Message Passing-based Inference in Hierarchical Autoregressive Models

Message Passing-based Inference in Hierarchical Autoregressive Models

Albert Podusenko

Message Passing-based Inference in Hierarchical Autoregressive Models

Albert Podusenko

A catalogue record is available from the Eindhoven University of Technology Library.
ISBN 978-90-386-5594-9

Keywords: Forney-Style Factor Graphs, Hierarchical Autoregressive Models, Generative Modeling, Message Passing, Probabilistic Programming

The research in this dissertation has been sponsored by the Netherlands Organization for Scientific Research (NWO).

LaTeX style template provided by Joos Buijs

Message Passing-based Inference in Hierarchical Autoregressive Models

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
rector magnificus prof.dr.ir. F.P.T. Baaijens, voor een
commissie aangewezen door het College voor Promoties,
in het openbaar te verdedigen op
dinsdag 20 december 2022 om 13:30 uur

door

Albert Podusenko

geboren te Nachodka, Russische Verre Oosten

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	prof.dr.ir. A.M.J. Koonen
1e promotor:	prof.dr.ir. Bert de Vries
co-promotor:	dr. Wouter M. Kouw
leden:	prof.dr.ir. Martijn Wisse (Technische Universiteit Delft)
	prof.dr. Siep Weiland
	dr.ir. Rik Vullings
	dr.ir. Sander Stuijk

Het onderzoek dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

Summary

Message Passing-based Inference in Hierarchical Autoregressive Models

This dissertation describes a research effort toward automating personalized design of hearing aid algorithms through in-the-field communication between a user and a portable intelligent agent. The traditional design cycle of hearing aid is inefficient as it requires many human professionals in the design loop who have to elicit and design for a hearing impaired person's unique and context-dependent preferences. In contrast, a wearable synthetic intelligent agent could possibly improve the quality of a hearing aid by on-the-spot suggestions for new hearing aid settings, rather than waiting for offline human expert intervention. To create such an agent, we take inspiration from a theoretical neuroscience framework called the Free Energy Principle, which explains how living brains effectively control their environment by online Bayesian learning of a model of their environment.

According to this hypothesis, an agent (such as a brain) holds a generative probabilistic model for its sensory input signals. Translated to the context of a synthetic agent and an acoustic environment with a hearing aid (HA) and a HA patient, the agent's generative model should comprise a model for both environmental acoustic signals and user appraisals for hearing aid behavior. These models ought to be learned under in-situ conditions through Bayesian inference, which offers a rigorous procedure for parameter estimation in probabilistic models.

Following the premise of the Free Energy Principle, the essence of our approach to automated HA design is that all engineering tasks can be formulated as a Bayesian inference on the generative probabilistic model. In particular, this dissertation focuses on a specific family of models for environmental acoustical signals, namely Hierarchical Autoregressive Models. In principle, the flexibility of these models supports describing complex non-stationary acoustic environments. Unfortunately, Bayesian parameter estimation in these models is not trivial, and inference

solutions do not exist in closed-form. Therefore, this work develops methods to automate Bayesian inference for both state and parameter updating in hierarchical autoregressive models.

The contributions of this thesis are the following. First, we explore different hierarchical autoregressive models such as continuous time-varying, switching, and coupled autoregressive models. We cast these models into a factor graph framework that provides a convenient visualization of the models. We show that hierarchical models build on a network of special building blocks that can be re-used to increase the expressiveness of other dynamical models. Second, we realize Bayesian inference by an efficient message passing-based algorithm on these probabilistic factor graphs. We obtain closed-form message passing update rules for hierarchical autoregressive models. Third, closing in on the final application, we make use of the developed tools for efficient inference in hierarchical autoregressive models to build a synthetic agent that tunes hearing aid parameters under situated conditions. The developed agent solves the classification of acoustic context, infers optimal trial design, and executes the HA signal processing algorithm all by automated Bayesian inference.

In summary, this thesis provides a generic framework for hybrid, efficient and automatable Bayesian inference on probabilistic graphical models representing hierarchical autoregressive models. All derivations for the inference procedures have been added to the open-source Julia package `ReactiveMP.jl` that focuses on efficient and scalable Bayesian inference.

Бабушке посвящаю.

Contents

List of Symbols

Mathematical notation

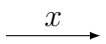
x	A scalar variable
$\boldsymbol{x}$	A vector variable
x_i	The *i*-th element of a vector $\boldsymbol{x}$
$\boldsymbol{x}_{1:n}$	A sequence up to and including index n
$\boldsymbol{X}$	A matrix variable
$\boldsymbol{X}^\top$	The transpose of matrix $\boldsymbol{X}$
$\boldsymbol{X}^{-1}$	The inverse of matrix $\boldsymbol{X}$
$x_{i,j}$	The *i,j*-th element of matrix $\boldsymbol{X}$
f	A factor function
p	A probability distribution
q	A variational distribution
$\vec{\mu}$	A forward sum-product message
$\vec{\nu}$	A forward variational message
$\mathbb{E}$	Expectation operator
Var	Variance
Cov	Covariance
tr	The matrix trace operator
$\log$	The natural logarithm
$\boldsymbol{e}_i$	The Cartesian unit vector where i-th element is 1
F	A Free energy functional
U	An average energy functional
H	The Entropy functional
m	A model
m_x	The mean of the univariate random variable x, i.e. $\mathbb{E}[x]$
v_x	The variance of the univariate random variable x, i.e. $\mathrm{Var}[x]$
$m_{\boldsymbol{x}}$	The mean of the multivariate random variable $\boldsymbol{x}$, i.e. $\mathbb{E}[\boldsymbol{x}]$
$V_{\boldsymbol{x}}$	The covariance of the multivariate random variable $\boldsymbol{x}$, i.e. $\mathrm{Cov}[\boldsymbol{x}]$
$m_{\boldsymbol{X}}$	The mean of the matrix-variate random variable $\boldsymbol{X}$, i.e. $\mathbb{E}[\boldsymbol{X}]$
I_M	Identity matrix of size $(M \times M)$.
$V(\gamma)$	Covariance matrix of the autoregressive model containing only one non-zero except for the first element, which is $1/\gamma$.

Probability distributions

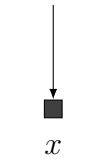
$\delta(y-m)$	Dirac delta centered on m
$\mathcal{N}(m,v)$	Gaussian distribution with mean m and v
$\Gamma(a,b)$	Gamma distribution with shape a and rate b
$\mathcal{W}(\boldsymbol{V},n)$	Wishart distribution with scale matrix $\boldsymbol{V}$ and degrees of freedom n
$\mathrm{Ber}(\pi)$	Bernoulli distribution with event probability π
$\mathrm{Cat}(\boldsymbol{\pi})$	Categorical distribution with event probability vector $\boldsymbol{\pi}$
$\mathrm{Dir}(\boldsymbol{\zeta})$	Dirichlet distribution with a concentration parameters vector $\boldsymbol{\zeta}$

Forney-style factor graph notation

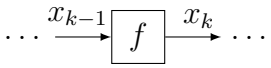
A factor node with node function f.



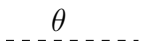
An edge that corresponds to a random variable x. Arrowheads on edges do not imply causal direction but only anchor message direction (forward with the arrow and backward against the arrow).



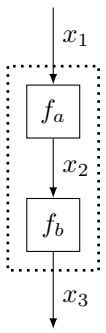
An edge that is clamped by an observation.



Three dots indicate a continuation of the displayed section.



Dashed edges indicate an equality constraint that extends over sections; parameters are usually denoted like this.



Subgraphs enclosed by dotted edges are summarized in a single composite node; in this case the composite node function is obtained by marginalizing over x_2.

Abbreviations

AR	Autoregressive
BFE	Bethe Free Energy
BI	Bayesian Inference
GMM	Gaussian Mixture Model
GM	Gaussian Mixture
HA	Hearing Aid
HAR	Hierarchical AR
DHA	Digital Hearing Aid
EEG	Electroencephalography
FE	Free Energy
FEP	Free Energy Principle
ML	Machine Learning
VAD	Voice Activity Detection
VFE	Variational Free Energy
VI	Variational Inference
TVAR	Time-varying AR
ΓMM	Gaussian Mixture Model
ΓM	Gaussian Mixture Model

Chapter 1

General Introduction

"If you are trying to look carefully at all equations which define conditional probability, you can understand something about reality; more than from your fantasy."

–Vladimir Vapnik

1.1 Motivation

Over the last few decades, research and development of machine learning (ML) techniques have led to scientific discoveries in diverse areas such as material science [1], astronomy [2], biology [3], chemistry [4], and mathematics [5]. ML also facilitated various engineering innovations, such as intelligent virtual assistants [6], dynamic robots [7], and autonomous vehicles [8]. ML helps in the medical and healthcare industries [9, 10]. Moreover, ML tools have immensely improved drug discovery [11] as well as the diagnosis of various rare diseases [12].

Generally speaking, in order for healthcare and other industries to benefit from ML-based algorithms, large amounts of training data must be available [13, 14]. Unfortunately, for many problems, these large amounts of data are not available or too costly to acquire, especially for situations that require personalized solutions. To cope with these issues, for several years, the members of the BIASlab research group[1] have been developing ML methods and tools to support automated, *situated* design of signal processing algorithms [15, 16, 17, 18]. The term "situated" here means that the training of these models takes place in real-time under in-situ conditions, and consequently, the reliance on a large training database is drastically reduced. In particular, much attention has been paid to situated design of hearing

[1] This thesis work was executed as a member of BIASlab research team, see `http://biaslab.org`.

aid (HA) algorithms. The ultimate goal of that research thread is to create personalized HA's automatically and solely (i.e., without pre-training on a large database) through in-the-field communication between a HA user and a portable ML agent. This dissertation describes a research effort to bring us closer to this goal.

In short, chapter 5 describes our design of said Bayesian agent that supports situated design of a HA algorithm. Chapters 2, 3 and 4 develop the needed Bayesian methods and tools in order to make the design of the Bayesian agent possible. In practice, this means that most of the work in this thesis will be focused on approximating Bayesian inference in hierarchical autoregressive models that we use for modeling acoustic signals.

We now return to motivate our problem. It is hard to overlook how hearing enriches our lives. Loss of auditory perception at different levels is believed to provoke both psychological [19] and neurological disorders [20]. Hearing loss (HL) can occur to almost anyone due to genetics, perinatal problems, environmental factors such as loud music, or simply aging processes [21, 22]. HA's are a great option that help mitigate HL. Unfortunately, the current design cycle of personalized commercially available HA's leaves much to be desired.

To illustrate this, imagine the following scenario. Suppose a HA patient becomes dissatisfied with her current HA while going about her business, e.g., walking in a busy traffic environment. In this case, she would typically go to an audiologist who will try to adjust the HA parameters to satisfy the patient. Tuning a HA is complicated though as patients experience HL with very individualized characteristics and degrees of severity. There is no fixed HA setting that satisfies a person in every environmental setting [23]. Modern digital HA algorithms are very complex and feature many signal processing modules, including voice activity detection (VAD), speech enhancement, noise reduction, feedback cancellation, and acoustic scene classification [24]. Therefore, an audiologist cannot perform an exhaustive search over all the tuning parameters of a HA for a single client. It would take years to traverse the whole space of HA parameters for a single client [25]. Not to mention that this approach would require a lot of effort from the user as she would likely have to submit an appraisal for each algorithm setting, which is an undesirable cognitive burden on the patient.

Users change their acoustic environment multiple times throughout the day. For example, they may move from and to their home, car, office, grocery store, train, bar, etc. These environments have different acoustic properties, such as echoes and reverberations. Therefore, it is not surprising that HA users favor different settings in different acoustic environments [26].

If the audiologist cannot resolve the problem manually by adjusting the parameters of the HA, the patient would have to wait for new algorithm releases from the HA manufacturer. These updates usually occur very infrequently and the newly proposed solution will likely be too late to satisfy the HA patient. The design of a new commercial HA algorithm must be accompanied by randomized controlled

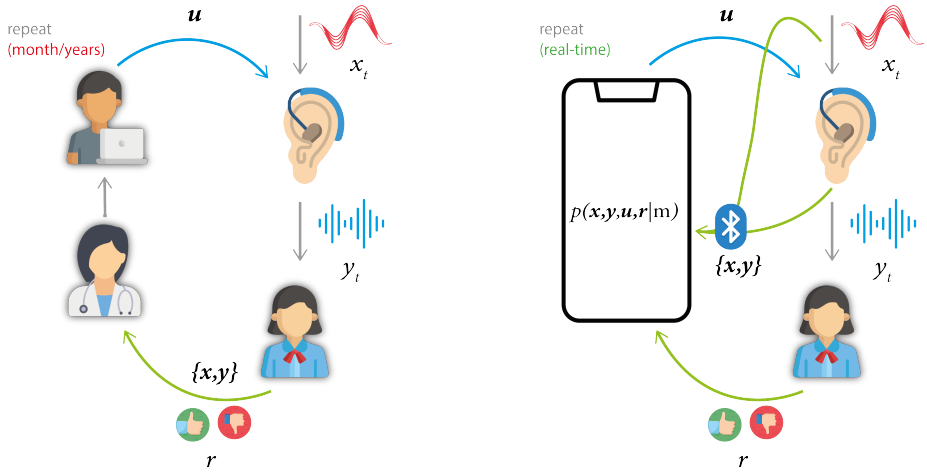


Figure 1.1: Comparison of the traditional and situated design cycle of HA algorithms. x_t and y_t are the input and output signals of a HA, respectively. r corresponds to appraisals by a human client. u denotes new settings for the HA. $\{\boldsymbol{x}, \boldsymbol{y}\}$ corresponds to the collection of input and output signals. (Left) In the traditional HA design cycle, the client refers to the audiologist for help, who, if necessary, turns to the engineers who develop new solutions for the HA client. This process often takes a very long time due to the number of human professionals in the design loop. (Right) The proposed *situated design cycle* of HA algorithms. In this case, professionals such as audiologists and engineers have been substituted by an AI agent that proposes sensible settings for the HA in real-time. The HA client becomes the center of the design loop by providing appraisals to the agent under in-situ conditions.

trials, which severely slows down the release cycle of HA solutions.

In short, the need for many many human professionals in the loop, as well as the large parameter space, leads to a rate of about 20% of HA patients that remain not fully satisfied with their HA's [27].

To improve these slow, inefficient (especially from the patient's point of view) design loops, HA's should ideally be tuned online, under *situated* conditions. In that case, the audiologist or engineer will not be present at the scene where the problems occur. Hence, we strive to substitute the "professional" HA practitioner by a synthetic real-time agent that interacts in real-time, under situated conditions, with a HA user (see Figure 1.1).

The central question then is how to design an agent that can intelligently tweak the tuning parameters of a hearing aid in real-time. Let us first discuss the task of such an agent. In general, a HA design agent may propose new HA settings

under two circumstances: Firstly (the reactive case), whenever the HA user is unhappy with the current performance of her HA, she is allowed to submit a negative appraisal to the agent, who in turn should respond instantly by changing the HA parameters to alternative settings. Secondly (the proactive case), the agent should update the HA settings without a user prompt if it anticipates that the user *will be* unhappy if no changes were made.

This type of HA design agent is challenged by a very difficult task. Assume that the HA has 10 tuning parameters and that we have 5 interesting settings for each parameter. The total number of parameter settings is then 5^{10}, which is equivalent to about 10 million different settings. Clearly, we do not want the agent to traverse randomly through such a large parameter space. In other words, the agent must first *learn* online the desired preferences for each acoustic environment of the HA patient and then take advantage of this knowledge by proposing the most interesting HA settings.

Apparently, the task of the agent exhibits a typical "exploration-exploitation" trade-off that is characteristic of reinforcement learning problems: the agent's proposals for a new HA setting should balance an information-seeking goal (to learn the preferences of the user) versus a utility-driven goal (to take advantage of this knowledge and propose the more preferred HA settings). Unfortunately, known implementations of reinforcement learning are famous for using many training examples.

In this thesis, we seek to further develop a new idea for these types of online learning problems. Our approach is motivated by the Free Energy Principle (FEP) [28]. The FEP and its realization by Active Inference (AIF), is a neuro-scientific theory that describes how living brains control their environment effectively by online learning of a model of their environment. At the BIASlab team, we work on transferring ideas from FEP-based learning to engineering systems. Synthetic FEP-based agents been successfully applied to multiple engineering domains such as robotics [29], reinforcement learning [30], neuroscience simulations [31], and audio signal processing [32].

Crucially, under the FEP protocol, agents learn a generative probabilistic model of their environment and plan future actions to maintain the states in that model to preferred settings. For instance, a fish will take actions to keep itself in the water, and a human will similarly take actions that are supportive of being alive, e.g., seek to drink something when thirsty. FEP is consistent with *the good regulator theorem* [33], which states that "every good regulator of a system must be a model of that system." In other words, a good, intelligent agent must embrace a model of its environment [33]. If an agent cannot make sense of its world, we cannot expect it to act upon it effectively. In the case of creating an effective FEP-based HA agent, the FEP claims that the agent needs to maintain a generative probabilistic model of its environment, which in this case are acoustic signals from the HA and user appraisals from the HA patient, see Figure 1.1.

A generative model for user responses is further discussed in chapter 5, but it is not the primary focus of this dissertation. In this dissertation, we focus on modeling the acoustic environment.

For a FEP-based HA agent, the acoustic environment may consist of speech and noise signals. In general, acoustic signals are hierarchically organized. Speech signals break down into sentences composed of phonemes, glottal pulses, and formants [34]. Noises depend on the acoustic context that usually evolves slowly over time, as the user can always move from one place to another. Each layer of these hierarchically structured signals operates at different time scales, e.g., acoustic context changes slower than speech. To account for all these features of the signals, this dissertation focuses on a specific class of models, namely hierarchical autoregressive (HAR) models.

In summary, the work in this thesis is ultimately motivated by a desire to lubricate the HA design cycle by introducing a machine learning-based agent that supports the real-time, situated design of HA algorithms. The task for the agent is very challenging, and our research interest is focused on transferring a theory about how brains control the world (the FEP) to engineering systems. In practice, this means we need to focus on developing generative probabilistic models for acoustic signals and user appraisals for HA behavior. We focus on the first task, namely, the development of generative probabilistic models for acoustic signals. The essence of the approach is that all engineering tasks (state estimation, parameter estimation, inferring the most interesting next HA setting, etc.) can all be framed as a Bayesian inference task on the generative model. We will focus on developing methods to automate these inference processes.

Next, we shortly discuss the focus on HAR models in this thesis.

1.2 Hierarchical autoregressive models

Autoregressive (AR) models have been known for decades in the statistical [35] and signal processing and control literature [36]. These models have been successfully used to model a wide variety of time series such as observations from texture representations [37], communication [38], EEG [39], and speech signals [40]. In a nutshell, an AR model is a model that predicts future observations of a system based on previous observations, plus a stochastic component (often referred to as noise) to account for modeling errors. Technically, the AR model is specified by

$$x_t = \sum_{k=1}^{K} \theta_k x_{t-k} + n_t$$

where $x_t, \theta_k, n_t \in \mathbb{R}$ are the states, AR coefficients, and the noise signal, respectively.

