psi^T\tilde{\Psi}\tilde{\Psi}^T = \frac{1}{m}\Phi(L + n\lambda I)^{-1}\tilde{L}\tilde{\Psi}^T = \frac{1}{m}\Phi A\tilde{\Psi}^T. \tag{5.20}$$

The evidence operator is estimated as

$$\begin{aligned}\hat{C}_{X|Y}\tilde{C}_{YY}(\hat{C}_{X|Y})^T &= \frac{1}{m}\Phi(L + n\lambda I)^{-1}\tilde{L}\tilde{\Psi}^T\Psi(L + n\lambda I)^{-1}\Phi^T \\ &= \frac{1}{m}\Phi(L + n\lambda I)^{-1}\tilde{L}\tilde{L}^T(L + n\lambda I)^{-1}\Phi^T \\ &= \frac{1}{m}\Phi AA^T\Phi^T.\end{aligned} \tag{5.21}$$

Finally, by definition 5.6, the DMO is estimated as

$$\begin{aligned}\bar{C}'_{X|Y} &= (\hat{C}_{X|Y}\tilde{C}_{YY})^T(\hat{C}_{X|Y}\tilde{C}_{YY}(\hat{C}_{X|Y})^T + \epsilon I)^{-1} \\ &= \left[\frac{1}{m}\Phi A\tilde{\Psi}^T\right]^T\left[\frac{1}{m}\Phi AA^T\Phi^T + \epsilon I\right]^{-1} \\ &= \left[\Phi A\tilde{\Psi}^T\right]^T\left[\Phi AA^T\Phi^T + m\epsilon I\right]^{-1} \\ &= \tilde{\Psi}A^T\Phi^T\left[\Phi AA^T\Phi^T + m\epsilon I\right]^{-1} \\ &= \tilde{\Psi}\left[A^T\Phi^T\Phi A + m\epsilon I\right]^{-1}A^T\Phi^T \\ &= \tilde{\Psi}\left[A^TKA + m\epsilon I\right]^{-1}A^T\Phi^T.\end{aligned} \tag{5.22}$$

□

5.13 Supporting Proofs for Section 5.5

Proof of Theorem 5.6. Each optimization is a standard regularized least squares problem. The first optimization over $\mathbf{v}$ can be written as

$$\hat{\mathbf{v}}[\mathbf{w}] = \arg\min_{\mathbf{v}\in\mathbb{R}^q} \|\Phi^T\mathbf{w} - \Psi^T\mathbf{v}\|^2 + n\lambda\|\mathbf{v}\|^2, \tag{5.23}$$

where $\mathbf{f} = \Phi^T\mathbf{w}$ is the target and Ψ is the feature matrix. This gives the solution $\hat{\mathbf{v}}[\mathbf{w}] = (\Psi\Psi^T + n\lambda I)^{-1}\Psi(\Phi^T\mathbf{w})$. Therefore, The second optimization over $\mathbf{w}$ can be written as

$$\begin{aligned}
\bar{\mathbf{w}} &= \arg\min_{\mathbf{w}\in\mathbb{R}^p} \|\tilde{\mathbf{z}} - \tilde{\Psi}^T\hat{\mathbf{v}}[\mathbf{w}]\|^2 + m\epsilon\|\mathbf{w}\|^2 \\
&= \arg\min_{\mathbf{w}\in\mathbb{R}^p} \|\tilde{\mathbf{z}} - \tilde{\Psi}^T(\Psi\Psi^T + n\lambda I)^{-1}\Psi\Phi^T\mathbf{w}\|^2 + m\epsilon\|\mathbf{w}\|^2 \\
&= \arg\min_{\mathbf{w}\in\mathbb{R}^p} \|\tilde{\mathbf{z}} - A^T\Phi^T\mathbf{w}\|^2 + m\epsilon\|\mathbf{w}\|^2 \\
&= \arg\min_{\mathbf{w}\in\mathbb{R}^p} \|\tilde{\mathbf{z}} - \Theta^T\mathbf{w}\|^2 + m\epsilon\|\mathbf{w}\|^2,
\end{aligned} \tag{5.24}$$

where we used $A := \Psi^T(\Psi\Psi^T + n\lambda I)^{-1}\tilde{\Psi}$ as per definition 5.8 and we define $\Theta := \Phi A$. This is now a regularized least squares problem with $\tilde{\mathbf{z}}$ as the target and $\Theta := \Phi A$ as the feature matrix. This gives the solution $\bar{\mathbf{w}} = (\Theta\Theta^T + m\epsilon I)^{-1}\Theta\tilde{\mathbf{z}} = (\Phi AA^T\Phi^T + m\epsilon I)^{-1}\Phi A\tilde{\mathbf{z}}$, which yields the parametric DME estimator in definition 5.8. □

Proof of Lemma 5.2. We first establish that the transformation matrix in definition 5.7 $A = (L + n\lambda I)^{-1}\tilde{L}$ is the same as the transformation matrix in definition 5.8 $A = \Psi^T(\Psi\Psi^T + n\lambda I)^{-1}\tilde{\Psi}$ via a special case of the Woodbury identity $B(CB + \delta I)^{-1} = (BC + \delta I)^{-1}B$ for appropriately sized matrices or operators B and C [Higham, 2002]. Consequently, $(L + n\lambda I)^{-1}\tilde{L} = (\Psi^T\Psi + n\lambda I)^{-1}\Psi^T\tilde{\Psi} = \Psi^T(\Psi\Psi^T + n\lambda I)^{-1}\tilde{\Psi}$.

From definition 5.7 we have $\bar{\boldsymbol{\alpha}} := A\big[A^TKA + m\epsilon I\big]^{-1}\tilde{\mathbf{z}}$ so that

$$\begin{aligned}
\Phi\bar{\boldsymbol{\alpha}} &= \Phi A\big[A^TKA + m\epsilon I\big]^{-1}\tilde{\mathbf{z}} \\
&= \Phi A\big[A^T\Phi^T\Phi A + m\epsilon I\big]^{-1}\tilde{\mathbf{z}} \\
&= \big[\Phi AA^T\Phi^T + m\epsilon I\big]^{-1}\Phi A\tilde{\mathbf{z}} \\
&= \bar{\mathbf{w}}.
\end{aligned} \tag{5.25}$$

This relationship is a direct consequence of the kernel trick, where we used $k(x, x') = \phi(x)^T\phi(x')$ such that $K = \Phi^T\Phi$. □

Proof of Theorem 5.7 Part 1. In this proof we provide the derivations for task transformed Bayesian linear regression (TTBLR). We first reiterate the priors and likelihoods used.