A hierarchical AR (HAR) model is an extension of an AR model, where the coefficients and/or noise signal are modeled by some "superior" hierarchical process. For example, one can think of letting AR coefficients slowly vary in time, yielding a continuously time-varying autoregressive (TVAR) process. We discuss the TVAR model in detail in Chapter 2. The flexibility of HAR models makes it possible to extend the class of time series that AR models can predict. Unlike AR models with fixed coefficients, HAR models support modeling of non-stationary signals such as speech [41] or cardiovascular responses [42].

In order to use a HAR model for modeling an environmental process inside a synthetic FEP agent, we need to represent the HAR model as a generative probabilistic model. In this dissertation, we favor a state-space model (SSM) representation of HAR models. We will introduce an SSM description for the HAR model in Chapter 2.

Principally, estimation and tracking of hidden states and parameters in generative probabilistic models can be realized through Bayesian inference (BI). As will be shown in Chapter 2, implementing BI in a HAR model is not straightforward. This dissertation will, for a large part, focus on the realization of efficient automatable Bayesian inference in HAR models.

Next, we shortly review the essence of Bayesian inference.

1.3 Bayesian Inference

BI rests on Bayes' rule, which provides a recipe for updating one's beliefs about quantities of interest when relevant new data becomes available. For example, in the context of HA design, the relevant information may comprise a sum of acoustic signals, including speech and babble noise. Let us assume a generative probabilistic model $p(y, x) = p(y|x)p(x)$, where y and x refer to the received sum-of-acoustic-signals and an unobserved (i.e., latent) constituent speech signal, respectively. Now, after receiving a specific signal $y = \hat{y}$, we should use Bayes' rule to infer the constituent speech signal x by

$$\underbrace{p(x|\hat{y})}_{\text{posterior}} \cdot \underbrace{\int p(\hat{y}|x)p(x)\mathrm{d}x}_{\text{evidence } p(\hat{y})} = \underbrace{p(\hat{y}|x)}_{\text{likelihood}} \cdot \underbrace{p(x)}_{\text{prior}} \tag{1.1}$$

Note that the inference process makes use of both the model assumption and the observations $y = \hat{y}$. The right-hand side of equation 1.1 states the model assumption and substitutes the observations $\hat{y}$ into that model. According to the product rule of Probability Theory, the right-hand side is mathematically equivalent to the left-hand side. The left-hand side comprises two factors. The factor $p(x|\hat{y})$ is called the posterior distribution for signal x, and it describes our state-of-knowledge, expressed by a probability distribution, about the speech signal x after having ob-

served $\hat{y}$. The second factor $p(\hat{y})$ is called model evidence, which can be interpreted as a model performance criterion. The evidence is theoretically computed by integrating all latent variables (x) from the generative model. *The computational challenge of BI is to compute the model representation as posterior times evidence from the likelihood times prior representation.*

Unfortunately, the employment of Bayes' rule within models that exhibit time-varying hierarchical structures often results in an analytically intractable posterior distribution [43, 44]. The intractability of the posterior distribution may occur as a result of (1) the need to integrate over a very large state space in the evidence term or (2) non-conjugate prior-posterior pairing[2]. To cope with these issues, the BI community has developed various tractable approximate inference techniques, such as variational inference (VI) [45] and Monte Carlo sampling-based methods [46].

Inference based on Monte Carlo sampling is a computationally demanding procedure. Consequently, its deployment as a real-time inference method in HAR models on small, low-power devices such as HA's is infeasible. Wearable devices with limited computational resources such as HA's cannot perform inference-by-sampling in real-time for any other than the simplest models. In contrast, Variational Inference (VI) casts Bayesian inference to an optimization problem (Chapter 2) that generally can be (partially) solved by a much lower computational load. If computational resources were not an issue, VI is less accurate than sampling-based inference. However, given our long-term goal to realize these methods in a wearable device, VI is still the more attractive option since it is faster and scales easier to inference in large models.

1.4 Research questions

This dissertation aims to establish a principled approach to building HAR models and running Bayesian inference within these models. The general question considered in this dissertation can be formulated as follows:

> *How can Bayesian inference be realized for hierarchical autoregressive models for signal processing applications?*

To answer this question, we will decompose each HAR model into sub-modules and represent the model by a factor graph. A factor graph represents a factorized probability distribution in the form of interconnected modules (*nodes*). Factor graphs provide a convenient visualization of the model. More importantly, factor graphs come together with a formal framework for efficient (variational) Bayesian inference, which is commonly named message-passing-based inference [47, Chapter 8].

[2] A conjugate prior and likelihood pairing leads to a closed-form solution for the posterior distribution, which is from the same distribution family as the prior distribution.

A second very important property of factor graphs is that the sub-modules and associated message passing rules can be stored in a table and re-used to create novel models, along with the inference processes for these models.

In this thesis, we choose a particular factor graph style, namely the Forney-style Factor Graph (FFG) framework, where the factors and variables of the factorized model are represented by nodes and edges, respectively. A more detailed description of factor graphs and message passing-based inference will be provided in chapter 2.

As a first milestone, we will focus on time-varying autoregressive (TVAR) models. The TVAR model is an AR model that allows coefficients to vary slowly over time. This (seemingly simple) extension allows TVAR models to process considerably more complex signals than the conventional AR model. A TVAR model can also be viewed as a subclass of HAR models that forms a hierarchy through higher-layer models for the time-varying coefficients. In principle, we do not want to limit ourselves to a fixed number of hierarchies in TVAR. Instead, we want to obtain a flexible solution for TVAR models such that they can be extended to an arbitrary number of hierarchical levels. However, this flexibility opens "Pandora's box" - the evidence term and consequently posterior distribution for states and parameters in TVAR model becomes analytically intractable (Chapter 2, Section 2.2).

As a result, we have to resort to an approximate inference method that delivers proxies for both model performance (the evidence term) and posterior distributions for the states and parameters of TVAR models. These considerations lead to the first concrete research question of this dissertation:

Q1. How can approximate Bayesian inference be implemented for time-varying autoregressive models?

We will answer this question in Chapter 2 by representing the TVAR model by an FFG [48].

We will localize the inference problem by examining a graph structure corresponding to the TVAR model. We will show that essentially the TVAR model builds on a network of special building blocks that we will label "AR nodes".

We will demonstrate how different TVAR models can be built by stacking multiple AR nodes. Most importantly, we will develop a formal procedure that allows us to make hierarchical models in the following chapters (Chapter 2, Section 2.4).

The next class of hierarchical autoregressive models we will explore is the Switching AutoRegressive (SwAR) model. SwAR models are well suited to model non-stationary regime-switching signals.

We will develop a variational message passing-based inference method on the FFG representation of the TVAR model. In variational inference, the evidence term is replaced by an approximation that in the machine learning literature is known as "Evidence Lower Bound" (ELBO), or equivalently in the physics literature as (negative) "variational Free Energy."

A simple example of a regime switch in a signal could be a transition between two acoustic environments, e.g., when moving from a train station to a crowded street. A conventional AR can track the signals induced by these acoustic environments separately. The SwAR model extends the functionality of AR models by introducing regime-switching dynamics. In other words, SwAR imposes a transition distribution on the parameters of the AR model. Usually, the parameters of the SwAR model are not known a priori, which implies that we need to assign a prior distribution to them. In the case of the TVAR model, we use the Normal and Gamma distributions as priors for the AR coefficients and parameters of the noise source model, respectively. It is natural to assume that the parameters of a SwAR model are generated from a mixture of a finite number of these distributions with unknown parameters. For example, the Gaussian mixture model (GMM) can specify a prior distribution for the AR coefficients. As for the noise parameters prior, we would need a Gamma mixture model (ΓMM).

Since our inference procedure is based on message passing in an FFG, we need to have access to FFG nodes for both the GMM and ΓMM. An FFG node for the GMM and its message passing-based inference procedure was developed in [25]. Unfortunately, similar work is not available for the ΓMM. Therefore, in this thesis, we focus on FFG-based inference for the Gamma mixture model (ΓMM) to ensure that we can use it to build an FFG representation of SwAR model. This leads to the second concrete research question:

Q2. How can Bayesian inference be implemented for tracking hidden states and parameters in a Gamma mixture model?

As in the case of TVAR models, we will develop the inference equations of the ΓMM model in the context of factor graphs. We will propose two inference algorithms for this model, each with advantages and shortcomings. As a result of answering **Q2**, we will obtain another module (similar to the AR node), namely a ΓM node, which we will use to build a SwAR model in an FFG representation. We will need to correctly integrate this node, along with the GMM node, into the SwAR model and make sure that BI in SwAR works as intended. This will lead to the third concrete research question of this dissertation:

Q3. How can approximate Bayesian inference be implemented in switching autoregressive models?

Armed with ΓM and AR nodes, we can proceed with modeling complicated acoustic environments that exhibit both continuously time-varying and regime-switching behaviors. In the motivation Section 1.1, we discussed the problem of automating the situated design of HA algorithms by FEP agents. To conclude this thesis, making use of the developed tools for efficient inference in HAR models, we will attempt in Chapter 5 to build an FEP-based agent that tunes HA parameters

under situated conditions. To achieve this goal, we need to construct an agent that comprises a model of its environment, i.e., a model for both user preferences and acoustic signals such as speech and noise signals. We also must design a protocol for interactions between the HA agent, its client, and the environment. These challenges lead to the final concrete research question:

Q4. How can hierarchical autoregressive models support the development of novel personalized hearing aid algorithms?

In the next chapters of this thesis, we attempt to develop solutions to the research questions **Q1**-**Q4**. Overall, our proposed methods will constitute a modular Bayesian approach to the situated design of signal processing algorithms, particularly when the environmental signals can be effectively modeled by HAR models.

1.5 Summary of Contributions

The following list presents a high-level overview of the novel contributions described by this dissertation:

- Development of a low-complexity message passing-based inference procedure for the tracking of states, parameters and free energy in latent time-varying autoregressive models (TVAR). (Chapter 2)
- Development of message passing-based inference in Gamma Mixture models (ΓMM). (Chapter 3)
- Realization of message passing-based inference for switching autoregressive models (SwAR). (Chapter 4)
- A probabilistic generative modeling approach to design of novel hearing aid algorithms. (Chapter 5)

1.6 Outline of the Dissertation

Chapter 2 addresses **Q1** by presenting a TVAR model using a Forney-style factor graph. An automated message passing algorithm handles the inference problem for states and parameters. We introduce a composite "AR node" with probabilistic observations that can be used as a plug-in module in complex hierarchical models. Our proposed solution for online model scoring includes tracking variational free energy (FE) as a Bayesian measure of TVAR model performance.

Chapters 3 and 4 explore the extension of hierarchical autoregressive models to context-switching regimes. *Chapter 3* serves as a prerequisite for building SwAR

models. By addressing **Q2**, this chapter develops a "Gamma Mixture node" that can operate as a prior for the precision parameter of a Gaussian distribution. Two variants of variational message passing-based inference in a Gamma mixture model are proposed. Finally, this chapter shows how the Gamma Mixture node can be used as a building block for modeling both univariate and multivariate observations. *Chapter 4* approaches **Q3** by introducing a fully Bayesian SwAR model that includes Gaussian mixture and Gamma mixture models. The SwAR model is well-suited to model regime switches in environmental acoustic signals.

Chapter 5 addresses **Q4** by fusing the results of previous chapters, yielding a model that encompasses both continuously time-varying and switching behavior of acoustic environments. Additionally, we develop an FEP-based agent that iteratively designs a personalized audio processing algorithm through situated interactions with a human client. The generative model of the FEP agent will be represented by a factor graph. All engineering tasks (parameter learning, acoustic context classification, trial design, etc.) are phrased as inference tasks on the generative model and can be automatically realized by a hybrid message passing on the factor graph.

Chapter 6 reflects on the results obtained in the dissertation and offers a perspective for future research on the topic.

Chapter 2

Message Passing-based Inference in Time-Varying Autoregressive Models

This chapter is based on the original work referenced below. Notations have been adjusted to reflect conventions throughout the dissertation.

> **Albert Podusenko**, Wouter M. Kouw, Bert de Vries, *Message Passing-based Inference for Time-Varying Autoregressive Models*, Special issue on Bayesian Inference in Probabilistic Graphical Models, Entropy, 2021

Abstract

Time-varying autoregressive (TVAR) models are widely used for modeling non-stationary signals. Unfortunately, the online joint adaptation of both states and parameters in these models remains challenging. In this paper, we represent the TVAR model by a factor graph and solve the inference problem by automated message passing-based inference for states and parameters. We derive structured variational update rules for a composite "AR node" with probabilistic observations that can be used as a plug-in module in hierarchical models, for example, to model the time-varying behavior of the hyper-parameters of a time-varying AR model. Our method includes tracking variational free energy (FE) as a Bayesian measure of TVAR model performance. The proposed methods are verified on a synthetic data set and validated on real-world data from temperature modeling and speech enhancement tasks.

2.1 Introduction

Autoregressive (AR) models are capable of describing a wide range of time series patterns [49, 50]. The extension to Time-Varying AR (TVAR) models, where the AR coefficients are allowed to vary over time, supports the tracking of non-stationary signals. TVAR models have been successfully applied to a wide range of applications, including speech signal processing [51, 41, 52], signature verification [53], cardiovascular response modeling [42], acoustic signature recognition of vehicles [54], radar signal processing [55], and EEG analysis [56, 57].

The realization of TVAR models in practice often poses some computational issues. For many applications, such as speech processing in a hearing aid, both a low computational load and high modeling accuracy are essential.

The problem of parameter tracking in TVAR models has been extensively explored in a non-Bayesian setting. For example, ref. [58] employs over-determined modified Yule-Walker equations and [59] applies the covariance method to track the parameters in a TVAR model. In [60], expressions for the mean vector and covariance matrix of TVAR model coefficients are derived, and [61] uses wavelets for TVAR model identification. Essentially, all these approaches provide maximum likelihood estimates of coefficients for TVAR models without measurement noise. In [62], autoregressive parameters were estimated from noisy observations by using a recursive least-squares adaptive filter.

We take a Bayesian approach since we are also interested in tracking Bayesian evidence (or an approximation thereof) as a model performance measure. Bayesian evidence can be used to track the optimal AR model order or, more generally, to compare the performance of a TVAR model to an alternative model. To date, Bayesian parameter tracking in AR models has mostly been achieved by Monte Carlo sampling methods [63, 64, 65, 66, 67]. The sampling-based inference is highly accurate, but it is often computationally too expensive for real-time processing on wearable devices such as hearing aids, smart watches, etc.

In this paper, we develop a low-complexity variational message passing-based (VMP) realization for tracking states, parameters, and free energy (an upper bound on Bayesian evidence) in TVAR models. All update formulas are closed-form, and the complete inference process can easily be realized.

VMP is a low-complexity distributed message passing-based realization of variational Bayesian inference on a factor graph representation of the model [68, 45]. Previous work on message passing-based inference for AR models include [69], but their work describes maximum likelihood estimation and therefore does not track proper posteriors and free energy. In [70], the variational inference is employed to estimate the parameters of a multivariate AR model, but their work does not take advantage of the factor graph representation.

The factor graph representation we employ in this paper provides some distinct advantages from other works on inference in TVAR models. First, a factor graph

formulation is by definition completely modular and supports re-using the derived TVAR inference equations as a plug-in module in other factor graph-based models. In particular, since we allow for measurement noise in the TVAR model specification, the proposed TVAR factor can easily be used as a latent module at any level in hierarchical dynamical models. Moreover, due to the modularity, VMP update rules can easily be mixed with different update schemes such as belief propagation and expectation [48, 71] in other modules, leading to hybrid message passing schemes for efficient inference in complex models. We have implemented the TVAR model in the open source and freely available factor graph toolbox `ForneyLab` [72] and `ReactiveMP` [73].

The rest of this paper is organized as follows. In Section 2.2, we specify the TVAR model as a probabilistic state space model and define the inference tasks that relate to tracking of states, parameters, and Bayesian evidence. After a short discussion on the merits of using Bayesian evidence as a model performance criterion (Section 2.3.1), we formulate Bayesian inference in the TVAR model as a set of sequential prediction-correction processes (Section 2.3.2). We will realize these processes as VMP update rules and proceed with a short review of Forney-style factor graphs and message passing in Section 2.4. Then, in Section 2.5, the VMP equations are worked out for the TVAR model and summarized in Table 2.1. Section 2.6 discusses a verification experiment on a synthetic data set and applications of the proposed TVAR model to temperature prediction and speech enhancement problems. Full derivations of the closed-form VMP update rules are presented in Appendix A.1.

2.2 Model Specification and Problem definition

In this section, we first specify TVAR model as a state-space model. This is followed by an inference problem formulation.

2.2.1 Model Specification

A TVAR model is specified as

$$\theta_{kt} \sim \mathcal{N}(\theta_{kt-1}, \omega) \tag{2.1a}$$

$$x_t \sim \mathcal{N}\Big(\sum_{k=1}^{K} \theta_{kt} x_{t-k}, \gamma^{-1}\Big) \tag{2.1b}$$

$$y_t \sim \mathcal{N}(x_t, \tau)\,, \tag{2.1c}$$

where $y_t \in \mathbb{R}$, $x_t \in \mathbb{R}$ and $\theta_{k,t} \in \mathbb{R}$ represent the the observation, state and parameters at time t, respectively. K denotes the order of the AR model. As a notational

convention, we use $\mathcal{N}(\boldsymbol{m}, \boldsymbol{V})$ to denote a Gaussian distribution with mean $\boldsymbol{m}$ and co-variance matrix $\boldsymbol{V}$. We can re-write (2.1) in state-space form as

$$\boldsymbol{\theta}_t \sim \mathcal{N}(\boldsymbol{\theta}_{t-1}, \omega \mathrm{I}_K) \tag{2.2a}$$

$$\boldsymbol{x}_t \sim \mathcal{N}\Big(A(\boldsymbol{\theta}_t)\boldsymbol{x}_{t-1}, V(\gamma)\Big) \tag{2.2b}$$

$$y_t \sim \mathcal{N}(\boldsymbol{e}_1^\intercal \boldsymbol{x}_t, \tau)\,, \tag{2.2c}$$

where $\boldsymbol{\theta}_t = [\theta_{1t}, \theta_{2t}, ..., \theta_{Kt}]^\intercal$, $\boldsymbol{x_t} = (x_t, x_{t-1}, \ldots, x_{t-K+1})^\intercal$, $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^\intercal$ is an K-dimensional unit vector, $V(\gamma) = (1/\gamma)\boldsymbol{e}_1\boldsymbol{e}_1^\intercal$, and

$$A(\boldsymbol{\theta}) = \begin{bmatrix} \boldsymbol{\theta}^\intercal & \\ \mathrm{I}_{K-1} & \mathbf{0}\,. \end{bmatrix} \tag{2.3}$$

Technically, a TVAR model usually assumes $\tau = 0$, indicating no measurement noise. Note that the presence of measurement noise in (2.2c) "hides" the states $\boldsymbol{x}_t$ in the generative model (2.2) from the observation sequence y_t, yielding a latent TVAR. We add measurement noise explicitly so the model can accept information from likelihood functions that are not constrained to be delta functions with hard observations. As a result, the AR model we define here can be used at any level in deep hierarchical structures such as [74] as a plug-in module.