Priors We first place priors on the weights of our linear models $g(y) = \mathbf{v}^T\psi(y)$ and $f(x) = \mathbf{w}^T\psi(x)$,

$$\begin{aligned} p(\mathbf{v}) &\sim \mathcal{N}(\mathbf{v}; \mathbf{0}, \beta^2 I), \\ p(\mathbf{w}) &\sim \mathcal{N}(\mathbf{w}; \mathbf{0}, \gamma^2 I). \end{aligned} \tag{5.26}$$

Likelihoods As we only observe from g and never from f directly, there is no need to add noise from $f(x)$ to z and we degenerate the likelihood to $z = f(x)$. The likelihood for g is the regular Gaussian likelihood due to observational noise. Together, we have

$$\begin{aligned} p(z|\mathbf{v}) &= \mathcal{N}(z; \mathbf{v}^T\psi(y), \sigma^2), \\ p(z|\mathbf{w}) &= \mathcal{N}(z; \mathbf{w}^T\phi(x), 0). \end{aligned} \tag{5.27}$$

Prior for g The prior on the weights of g is

$$p(\mathbf{v}) = \mathcal{N}(\mathbf{v}; \mathbf{0}, \beta^2 I). \tag{5.28}$$

Likelihood for g In task transformed learning, the pairs $(\mathbf{y}, \mathbf{z})$ are used to learn g, and $(\tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ are the query points for g. Although $\mathbf{z}$ is not directly available, they are propagated through from f. We also refer to $\mathbf{z}$ as the pseudo-training targets. This leads to the following likelihood,

$$\begin{aligned} p(\mathbf{z}|\mathbf{v}) &= \mathcal{N}(\mathbf{z}; \Psi^T\mathbf{v}, \sigma^2 I), \\ p(\tilde{\mathbf{z}}|\mathbf{v}) &= \mathcal{N}(\tilde{\mathbf{z}}; \tilde{\Psi}^T\mathbf{v}, \sigma^2 I). \end{aligned} \tag{5.29}$$

Marginal Likelihood for g The marginal likelihood of observing the pseudo-training targets $\mathbf{z}$ is

$$\begin{aligned} p(\mathbf{z}) &= \int_{\mathbb{R}^q} p(\mathbf{z}|\mathbf{v})p(\mathbf{v})d\mathbf{v} \\ &= \mathcal{N}(\mathbf{z}; \mathbf{0}, \beta^2\Psi^T\Psi + \sigma^2 I). \end{aligned} \tag{5.30}$$

Posterior for g The posterior of the weights given the pseudo-training targets $\mathbf{z}$ is

$$\begin{aligned} p(\mathbf{v}|\mathbf{z}) &= \frac{p(\mathbf{z}|\mathbf{v})p(\mathbf{v})}{p(\mathbf{z})} \\ &= \mathcal{N}\left(\mathbf{v}; \left(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I\right)^{-1}\Psi\mathbf{z}, \sigma^2\left(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I\right)^{-1}\right). \end{aligned} \tag{5.31}$$

Predictive distribution for g The posterior predictive distribution of $\tilde{\mathbf{z}}$ given the pseudo-training targets $\mathbf{z}$ is

$$\begin{aligned} p(\tilde{\mathbf{z}}|\mathbf{z}) &= \int_{\mathbb{R}^q} p(\tilde{\mathbf{z}}|\mathbf{v})p(\mathbf{v}|\mathbf{z})d\mathbf{v} \\ &= \mathcal{N}\left(\tilde{\mathbf{z}}; \tilde{\Psi}^T\left(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I\right)^{-1}\Psi\mathbf{z}, \sigma^2\tilde{\Psi}^T\left(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I\right)^{-1}\tilde{\Psi} + \sigma^2 I\right) \\ &= \mathcal{N}(\tilde{\mathbf{z}}; A^T\mathbf{z}, \Sigma), \end{aligned} \tag{5.32}$$

where $A = \Psi^T(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I)^{-1}\tilde{\Psi}$ and $\Sigma = \sigma^2\tilde{\Psi}^T(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I)^{-1}\tilde{\Psi} + \sigma^2 I$.

Importantly, the MAP solution for learning g amount to just taking the posterior mean $\hat{\mathbf{v}} = (\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I)^{-1}\Psi\mathbf{z}$ as a point estimate. In this case, the predictive covariance would simplify to $\Sigma = \sigma^2 I$.

Prior for f The prior on the weights of f is

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \gamma^2 I). \tag{5.33}$$

Likelihood for f As targets z are never directly observed from f, the likelihood is a noiseless Gaussian likelihood,

$$\begin{aligned} p(\mathbf{z}|\mathbf{w}) &= \mathcal{N}(\mathbf{z}; \Phi^T\mathbf{w}, 0I), \\ p(\mathbf{z}^\star|\mathbf{w}) &= \mathcal{N}(\mathbf{z}^\star; (\Phi^\star)^T\mathbf{w}, 0I). \end{aligned} \tag{5.34}$$

Propagating this likelihood through the predictive distribution of g, we have

$$\begin{aligned} p(\tilde{\mathbf{z}}|\mathbf{w}) &= \int_{\mathbb{R}^n} p(\tilde{\mathbf{z}}|\mathbf{z})p(\mathbf{z}|\mathbf{w})d\mathbf{z} \\ &= \mathcal{N}(\tilde{\mathbf{z}}; A^T\Phi^T\mathbf{w}, \Sigma). \end{aligned} \tag{5.35}$$

The above prior-likelihood pair describes a TBLR with $M = A = \Psi^T(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I)^{-1}\tilde{\Psi}$ as the transformation matrix and $\Sigma = \sigma^2\tilde{\Psi}^T(\Psi\Psi^T + \frac{\sigma^2}{\beta^2}I)^{-1}\tilde{\Psi} + \sigma^2 I$ as the noise covariance. As such, the remaining distributions exhibit the same forms as shown in table 5.3.

Marginal Likelihood for f The marginal likelihood for the observed targets $\tilde{\mathbf{z}}$ is

$$\begin{aligned} p(\tilde{\mathbf{z}}) &= \int_{\mathbb{R}^p} p(\tilde{\mathbf{z}}|\mathbf{w})p(\mathbf{w})d\mathbf{w} \\ &= \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, \gamma^2 A^T\Phi^T\Phi A + \Sigma) \\ &= \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, [\Sigma^{-1} - \Sigma^{-1}A^T\Phi^T C\Phi A\Sigma^{-1}]^{-1}), \end{aligned} \tag{5.36}$$

TABLE 5.3: Summary of TBLR and TBKR, where we define the shorthand $S := M^T K M + \Sigma$, $C := [\Phi M \Sigma^{-1} M^T \Phi^T + \frac{1}{\gamma^2} I]^{-1}$, and $\mathbf{m} := C\Phi M \Sigma^{-1} \tilde{\mathbf{z}}$.

Density	Transformed Bayesian Linear Regression	Transformed Bayesian Kernel Regression
Prior	$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \gamma^2)$	$p(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \mathbf{0}, K)$
Likelihood	$p(\tilde{\mathbf{z}}\|\mathbf{w}) = \mathcal{N}(\tilde{\mathbf{z}}; M^T \Phi^T \mathbf{w}, \Sigma)$	$p(\tilde{\mathbf{z}}\|\mathbf{f}) = \mathcal{N}(\tilde{\mathbf{z}}; M^T \mathbf{f}, \Sigma)$
Evidence	$p(\tilde{\mathbf{z}}) = \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, [\Sigma^{-1} - \Sigma^{-1} M^T \Phi^T C \Phi M \Sigma^{-1}]^{-1})$	$p(\tilde{\mathbf{z}}) = \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, M^T K M + \Sigma)$
Posterior	$p(\mathbf{w}\|\tilde{\mathbf{z}}) = \mathcal{N}(\mathbf{w}; \mathbf{m}, C)$	$p(\mathbf{f}\|\tilde{\mathbf{z}}) = \mathcal{N}(\mathbf{f}; KMS^{-1}\tilde{\mathbf{z}}, K - KMS^{-1}M^T K)$
Predictive	$p(\mathbf{z}^\star\|\tilde{\mathbf{z}}) = \mathcal{N}(\mathbf{z}^\star; \Phi^{\star T}\mathbf{m}, \Phi^{\star T} C \Phi^\star)$	$p(\mathbf{z}^\star\|\tilde{\mathbf{z}}) = \mathcal{N}(\mathbf{z}^\star; K^{\star T} M S^{-1} \tilde{\mathbf{z}}, K^{\star\star} - K^{\star T} M S^{-1} M^T K^\star)$

where $C = [\Phi A \Sigma^{-1} A^T \Phi^T + \frac{1}{\gamma^2} I]^{-1}$. The last line is an alternative form that is more computationally efficient when the number of features is less than $p < m$ where p is the dimensionality of the feature $\phi(x)$ for f.

Posterior for f The posterior of the weights $\mathbf{w}$ given the observed targets $\tilde{\mathbf{z}}$ is

$$\begin{aligned} p(\mathbf{w}|\tilde{\mathbf{z}}) &= \frac{p(\tilde{\mathbf{z}}|\mathbf{w})p(\mathbf{w})}{p(\tilde{\mathbf{z}})} \\ &= \mathcal{N}(\mathbf{w}; \mathbf{m}, C), \end{aligned} \tag{5.37}$$

where $\mathbf{m} := C\Phi A \Sigma^{-1} \tilde{\mathbf{z}}$.

Predictive distribution for f Finally, the overall predictive distribution of query targets $\mathbf{z}^\star$ given the observed targets $\tilde{\mathbf{z}}$ is

$$\begin{aligned} p(\mathbf{z}^\star|\tilde{\mathbf{z}}) &= \int_{\mathbb{R}^p} p(\mathbf{z}^\star|\mathbf{w}) p(\mathbf{w}|\tilde{\mathbf{z}}) d\mathbf{w} \\ &= \mathcal{N}(\mathbf{z}^\star; \Phi^{\star T}\mathbf{m}, \Phi^{\star T} C \Phi^\star). \end{aligned} \tag{5.38}$$

Consider the posterior mean $\mathbf{m} := C\Phi A \Sigma^{-1} \tilde{\mathbf{z}} = [\Phi A \Sigma^{-1} A^T \Phi^T + \frac{1}{\gamma^2} I]^{-1} \Phi M \Sigma^{-1} \tilde{\mathbf{z}}$, which would also be the MAP solution for f. Using the MAP solution for learning g such that $\Sigma = \sigma^2 I$, we have $\mathbf{m} := [\Phi A A^T \Phi^T + \frac{\sigma^2}{\gamma^2} I]^{-1} \Phi A \tilde{\mathbf{z}}$. This is the same form as the weights $\tilde{\mathbf{w}}$ of the parametric DME estimator (definition 5.8) with $\lambda = \frac{\sigma^2}{n\beta^2}$ and $\epsilon = \frac{\sigma^2}{m\gamma^2}$. □

Proof of Theorem 5.7 Part 2. In this proof we provide the derivations for task transformed Bayesian kernel regression (TTBKR), also named task transformed Gaussian process regression (TTGPR), whose graphical model is provided in figure 5.2. We first reiterate the priors and likelihoods used.

Priors We place GP priors on the functions g and f directly,

$$\begin{aligned} g &\sim \mathcal{GP}(0, \ell), \\ f &\sim \mathcal{GP}(0, k). \end{aligned} \tag{5.39}$$

Likelihoods As we only observe from g and never from f directly, there is no need to add noise from $f(x)$ to z and we degenerate the likelihood to $z = f(x)$. The likelihood for g is the regular Gaussian likelihood due to observational noise. Together, we have

$$\begin{aligned} p(z|g) &= \mathcal{N}(z; g(y), \sigma^2), \\ p(z|f) &= \mathcal{N}(z; f(x), 0). \end{aligned} \tag{5.40}$$

Prior for g The prior of g at $\mathbf{y}$ is

$$p(\mathbf{g}) = \mathcal{N}(\mathbf{g}; \mathbf{0}, L). \tag{5.41}$$

Likelihood for g In task transformed learning, the pairs $(\mathbf{y}, \mathbf{z})$ are used to learn g, and $(\tilde{\mathbf{y}}, \tilde{\mathbf{z}})$ are the query points for g. Although $\mathbf{z}$ is not directly available, they are propagated through from f. We also refer to $\mathbf{z}$ as the pseudo-training targets. The likelihood of observing $\mathbf{z}$ at $\mathbf{y}$ is,

$$p(\mathbf{z}|\mathbf{g}) = \mathcal{N}(\mathbf{z}; \mathbf{g}, \sigma^2 I). \tag{5.42}$$

Marginal Likelihood for g The marginal likelihood of observing the psuedo-training targets $\mathbf{z}$ is:

$$\begin{aligned} p(\mathbf{z}) &= \int_{\mathbb{R}^n} p(\mathbf{z}|\mathbf{g}) p(\mathbf{g}) d\mathbf{g} \\ &= \mathcal{N}(\mathbf{z}; \mathbf{0}, L + \sigma^2 I). \end{aligned} \tag{5.43}$$

Posterior for g The posterior of the latent function evaluations $\mathbf{g}$ at $\mathbf{y}$ given the pseudo-training targets $\mathbf{z}$ is

$$\begin{aligned} p(\mathbf{g}|\mathbf{z}) &= \frac{p(\mathbf{z}|\mathbf{g}) p(\mathbf{g})}{p(\mathbf{z})} \\ &= \mathcal{N}(\mathbf{g}; L(L + \sigma^2 I)^{-1}\mathbf{z}, L - L(L + \sigma^2 I)^{-1} L). \end{aligned} \tag{5.44}$$

Predictive distribution for g To obtain the predictive distribution, we first condition the GP field for on the latent function evaluations $\mathbf{g}$ at $\mathbf{y}$ and to obtain the conditional distribution for $\tilde{\mathbf{g}}$ at $\tilde{\mathbf{y}}$ given $\mathbf{g}$ at $\mathbf{y}$,

$$p(\tilde{\mathbf{g}}|\mathbf{g}) = \mathcal{N}(\tilde{\mathbf{g}}; \tilde{L}^T L^{-1}\mathbf{g}, \tilde{\tilde{L}} - \tilde{L}^T L^{-1}\tilde{L}). \tag{5.45}$$

where $\tilde{\tilde{L}} := \tilde{\Psi}^T\tilde{\Psi}$. Now, marginalize the conditional field against the posterior,

$$\begin{aligned} p(\tilde{\mathbf{g}}|\mathbf{z}) &= \int_{\mathbb{R}^n} p(\tilde{\mathbf{g}}|\mathbf{g})p(\mathbf{g}|\mathbf{z})d\mathbf{g} \\ &= \mathcal{N}(\tilde{\mathbf{g}}; \tilde{L}^T(L+\sigma^2 I)^{-1}\mathbf{z}, \tilde{\tilde{L}} - \tilde{L}^T(L+\sigma^2 I)^{-1}\tilde{L}) \end{aligned} \tag{5.46}$$