In a time-invariant AR model, $\boldsymbol{\theta}$ are part of the system's parameters. In a time-varying AR model, we consider $\boldsymbol{\theta}_t$ and $\boldsymbol{x}_t$ together the set of time-varying states. The parameters of the TVAR model are $\{\boldsymbol{\theta}_0, \boldsymbol{x}_0, \omega, \gamma, \tau\}$.

At the heart of the TVAR model is the transition model (2.2b), where $A(\boldsymbol{\theta}_t)$ is a companion matrix with AR coefficients. The multiplication $A(\boldsymbol{\theta})\boldsymbol{x}_{t-1}$ performs two operations: a dot product $\boldsymbol{\theta}_t^\intercal \boldsymbol{x}_{t-1}$ and a vector shift of $\boldsymbol{x}_{t-1}$ by one time step. The latter operation can be interpreted as bookkeeping, as it shifts each entry of $\boldsymbol{x}_{t-1}$ one position down and discards x_{t-K}.

2.2.2 Problem Definition

For a given time series $\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$, we are firstly interested in recursively updating posteriors for the states $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:l})$ and $p(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:i})$. In this context, prediction, filtering and smoothing are recovered for $i < t$, $i = t$ and $i > t$, respectively. Furthermore, we are interested in computing posteriors for the parameters $p(\boldsymbol{\theta}_0|\boldsymbol{y})$, $p(\boldsymbol{x}_0|\boldsymbol{y})$, $p(\omega|\boldsymbol{y})$, $p(\gamma|\boldsymbol{y})$ and $p(\tau|\boldsymbol{y})$.

Finally, we are interested in scoring the performance of a proposed TVAR model m with specified priors for the parameters. In this paper, we take a fully Bayesian approach and select Bayesian evidence $p(\boldsymbol{y}|m)$ as the performance criterion. Section 2.3.1 discusses the merits of Bayesian evidence as a model performance criterion.

2.3 Inference in TVAR Models

In this section, we first discuss some of the merits of using Bayesian evidence as a model performance criterion. This is followed by an exposition of computing Bayesian evidence and the desired posteriors in the TVAR model.

2.3.1 Bayesian Evidence as a Model Performance Criterion

Consider a model m with parameters $\boldsymbol{\theta}$ and observations y. Bayesian evidence $p(\boldsymbol{y}|\mathrm{m})$ is considered an excellent model performance criterion. Note the following decomposition [75]:

$$\begin{aligned}\log p(\boldsymbol{y}|\mathrm{m}) &= \log \frac{p(\boldsymbol{y}|\boldsymbol{\theta},\mathrm{m})p(\boldsymbol{\theta}|\mathrm{m})}{p(\boldsymbol{\theta}|\boldsymbol{y},\mathrm{m})} \qquad \text{(use Bayes rule)} \\ &= \int p(\boldsymbol{\theta}|\boldsymbol{y},\mathrm{m}) \cdot \underbrace{\log \frac{p(\boldsymbol{y}|\boldsymbol{\theta},\mathrm{m})p(\boldsymbol{\theta}|\mathrm{m})}{p(\boldsymbol{\theta}|\boldsymbol{y},\mathrm{m})}}_{\log p(\boldsymbol{y}|\mathrm{m}) \text{ is not a function of } \boldsymbol{\theta}} \mathrm{d}\boldsymbol{\theta} \\ &= \underbrace{\int p(\boldsymbol{\theta}|\boldsymbol{y},\mathrm{m}) \log p(\boldsymbol{y}|\boldsymbol{\theta},\mathrm{m}) \mathrm{d}\boldsymbol{\theta}}_{\text{data fit}} - \underbrace{\int p(\boldsymbol{\theta}|\boldsymbol{y},\mathrm{m}) \log \frac{p(\boldsymbol{\theta}|\boldsymbol{y},\mathrm{m})}{p(\boldsymbol{\theta}|\mathrm{m})} \mathrm{d}\boldsymbol{\theta}}_{\text{complexity}} \end{aligned} \tag{2.4}$$

The first term (data fit or sometimes called accuracy) measures how well the model predicts the data $\boldsymbol{y}$, after learning from the data. We want this term to be large (although only focusing on this term could lead to over-fitting). The second term (complexity) quantifies the amount of information that the model absorbed through learning by moving parameter beliefs from $p(\boldsymbol{\theta}|\mathrm{m})$ to $p(\boldsymbol{\theta}|\boldsymbol{y},\mathrm{m})$. To see this, note that the mutual information between two variables $\boldsymbol{\theta}$ and $\boldsymbol{y}$, which is defined as

$$I[\boldsymbol{\theta};\boldsymbol{y}] = \iint p(\boldsymbol{\theta},\boldsymbol{y}) \log \frac{p(\boldsymbol{\theta}|\boldsymbol{y})}{p(\boldsymbol{\theta})} \mathrm{d}\boldsymbol{\theta}\mathrm{d}\boldsymbol{y}\,,$$

can be interpreted as expected complexity. The complexity term regularizes the Bayesian learning process automatically. Preference for models with high Bayesian evidence implies a preference for models that get a good data fit without the need to learn much from the data set. These types of models are said to *generalize* well since they can be applied to different data sets without specific adaptations for each data set. Therefore, Bayesian learning automatically leads to models that tend to generalize well.

Note that Bayesian evidence for a model m, given a full times series $\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$, can be computed by multiplication of the sample-based evidences:

$$p(\boldsymbol{y}|\mathrm{m}) = \prod_{t=1}^{T} p(y_t|\boldsymbol{y}_{1:t-1},\mathrm{m})\,. \tag{2.5}$$

2.3.2 Inference as a Prediction-Correction Process

To illustrate the type of calculations needed for computing Bayesian model evidence and the posteriors for states and parameters, we now proceed to write out the required calculations for the TVAR model in a filtering context.

Assume that at the beginning of time step t, we are given the state posteriors $q(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1})$, $q(\boldsymbol{\theta}_{t-1}|\boldsymbol{y}_{1:t-1})$. We will denote the inferred probabilities by $q(\cdot)$, in contrast to factors from the generative model that are written as $p(\cdot)$. We start the procedure by setting the state priors for the generative model at step t to the posteriors of the previous time step

$$p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}) := q(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}) \tag{2.6}$$

$$p(\boldsymbol{\theta}_{t-1}|\boldsymbol{y}_{1:t-1}) := q(\boldsymbol{\theta}_{t-1}|\boldsymbol{y}_{1:t-1}) \tag{2.7}$$

Given a new observation y_t, we are now interested inferring the evidence $q(y_t|\boldsymbol{y}_{t-1})$, and in inferring posteriors $q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$ and $q(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:t})$.

This involves a prediction (forward) pass through the system that leads to the evidence update, followed by a correction (backward) pass that updates the states. We work this out in detail below. For clarity of exposition, in this section we call $\boldsymbol{x}_t$ states and $\boldsymbol{\theta}_t$ parameters. Starting with the forward pass (from latent variables toward observation), we first compute a parameter prior predictive:

$$\underbrace{q(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:t-1})}_{\substack{\text{parameter}\\ \text{prior predictive}}} = \int \underbrace{p(\boldsymbol{\theta}_t|\boldsymbol{\theta}_{t-1})}_{\substack{\text{parameter}\\ \text{transition}}} \underbrace{p(\boldsymbol{\theta}_{t-1}|\boldsymbol{y}_{1:t-1})}_{\substack{\text{parameter}\\ \text{prior}}} \mathrm{d}\boldsymbol{\theta}_{t-1}\,. \tag{2.8}$$

Then the prior predictive for the state transition becomes:

$$\underbrace{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}_{1:t-1})}_{\substack{\text{state transition}\\ \text{prior predictive}}} = \int \underbrace{p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{\theta}_t)}_{\text{state transition}} \underbrace{q(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:t-1})}_{\substack{\text{parameter}\\ \text{prior predictive}}} \mathrm{d}\boldsymbol{\theta}_t\,. \tag{2.9}$$

Note that the state transition prior predictive, due to its dependency on time-varying θ_t, is a function of the observed data sequence. The state transition prior predictive can be used together with the state prior to inferring the state prior predictive:

$$\underbrace{q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})}_{\substack{\text{state}\\ \text{prior predictive}}} = \int \underbrace{q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{y}_{1:t-1})}_{\substack{\text{state transition}\\ \text{prior predictive}}} \underbrace{p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1})}_{\text{state prior}} \mathrm{d}\boldsymbol{x}_{t-1}\,. \tag{2.10}$$

The evidence for model m that is provided by observation y_t is then given by

$$\underbrace{q(y_t|\boldsymbol{y}_{1:t-1})}_{\text{evidence}} = \int \underbrace{p(y_t|\boldsymbol{x}_t)}_{\substack{\text{state}\\ \text{likelihood}}} \underbrace{q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})}_{\substack{\text{state prior}\\ \text{predictive}}} \mathrm{d}\boldsymbol{x}_t\,. \tag{2.11}$$

When y_t has not yet been observed, $q(y_t|\boldsymbol{y}_{1:t-1})$ is a prediction for y_t. After plugging in the observed value for y_t, the evidence is a scalar that scores how well the model performed in predicting y_t. As discussed in (2.5), the results $q(y_t|\boldsymbol{y}_{1:t-1})$ for $t = 1, 2, \ldots, T$ in (2.11) can be used to score the model performance for a given time series $\boldsymbol{y} = (y_1, y_2, \ldots, y_T)$. Note that to update the evidence, we need to integrate over all latent variables $\boldsymbol{\theta}_{t-1}$, $\boldsymbol{\theta}_t$, $\boldsymbol{x}_{t-1}$ and $\boldsymbol{x}_t$ (by (2.8)–(2.11)). In principle, this scheme needs to be extended with integration over the parameters ω, γ, and τ.

Once we have inferred the evidence, we proceed by a backward corrective pass through the model to update the posterior over the latent variables given the new observation y_t. The state posterior can be updated by the Bayes rule:

$$\underbrace{q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})}_{\text{state posterior}} = \frac{\overbrace{p(y_t|\boldsymbol{x}_t)}^{\substack{\text{state}\\\text{likelihood}}}\overbrace{q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1})}^{\substack{\text{state prior}\\\text{predictive}}}}{\underbrace{q(y_t|\boldsymbol{y}_{1:t-1})}_{\text{evidence}}} \tag{2.12}$$

Next, we need to compute a likelihood function for the parameters. Fortunately, we can re-use some intermediate results from the forward pass. The likelihood for the parameters is given by

$$\underbrace{q(y_t|\boldsymbol{\theta}_t, \boldsymbol{y}_{1:t-1})}_{\substack{\text{parameter}\\\text{likelihood}}} = \int \underbrace{p(y_t|\boldsymbol{x}_t)}_{\substack{\text{state}\\\text{likelihood}}} \underbrace{q(\boldsymbol{x}_t|\boldsymbol{\theta}_t, \boldsymbol{y}_{1:t-1})}_{\substack{\text{state prior}\\\text{predictive}}} \mathrm{d}\boldsymbol{x}_t \tag{2.13}$$

The parameter posterior then follows from Bayes rule:

$$\underbrace{q(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:t})}_{\text{parameter posterior}} = \frac{\overbrace{q(y_t|\boldsymbol{\theta}_t, \boldsymbol{y}_{1:t-1})}^{\substack{\text{parameter}\\\text{likelihood}}}\overbrace{q(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:t-1})}^{\substack{\text{parameter}\\\text{prior predictive}}}}{\underbrace{q(y_t|\boldsymbol{y}_{1:t-1})}_{\text{evidence}}} \tag{2.14}$$

Equations (2.11), (2.12) and (2.14) contain the solutions to our inference task. Note that the evidence $q(y_t|\boldsymbol{y}_{1:t-1})$ is needed to normalize the latent variable posteriors in (2.12) and (2.14). Moreover, while we integrate over the states by (2.11) to compute the evidence, (2.14) reveals that the evidence can alternatively be computed by integrating over the parameters through

$$\underbrace{q(y_t|\boldsymbol{y}_{1:t-1})}_{\text{evidence}} = \int \underbrace{q(y_t|\boldsymbol{\theta}_t, \boldsymbol{y}_{1:t-1})}_{\substack{\text{parameter}\\\text{likelihood}}} \underbrace{q(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:t-1})}_{\substack{\text{parameter}\\\text{prior predictive}}} \mathrm{d}\boldsymbol{\theta}_t \,. \tag{2.15}$$

This latter method of evidence computation may be useful if re-using (2.11) in (2.14) leads to numerical rounding issues.

Unfortunately, many of Equations (2.8) through (2.14) are not analytically tractable for the TVAR model. This happens due to (1) integration over large state spaces, (2) non-conjugate prior-posterior pairing, and (3) the absence of a closed-form solution for the evidence factor.

To overcome this challenge, we will perform inference by a hybrid message passing scheme in a factor graph. In the next section, we give a short review of Forney-Style Factor Graphs (FFG), and Message-Passing (MP) based inference techniques.

2

2.4 Factor Graphs and Message Passing-Based Inference

In this section, we make a brief introduction of Forney-Style Factor graph (FFG) and sum-product (SP) algorithm. After that we review the minimization of variational free energy and Variational Message Passing (VMP) algorithm.

2.4.1 Forney-Style Factor Graphs

A Forney-style Factor graph is a representation of a factorized function where the factors and variables are represented by nodes and edges, respectively. An edge is connected to a node if and only if the (edge) variable is an argument of the node function. In our work, we use FFGs to represent factorized probability distributions. FFGs provide both an attractive visualization of the model and a highly efficient and modular inference method based on message passing. An important component of the FFG representation is the equality node. If a variable x is shared between more than two nodes, then we introduce two auxiliary variables x' and x'' and use an equality node

$$f_=(x, x', x'') = \delta(x - x')\delta(x - x'') \tag{2.16}$$

to constrain the marginal beliefs over x, x', x'' to be equal. With this mechanism, any factorized function can be represented as an FFG.

An FFG visualization of the TVAR model is depicted in Figure 2.3, but for illustrative purposes, we first consider an example factorized distribution

$$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_3) \tag{2.17}$$

This distribution can be visualized by an FFG shown in Figure 2.1. An FFG is, in principle, an undirected graph, but we often draw arrows on the edges in the "generative" direction, which is the direction that describes how the observed data

is generated. Assume that we are interested in computing the marginal for x_2, given by

$$p(x_2) = \iiint p(x_1, x_2, x_3, x_4)\mathrm{d}x_1\mathrm{d}x_3\mathrm{d}x_4 \tag{2.18}$$

We can reduce the complexity of computing this integral by rearranging the factors over the integration signs as

$$p(x_2) = \underbrace{\int \underbrace{p(x_1)}_{\vec{\mu}_1(x_1)} p(x_2|x_1)\mathrm{d}x_1}_{\vec{\mu}_2(x_2)} \cdot \underbrace{\Big(\int p(x_3|x_2) \underbrace{\Big(\int p(x_4|x_3)\mathrm{d}x_3 \Big)}_{\overleftarrow{\mu}_3(x_3)} \mathrm{d}x_3 \Big)}_{\overleftarrow{\mu}_2(x_2)} \tag{2.19a}$$

$$= \vec{\mu}_2(x_2) \cdot \overleftarrow{\mu}_2(x_2)\,. \tag{2.19b}$$

These re-arranged integrals can be interpreted as messages passed over the edges, see Figure 2.1. It is a notational convention to call a message $\vec{\mu}(\cdot)$ that aligns with the direction of the edge arrow a forward message and similarly, a message $\overleftarrow{\mu}(\cdot)$ that opposes the direction of the edge is called a backward message.

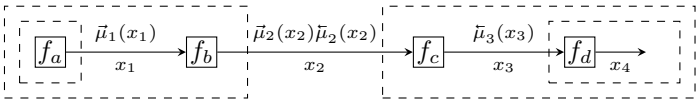


Figure 2.1: An FFG corresponding to model (2.17), including messages as per (2.19). For graphical clarity, we defined $f_a(x_1) = p(x_1)$, $f_b(x_1, x_2) = p(x_2|x_1)$, $f_c(x_2, x_3) = p(x_3|x_2)$ and $f_d(x_3, x_4) = p(x_4|x_3)$.

This message passed-based algorithm of computing the marginal is called belief propagation (BP) or the sum-product algorithm. As can be verified in (2.19), for a node with factor $f(y, x_1, \ldots, x_n)$, the outgoing BP message $\vec{\mu}(y)$ to variable y can be expressed as

$$\vec{\mu}_y(y) = \int \cdots \int f(y, x_1, \ldots, x_n) \prod_{i=1}^{n} \vec{\mu}_i(x_i)\mathrm{d}x_i\,. \tag{2.20}$$

where $\vec{\mu}_i(x_i)$ is an incoming message over edge x_i. If the factor graph is a tree, meaning that the graph contains no cycles, then BP leads to exact Bayesian inference. A more detailed explanation of belief propagation message passing in FFGs can be found in [48].

2.4.2 Free Energy and Variational Message Passing

Technically, BP is a message passing algorithm that belongs to a family of message passing algorithms that minimize a constrained variational free energy func-

tional [76]. Unfortunately, the sum-product rule (2.20) only has a closed-form solution for Gaussian incoming messages $\vec{\mu}_i(x_i)$ and linear variable relations in $f(y, x_1, \ldots, x_n)$. Another important member of the free energy minimizing algorithms is the Variational Message Passing (VMP) algorithm [68]. VMP enjoys a wider range of analytically computable message update rules.

We will shortly review variational Bayesian inference and VMP next. Consider a model $p(\boldsymbol{y}, \boldsymbol{x})$ with observations $\boldsymbol{y}$ and unobserved (latent) variables $\boldsymbol{x}$. We are interested in inferring the posterior distribution $p(\boldsymbol{x}|\boldsymbol{y})$. In variational inference we introduce an approximate posterior $q(\boldsymbol{x})$ and define a variational free energy functional as

$$F[q] \triangleq \int q(\boldsymbol{x}) \log \frac{q(\boldsymbol{x})}{p(\boldsymbol{y}, \boldsymbol{x})} \mathrm{d}\boldsymbol{x} = \underbrace{\int q(\boldsymbol{x}) \log \frac{q(\boldsymbol{x})}{p(\boldsymbol{x}|\boldsymbol{y})} \mathrm{d}\boldsymbol{x}}_{\text{KL divergence } D_{\mathrm{KL}}(q,p)} - \underbrace{\log p(\boldsymbol{y})}_{\text{log-evidence}} . \tag{2.21}$$

The second term in (2.21) (log-evidence) is not a function of the argument of F. The first term is a KL-divergence, which is by definition non-negative and only equals zero for $q(\boldsymbol{x}) = p(\boldsymbol{x}|\boldsymbol{y})$. As a result, variational inference by minimization of $F[q]$ provides

$$q^*(\boldsymbol{x}) = \arg\min_q F[q] \tag{2.22}$$

which is an approximation to the Bayesian posterior $p(\boldsymbol{x}|\boldsymbol{y})$. Moreover, the minimized free energy $F[q^*]$ is an upper bound for minus log-evidence and, in practice, is used as a model performance criterion. Similarly to (2.4), the free energy can be decomposed as

$$F[q] = \underbrace{\int q(\boldsymbol{x}) \log p(\boldsymbol{y}|\boldsymbol{x}, \mathrm{m}) \mathrm{d}\boldsymbol{x}}_{\text{accuracy}} - \underbrace{\int q(\boldsymbol{x}) \log \frac{q(\boldsymbol{x})}{\underbrace{p(\boldsymbol{x}|\mathrm{m})}_{\text{prior}}} \mathrm{d}\boldsymbol{x}}_{\text{complexity}} \tag{2.23}$$

which underwrites its usage as a performance criterion for model m, given observations $\boldsymbol{y}$.