Finally, marginalize the likelihood $p(\tilde{\mathbf{g}}|\mathbf{z})$ with the predictive distribution of the latent evaluations $\tilde{\mathbf{g}}$ to get the final predictive distribution of the observations $\tilde{\mathbf{z}}$,

$$\begin{aligned} p(\tilde{\mathbf{z}}|\mathbf{z}) &= \int_{\mathbb{R}^m} p(\tilde{\mathbf{z}}|\tilde{\mathbf{g}})p(\tilde{\mathbf{g}}|\mathbf{z})d\tilde{\mathbf{g}} \\ &= \mathcal{N}(\tilde{\mathbf{z}}; \tilde{L}^T(L+\sigma^2 I)^{-1}\mathbf{z}, \tilde{\tilde{L}} + \sigma^2 I - \tilde{L}^T(L+\sigma^2 I)^{-1}\tilde{L}) \\ &= \mathcal{N}(\tilde{\mathbf{z}}; A^T\mathbf{z}, \Sigma), \end{aligned} \tag{5.47}$$

where $A = (L+\sigma^2 I)^{-1}\tilde{L}$ and $\Sigma = \tilde{\tilde{L}} + \sigma^2 I - \tilde{L}^T(L+\sigma^2 I)^{-1}\tilde{L}$.

Importantly, the MAP solution for learning g amount to just taking the posterior mean $\tilde{\mathbf{g}} = \tilde{L}^T(L+\sigma^2 I)^{-1}\mathbf{z}$ as a point estimate. In this case, the predictive covariance would simplify to $\Sigma = \sigma^2 I$.

Prior for f The prior of f at $\mathbf{x}$ is

$$p(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \mathbf{0}, K). \tag{5.48}$$

Likelihood for f As targets z are never directly observed from f, the likelihood is a noiseless Gaussian likelihood,

$$p(\mathbf{z}|\mathbf{f}) = \mathcal{N}(\mathbf{z}; \mathbf{f}, 0I). \tag{5.49}$$

Propagating this likelihood through the predictive distribution of g, we have

$$\begin{aligned} p(\tilde{\mathbf{z}}|\mathbf{f}) &= \int_{\mathbb{R}^n} p(\tilde{\mathbf{z}}|\mathbf{z})p(\mathbf{z}|\mathbf{f})d\mathbf{z} \\ &= \mathcal{N}(\tilde{\mathbf{z}}; A^T\mathbf{f}, \Sigma). \end{aligned} \tag{5.50}$$

The above prior-likelihood pair describes a TBKR with $M = A = (L+\sigma^2 I)^{-1}\tilde{L}$ as the transformation matrix and $\Sigma = \tilde{\tilde{L}} + \sigma^2 I - \tilde{L}^T(L+\sigma^2 I)^{-1}\tilde{L}$ as the noise

covariance. As such, the remaining distribution exhibit the same forms as shown in table 5.3.

Marginal Likelihood for f The marginal likelihood for the observed targets $\tilde{\mathbf{z}}$ is

$$\begin{aligned} p(\tilde{\mathbf{z}}) &= \int_{\mathbb{R}^n} p(\tilde{\mathbf{z}}|\mathbf{f})p(\mathbf{f})d\mathbf{f} \\ &= \mathcal{N}(\tilde{\mathbf{z}}; \mathbf{0}, A^T K A + \Sigma). \end{aligned} \tag{5.51}$$

Posterior for f The posterior of the function evaluations $\mathbf{f}$ at $\mathbf{x}$ given the observed targets $\tilde{\mathbf{z}}$ is

$$\begin{aligned} p(\mathbf{f}|\tilde{\mathbf{z}}) &= \frac{p(\tilde{\mathbf{z}}|\mathbf{f})p(\mathbf{f})}{p(\tilde{\mathbf{z}})} \\ &= \mathcal{N}(\mathbf{f}; KA(A^T K A + \Sigma)^{-1}\tilde{\mathbf{z}}, K - KA(A^T K A + \Sigma)^{-1}A^T K). \end{aligned} \tag{5.52}$$

Predictive distribution for f Finally, to obtain the predictive distribution we first condition the GP field on the latent function evaluations $\mathbf{f}$ to obtain the conditional distribution for $\mathbf{f}^\star$ at $\mathbf{x}^\star$ given $\mathbf{f}$ at $\mathbf{x}$.

$$p(\mathbf{f}^\star|\mathbf{f}) = \mathcal{N}(\mathbf{f}^\star; (K^\star)^T K^{-1}\mathbf{f}, K^{\star\star} - (K^\star)^T K^{-1} K^\star). \tag{5.53}$$

Now, marginalize the conditional field against the posterior,

$$\begin{aligned} p(\mathbf{f}^\star|\tilde{\mathbf{z}}) &= \int_{\mathbb{R}^n} p(\mathbf{f}^\star|\mathbf{f})p(\mathbf{f}|\tilde{\mathbf{z}})d\mathbf{f} \\ &= \mathcal{N}(\mathbf{f}^\star; (K^\star)^T A(A^T K A + \Sigma)^{-1}\tilde{\mathbf{z}}, K^{\star\star} - (K^\star)^T A(A^T K A + \Sigma)^{-1}A^T K^\star). \end{aligned} \tag{5.54}$$

Finally, the overall predictive distribution of query targets $\mathbf{z}^\star$ given the observed targets $\tilde{\mathbf{z}}$ is

$$\begin{aligned} p(\mathbf{z}^\star|\tilde{\mathbf{z}}) &= \int_{\mathbb{R}^n} p(\mathbf{z}^\star|\mathbf{f}^\star)p(\mathbf{f}^\star|\tilde{\mathbf{z}})d\mathbf{f}^\star \\ &= \mathcal{N}(\mathbf{z}^\star; (K^\star)^T A(A^T K A + \Sigma)^{-1}\tilde{\mathbf{z}}, K^{\star\star} - (K^\star)^T A(A^T K A + \Sigma)^{-1}A^T K^\star). \end{aligned} \tag{5.55}$$

Consider the posterior predictive mean at a particular query point x, $\bar{f}(x) = (\mathbf{k}(x))^T A(A^T K A + \Sigma)^{-1}\tilde{\mathbf{z}} = \tilde{\mathbf{z}}^T(A^T K A + \Sigma)^{-1}A^T\mathbf{k}(x)$. Using the MAP solution for learning g such that $\Sigma = \sigma^2 I$, we have $\bar{f}(x) = \tilde{\mathbf{z}}^T(A^T K A + \sigma^2 I)^{-1}A^T\mathbf{k}(x)$. This is the same form as the nonparametric DME estimator (definition 5.8) with $\lambda = \frac{\sigma^2}{n}$ and $\epsilon = \frac{\sigma^2}{m}$. □

5.14 Supporting Proofs for Section 5.6

Proof of Theorem 5.8. We first factorize the joint operator $C_{YX} = (C_{XY})^T$ in both directions,

$$C_{Y|X}C_{XX} = C_{YX} = (C_{XY})^T = (C_{X|Y}C_{YY})^T = C_{YY}(C_{X|Y})^T. \tag{5.56}$$

This is analogous to the equation $p(y|x)p(x) = p(y,x) = p(x,y) = p(x|y)p(y) = p(y)p(x|y)$.

Since $C_{X|Y}C_{Y|X}C_{XX} = C_{XX}$, we then apply $C_{X|Y}$ on both sides to cancel out $C_{Y|X}$ and obtain the equation for C_{XX},

$$C_{XX} = (C_{X|Y}C_{Y|X})C_{XX} = C_{X|Y}(C_{Y|X}C_{XX}) = C_{X|Y}C_{YY}(C_{X|Y})^T. \tag{5.57}$$

This is analogous to the equation $p(x) = \int_{\mathcal{Y}} p(y|x)dyp(x) = \int_{\mathcal{Y}} p(y|x)p(x)dy = \int_{\mathcal{Y}} p(y)p(x|y)dy$.

Hence,

$$C'_{XX} := C_{X|Y}C_{YY}(C_{X|Y})^T = C_{XX}. \tag{5.58}$$

Finally, from theorem 5.3 we have

$$C'_{X|Y} = (C_{X|Y}C_{YY})^T(C_{X|Y}C_{YY}(C_{X|Y})^T)^{-1} = C_{YX}C_{XX}^{-1} = C_{Y|X}. \tag{5.59}$$

□

Proof of Theorem 5.9. Since $m = n$ and $\tilde{\mathbf{y}} = \mathbf{y}$, we have that $\tilde{L} = L$, $\tilde{\Psi} = \Psi$. Consequently, $\lim_{\lambda\to 0^+} A = \lim_{\lambda\to 0^+}(L+n\lambda I)^{-1}L = I$. Substituting this into (5.5) we have

$$\begin{aligned}\lim_{\lambda\to 0^+} \bar{C}'_{X|Y} &= \lim_{\lambda\to 0^+} \tilde{\Psi}\big[A^TKA + m\epsilon I\big]^{-1}A^T\Phi^T \\ &= \Psi\big[I^TKI + n\epsilon I\big]^{-1}I^T\Phi^T \\ &= \Psi\big[K + n\epsilon I\big]^{-1}\Phi^T.\end{aligned} \tag{5.60}$$

Reversing the roles of X and Y in (5.5) and replacing the notation λ with ϵ, we have that $\hat{C}_{Y|X} = \Psi\big[K + n\epsilon I\big]^{-1}\Phi^T$. This concludes the proof. □

5.15 Supporting Proofs for Section 5.9

Proof of Theorem 5.10. Consider the absolute difference between $\bar{q}(\mathbf{y})$ and $p_\epsilon(\mathbf{y})$,

$$|\bar{q}(\mathbf{y}) - p_\epsilon(\mathbf{y})| \leq |\bar{q}(\mathbf{y}) - q(\mathbf{y})| + |q(\mathbf{y}) - p_\epsilon(\mathbf{y})|. \tag{5.61}$$

where $q(\mathbf{y}) := \langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\mathbf{\Theta}} \mu_{\mathbf{\Theta}} \rangle_{\mathcal{H}_k} = \mathbb{E}[\langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\mathbf{\Theta}} \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}]$. The first term is

$$\begin{aligned} |\bar{q}(\mathbf{y}) - q(\mathbf{y})| &= |\langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\mathbf{\Theta}} (\tilde{\mu}_{\mathbf{\Theta}} - \mu_{\mathbf{\Theta}}) \rangle_{\mathcal{H}_k}| = |\langle (\hat{C}_{\mathbf{X}|\mathbf{\Theta}})^T \kappa_\epsilon(\mathbf{y}, \cdot), (\tilde{\mu}_{\mathbf{\Theta}} - \mu_{\mathbf{\Theta}}) \rangle_{\mathcal{H}_\ell}| \\ &\leq \|(\hat{C}_{\mathbf{X}|\mathbf{\Theta}})^T \kappa_\epsilon(\mathbf{y}, \cdot)\|_{\mathcal{H}_\ell} \|(\tilde{\mu}_{\mathbf{\Theta}} - \mu_{\mathbf{\Theta}})\|_{\mathcal{H}_\ell} \\ &\leq c \|(\tilde{\mu}_{\mathbf{\Theta}} - \mu_{\mathbf{\Theta}})\|_{\mathcal{H}_\ell}. \end{aligned} \tag{5.62}$$

for some constant c since $\hat{C}_{\mathbf{X}|\mathbf{\Theta}}$ is a bounded operator for all n. Hence, $|\bar{q}(\mathbf{y}) - q(\mathbf{y})|$ decays at $O(m^{-\frac{1}{2}})$.

For the second term, we have $p_\epsilon(\mathbf{y}) = \mathbb{E}[p_\epsilon(\mathbf{y}|\mathbf{\Theta})] = \mathbb{E}[\langle \kappa_\epsilon(\mathbf{y}, \cdot), \mu_{\mathbf{X}|\mathbf{\Theta}=\mathbf{\Theta}} \rangle_{\mathcal{H}_k}] = \mathbb{E}[\langle \kappa_\epsilon(\mathbf{y}, \cdot), C_{\mathbf{X}|\mathbf{\Theta}} \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}]$, similar to $q(\mathbf{y}) = \mathbb{E}[\langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\mathbf{\Theta}} \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}]$. Since we use bounded kernels, define $\bar{\ell} := \sup_{\boldsymbol{\theta}} \|\ell(\boldsymbol{\theta}, \cdot)\|_{\mathcal{H}_\ell}$ and $\bar{\kappa}_\epsilon := \sup_{\mathbf{y}} \|\kappa_\epsilon(\mathbf{y}, \cdot)\|_{\mathcal{H}_k}$. The second term becomes

$$\begin{aligned} |q(\mathbf{y}) - p_\epsilon(\mathbf{y})| &= |\mathbb{E}[\langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\mathbf{\Theta}} \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}] - \mathbb{E}[\langle \kappa_\epsilon(\mathbf{y}, \cdot), C_{\mathbf{X}|\mathbf{\Theta}} \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}]| \\ &\leq \mathbb{E}[|\langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\mathbf{\Theta}} \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}] - \mathbb{E}[\langle \kappa_\epsilon(\mathbf{y}, \cdot), C_{\mathbf{X}|\mathbf{\Theta}} \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}|] \\ &= \mathbb{E}[|\langle \kappa_\epsilon(\mathbf{y}, \cdot), (\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}) \ell(\mathbf{\Theta}, \cdot) \rangle_{\mathcal{H}_k}|] \\ &\leq \mathbb{E}[\|\kappa_\epsilon(\mathbf{y}, \cdot)\|_{\mathcal{H}_k} \|(\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}) \ell(\mathbf{\Theta}, \cdot)\|_{\mathcal{H}_k}] \\ &= \|\kappa_\epsilon(\mathbf{y}, \cdot)\|_{\mathcal{H}_k} \mathbb{E}[\|(\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}) \ell(\mathbf{\Theta}, \cdot)\|_{\mathcal{H}_k}] \\ &= \bar{\kappa}_\epsilon \mathbb{E}[\|(\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}) \ell(\mathbf{\Theta}, \cdot)\|_{\mathcal{H}_k}] \\ &\leq \bar{\kappa}_\epsilon \mathbb{E}[\|\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}\|_{HS} \|\ell(\mathbf{\Theta}, \cdot)\|_{\mathcal{H}_\ell}] \\ &= \bar{\kappa}_\epsilon \mathbb{E}[\|\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}\|_{HS} \sqrt{\ell(\mathbf{\Theta}, \mathbf{\Theta})}] \\ &= \bar{\kappa}_\epsilon \mathbb{E}[\sqrt{\ell(\mathbf{\Theta}, \mathbf{\Theta})}] \|\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}\|_{HS} \\ &\leq \bar{\kappa}_\epsilon \mathbb{E}[\bar{\ell}] \|\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}\|_{HS} \\ &= \bar{\kappa}_\epsilon \bar{\ell} \|\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}\|_{HS}. \end{aligned} \tag{5.63}$$