In an FFG context, the model $p(\boldsymbol{y}, \boldsymbol{x})$ is represented by a set of connected nodes. Consider a generic node of the FFG, given by $f(y, x_1, \ldots, x_n)$ where in the case of VMP, the incoming messages are approximations to the marginals $q_i(x_i), i = 1, \ldots, n$, see Figure 2.2.

It can be shown that the outgoing VMP message of f towards edge y is given by [77]

$$\vec{\nu}(y) \propto \exp\left(\int \cdots \int \log f(y, x_1, \ldots, x_n) \prod_{i=1} q\left(x_i\right) \mathrm{d}x_i \right) . \tag{2.24}$$

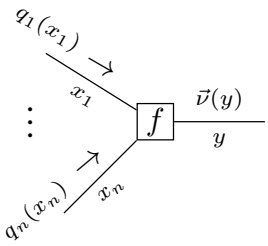


Figure 2.2: A generic node $f(y, x_1, \ldots, x_n)$ with incoming variational messages $q_i(x_i)$ and outgoing variational message $\vec{\nu}(y)$, see Equation (2.24). Note that the marginals $q(\cdot)$ propagate in the graph as messages.

In this paper, we adopt the notational convention to denote belief propagation messages (computed by (2.20)) by μ and VMP messages (computed by (2.24)) by ν. The approximate marginal $q(y)$ can be obtained by multiplying incoming and outgoing messages on the edge for y

$$q(y) \propto \vec{\nu}(y)\overleftarrow{\nu}(y)\,. \tag{2.25}$$

This process (compute forward and backward messages for an edge and update the marginal) is executed sequentially and repeatedly for all edges in the graph until convergence. In contrast to BP-based inference, the VMP and marginal update rules (2.24) and (2.25) lead to closed-form expressions for a large set of conjugate node pairs from the exponential family of distributions. For instance, updating the variance parameter of a Gaussian node with a connected inverse-gamma distribution node results in closed-form VMP updates.

In short, both BP- and VMP-based message passing can be interpreted as minimizing variational free energy, albeit under a different set of local constraints [76]. Typical constraints include factorization and form constraints on the posterior such as $q(\boldsymbol{x}) = \prod_i q_i(x_i)$ and $q(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$, respectively. Since the constraints are local, BP and VMP can be combined in a factor graph to create hybrid message passing-based variational inference algorithms. For a more detailed explanation of VMP in FFGs, we refer to [77]. Note that hybrid message passing does in general not guarantee to minimize variational free energy [76]. However, in our experiments in Section 2.6 we will show that iterating our stationary solutions by message passing does lead to free energy minimization.

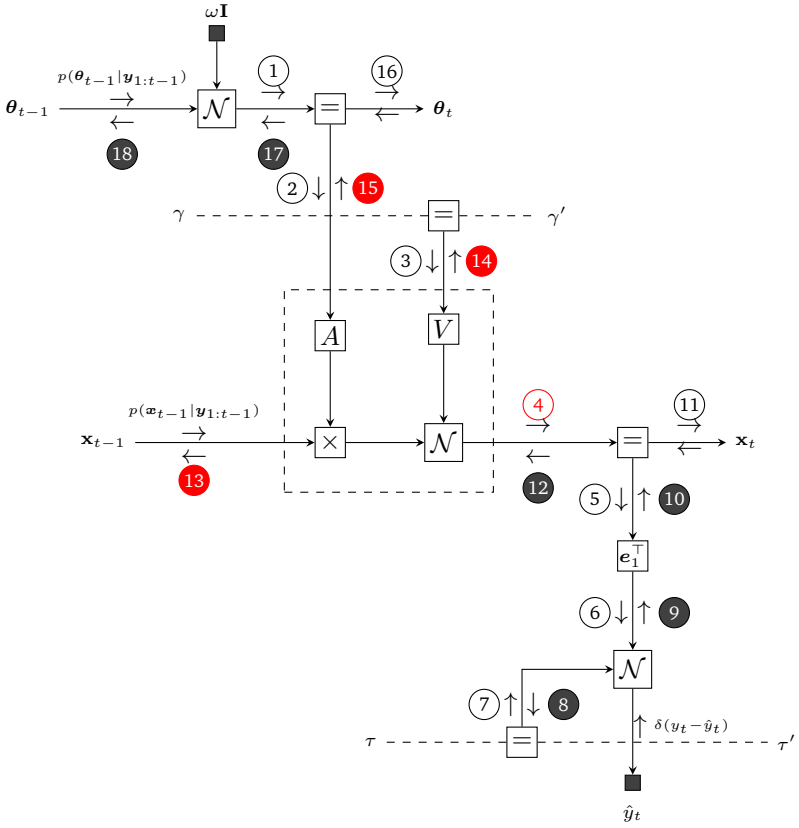


Figure 2.3: One time segment of an FFG corresponding to the TVAR model. We use small black nodes to denote observations and fixed given parameter values. The observation node for y_t sends a message $\delta(y_t - \hat{y}_t)$ into the graph to indicate that $y_t = \hat{y}_t$ has been observed. Dashed undirected edges denote time-invariant variables. Circled numbers indicate a selected computation schedule. Backward messages are marked by black circles. The intractable messages are labeled with red. The dashed box represents a composite AR node as specified by (2.30). Here $\mathbf{I} = \mathrm{I}_K$

2.5 Variational Message Passing for TVAR Models

2.5.1 Message Passing-based Inference in the TVAR model

The TVAR model at time step t can be represented by an FFG as shown in Fig. 2.3. We are interested in providing a message passing solution to the inference tasks as

specified by equations (2.8) - (2.14). At the left-hand side of Fig. 2.3, the incoming messages are the priors $p(\boldsymbol{\theta}_{t-1}|\boldsymbol{y}_{1:t-1})$ and $p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1})$. At the bottom of the graph, there is a new observation y_t. The goal is to pass messages in the graph to compute posteriors $q(\boldsymbol{\theta}_t|\boldsymbol{y}_{1:t})$ (message (16)) and $q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$ (message (11)). In order to support smoothing algorithms, we also want to be able to pass incoming prior messages from the right-hand side to outgoing messages 13 and 18 at the left-hand side. Forward and backward messages are drawn as open and closed circles respectively.

Technically, the generative model (2.2) at time step t for the TVAR model can shortly be written as $p(y_t|\boldsymbol{z}_t)p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})$, where $\boldsymbol{z}_t = \{\boldsymbol{x}_t, \boldsymbol{\theta}_t, \omega, \gamma, \tau\}$ are the latent variables. On this view, we can write the free energy functional for the TVAR model at time step t as

$$F[q(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t|\boldsymbol{y}_{1:t})] = \iint q(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t|\boldsymbol{y}_{1:t}) \log \frac{\overbrace{q(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t|\boldsymbol{y}_{1:t})}^{\text{posterior}}}{\underbrace{p(y_t|\boldsymbol{z}_t)p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})}_{\text{generative model}} \underbrace{p(\boldsymbol{z}_{t-1}|\boldsymbol{y}_{1:t-1})}_{\text{prior from past}}} \mathrm{d}\boldsymbol{z}_{t-1}\mathrm{d}\boldsymbol{z}_t \,. \tag{2.26}$$

and minimize $F[q]$ by message passing. In a smoothing context, we would include a prior from the future $p(\boldsymbol{z}_t|\boldsymbol{y}_{t+1:t+T}) := q(\boldsymbol{z}_t|\boldsymbol{y}_{t+1:t+T})$, yielding a FE functional

$$F[q(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t|\boldsymbol{y}_{1:T})] = \tag{2.27}$$

$$\iint q(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t|\boldsymbol{y}_{1:T}) \log \frac{\overbrace{q(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t|\boldsymbol{y}_{1:T})}^{\text{posterior}}}{\underbrace{p(y_t|\boldsymbol{z}_t)p(\boldsymbol{z}_t|\boldsymbol{z}_{t-1})}_{\text{generative model}} \underbrace{p(\boldsymbol{z}_{t-1}|\boldsymbol{y}_{1:t-1})}_{\substack{\text{prior}\\ \text{from past}}} \underbrace{p(\boldsymbol{z}_t|\boldsymbol{y}_{t+1:t+T})}_{\substack{\text{prior}\\ \text{from future}}}} \mathrm{d}\boldsymbol{z}_{t-1}\mathrm{d}\boldsymbol{z}_t.$$

In a filtering context, $q(\boldsymbol{z}_t|\boldsymbol{y}_{t+1:t+T}) \propto 1$ and the functional (2.27) simplifies to (2.26).

2.5.2 Intractible Messages and the Composite AR node

The modularity of message passing in FFGs allows us to focus on only the intractable message and marginal updates. For instance, while there is no problem with the analytical computation of the backward message 12, the corresponding forward message (4),

$$\vec{\mu}(\boldsymbol{x}_t) = \int \mathcal{N}\Big(\boldsymbol{x}_t|A(\boldsymbol{\theta}_t)\boldsymbol{x}_{t-1}, V(\gamma)\Big) \underbrace{\vec{\mu}(\boldsymbol{x}_{t-1})\vec{\mu}(\boldsymbol{\theta}_t)\vec{\mu}(\gamma)}_{\text{Gaussian messages}} \mathrm{d}\gamma\mathrm{d}\boldsymbol{\theta}_t\boldsymbol{x}_{t-1} \tag{2.28}$$

cannot be solved analytically [78]. Similarly, some other messages 13, 14 and 15 do not have a closed-form solution in the constrained free energy minimization framework. For the purpose of identification, in Fig. 2.3 intractable messages are marked in red color.

In an FFG framework, we can isolate the problematic part of the TVAR model (Figure 2.3) by introducing a "composite" AR node. Composite nodes conceal their internal operations from the rest of the graph. As a result, inference can proceed as long as each composite node follows proper message-passing communication rules at its interfaces to the rest of the graph. The composite AR node

$$f_{\text{AR}}(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \boldsymbol{\theta}_t, \gamma) = \mathcal{N}\left(\boldsymbol{x}_t | A(\boldsymbol{\theta}_t)\boldsymbol{x}_{t-1}, V(\gamma)\right) \tag{2.29}$$

is indicated in Fig. 2.3 by a dashed box. Note that the internal shuffling of the parameters $\boldsymbol{\theta}_t$ and γ, respectively by means of $A(\boldsymbol{\theta}_t)$ and $V(\gamma)$, is hidden from the network outside the composite AR node.

2.5.3 VMP Update Rules for the Composite AR Node

In this section we isolate the composite AR node by the specification

$$f_{\text{AR}}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}, \gamma) = \mathcal{N}\left(\boldsymbol{y} | A(\boldsymbol{\theta})\boldsymbol{x}, V(\gamma)\right) , \tag{2.30}$$

where, relative to (2.29), we used substitutions $\boldsymbol{y} = \boldsymbol{x}_t, \boldsymbol{x} = \boldsymbol{x}_{t-1}, \boldsymbol{\theta} = \boldsymbol{\theta}_t$.

Under the structural factorization constraint[1]

$$q(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}, \gamma) = q(\boldsymbol{y}, \boldsymbol{x})q(\boldsymbol{\theta})q(\gamma) , \tag{2.31}$$

and consistency constraints

$$q(\boldsymbol{y}) = \int q(\boldsymbol{y}, \boldsymbol{x}) \mathrm{d}\boldsymbol{x} , \quad q(\boldsymbol{x}) = \int q(\boldsymbol{y}, \boldsymbol{x}) \mathrm{d}\boldsymbol{y} \tag{2.32}$$

the marginals $q(\boldsymbol{\theta})$, $q(\boldsymbol{x})$, $q(\boldsymbol{y})$ and $q(\gamma)$ can be obtained from the minimisation of the composite-AR free energy functional

$$F_{\text{AR}}[q] = \int q(\boldsymbol{y}, \boldsymbol{x})q(\boldsymbol{\theta})q(\gamma) \log \frac{\overbrace{q(\boldsymbol{y}, \boldsymbol{x})q(\boldsymbol{\theta})q(\gamma)}^{\text{posterior}}}{\underbrace{f_{\text{AR}}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}, \gamma)}_{\text{AR node}}} \mathrm{d}\boldsymbol{y}\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\theta}\mathrm{d}\gamma . \tag{2.33}$$

Recalling (2.25), we can write the minimizer of FE functional (2.33) with respect to $\boldsymbol{\theta}$ as

$$q(\boldsymbol{\theta}) \propto \vec{\nu}(\boldsymbol{\theta})\overleftarrow{\nu}(\boldsymbol{\theta}) \tag{2.34}$$

[1] See Appendix A.2 for more on structural VMP.

where $q(\boldsymbol{\theta})$ is associated with the incoming message to AR node and $\vec{\nu}(\boldsymbol{\theta})$ is a variational outgoing message. Hence, the outgoing message from the AR node toward $\boldsymbol{\theta}$ can be written as

$$\vec{\nu}(\boldsymbol{\theta}) \propto \exp\left(\mathbb{E}_{q(\boldsymbol{y},\boldsymbol{x})q(\boldsymbol{\theta}_t)q(\gamma)} \log\left[\mathcal{N}\Big(\boldsymbol{y}|A(\boldsymbol{\theta})\boldsymbol{x}, V(\gamma)\Big)\right]\right) \tag{2.35}$$

In Appendix A.1 we work out a closed-form solution for this and all other update rules, plus an evaluation of free energy for the composite AR node. The results are reported in Table 2.1. With these rules in hand, the composite AR node can be plugged into any factor graph and take part in a message passing-based free energy minimization process.

2.6 Experiments

In this section, we first verify the proposed methodology by a simulation of the proposed TVAR model on synthetic data, followed by validation experiments on two real-world problems. We implemented all derived message passing rules in the open source Julia package `ForneyLab.jl` [72] and `ReactiveMP.jl` [73]. The code for the experiments and for the AR node can be found in public Github repositories. `https://github.com/biaslab/TVAR_FFG`, accessed on 27 May 2021, `https://github.com/biaslab/LAR`, accessed on 27 May 2021) We used the following computer configuration to run the experiments. *Operation system*: macOS Big Sur, *Processor*: 2, 7 GHz Quad-Core Intel Core $i7$, *RAM*: 16 GB.

2.6.1 Verification on a Synthetic Data Set

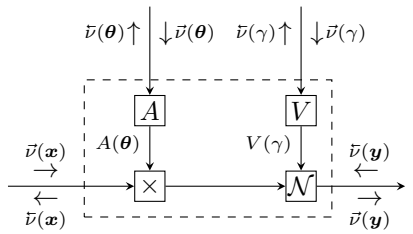
To verify the proposed TVAR inference methods, we synthesized data from two generative models m_1 and m_2, as follows:

$$\boldsymbol{\theta}_t \begin{cases} = \boldsymbol{\theta}_{t-1} & \text{if m} = \text{m}_1 \\ \sim \mathcal{N}(\boldsymbol{\theta}_{t-1}, \omega \text{I}_K) & \text{if m} = \text{m}_2 \end{cases} \tag{2.36a}$$

$$\boldsymbol{x}_t \sim \mathcal{N}\Big(A(\boldsymbol{\theta}_t)\boldsymbol{x}_{t-1}, V(\gamma)\Big) \tag{2.36b}$$

$$y_t \sim \mathcal{N}(\boldsymbol{e}_1^T \boldsymbol{x}_t, \tau) \tag{2.36c}$$

Table 2.1: Variational message update rules for the autoregressive (AR) node (dashed box) of Equation (2.30).

Factor graph	
	
Marginals	**Functional form**
$q(\boldsymbol{\theta})$	$\mathcal{N}\left(\boldsymbol{\theta}\middle\vert\hat{\boldsymbol{m}}_{\boldsymbol{\theta}}, \hat{\boldsymbol{V}}_{\boldsymbol{\theta}}\right)$
$q(\gamma)$	$\Gamma\left(\gamma\middle\vert\hat{\alpha}, \hat{\beta}\right)$
$q(\boldsymbol{y}, \boldsymbol{x})$	$\mathcal{N}\left(\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{x}\end{bmatrix}\middle\vert\middle\vert \begin{bmatrix}\hat{\boldsymbol{m}}_{\boldsymbol{y}}\\ \hat{\boldsymbol{m}}_{\boldsymbol{x}}\end{bmatrix}, \begin{bmatrix}\hat{\boldsymbol{V}}_{\boldsymbol{y}} & \hat{\boldsymbol{V}}_{\boldsymbol{yx}}\\ \hat{\boldsymbol{V}}_{\boldsymbol{xy}} & \hat{\boldsymbol{V}}_{\boldsymbol{x}}\end{bmatrix}\right)$ (App. A.8)
Messages	**Functional form**
$\vec{\nu}(\boldsymbol{y})$	$\mathcal{N}\left(\boldsymbol{y}\middle\vert\boldsymbol{z}_0, \boldsymbol{\Sigma}\right)$
$\overleftarrow{\nu}(\boldsymbol{x})$	$\mathcal{N}\left(\boldsymbol{x}\middle\vert\boldsymbol{\Lambda}_1^{-1}\boldsymbol{z}_1, \boldsymbol{\Lambda}_1^{-1}\right)$
$\overleftarrow{\nu}(\boldsymbol{\theta})$	$\mathcal{N}\left(\boldsymbol{\theta}\middle\vert\boldsymbol{\Lambda}_2^{-1}\boldsymbol{z}_2, \boldsymbol{\Lambda}_2^{-1}\right)$
$\overleftarrow{\nu}(\gamma)$	$\Gamma\left(\gamma\middle\vert 1.5, b/2\right)$
Auxiliaries	**Functional form**
b	$\boldsymbol{e}_1^\intercal\left[\hat{\boldsymbol{V}}_{\boldsymbol{y}} + \hat{\boldsymbol{m}}_{\boldsymbol{y}}(\hat{\boldsymbol{m}}_{\boldsymbol{y}})^\intercal - 2\hat{\boldsymbol{m}}_{\boldsymbol{A}}(\hat{\boldsymbol{V}}_{\boldsymbol{xy}} + \hat{\boldsymbol{m}}_{\boldsymbol{x}}(\hat{\boldsymbol{m}}_{\boldsymbol{y}})^\intercal)\right]\boldsymbol{e}_1$ $+ \boldsymbol{e}_1^\intercal\left[\boldsymbol{m}_{\boldsymbol{A}}(\hat{\boldsymbol{V}}_{\boldsymbol{x}} + \hat{\boldsymbol{m}}_{\boldsymbol{x}}(\hat{\boldsymbol{m}}_{\boldsymbol{x}})^\intercal)\boldsymbol{m}_{\boldsymbol{A}}^\intercal\right]\boldsymbol{e}_1$ $+ \text{tr}(\boldsymbol{V}_{\boldsymbol{\theta}}\left(\hat{\boldsymbol{V}}_{\boldsymbol{x}} + \hat{\boldsymbol{m}}_{\boldsymbol{x}}(\hat{\boldsymbol{m}}_{\boldsymbol{x}})^\intercal\right))$
$\boldsymbol{\Sigma}$	$\boldsymbol{m}_{\boldsymbol{A}}(\boldsymbol{V}_{\boldsymbol{x}}^{-1} + m_\gamma V_{\boldsymbol{\theta}})^{-1}\boldsymbol{m}_{\boldsymbol{A}}^\intercal + \boldsymbol{m}_{\boldsymbol{V}}$
$\boldsymbol{z}_0$	$\boldsymbol{m}_{\boldsymbol{A}}(V_{\boldsymbol{x}}^{-1} + m_\gamma V_{\boldsymbol{\theta}})^{-1}\boldsymbol{V}_{\boldsymbol{x}}^{-1}\boldsymbol{m}_{\boldsymbol{x}}$
$\boldsymbol{\Lambda}_1$	$\boldsymbol{m}_{\boldsymbol{A}}^\intercal\left(\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{V}}\right)^{-1}\boldsymbol{m}_{\boldsymbol{A}} + m_\gamma\boldsymbol{V}_{\boldsymbol{\theta}}$
$\boldsymbol{z}_1$	$\boldsymbol{m}_{\boldsymbol{A}}^\intercal\left(\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{V}}\right)^{-1}\boldsymbol{m}_{\boldsymbol{y}}$
$\boldsymbol{\Lambda}_2$	$\hat{m}_\gamma(\hat{\boldsymbol{V}}_{\boldsymbol{x}} + \hat{\boldsymbol{m}}_{\boldsymbol{x}}(\hat{\boldsymbol{m}}_{\boldsymbol{x}})^\intercal)$
$\boldsymbol{z}_2$	$(\hat{\boldsymbol{V}}_{\boldsymbol{xy}} + \hat{\boldsymbol{m}}_{\boldsymbol{x}}(\hat{\boldsymbol{m}}_{\boldsymbol{y}})^\intercal)\boldsymbol{e}_1 m_\gamma$
Free energy $F[q]$	
$\frac{\hat{m}_\gamma}{2}\left(\hat{\sigma}_y^2 + \hat{m}_y^2 - 2\left[\hat{V}_{y\boldsymbol{x}^\intercal} + \hat{\boldsymbol{m}}_{\boldsymbol{y}}\hat{\boldsymbol{m}}_{\boldsymbol{x}}^\intercal\right]\hat{\boldsymbol{m}}_{\boldsymbol{\theta}} + \text{tr}\left[(\hat{\boldsymbol{V}}_{\boldsymbol{\theta}} + \boldsymbol{m}_{\boldsymbol{\theta}}\boldsymbol{m}_{\boldsymbol{\theta}}^\intercal)\hat{\boldsymbol{V}}_{\boldsymbol{x}}\right]\right)$ $+ \frac{\hat{m}_\gamma}{2}\left(\hat{\boldsymbol{m}}_{\boldsymbol{\theta}}^\intercal(\hat{\boldsymbol{V}}_{\boldsymbol{x}} + \hat{\boldsymbol{m}}_{\boldsymbol{x}}(\hat{\boldsymbol{m}}_{\boldsymbol{x}})^\intercal)\hat{\boldsymbol{m}}_{\boldsymbol{\theta}}\right) - \frac{1}{2}\left[\psi(\hat{\alpha}) - \log\hat{\beta}\right] + \frac{1}{2}\log 2\pi$	
Auxiliary expectations	
$\hat{m}_\gamma = \frac{\hat{\alpha}}{\hat{\beta}} \qquad \boldsymbol{m}_{\boldsymbol{A}} = \mathbb{E}_{q(\boldsymbol{\theta})}[A(\boldsymbol{\theta})]$ $\sigma_y^2 = \boldsymbol{e}_1^\intercal\hat{\boldsymbol{V}}_{\boldsymbol{y}}\boldsymbol{e}_1 \qquad m_y = \boldsymbol{e}_1^\intercal\hat{\boldsymbol{m}}_{\boldsymbol{y}}\boldsymbol{e}_1 \qquad V_{y\boldsymbol{x}} = \hat{\boldsymbol{V}}_{\boldsymbol{yx}}\boldsymbol{e}_1$	

with priors

$$p(K = i) = \prod_{i=1}^{10} 0.1^{[K=i]} \tag{2.37a}$$

$$\boldsymbol{\theta}_0 \sim \begin{cases} \mathcal{N}(\mathbf{0}, \boldsymbol{I}) & \text{if m} = \text{m}_1 \\ \mathcal{N}(\mathbf{0}, 1e12\boldsymbol{I}) & \text{if m} = \text{m}_2 \end{cases} \tag{2.37b}$$

$$\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, 1e12\boldsymbol{I}) \tag{2.37c}$$

$$\gamma \sim \Gamma(1.0, 1e-5) \tag{2.37d}$$

$$\tau = 1.0 \tag{2.37e}$$

$$\omega = 0.01 \tag{2.37f}$$

where K is the number of AR coefficients. Although these models differ only with respect to the properties of the AR coefficients $\boldsymbol{\theta}$, this variation has an important influence on the data generative process. The first model m_1 specifies a stationary AR process, since $\delta(\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-1})$ in (2.36a) indicates that θ is not time-varying in m_1. The second model m_2 represents a proper TVAR process as the prior evolution of the AR coefficients follows a random walk. One-time segment FFGs corresponding to the Equations (2.36) are depicted in Figure 2.4.

For each model, we generated a data set of 100 different time series, each of length 100 (so we have $2 \times 100 \times 100$ data points). For each time series, as indicated by (2.37a), the AR order M of the generative process was randomly drawn from the set $\{1, 2, \ldots, 10\}$. We used rather non-informative/broad priors for states and parameters for both models, see (2.37). This was done to ensure that the effect of the prior distributions is negligible relative to the information in the data set.

These time series were used in the following experiments. We selected two recognition models m_1 and m_2 with the same specifications as were used for generating the data set. The recognition models were trained on time series that were generated by models with the same AR order.

We proceeded by computing the quantities $q(\boldsymbol{x}_{1:T}|\boldsymbol{y}_{1:T})$, $q(\boldsymbol{\theta}_{1:T}|\boldsymbol{y}_{1:T})$, $q(\gamma|\boldsymbol{y}_{1:T})$ and $F[q(\boldsymbol{z}_{t-1}, \boldsymbol{z}_t|\boldsymbol{y}_{1:T})]$ (where $\boldsymbol{z}$ comprises all latent states and parameters) for both models, following the proposed rules from Table 2.1.

As a verification check, we first want to ensure that inference recovers the hidden states x_t for each $t \in (1, 2, \ldots 100)$. Secondly, we want to verify the convergence of FE. As we have not used any approximations along the derivations of variational messages, we expect a smoothly decreasing curve for FE until convergence. The results of the verification stage are highlighted for a typical case in Figure 2.5. The figure confirms that states x_t are accurately tracked and that a sliding average of the AR coefficients $\boldsymbol{\theta}_t$ is also nicely tracked. Figure 2.5 also indicates that the FE uniformly decreases towards lower values as we spend more computational power.

We note that the FE score by itself does not explain whether the model is good

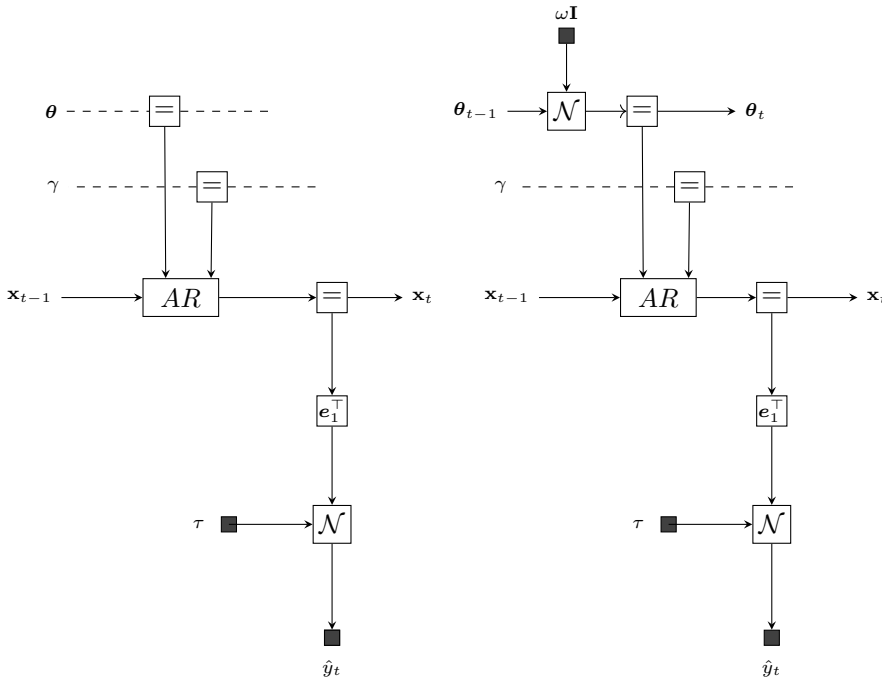


Figure 2.4: Forney-style Factor Graphs corresponding to Equation (2.36). (**Left**) model m_1. (**Right**) model m_2.

or not, but it serves as a good measure for model comparison. In the following subsection, we demonstrate how FE scores can be used for model selection.

2.6.2 Temperature Modeling

AR models are well-known for predicting different weather conditions such as wind, temperature, precipitation, etc. Here, we will revisit the problem of modeling daily temperature. We used a data set of daily minimum temperatures (in C°) in Melbourne, Australia, 1981–1990 (3287 days) (`https://www.kaggle.com/paulbrabban/daily-minimum-temperatures-in-melbourne`, accessed on 27 May 2021). We then corrupted the data set by adding random noise sampled from $\mathcal{N}(0, 10.0)$ to the actual temperatures. A fragment of the time series is depicted in Figure 2.6.

To estimate the actual temperature based on past noisy observations by computing $q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$, we use a TVAR model with measurement noise to simulate uncertainty about corrupted observations. The model is specified by the following

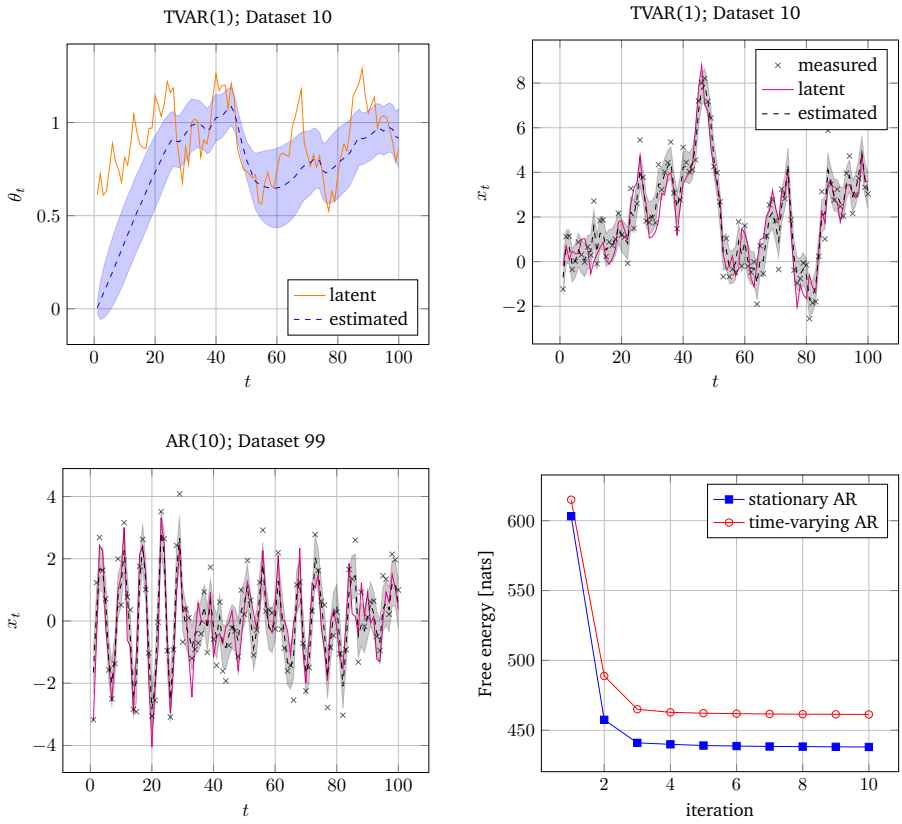


Figure 2.5: Verification results. The solid line corresponds to the value of the latent (hidden) states in the generative processes. The dashed line corresponds to the expected mean value of the posterior estimates of hidden states $q(\cdot|\boldsymbol{y}_{1:100})$ in the recognition models. The shadowed regions correspond to one standard deviation of the posteriors in the recognition models below and above the estimated mean. The top two plots show inference results for the coefficients $\boldsymbol{\theta}_t$ (top-left) and states x_t (top-right) of TVAR(1) (model m_2, AR order $K = 1$) for time series ♯10. (bottom-left) State trajectory $q(x_t|\boldsymbol{y}_{1:100})$ model m_1, AR order $K = 1$ on time series ♯99. (Bottom-right) Evolution of FE for m_1 (AR) and m_2 (TVAR), averaged over their corresponding time series. The iteration number at the abscissa steps through a single marginal update for all edges in the graph.

equation set

$$\boldsymbol{\theta}_t \sim \mathcal{N}(\boldsymbol{\theta}_{t-1}, \mathrm{I}_M) \tag{2.38a}$$

$$\boldsymbol{x}_t \sim \mathcal{N}\Big(A(\boldsymbol{\theta}_t)\boldsymbol{x}_{t-1} + \boldsymbol{e}_1\eta, V(\gamma)\Big) \tag{2.38b}$$

$$y_t \sim \mathcal{N}(\boldsymbol{e}_1^\intercal \boldsymbol{x}_t, \tau^{-1}) \tag{2.38c}$$

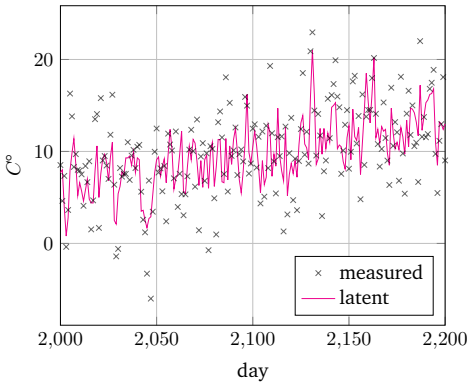


Figure 2.6: Temperature time-series from days 2000 to 2200. Crosses denote the thermometer readings plus added noise. The solid line corresponds to the latent (hidden) daily temperature.

with priors

$$\boldsymbol{\theta}_0 \sim \mathcal{N}(\mathbf{0},\boldsymbol{I}) \qquad \boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0},\boldsymbol{I}) \qquad \eta \sim \mathcal{N}(0.0, 10.0) \tag{2.39a}$$

$$\gamma \sim \Gamma(1.0, 1.0) \qquad \tau \sim \Gamma(0.1, 1.0) \tag{2.39b}$$

where $\boldsymbol{I} = \mathrm{I}_K$. Note that we use τ as a precision parameter in this experiment. Since the temperature data is not centered around 0 °C, we added a bias term η to the state $\boldsymbol{x}_t$. The corresponding FFG is depicted in Figure 2.7.

Note that we put a Gamma prior on the measurement noise precision τ, meaning that we are uncertain about the size of the error of the thermometer reading. The inference task for the model is computing $q(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$, in other words, we track the states based only on past data. Of course, after training, we could use the model for temperature prediction by tracking $q(\boldsymbol{x}_{t+k}|\boldsymbol{y}_{1:t})$ for $k \geq 1$. We compare the performance of four TVAR models with AR orders $K = \{1, 2, 3, 4\}$. To choose the best model, we computed the average FE score for each TVAR(K) model.

Figure 2.8 shows that on average TVAR(3) outperforms its competitors. The complexity vs accuracy decomposition (2.23) of FE explains why a lower order model may outperform higher order models. TVAR(4) maybe as accurate or more accurate than TVAR(3) but the increase in accuracy is more than offset by the increase in complexity. For the lower order models, it is the other way around: they are less complex and involve fewer computations than TVAR(3), but the loss in model complexity leads to too much loss in data modeling accuracy. Overall, TVAR(3) is the best model for this data set. Practically, we always favor the model that features the lowest FE score. In the next subsection we will use this technique (scoring FE) for online model selection.

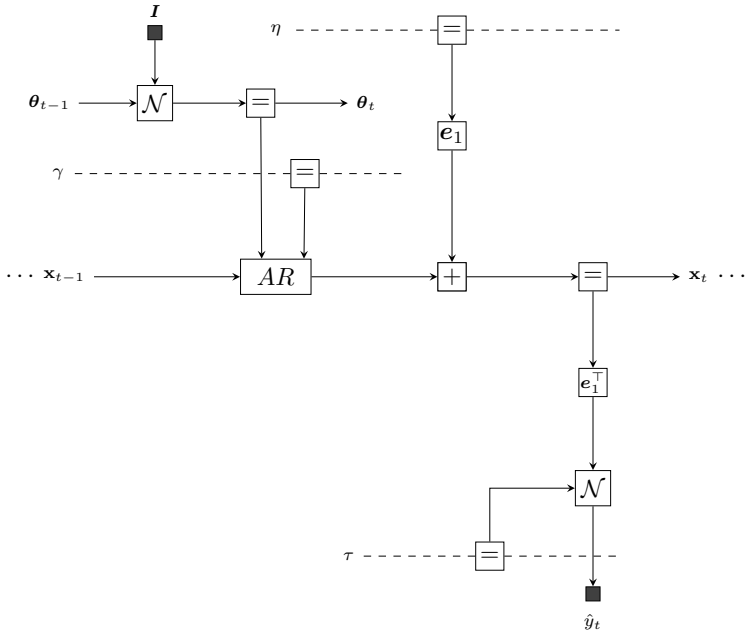


Figure 2.7: One time segment of a Forney-style factor graph (FFG) for the TVAR model in the temperature modeling task (2.38).

2.6.3 Single-Channel Speech Enhancement

Single-channel speech enhancement (SCSE) is a well-known challenging task that aims to enhance noisy speech signals that were recorded by a single microphone. In single microphone recordings, we cannot use any spatial information that is commonly used in beamforming applications. Much work has been done to solve the SCSE task, ranging from Wiener filter-inspired signal processing techniques [79, 80] to deep learning neural networks [81]. In this paper, we use data from the speech corpus (NOIZEUS) (`https://ecs.utdallas.edu/loizou/speech/noizeus/`, accessed on 27 May 2021) [82] and corrupted clean speech signals with white Gaussian noise, leading to a signal-to-noise ratio (SNR)

$$\text{SNR}(\boldsymbol{s}_{1:T}, \boldsymbol{y}_{1:T}) = 10 \log_{10} \left[\frac{\sum_t^T s_t^2}{\sum_t^T (s_t - y_t)^2} \right] \approx 13.36\,\text{dB} \tag{2.40}$$

where $\boldsymbol{s}_{1:T} = (s_1, \ldots, s_T)$ and $\boldsymbol{y}_{1:T} = (y_1, \ldots, y_T)$ are clean and corrupted speech signals. s_t is a speech signal at time t and T is the length of the signal.