Hence, in the worst case $|q(\mathbf{y}) - p_\epsilon(\mathbf{y})|$ decays at the rate $\|\hat{C}_{\mathbf{X}|\mathbf{\Theta}} - C_{\mathbf{X}|\mathbf{\Theta}}\|_{HS}$ decays, which is $O_p((n\lambda)^{-\frac{1}{2}} + \lambda^{\frac{1}{2}})$. Together with the first term, we have the claimed convergence rate.

Finally, the empirical form is obtained from substituting the empirical forms for the likelihood CMO and prior embedding, $\bar{q}(\mathbf{y}) := \langle \kappa_\epsilon(\mathbf{y}, \cdot), \hat{C}_{\mathbf{X}|\mathbf{\Theta}} \tilde{\mu}_{\mathbf{\Theta}} \rangle_{\mathcal{H}_k} = \langle \kappa_\epsilon(\mathbf{y}, \cdot), (\Phi(L + n\lambda I)^{-1} \Psi^T)(\frac{1}{m} \tilde{\Psi} \mathbf{1}_m) \rangle_{\mathcal{H}_k} = \frac{1}{m} \boldsymbol{\kappa}_\epsilon^T A \mathbf{1}_m$. □

Chapter 6

Conclusion

Learning and inference form the foundation of an intelligent system. Yet, doing so can be tremendously challenging in the real world, due to the richness and diversity of our environment. In order to successfully discover and act upon important relationships in complex systems, we must be careful not to place unnecessary restrictions on the way we represent our knowledge, despite the simplification it may bring. In this thesis, we represent conditional relationships with conditional kernel mean embeddings, and derive frameworks to learn and make inferences with them to leverage their high representational capacity.

The central theme of this thesis revolves around formulating coherent hyperparameter learning and probabilistic inference frameworks around CMEs using Bayesian principles. It presents novel frameworks for doing so in three general problem settings – classification (chapter 3), inference (chapter 4), and regression (chapter 5), painting three different perspectives of this theme. They are motivated by the core philosophy that Bayesian principles would guide the development of a natural hyperparameter learning algorithm for the particular probabilistic inference task, enabling a holistic modeling framework around the CME.

6.1 Themes and Connections

While the frameworks proposed in chapters 3 to 5 are developed in different settings with different formulations and methodologies, there are overarching themes and connections between them. In this discussion, we explore these connections by focusing on different aspects of the framework.

The core advantage of developing frameworks that leverage CMEs is that it allows for extremely flexible nonparametric representations of conditional distributions without assuming any particular parametric form. Consequently, in all three

chapters, CMEs enable solutions to their respective problem settings in their most general and least restrictive form:

- In chapter 3, the resulting classification framework is naturally probabilistic and multiclass, subsuming the non-probabilistic and binary setting. It also makes little assumption on the feature maps, enabling a wide range of possible architectures, from shallow to deep constructions with narrow to wide layers.
- In chapter 4, the resulting inference framework does not require tractability in likelihood evaluations, allowing approximate Bayesian inference to be performed within challenging and intractable settings. This is possible as our framework form nonparametric representations using only samples.
- In chapter 5, the resulting regression framework is established from multiple dimensions – from parametric to nonparametric for representation of features, from non-Bayesian to Bayesian for inference on latents, and from non-transformed to transformed for observations of targets. The learning algorithm is general and unifies all these settings.

The three chapters of this thesis leverage Bayesian principles at different levels. We can describe their differences firstly in terms of the how their hyperparameter learning algorithms are developed and secondly in terms of how the probabilistic inference tasks are performed.

Firstly, while all three chapters are motivated by formulating a marginal likelihood objective for the inference task at hand, the level at which the marginal likelihood objective aligns with both the inference task and the CME itself increases by each chapter:

- In chapter 3, the hyperparameter learning objective does not correspond to any known marginal likelihood, but emulates its desirable properties via learning theoretic bounds, such as the critical balance between the data fit error and a data-dependent model complexity. Consequently, in the strict sense this formulation is not Bayesian, but attempts to emulate Bayesian consequences using statistical learning theory.
- In chapter 4, the hyperparameter learning objective is a surrogate that approximates the marginal likelihood of the likelihood-free inference task. This arises from the realization that conditional kernel mean embeddings are natural surrogates for likelihood-free models, which can be used for approximate Bayesian inference problems with intractable likelihoods. Consequently, there already exists a natural Bayesian system from the problem

setup, which, while intractable, can be approximated by surrogates using conditional kernel mean embeddings.

- In chapter 5, the hyperparameter learning objective is exactly the marginal likelihood of the task transformed regression task itself. This is possible as we introduce, formulate, and establish deconditional kernel mean embeddings as the natural counterpart to conditional kernel mean embeddings, show that it solves the task transformed regression problem, and reveal their Bayesian extensions and connections. Consequently, the marginal likelihood objective is both aligned to the inference task and the conditional and deconditional kernel mean embedding themselves.

Secondly, the three chapters also differ in the level of Bayesian treatment the inference task receives, in part because the hyperparameter learning algorithms are formulated with respect to the inference task and, as described above, they differ in the level at which they leverage Bayesian principles:

- In chapter 3, the multiclass predictions are probabilistic, but not Bayesian, since no uncertainty quantification is performed on latent variables or functions.
- In chapter 4, the likelihood-free inference task is an approximate Bayesian inference task itself, so uncertainty quantification is performed for simulator parameters as a core and defining component of the inference task, since it is the main purpose of this problem setting. However, no uncertainty quantification is performed on latent variables or functions of the surrogates themselves, including the conditional kernel mean embedding itself.
- In chapter 5, uncertainty quantification is performed on the solution the conditional and deconditional kernel mean embedding provides, which gives rise to uncertainty quantification for the task transformed regression task. This is enabled by regression perspectives to the estimators that conditional and deconditional kernel mean embeddings provide.

6.2 Implications and Future Work

This thesis makes advances to our understanding of CMEs and their applicability through the lens of three cornerstone settings of machine learning – classification, inference, and regression. In addition to the list of contributions that was introduced in section 1.3, the thesis also makes softer contributions by demonstrating novel approaches or ideas that could be generalized beyond the frameworks formulated.

Chapter 3 investigates learning theoretic bounds from statistical learning theory to establish expected risk bounds for CMEs. Traditionally, expected risk bounds are specified with respect to a class of functions, instead of specified with respect to a particular function. In our case, the class of functions are CMEs across a set of hyperparameter settings, and the particular function is a single CME with one set of hyperparameter setting. In our proofs, we showed how to construct expected risk bounds for the latter from the former. The critical element being that the particular function defines a space of functions via an upper bound. Furthermore, due to its probabilistic and data-dependent nature, the expected risk bound can be instantiated with only batch subsets of the data at each iteration instead, dramatically lowering computational costs. Those ideas by themselves do not rely on CMEs and can be potentially generalized beyond CMEs.