Historically, AR models have shown to perform well for modeling speech sig-

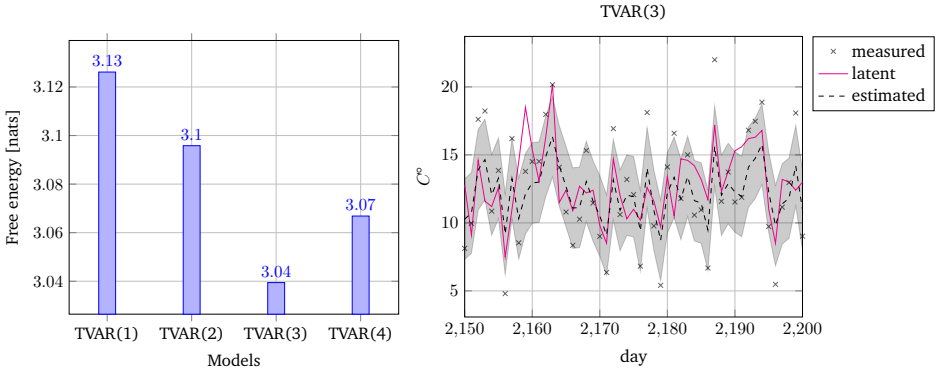


Figure 2.8: (**Left**) Comparison of four TVAR(M) models for the temperature filtering problem. Bars correspond to the averaged (over 3287 days) FE score for each model. (**Right**) Inference example of the best performing model (TVAR(3)). Crosses denote the thermometer reading plus added noise. The solid line corresponds to the latent (hidden) daily temperature. The dashed line corresponds to the mean of the posterior estimates of hidden temperature and the shadowed region corresponds to one standard deviation below and above the estimated temperature.

nals in the time (waveform) domain [83, 84]. Despite the fact that speech is a highly nonstationary signal, we may assume it to be stationary within short time intervals (frames) of about 10 [ms] each [85]. Since voiced, unvoiced and silence frames have very different characteristics, we used 5 different models (a random walk model (RW), AR(1), AR(2), TVAR(1) and TVAR(2)) for each frame of 10 [ms] with 2.5 [ms] overlap. Given a sampling frequency of 8 [kHZ], each frame results into 80 samples with 20 samples overlap. The AR and TVAR models were specified by Equations (2.36). For each frame, we evaluated the model performance by minimized FE and selected the model with minimal FE score. We used identical prior parameters for all models where possible. To recover the speech signal we computed the mean values of $q(\boldsymbol{x}_t|y_{1:T})$ of the selected model for each frame. The SNR gain of this SCSE system was

$$\text{SNR}(\boldsymbol{s}_{1:T}, \boldsymbol{x}_{1:T}) - \text{SNR}(\boldsymbol{s}_{1:T}, \boldsymbol{y}_{1:T}) \approx 3.7\,\text{dB}\,. \tag{2.41}$$

Figure 2.9 show the spectrograms of the clean, noisy, and filtered signal respectively.

Next, we analyze the inference results in a bit more detail. Table 2.2 shows the percentage of winning models for each frame based on the free energy score. As we can see, for more than 30% of all frames, the random walk model performs best. This happens mostly because the AR model gets penalized by its complexity

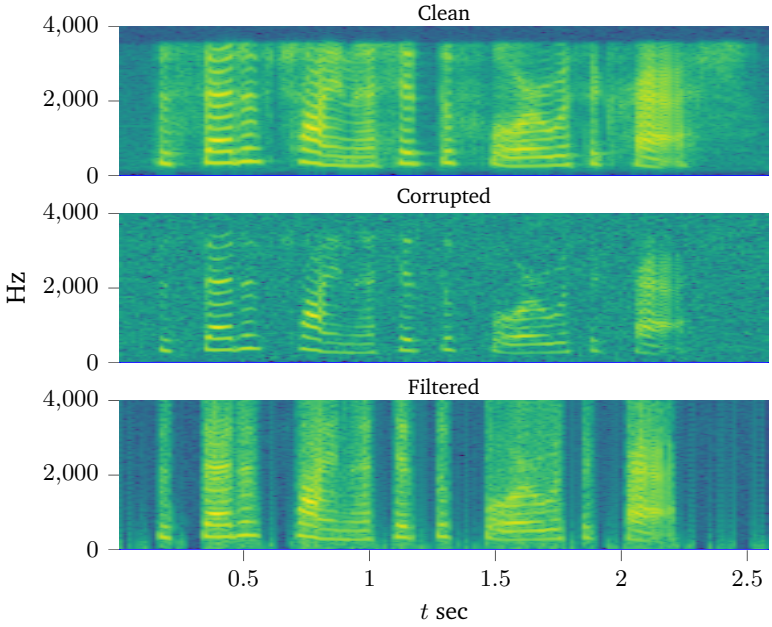


Figure 2.9: Spectrogram of recovered speech signal in the experiment of Section 2.6.3.

Table 2.2: Percentage of preferred models (based on FE scores) for all frames on the speech enhancement task.

	RW	**AR(1)**	**AR(2)**	**TVAR(1)**	**TVAR(2)**
Ratio	32.2%	54.3%	10.7%	1.2%	0.5%

term for a silent frame. We recognize that the best models in about 90% of the frames are AR(1) and RW. On the other hand, for the frames where the speech signal transitions from silent or unvoiced to voiced, these fixed models start to fail, and the time-varying AR models perform better. This effect can be seen in Figure 2.10.

Figure 2.11 shows the performance of the AR(2) and RW models on a frame with a voiced speech signal. For this case, the AR(2) model performs better.

Finally, Figure 2.12 shows how the TVAR(2) model compares to the RW model on one of the unvoiced/silence frames. While TVAR(2) estimates appear to be more accurate, it pays a higher "price" for the model complexity term in the FE score, and the RW model wins the frame.

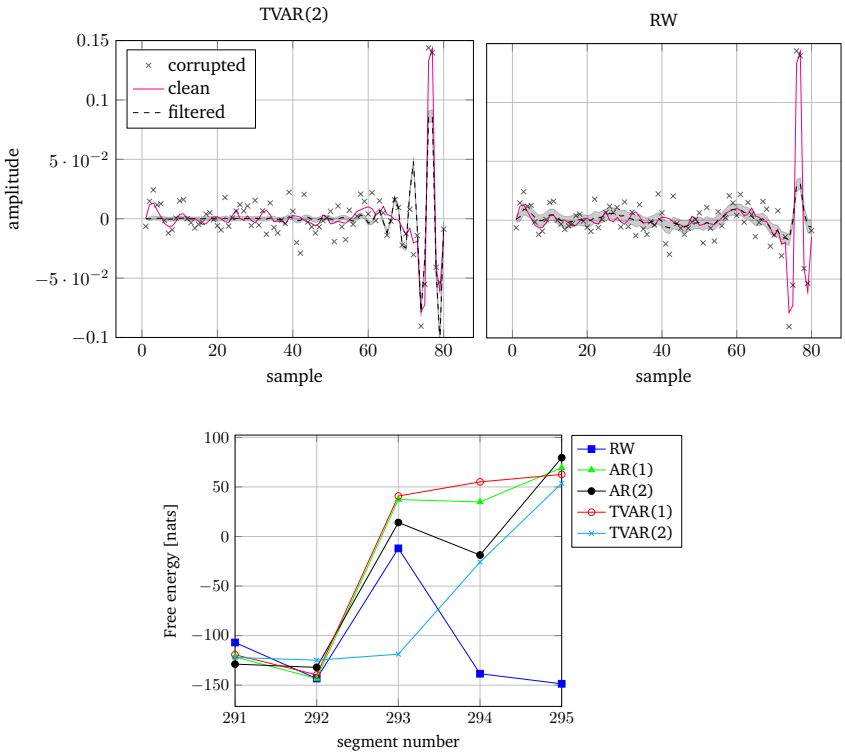


Figure 2.10: (Top) (**Top-left**) Inference by TVAR(2) for the segment 293. (**Top-right**) Inference by RW for the segment 293. Note how the TVAR model is able to follow the transitions at the end of the frame, while the RW cannot adapt within one frame. (**Bottom**) FE scores from segment 291 to 295. TVAR(2) wins frame 293 as it has the lowest FE score.

2.7 Discussion

We have introduced a TVAR model that includes efficient joint variational Bayesian tracking of states, parameters, and free energy. The system can be used as a plug-in module in factor graph-based representations of other models. At several points in this paper, we have made some design decisions that we shortly review here.

While FE computation for the AR node provides a convenient performance criterion for model selection, we noticed in the speech enhancement simulation that separate FE tracking for each candidate model leads to a large computational overhead. There are ways to improve the model selection process that we used in the

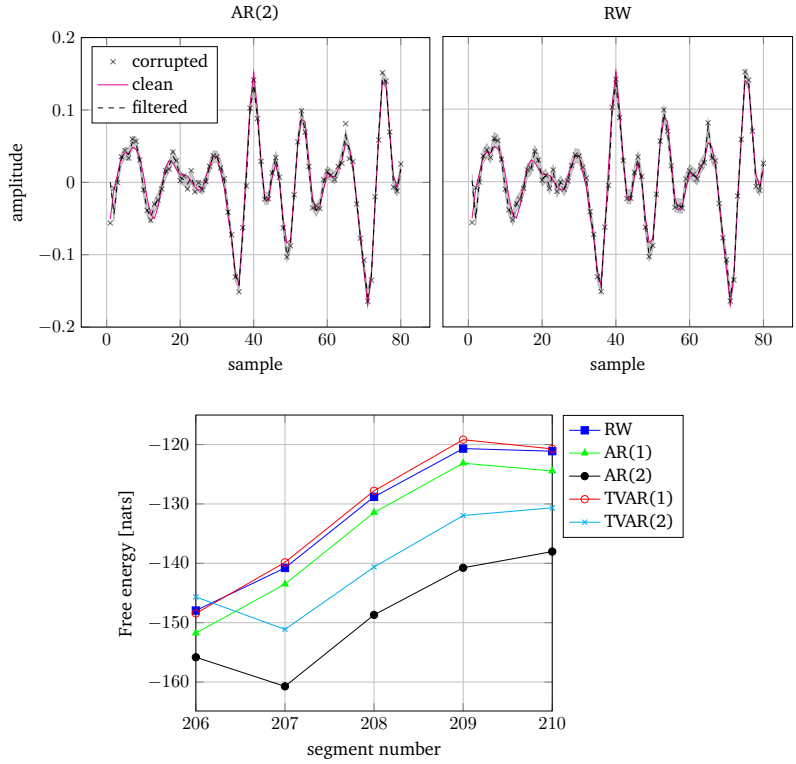


Figure 2.11: Comparison of AR(2) and RW models for a voiced signal frame. (**Top-left**) Inference by AR(2) for the segment 208. (**Top-right**) Inference by RW for the segment 208. (**Bottom**) FE scores from segment 206 to 210. The AR(2) model wins frame 208.

speech enhancement simulation. One way is to consider a mixture model of candidate models and track the posterior over the mixture coefficients [86]. Alternatively, a very cheap method for online Bayesian model selection may be the recently developed Bayesian Model Reduction (BMR) method [87]. The BMR method is based on a generalization of the Savage-Dickey Density Ratio and supports the tracking of free energy of multiple nested models with almost no computational overhead. Both methods seem to integrate well with a factor graph representation and we plan to study this issue in future work.

In this paper, the posterior factorization (2.31) supports the modeling of temporal dependencies between input and output of the AR node in the posterior. Technically, (2.31) corresponds to a structural VMP assumption, in contrast to the more

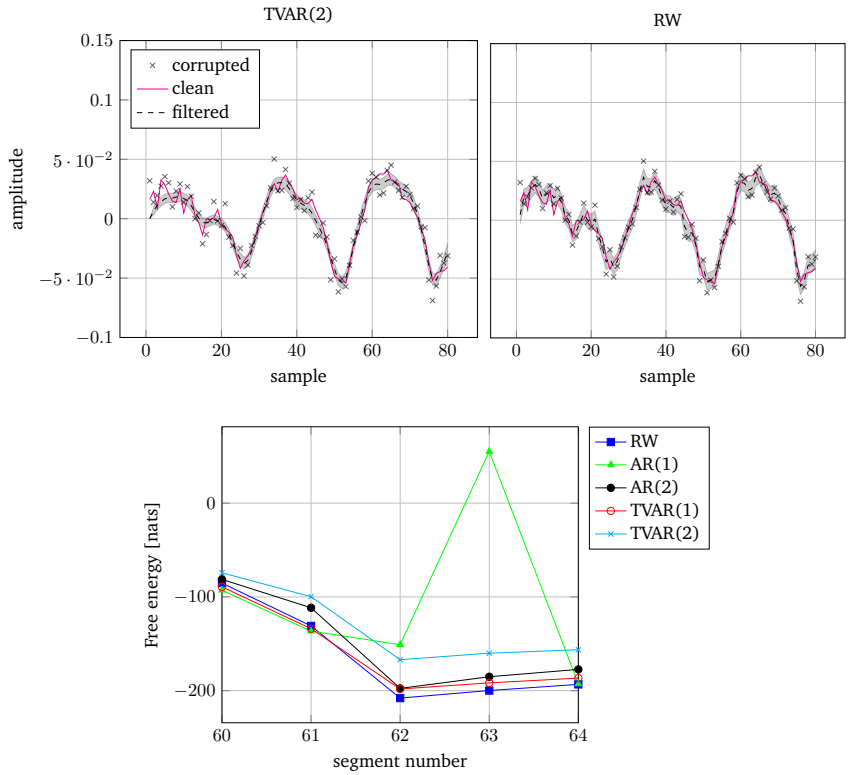


Figure 2.12: Comparison of TVAR(2) and RW models for an unvoiced/silence frame. (**Top-left**) Inference by TVAR(2) for the frame 62. (**Top-right**) Inference by RW for the frame 62. **(Bottom)** FE scores from segment 60 to 64. The RW model scores best on frame 62 due to its low complexity.

constrained mean-field VMP algorithm that would be based on $q(\boldsymbol{z}) = \prod_i q_i(z_i)$, where $\boldsymbol{z}$ is the set of all latent variables [88]. We could have also worked out alternative update rules for the assumption of a joint factorization of precision γ and AR coefficients $\boldsymbol{\theta}$. In that case, the prior (incoming message $\vec{\nu}(\boldsymbol{\theta}, \gamma)$ to AR node) would be in the form of a Normal-Gamma distribution. While any of these assumptions are technically valid, each choice accepts a different trade-off in the accuracy vs. complexity space. We review structural VMP in Appendix A.2.

In the temperature modeling task, we added some additional random variables (bias, measurement noise precision). To avoid identifiability issues, in (2.38a) we fixed the covariance matrix of the time-varying AR coefficient to the identity matrix.

In principle, this constraint can be relaxed. For example, an Inverse-Wishart prior distribution can be added to the covariance matrix.

In our speech enhancement experiments in Section 2.6.3, we assume that the measurement noise variance is known. In a real-world scenario, this information is usually not accessible. However, online tracking of measurement noise or other (hyper-)parameters is usually not a difficult extension when the process is simulated in a factor graph toolbox such as ForneyLab [72]. If so desired, we could add a prior on the measurement noise variance and track the posterior. The online free energy criterion (2.23) can be used to determine if the additional computational load (complexity) of Bayesian tracking of the variance parameter has been compensated by the increase in modeling accuracy.

The realization of the TVAR model in ForneyLab comes with some limitations. For large smoothing problems (say, $>$1000 data points), the computational load of message passing in ForneyLab becomes too heavy for a standard laptop (as was used in the paper). Consequently, in the current implementation, it is difficult to employ the AR node for processing large time series on a standard laptop. To circumvent this issue, when using ForneyLab, one can combine filtering and smoothing solutions into a batch learning procedure. In future work, we plan to remedy this issue by some ForneyLab refactoring work. Additionally, the implemented AR node does not provide a closed-form update rule for the marginal distribution when the probability distribution types of the incoming messages (priors) are different from the ones used in our work. Fortunately, ForneyLab supports resorting to (slower) sampling-based update rules when closed-form update rules are not available.

2.8 Conclusion

We presented a variational message-passing approach to tracking states and parameters in latent TVAR models. The required update rules have been summarized and implemented in the factor graph packages `ForneyLab.jl` and `ReactiveMP.jl`, thus making transparent usage of TVAR factors available in freely definable stochastic dynamical systems. Aside from VMP update rules, we derived a closed-form expression for the variational free energy (FE) of an AR factor. Free Energy can be used as a proxy for Bayesian model evidence and allows for model performance comparisons between the TVAR models and alternative structures. Owing to the locality and modularity of the FFG framework, we demonstrated how AR nodes could be applied as plug-in modules in various dynamic models. We verified the correctness of the rules on a synthetic data set and used the proposed TVAR model for a few relatively simple but different real-world problems. In future work, we plan to extend the current factor graph-based framework to efficient and transparent tracking of AR model order and online model comparison and selection with alternative models.

Chapter 3

Message Passing-based Inference in Gamma-Mixture Models

This chapter is based on the original work referenced below. Contributions are split evenly among the first four authors; namely, the original idea, supporting software, simulations, and text has been established in close collaboration. Notations have been adjusted to reflect conventions throughout the dissertation.

> **Albert Podusenko**, **Bart van Erp**, **Dmitry Bagaev**, **Ismail Senoz**, Bert de Vries, *Message Passing-Based Inference in the Gamma Mixture Model*. In 2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP) - Proceedings

Abstract

The Gamma mixture model is a flexible probability distribution representing beliefs about scale variables such as precisions. However, inference in the Gamma mixture model for all latent variables is non-trivial as it leads to intractable equations. This paper presents two variants of variational message passing-based inference in a Gamma mixture model. We use moment matching and alternatively expectation-maximization to approximate the posterior distributions. The proposed method supports automated inference in factor graphs for large probabilistic models that contain multiple Gamma mixture models as plug-in factors. The Gamma mixture model has been implemented in a factor graph package, and we present experimental results for both synthetic and real-world data sets.

3.1 Introduction

Mixture models are commonly used in the literature to model probability density functions that are outside the exponential family. Gaussian mixture models are often used, especially in the field of natural language processing [89]. However, this paper will focus on the less common Gamma mixture models (ΓMMs). The ΓMMs allow us to efficiently model skewed distributions with positive support [90]. For example, this model can be used as the conjugate prior for the precision parameter of a Gaussian distribution. In that case, the conjugate relationship supports the modeling of processes with switching noise levels.

The ΓMM has been used in a variety of applications, such as in the detection of COVID-19 in medical images [91]. The literature describes a few approaches for performing inference in the ΓMM, or the generalized ΓMM, most notably a sampling approach [92] and a variational expectation-maximization method [90]. Unfortunately, these approaches are not modular by nature, which often leads to tedious and error-prone manual derivations when extending or applying the models in a different context. In this paper, we propose a modular message passing-based probabilistic inference method for ΓMMs.