In our work, we relied on the cross entropy loss and the convergence properties we proved for the MCE. However, it is conceivable that for other classifiers, potentially not constructed from CMEs and have different convergence properties, could be applied in conjunction with a potentially different loss.

Chapter 4 proposes a complete framework for likelihood-free inference (LFI) encapsulating model, learning, and inference. While KELFI is an efficient and effective framework given simulator samples, it has yet to play an active role in choosing parameter settings to run the simulator and collect samples from. In our discussion we alluded to the potential of Bayesian optimization techniques via the connection of the KML to GPR. Alternatively, a sequential strategy may be adopted where the posterior samples from KELFI serve to be the proposal prior for the next batch of simulator runs.

Another potential addition to the KELFI framework is to explicitly formulate the KML to learn or alleviate summary statistics. While we have done this for specific data formats such as with the iid-KML or the ST-KML, it may be possible to construct the CME used in the KML from highly flexible function classes such as neural networks and treat the summary operations as a learnable mapping.

Chapter 5 introduces the task transformed regression (TTR) problem. In our work, we propose linear and kernel solutions to the task transformed regression (TTR) problem to obtain variants of the DME estimators, as well as Bayesian extensions thereof with connections to the TTGP. Nevertheless, it is possible to posit other families of models or architectures to be trained under the TTR losses to arrive at solutions other than the standard DME that we propose.

Furthermore, it is possible to generalize the task transformed problem to classification or other prediction tasks by using a the appropriate loss. We call this general setting *task transformed learning*, which in essence transforms the task of predicting a target from old features to predicting a target from new features when only pairs between targets and old features and pairs between old features and new

features are available. The crucial element is not to treat them as a cascaded or transitive stages of prediction, which as we have shown is ineffective in the TTR case. Instead, the philosophy is to treat it as two processes from each set of features, old or new, to the target, and match them correspondingly for inference to propagate, with an attention to which variables are latent and which are not. In the formulation of the TTGP, due to the GP prior and Gaussian likelihoods, this propagation has closed form solutions. It is thus conceivable that generalizations to the TTR problem could in general not admit closed form solutions for such propagations, and approximations would be necessary.

Potential applications of the task transformed learning frameworks include supervised dimensionality reduction and learning under missing data. In the former, the new features have lower dimensionality than the old features. In the latter, we have three cases depending on the goal – either the old or new features contain missing values and the other is the completed version of it, or both are vary in what they are missing from the completed version. Evidently, a more formal problem definition is required to establish the appropriate formulation.

The task transformed learning problem may also bear connections to explore with transfer learning and meta learning, as they all address learning between or across tasks.

In summary, Bayesian principles and perspectives to CMEs enable more automatic and powerful technique for learning and inference. This thesis contributed towards this goal in three very cornerstone settings in machine learning, and open up avenues in multiple directions for further investigations.

Bibliography

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). TensorFlow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, Georgia, USA*.

Aeberhard, S., Coomans, D., and De Vel, O. (1992). Comparison of classifiers in high dimensional settings. *Dept. Math. Statist., James Cook Univ., North Queensland, Australia, Tech. Rep*, (92-02).

Aly, M. (2005). Survey on multiclass classification methods. *Neural Netw*, pages 1–9.

Andrieu, C., Roberts, G. O., et al. (2009). The pseudo-marginal approach for efficient monte carlo computations. *The Annals of Statistics*, 37(2):697–725.

Bache, K. and Lichman, M. (2013). UCI machine learning repository.

Bartlett, P. L. and Mendelson, S. (2002). Rademacher and Gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482.

Beaumont, M. A. (2010). Approximate bayesian computation in evolution and ecology. *Annual review of ecology, evolution, and systematics*, 41:379–406.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

Carmeli, C., De Vito, E., Toigo, A., and Umanitá, V. (2010). Vector valued reproducing kernel hilbert spaces and universality. *Analysis and Applications*, 8(01):19–61.

Chen, Y., Welling, M., and Smola, A. (2010). Super-samples from kernel herding. In *The Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pages 109–116. AUAI Press.

Cortes, C., Kloft, M., and Mohri, M. (2013). Learning kernels using local Rademacher complexity. In *Advances in neural information processing systems*, pages 2760–2768.

Dai, B., He, N., Pan, Y., Boots, B., and Song, L. (2017). Learning from Conditional Distributions via Dual Embeddings. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1458–1467. PMLR.

Fisher, R. A. (1922). On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 222(594-604):309–368.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188.

Flaxman, S., Sejdinovic, D., Cunningham, J. P., and Filippi, S. (2016). Bayesian learning of kernel embeddings. In *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pages 182–191. AUAI Press.

Freire, A. L., Barreto, G. A., Veloso, M., and Varela, A. T. (2009). Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. In *Robotics Symposium (LARS), 2009 6th Latin American*, pages 1–6. IEEE.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

Fukumizu, K., Bach, F. R., and Jordan, M. I. (2004). Dimensionality reduction for supervised learning with reproducing kernel Hilbert spaces. *Journal of Machine Learning Research*, 5(Jan):73–99.

Fukumizu, K., Gretton, A., Lanckriet, G. R., Schölkopf, B., and Sriperumbudur, B. K. (2009). Kernel choice and classifiability for rkhs embeddings of probability distributions. In *Advances in neural information processing systems*, pages 1750–1758.

Fukumizu, K., Song, L., and Gretton, A. (2013). Kernel Bayes' rule: Bayesian inference with positive definite kernels. *Journal of Machine Learning Research*, 14(1):3753–3783.

Genton, M. G. (2001). Classes of kernels for machine learning: a statistics perspective. *Journal of machine learning research*, 2(Dec):299–312.

Gretton, A., Borgwardt, K. M., Rasch, M., Schölkopf, B., and Smola, A. J. (2007). A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520.

Gretton, A., Sejdinovic, D., Strathmann, H., Balakrishnan, S., Pontil, M., Fukumizu, K., and Sriperumbudur, B. K. (2012). Optimal kernel choice for large-scale two-sample tests. In *Advances in neural information processing systems*, pages 1205–1213.

Grünewälder, S., Lever, G., Baldassarre, L., Patterson, S., Gretton, A., and Pontil, M. (2012). Conditional mean embeddings as regressors. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, volume 2, pages 1823–1830.

Halton, J. H. (1960). On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. *Numerische Mathematik*, 2(1):84–90.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications.

Higham, N. J. (2002). *Accuracy and stability of numerical algorithms*. SIAM.

Horton, P. and Nakai, K. (1996). A probabilistic classification system for predicting the cellular localization sites of proteins. In *Ismb*, volume 4, pages 109–115.

Jitkrittum, W., Szabó, Z., Chwialkowski, K. P., and Gretton, A. (2016). Interpretable Distribution Features with Maximum Testing Power. In *Advances In Neural Information Processing Systems*, pages 181–189.