We represent the ΓMM as a composite factor (node) in a Forney-style Factor Graph (FFG) [93, 94]. A benefit of the FFG representation is that all (message passing) computations are local and, as a result, the ΓMM factor can be used as a plug-in module in larger probabilistic models. More details on the FFGs will be provided in Section 3.2, where we also specify the Gamma mixture (ΓM) model.

This follows the problem that we solve in this paper: how to perform message passing-based inference in the ΓMM. A solution proposal is presented in Section 3.3. Specifically, in Section 3.3.3 we provide a local expectation-maximization extension to variational message passing, and in Section 3.3.4 we propose a moment matching-based non-conjugate variational message passing method. These solutions are verified and validated in Section 3.4. We discuss our findings and conclude the paper in Section 3.5.

3.2 Model Specification and Problem definition

Let $\boldsymbol{x} \triangleq [x_1, \ldots, x_T]$, where $x_t \in \mathbb{R}_{>0}$ for every $t = 1, \ldots, T$, denote a vector of strictly positive independent and identically distributed (IID) observations. The likelihood for a ΓMM with L mixture components is given by

$$x_t \sim \prod_{t=1}^{L} \Gamma(a_l, b_l)^{\mathbf{c}_t} , \tag{3.1}$$

where $\Gamma(a_l, b_l)$ specifies the Gamma distribution for x_t with shape and rate parameters a_l and b_l, respectively. $\boldsymbol{a} \triangleq [a_1, \ldots, a_L]$ and $\boldsymbol{b} \triangleq [b_1, \ldots, b_L]$ are vectors of the

parameters of the Gamma distributions such that $a_l, b_l \in \mathbb{R}_{>0}$ for every $l = 1, \ldots, L$. For each observation x_t we have a corresponding latent selector variable $\boldsymbol{c}_t$ comprising a 1-of-L binary vector with elements $c_{tl} \in \{0, 1\}$, which are constrained by $\sum_l c_{tl} = 1$. We denote the vector of selector variables by $\boldsymbol{c} \triangleq [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_T]$.

To complete the specification of the ΓMM we need to specify priors on $\boldsymbol{a}, \boldsymbol{b}$ and $\boldsymbol{c}$. We choose the priors as

$$a_l \sim \Gamma\left(\alpha_l^{(a)}, \beta_l^{(a)}\right) \quad \alpha_l^{(a)}, \beta_l^{(a)} \in \mathbb{R}_{>0} \tag{3.2}$$

$$b_l \sim \Gamma\left(\alpha_l^{(b)}, \beta_l^{(b)}\right) \quad \alpha_l^{(b)}, \beta_l^{(b)} \in \mathbb{R}_{>0} \tag{3.3}$$

$$\boldsymbol{c} \sim \mathrm{Cat}(\boldsymbol{\pi}) = \prod_{l=1}^{L} \pi_l^{c_l} \quad \text{where } \sum_{l=1}^{L} \pi_{l0} = 1, \tag{3.4}$$

and we choose a Dirichlet prior for the event probabilities $\boldsymbol{\pi} \triangleq [\pi_1, \ldots, \pi_L]$ as

$$\boldsymbol{\pi} \sim \mathrm{Dir}(\boldsymbol{\zeta}), \tag{3.5}$$

where $\boldsymbol{\zeta} = [\zeta_1, \ldots, \zeta_L]$ are the concentration parameters with $\zeta_l \in \mathbb{R}_{>0}$ for every $l = 1, \ldots, L$. The full ΓMM is then given by the joint distribution

$$p(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{\pi}) = p(\boldsymbol{x}|\boldsymbol{c}, \boldsymbol{a}, \boldsymbol{b}) p(\boldsymbol{a}) p(\boldsymbol{b}) p(\boldsymbol{c}|\boldsymbol{\pi}) p(\boldsymbol{\pi}). \tag{3.6}$$

An FFG is an undirected graph in which nodes represent factors of a global function and edges represent random variables [93]. In an FFG, each edge can be connected to a maximum of 2 factors, whereas a node can be connected to an arbitrary number of edges. Hence, FFGs usually contain multiple *equality nodes* with factors $\delta(x - x')\delta(x - x'')$ that constrain the beliefs over two "copy variables" x' and x'' to be equal to the belief over x [69]. As a matter of notational convention, in an FFG, factors are represented by square (unfilled) nodes and observations or fixed variables in these graphs are represented by small black squares, whose factors can be regarded as Dirac delta functions centered on the observed value. For a detailed explanation of the FFG formalism, we refer to [93, 94, 48]. FFGs corresponding to the ΓMM of (3.6) are presented in Table 3.1 and Fig. 3.1.

Given the ΓMM and a collection of observations $\boldsymbol{x}$ we are interested in obtaining the posterior distributions $p(\boldsymbol{a}|\boldsymbol{x}), p(\boldsymbol{b}|\boldsymbol{x}), p(\boldsymbol{c}|\boldsymbol{x})$ and $p(\boldsymbol{\pi}|\boldsymbol{x})$. Computation of the posteriors requires the integration and summation of the model (3.6) with respect to all remaining model variables:

$$p(\boldsymbol{a}|\boldsymbol{x}) = \frac{\sum_{\boldsymbol{c}} \int p(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{\pi}) \, \mathrm{d}\boldsymbol{b} \, \mathrm{d}\boldsymbol{\pi}}{\sum_{\boldsymbol{c}} \int p(\boldsymbol{x}, \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{\pi}) \, \mathrm{d}\boldsymbol{a} \, \mathrm{d}\boldsymbol{b} \, \mathrm{d}\boldsymbol{\pi}}. \tag{3.7}$$

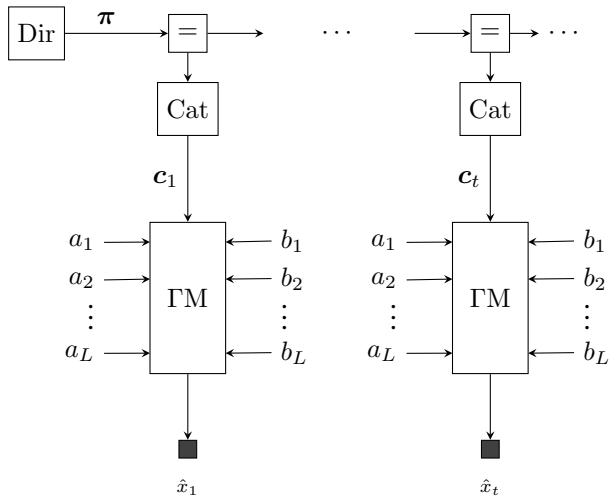


Figure 3.1: An FFG representation of the ΓMM in (3.6). Dir and Cat denote Dirichlet and Categorical distributions respectively. The '=' nodes represent equality factors. Small black nodes denote observations. For brevity, we did not add the nodes corresponding to the distributions of shape a_l and rate b_l parameters. The inside of the ΓM node is further worked out in Table 3.1.

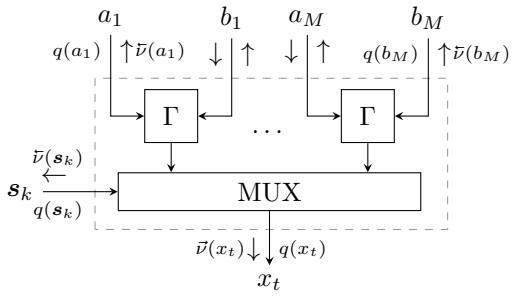
Even though (3.7) is the exact solution to one of the inference tasks, it is intractable because the integrals involving $\boldsymbol{a}$ and $\boldsymbol{b}$ do not yield known analytical solutions. In this paper, the problem we address is how to compute approximate posteriors for the ΓMM.

3.3 Approximate message passing-based inference

In this section, we first introduce message passing in an FFG as a probabilistic inference methodology. Next, we will derive messages for the Gamma mixture (ΓM) node using variational message passing (VMP) [68, 77], which allows us to perform probabilistic inference in the ΓMM. However, one of the VMP messages leads to an approximate posterior distribution, whose closed-form solution is the result of a non-conjugate multiplication that cannot be normalized analytically. We propose two approaches to resolve this problem. First, we propose using expectation maximization to bypass the need to calculate the normalization constant. Second, we apply moment matching to approximate the moments of the approximate posterior distribution through importance sampling [95, Ch.7].

Table 3.1: Table containing (top) the Forney-style factor graph representation of the Gamma mixture node. The node indicated by MUX represents a multiplexer node, which selects the mixture component. (middle) An overview of the chosen approximate posterior distributions. Here the $\hat{\cdot}$ accent refers to the parameters of these distributions. The choice of functional form for $q(a_l)$ depends on the approximation method (Section 3.3). (bottom) The derived messages for the Gamma mixture node. The definitions of ζ_{km} and ρ_{km} are presented in the supplementary material at `http://github.com/mlsp2021-gmm`.

Factor graph



Marginals	**Functional form**
$q(a_l)$	$\delta(a_l - \hat{a}_l)$ or $\Gamma\left(a_l \mid \hat{\alpha}_l^{(a)}, \hat{\beta}_l^{(a)}\right)$ $\hat{\alpha}_l^{(a)}, \hat{\beta}_l^{(a)} \in \mathbb{R}_{>0}$
$q(b_l)$	$\Gamma\left(b_l \mid \hat{\alpha}_l^{(b)},\ \hat{\beta}_l^{(b)}\right)$ $\hat{\alpha}_l^{(b)}, \hat{\beta}_l^{(b)} \in \mathbb{R}_{>0}$
$q(\boldsymbol{s}_t)$	$\prod_{l=1}^{L} \hat{\pi}_l^{s_{km}}$ such that $\sum_{m=1}^{M} \hat{\pi}_l = 1$
$q(x_t)$	$\Gamma\left(x_t \mid \hat{\alpha}_t^{(x)},\ \hat{\beta}_t^{(x)}\right)$ $\hat{\alpha}_t^{(x)}, \hat{\beta}_t^{(x)} \in \mathbb{R}_{>0}$
Messages	**Functional form**
$\overleftarrow{\nu}(a_l)$	$\exp\left(\hat{\pi}_t \left(a_l \zeta_{km} - \log \Gamma(a_l)\right)\right)$
$\overleftarrow{\nu}(b_l)$	$\Gamma\left(b_l \middle\vert 1 + \hat{\pi}_l \mathrm{E}\left[a_l\right],\ \hat{\pi}_l \frac{\hat{\alpha}_t^{(x)}}{\hat{\beta}_t^{(x)}}\right)$
$\overleftarrow{\nu}(\boldsymbol{s}_t)$	$\prod_{l=1}^{L} \rho_{km}^{s_{km}}$ such that $\sum_{m=1}^{M} \rho_{km} = 1$
$\vec{\nu}(x_t)$	$\Gamma\left(x_t \middle\vert \sum_{m=1}^{M} \hat{\pi}_l \mathrm{E}\left[a_l\right],\ \sum_{m=1}^{M} \hat{\pi}_l \frac{\hat{\alpha}_l^{(b)}}{\hat{\beta}_l^{(b)}}\right)$

3.3.1 Variational message passing

Because of the conditional independencies in the generative model we can perform execution of (3.7) through a distributed set of smaller local computations called messages. Unfortunately, the intractability in these computations limits us in performing exact message passing-based inference, also known as the sum-product algorithm [96] or belief propagation [97]. To resolve this, we will resort to VMP [68, 77]. Consider the generative model $p(\boldsymbol{x}, \boldsymbol{z})$ for the ΓMM, where $\boldsymbol{z} \triangleq [\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{\pi}]$, with intractable posterior distribution $p(\boldsymbol{z}|\boldsymbol{x})$, in which $\boldsymbol{x}$ and $\boldsymbol{z}$ are the observed and latent variables, respectively. Variational inference approximates the exact posterior distribution $p(\boldsymbol{z}|\boldsymbol{x})$ by a tractable approximate posterior distribution $q(\boldsymbol{z})$ through minimization of the variational free energy (VFE) functional

$$F[q] = D_{\mathrm{KL}}[q(\boldsymbol{z})||p(\boldsymbol{z}|\boldsymbol{x})] - \log p(\boldsymbol{x})\,. \tag{3.8}$$

where D_{KL} is the Kullback-Leibler divergence. In VMP the variational free energy is optimized by iteratively updating the approximate posterior distributions. In order to facilitate optimization of VFE, $q(\boldsymbol{z})$ is often constrained by a mean-field factorization

$$q(\boldsymbol{z}) = \prod_i q(z_i)\,. \tag{3.9}$$

For a generic node $f(z_1, z_2, \ldots, z_M)$ the outgoing variational message $\vec{\nu}(z_j)$, under the mean-field assumption, can be evaluated as [77]

$$\vec{\nu}(z_j) \propto \exp \int \prod_{i \neq j} q(z_i) \log f(z_1, z_2, \ldots, z_M) \mathrm{d}\boldsymbol{z}_{\setminus j}. \tag{3.10}$$

The approximate posterior can then be updated by the normalized multiplication of the messages on the corresponding edge as

$$q(z_j) = \frac{\vec{\nu}(z_j)\overleftarrow{\nu}(z_j)}{\displaystyle\int \vec{\nu}(z_j)\overleftarrow{\nu}(z_j)\mathrm{d}z_j}\,. \tag{3.11}$$

In the VMP algorithm, (3.10) and (3.11) are iteratively repeated for all variables until convergence [77].

3.3.2 Variational message passing in the Gamma mixture node

The ΓM node of (3.1) has been visualized in Table 3.1. We will assume a mean-field factorization over the joint approximate posterior distribution as

$$q(x_t, \boldsymbol{c}_t, \boldsymbol{a}, \boldsymbol{b}) = q(x_t)q(\boldsymbol{c}_t) \prod_{m=1}^{M} q(a_m)q(b_m), \tag{3.12}$$

where the distributions of the individual factors are presented in Table 3.1. To support modular usage of the ΓM node, the variable x_t is not assumed to be observed for the derivations of the messages. The variational messages of Table 3.1 have been derived by the substitution of the approximate posterior distributions into (3.10).[1]

All messages, except for $\overleftarrow{\nu}(a_m)$, are of the same functional form as the corresponding approximate posterior distribution. Since the Gamma and categorical distributions are closed under multiplication, the resulting updated approximate posterior distributions remain in the same family of distributions. However, the message $\overleftarrow{\nu}(a_m)$ has a functional form that makes a closed-form result for the approximate posterior distribution infeasible. Therefore, to make the calculations tractable we will approximate $q(a_m)$ by a parametric distribution, see Table 3.1. In the remainder of this section, we will propose two solutions: (1) expectation-maximization and (2) moment matching.

3.3.3 Solution 1: Expectation-maximization (VMP-EM)

The first proposed solution uses VMP in conjunction with expectation-maximization (VMP-EM) to approximate the resulting posterior distribution of a_m using message passing, inspired by [71]. Here the posterior distribution $q(a_l)$ is fixed to a Dirac delta function

$$q(a_l) = \delta(a_l - \hat{a}_l) \tag{3.13}$$

instead of the Gamma distribution from Table 3.1. This distribution is located at $\hat{a}_l$, whose value is obtained through expectation-maximization using message passing according to [71]. The location $\hat{a}_l$ is determined by

$$\hat{a}_l = \underset{a_l}{\operatorname{argmax}}(\log \vec{\nu}(a_l) + \log \overleftarrow{\nu}(a_l))\,, \text{ s.t. } a_l > 0\,, \tag{3.14}$$

where the message $\overleftarrow{\nu}(a_l)$ represents the variational message from Table 3.1.

Theorem 1. *The solution of the constrained maximization problem given by* (3.14) *exists and is unique.*

Proof. From Table 3.1 we know $\log \overleftarrow{\nu}(a_l) = \hat{\pi}_t \left(a_l \zeta_{kl} - \log \Gamma(a_l)\right)$. Since the logarithm of the Gamma function is strictly convex when restricted to positive real numbers (Bohr-Mollerup theorem) [98], $\log \overleftarrow{\nu}(a_m)$ is strictly concave as it is a summation of affine and a strictly concave term [99, Ch. 2.3]. Because the prior message $\vec{\nu}(a_m)$ is proportional to a Gamma distribution, $\log \vec{\nu}(a_l)$ is either affine or concave depending on the shape parameter. Hence, $\log\left(\vec{\nu}(a_l)\overleftarrow{\nu}(a_l)\right)$ is always strictly concave. Because it is concave the maximum exists by strong duality [99, Ch. 5.3.2] and is unique because concavity is strict. □

[1] The derived messages are available in the supplementary material at `https://github.com/mlsp2021-gmm/gmm-experiments`.

3.3.4 Solution 2: Moment matching (VMP-MM)

Expectation-maximization provides us with a single estimate of the parameter a_l. If instead, we would like to retain uncertainty about this parameter, we could approximate the resulting marginal distribution by a Gamma distribution using VMP with moment matching (VMP-MM), realized by importance sampling (IS) [95, Ch.7]. The IS procedure approximates the target distribution $q(a_l)$ by drawing M samples $a_l^{(m)}$ from an *importance distribution* $\tilde{q}(a_m)$ as

$$a_l^{(m)} \sim \tilde{q}(a_l) = \frac{\vec{\nu}(a_l)}{\int_{\mathbb{R}_{>0}} \vec{\nu}(a_l)\mathrm{d}a_l}, m = 1, \ldots, M. \tag{3.15}$$

We choose the normalized forward message $\tilde{q}(a_l)$ as the importance distribution. We can make this choice because the support of the importance distribution is $\mathbb{R}_{>0}$, which coincides with the support of the multiplication $\vec{\nu}(a_l)\overleftarrow{\nu}(a_l)$. The mean and variance of a_l can then be approximated by

$$\mathbb{E}[a_l] \approx \sum_{m=1}^{M} a_l^{(m)} \vec{\nu}(a_l^{(m)})/Z \tag{3.16a}$$

$$\mathrm{Var}[a_l] \approx \sum_{m=1}^{M} (a_l^{(m)} - \mathbb{E}[a_l])^2 \vec{\nu}(a_l^{(m)})/Z\,, \tag{3.16b}$$

where $Z = \sum_{m=1}^{M} \vec{\nu}(a_l^{(m)})$ is the normalization constant. In our implementation, we employ adaptive resampling [95, Ch.7] to avoid the degeneracy problem for the estimates obtained by (3.16a) and (3.16b).

Theorem 2. *For $M \to \infty$ the summations given by* (3.16) *converge to the true mean and variance of $q(a_l)$.*

Proof. The numerator of (3.16a) $\sum_{m=1}^{M} a_l^{(m)} \vec{\nu}(a_l^{(m)})$ is the average of $a_l q(a_l^{(m)})/\tilde{q}(a_l^{(m)})$ under-sampling from $\tilde{q}(a_l^{(m)})$. These numerators for different M are independent and identically distributed random variables with mean $\mathbb{E}[a_l]$ [100]. The strong law of large numbers gives

$$\mathbb{P}\left\{ \lim_{M\to\infty} \sum_{m=1}^{M} a_l^{(m)} \vec{\nu}(a_l^{(m)})/Z = \mathbb{E}[a_l] \right\} = 1. \tag{3.17}$$

The denominator of (3.16a) Z converges to 1. □

With the mean and the variance the parameters of the Gamma distribution $q(a_l)$ from Table 3.1 can be determined as

$$\hat{\alpha}_l^{(a)} = \frac{\mathbb{E}[a_l]^2}{\mathrm{Var}[a_l]}, \qquad \hat{\beta}_l^{(a)} = \frac{\mathbb{E}[a_l]}{\mathrm{Var}[a_l]}\,. \tag{3.18}$$

Note that, unlike VMP-EM which yields a point estimate by determining (3.14), VMP-MM results in a proper posterior distribution for a_m.