Kajihara, T., Yamazaki, K., Kanagawa, M., and Fukumizu, K. (2018). Kernel Recursive ABC: Point Estimation with Intractable Likelihood. *arXiv preprint arXiv:1802.08404*.

Kanagawa, M. and Fukumizu, K. (2014). Recovering Distributions from Gaussian RKHS Embeddings. In *AISTATS*, pages 457–465.

Kanagawa, M., Nishiyama, Y., Gretton, A., and Fukumizu, K. (2016). Filtering with state-observation examples via kernel monte carlo filter. *Neural computation*, 28(2):382–444.

Kaya, E., Yasar, A., and Saritas, I. (2016). Banknote Classification Using Artificial Neural Network Approach. *International Journal of Intelligent Systems and Applications in Engineering*, 4(1):16–19.

Kearns, M., Mansour, Y., Ng, A. Y., and Ron, D. (1997). An experimental and theoretical comparison of model selection methods. *Machine Learning*, 27(1):7–50.

Kimeldorf, G. and Wahba, G. (1971). Some results on Tchebycheffian spline functions. *Journal of mathematical analysis and applications*, 33(1):82–95.

Kingma, D. and Ba, J. (2016). Adam: A method for stochastic optimization. *The International Conference on Learning Representations (ICLR)*.

Kloft, M. and Blanchard, G. (2011). The local Rademacher complexity of lp-norm multiple kernel learning. In *Advances in Neural Information Processing Systems*, pages 2438–2446.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Ledoux, M. and Talagrand, M. (2013). *Probability in Banach Spaces: isoperimetry and processes*. Springer Science & Business Media.

Marin, J.-M., Pudlo, P., Robert, C. P., and Ryder, R. J. (2012). Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180.

Marjoram, P., Molitor, J., Plagnol, V., and Tavaré, S. (2003). Markov chain monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328.

McCalman, L., O'Callaghan, S., and Ramos, F. (2013). Multi-modal estimation with kernel embeddings for learning motion models. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2845–2852. IEEE.

Meeds, E. and Welling, M. (2014). GPS-ABC: Gaussian process surrogate approximate Bayesian computation. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 593–602. AUAI Press.

Mitrovic, J., Sejdinovic, D., and Teh, Y. W. (2016). DR-ABC: Approximate Bayesian Computation with kernel-based distribution regression.

Moreno, A., Adel, T., Meeds, E., Rehg, J. M., and Welling, M. (2016). Automatic variational ABC. *arXiv preprint arXiv:1606.08549*.

Muandet, K., Balduzzi, D., and Schölkopf, B. (2013). Domain Generalization via Invariant Feature Representation. In *ICML (1)*, pages 10–18.

Muandet, K., Fukumizu, K., Sriperumbudur, B., Schölkopf, B., et al. (2017). Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141.

Murray-Smith, R. and Pearlmutter, B. A. (2005). Transformations of gaussian process priors. In *Deterministic and Statistical Methods in Machine Learning*, pages 110–123. Springer.

Nakagome, S., Fukumizu, K., and Mano, S. (2013). Kernel approximate Bayesian computation in population genetic inferences. *Statistical applications in genetics and molecular biology*, 12(6):667–678.

Ong, V. M., Nott, D. J., Tran, M.-N., Sisson, S. A., and Drovandi, C. C. (2018). Variational Bayes with synthetic likelihood. *Statistics and Computing*, 28(4):971–988.

Pahikkala, T., Airola, A., Gieseke, F., and Kramer, O. (2012). Unsupervised multi-class regularized least-squares classification. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 585–594. IEEE.

Papamakarios, G. and Murray, I. (2016). Fast ε-free inference of simulation models with bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pages 1028–1036.

Park, M., Jitkrittum, W., and Sejdinovic, D. (2016). K2-ABC: Approximate Bayesian Computation with Kernel Embeddings. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pages 398–407. PMLR.

Pontil, M. and Maurer, A. (2013). Excess risk bounds for multitask learning with trace norm regularization. In *Conference on Learning Theory*, pages 55–76.

Price, L. F., Drovandi, C. C., Lee, A., and Nott, D. J. (2017). Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, pages 1–11.

Pritchard, J. K., Seielstad, M. T., Perez-Lezaun, A., and Feldman, M. W. (1999). Population growth of human y chromosomes: a study of y chromosome microsatellites. *Molecular biology and evolution*, 16(12):1791–1798.

Rahimi, A. and Recht, B. (2008). Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian processes for machine learning*. The MIT Press.

Rifkin, R., Yeo, G., Poggio, T., et al. (2003). Regularized least-squares classification. *Nato Science Series Sub Series III Computer and Systems Sciences*, 190:131–154.

Schölkopf, B. and Smola, A. J. (2002). *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press.

Sisson, S. A., Fan, Y., and Tanaka, M. M. (2007). Sequential monte carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 104(6):1760–1765.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959.

Song, L., Fukumizu, K., and Gretton, A. (2013). Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *IEEE Signal Processing Magazine*, 30(4):98–111.

Song, L., Huang, J., Smola, A., and Fukumizu, K. (2009). Hilbert space embeddings of conditional distributions with applications to dynamical systems. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 961–968. ACM.

Song, L., Zhang, X., Smola, A., Gretton, A., and Schölkopf, B. (2008). Tailoring density estimation via reproducing kernel moment matching. In *Proceedings of the 25th international conference on Machine learning*, pages 992–999. ACM.

Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. (2010a). On the relation between universality, characteristic kernels and RKHS embedding of measures. In *AISTATS*, pages 773–780.

Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. (2011). Universality, characteristic kernels and RKHS embedding of measures. *Journal of Machine Learning Research*, 12(Jul):2389–2410.

Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Schölkopf, B., and Lanckriet, G. R. (2010b). Hilbert space embeddings and metrics on probability measures. *Journal of Machine Learning Research*, 11(Apr):1517–1561.

Toni, T., Welch, D., Strelkowa, N., Ipsen, A., and Stumpf, M. P. (2009). Approximate bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31):187–202.

Tran, D., Ranganath, R., and Blei, D. (2017a). Hierarchical implicit models and likelihood-free variational inference. In *Advances in Neural Information Processing Systems*, pages 5523–5533.

Tran, M.-N., Nott, D. J., and Kohn, R. (2017b). Variational Bayes with intractable likelihood. *Journal of Computational and Graphical Statistics*, 26(4):873–882.

Wilkinson, R. (2014). Accelerating abc methods using gaussian processes. In *Artificial Intelligence and Statistics*, pages 1015–1023.

Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310):1102.

Xu, C., Liu, T., Tao, D., and Xu, C. (2016). Local Rademacher complexity for multi-label learning. *IEEE Transactions on Image Processing*, 25(3):1495–1507.

Xu, X. (2017). Generalization of the sherman–morrison–woodbury formula involving the schur complement. *Applied Mathematics and Computation*, 309:183–191.

Xu, Y. and Zhang, H. (2009). Refinement of reproducing kernels. *Journal of Machine Learning Research*, 10(Jan):107–140.

Yu, H.-f., Jain, P., Kar, P., and Dhillon, I. (2014). Large-scale Multi-label Learning with Missing Labels. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 593–601.

Zhou, Z.-H., Wei, D., Li, G., and Dai, H. (2004). On the size of training set and the benefit from ensemble. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 298–307. Springer.