3.4 Experiments

All experiments were implemented in the Julia programming language [101].[2] We used the following computer configuration: *Operating system*: macOS Big Sur, *Processor*: 2,7 GHz Quad-Core Intel Core i7, *RAM*: 16GB.

3.4.1 Verification

For the verification stage, we followed the setup from [92], where the Markov chain Monte Carlo was used for inference in a ΓMM. We generated data using three distinct ΓMMs, each specified by likelihood (3.1) with a different number of mixture components $L = \{2, 3, 4\}$. We fixed the shape and rate parameters a_l and b_l to the values in Table 3.2.

Table 3.2: Shape and rate parameters of the ΓMMs used for data generation.

	$\boldsymbol{a}$	$\boldsymbol{b}$
$L = 2$	[9, 90]	[27, 270]
$L = 3$	[40, 6, 200]	[20, 1, 20]
$L = 4$	[200, 400, 600, 800]	[100, 100, 100, 100]

Each of these models exhibits different behavior, as illustrated in Fig. 3.2. For $L = 2$, the mixture components have equal means, but different variances. For $L = 3$, two mixture components are well separated and have low variances. The third mixture has a large variance and overlaps with the other two mixtures. Finally, for $L = 4$ we have four well-separated mixtures. For each model, we generated 10 distinct data sets with different mixing coefficients. These mixing coefficients were sampled from a standard uniform distribution and normalized by dividing by the sum of the coefficients. Each data set contains $T = 2500$ observations (in total $3 \times 10 \times 2500$ data points). To verify the proposed inference method, we selected three generative models for which we assumed the number of components to be known. We then performed probabilistic inference through message passing for two situations. The first situation (known shape-rate) uses informative priors for $\boldsymbol{a}$ and $\boldsymbol{b}$ and a vague prior for $\boldsymbol{\pi}$. The second setup (known mixing) employs an informative prior for the mixing coefficient $\boldsymbol{\pi}$, but uninformative priors for $\boldsymbol{a}$ and $\boldsymbol{b}$. With informative priors, we imply that the distributions are centered at an ϵ-area ($\epsilon > 0$, $\epsilon^2 \approx 0$) of the values that were used for data generation. Priors were chosen so as not to violate the properties of the corresponding distributions. We motivate the usage of informative priors for either mixing coefficients or parameters of the gamma distribution for two reasons. First, based on a Bayesian analysis of the Gamma distribution [102], the choice of non-informative priors for small data sets

[2] Experiments are available at `https://github.com/mlsp2021-gmm/gmm-experiments`.

generally leads to low accuracy. We should choose the priors of the ΓMM carefully, as its parameter space is significantly larger than that of a single Gamma distribution. Secondly, due to the non-convexity of the mean-field assumption, we have multiple solutions for our inference problem [103, Ch.5]. Thus, the initialization of vague priors for all parameters of ΓMM can lead to undesirable local minima. The inference task, as specified in Section 3.2, computes the quantities $q(\boldsymbol{a}|\boldsymbol{x}_{1:T})$, $q(\boldsymbol{b}|\boldsymbol{x}_{1:T})$, $q(\boldsymbol{s}|\boldsymbol{x}_{1:T})$ and $q(\boldsymbol{\pi}|\boldsymbol{x}_{1:T})$. The notation $q(\cdot|\boldsymbol{x}_{1:T})$ refers to the marginals after observing the data. In this experiment, we first want to ensure that the proposed algorithm recovers the unknown parameters of the mixture components. Additionally, we want to verify the convergence of the proposed methodology by monitoring the VFE $F[q(\cdot)]$. We now highlight the results of the verification stage in Fig. 3.2. For the VMP-MM approach we used $M = 5000$. Both algorithms recover the parameters of the ΓMM in the aforementioned situations. Both algorithms converge, which is reflected by the evolution of the VFE in Fig. 3.3. The VMP-EM approach converges more slowly than the VMP-MM approach as a function of iteration count, but for this experimental setup, VMP-MM is on average, approximately 30 times slower in evaluation time than VMP-EM due to the relatively expensive sampling procedure.

3.4.2 Validation

For the validation of our model, we used the country data set from Kaggle.[3] This data set contains socio-economic and health data for all countries worldwide. The task is to categorize the countries based on data features. Most of the individual features represent positive real values. Therefore the ΓMM appears as a possible approach to modeling. For the brevity of the experiment, we transformed the "inflation" feature (4.8% entries are negative) to a positive real range. Unlike the experiments on the generated data sets, we now have to deal with multivariate observations. Each observation x_t is now represented by a vector of $N = 7$ features as $x_t = [x_t^{(1)}, ..., x_t^{(N)}]$, where the superscript denotes the feature, indexed by n, and where $x_t^{(n)} \in \mathbb{R}_{>0}$ for all $n = 1, \ldots, N$. For modeling the multivariate observations, we model each feature independently using a separate ΓMM, where the features are modeled by the same selector variable $\boldsymbol{s}_t$. The likelihood of each component of x_t then becomes

$$x_t \sim \prod_{l=1}^{L} \prod_{n=1}^{N} \Gamma\left(a_l^{(n)}, b_l^{(n)}\right)^{\boldsymbol{c}_t} \tag{3.19}$$

and we change (3.2) and (3.3) to contain $M \times N$ independent mixture components, such that each feature is modelled by its own set of mixture components.

In this setup, we do not have any prior information about the mixing coefficients. To obtain informative priors for the shape and rate parameters of the mix-

[3] https://www.kaggle.com/rohan0301/unsupervised-learning-on-country-data

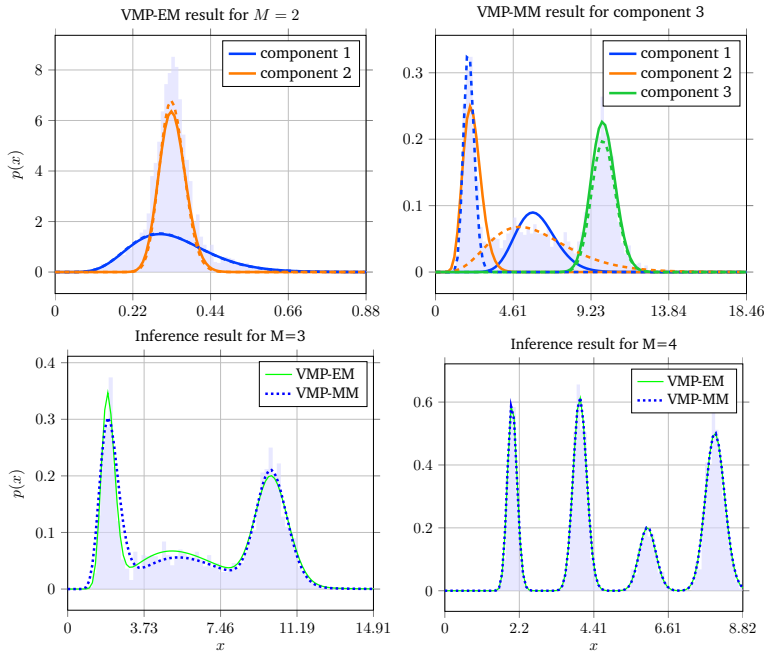


Figure 3.2: Verification results. The shaded light-blue bar plots in the background denote the normalized histograms of the generated data. (Top) The dashed and solid lines denote the actual and estimated density functions, respectively. (Top-Left) Inference results for the VMP-EM approach for two components with informative shape and rate parameters. The estimated and actual densities match, meaning that the mixing coefficient is inferred properly. (Top-Right) Inference results of the VMP-MM approach for three components with known mixing coefficients. The estimated mixture components 1 and 2 were swapped. The variance of the estimated component 1 is lower than the corresponding actual component 2. In contrast, the estimated component 2 has a larger variance than the actual component 1. The estimated component 3 features shape and rate parameters that are close to the parameters of the corresponding generated mixture. (Bottom) The dashed and solid lines denote the density functions estimated by VMP-EM and VMP-MM, respectively. (Bottom-Left) Comparison of both algorithms for three components with informative mixing coefficients. Both algorithms provide reasonable estimates of the shape and rate parameters for each mixture. (Bottom-Right) Comparison of two algorithms for a mixture of four components with informative shape and rate parameters. Both algorithms lead to correct mixing posteriors.

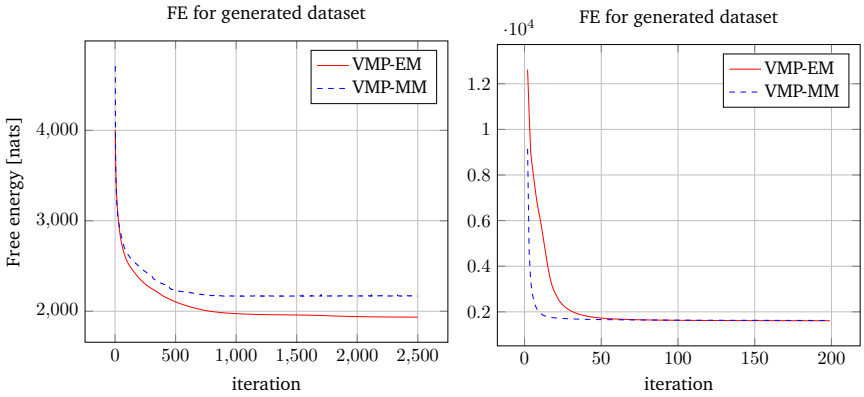


Figure 3.3: Verification results. Evolution of the variational free energy for the VMP-EM and VMP-MM algorithms averaged over their corresponding data sets. (Left) Situation with informative mixing coefficients. (Right) The situation with informative shape and rate parameters

ture components, we extracted the empirical means and variances of each feature and converted those to the shape and rate parameters of a Gamma distribution using (3.18). To disentangle the priors of shape and rate parameters, we added a positive random jitter term to each shape and rate parameter of the prior distributions. To determine the optimal number of mixture components, we tracked the values of mixing coefficients $\boldsymbol{\pi}$ for different numbers of mixture components $L = \{2, \ldots, 10\}$. Mixing coefficients that converge to 0 indicates the absence of the corresponding cluster [47, Ch. 10]. We highlight the inference results of the proposed algorithms in Figure 3.4. Based on this approach, both VMP-MM and VMP-EM experiments show that $L = 3$ is the optimal number of components. To visualize the inferred components, we used the t-distributed stochastic neighbor embedding (tSNE) [104]. tSNE provides an intuition of how the high-dimensional data is arranged by mapping the data onto a lower-dimensional space. Figure 3.5 shows the result of the tSNE projection for the country data set. We colored the data points according to $\underset{\boldsymbol{c}_t}{\operatorname{argmax}}(q(\boldsymbol{c}_t))$, i.e., the most likely mixture component of the corresponding marginal. In this way, the labels provided by VMP-EM and VMP-MM are identical.

3.5 Discussion and conclusions

The proposed inference methods, VMP-EM and VMP-MM, converge and correctly identify the parameters of ΓMM. However, although VMP-MM yields a "full" poste-

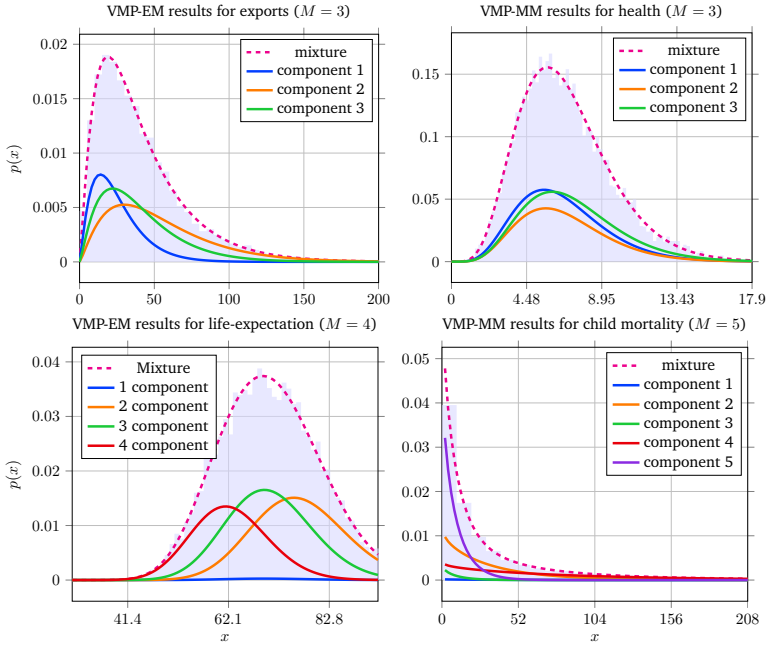


Figure 3.4: Validation results for separated features. The dashed line denotes estimated ΓM. Solid lines correspond to individual mixture components. (Top-left) Inference result of VMP-EM algorithm for "exports" feature when $L = 3$. (Top-right) Inference result of VMP-MM algorithm for health feature when $L = 3$. (Bottom-left) Inference result of VMP-EM algorithm for "life expectation" feature when $L = 4$. (Bottom-right) Inference result of VMP-MM algorithm for "child mortality" feature when $L = 5$.

rior distribution, it suffers from a slower evaluation time. In contrast, while VMP-EM enjoys a relatively fast evaluation time, it provides only point estimates for the shape parameters of the mixture components. It makes VMP-EM challenging to employ in an online learning scenario when new observations become available in sequential order.

For the validation experiments, we transformed the "inflation" feature to a real positive range, although this approach is undesirable as it breaks the natural support of the corresponding random variable. Alternatively, we could have substituted the ΓM node that models "inflation" with a Gaussian Mixture (GM) node [72], leading to a hybrid model that connects ΓM and GM nodes through selector variables.

We presented a variational message-passing approach for inferring the parameters in Gamma mixture models. The required variational messages are summarized in Table 3.1. We proposed two approaches for computing the marginal distribution

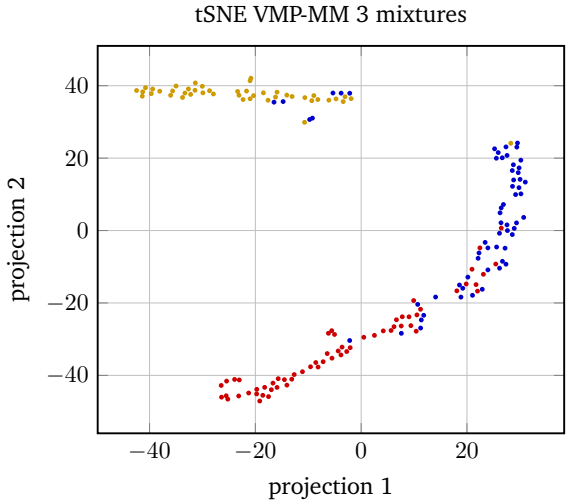


Figure 3.5: tSNE visualization of validation experiments. The data points are colored according to the most likely mixture component of the corresponding marginal.

of the shape parameters of the Gamma mixture model. Furthermore, we demonstrated the convergence of the inference procedure through the minimization of variational free energy. The correctness of the message-passing scheme was verified on a synthetic data set. The Gamma mixture node can now be used as a plug-in node in any graphical model that supports message passing-based inference. Owing to the locality and modularity of the FFG framework, we showed how the Gamma mixture model can be easily extended to tackle multi-dimensional problems such as the clustering of countries. In future work, we plan to use the Gamma mixture node for probabilistic modeling of time series that exhibit switching behavior.

Chapter 4

Message Passing-based Inference in Switching Autoregressive Models

This chapter is based on the original work referenced below. Notations have been adjusted to reflect conventions throughout the dissertation.

> **Albert Podusenko**, Bart van Erp, Dmitry Bagaev, Ismail Senoz, Bert de Vries, *Message Passing-Based Inference in Switching Autoregressive Models*. In 30th European Signal Processing Conference (EUSIPCO 2022) - Proceedings

Abstract

The switching autoregressive model is a flexible model for signals generated by non-stationary processes. Unfortunately, the evaluation of the exact posterior distributions of the latent variables for a switching autoregressive model is analytically intractable. This limits the applicability of switching autoregressive models in practical signal processing tasks. This paper presents a message passing-based approach for computing approximate posterior distributions in the switching autoregressive model. Our solution tracks approximate posterior distributions in a modular way and easily extends to more complicated model variations. The proposed message passing algorithm is verified and validated on synthetic and acoustic data sets respectively.

4.1 Introduction

Autoregressive (AR) models have been widely used to represent acoustic signals, such as speech signals [105, 83] or background noise [106, 107]. In order to take into account non-stationary behavior, switching autoregressive (SwAR) models have been developed as an extension to standard AR models [108], [66, Ch. 24.6]. This extension from the original AR model may lead to increasing model performance but also leads to a more complicated inference procedure.

Technically, the SwAR model only differs from the regular AR model through its prior distributions on the parameters, as will be specified in detail in Section 4.2. Instead of deriving all update equations for state and parameter estimation in this specific model by hand, as was done for the simplified model in [66, Ch. 24.6], we automate inference by message passing in (Forney-style) factor graph (FFG) representation of the model [93, 94]. The local message update equations have been pre-derived for the constituent factor nodes of the SwAR model in earlier works [25, Apps. 2 & 9], [109], which allows us to automatically generate an inference algorithm for the SwAR model.

This paper describes a message passing-based approach for performing probabilistic inference in the switching autoregressive model. We make the following contributions:

- A switching autoregressive model is specified where both states and parameters are treated as latent variables in Section 4.2.
- The basic SwAR model is extended with temporal dynamics for the active switching states evolving over a different time scale in Section 4.2.
- We state our problem definition as an inference task on the SwAR model in Section 4.2, and show how this inference task can be realized through message passing-based inference in an FFG in Section 4.4.
- We demonstrate our proposed methodology through a set of verification and validation experiments in Section 4.5.

Finally, we discuss the obtained results and conclude the paper in Section 4.6.

4.2 Model specification

Let $\boldsymbol{y}_t \triangleq [y_t, \ldots, y_{t-K+1}]^\top \in \mathbb{R}^K$, denote a vector of the K latest observations at time t. The likelihood function of a SwAR model is defined as

$$y_t \sim \mathcal{N}\left(\boldsymbol{\theta}_i^\top \boldsymbol{y}_{t-1}, \gamma_i^{-1}\right), \tag{4.1}$$

where we use $\mathcal{N}(m, \gamma^{-1})$ to denote a Gaussian distribution with mean m and precision γ. $\boldsymbol{\theta}_i = [\theta_{1i}, ..., \theta_{Ki}]^\top \in \mathbb{R}^N$ and $\gamma_i \in \mathbb{R}_{>0}$ denote the autoregressive coefficients and process noise precision of the N^{th}-order SwAR model, respectively.

The vector of previous observations $\boldsymbol{y}_{t-1}$ is updated with the next observation y_t according to [110] by

$$\boldsymbol{y}_t = \boldsymbol{S}\boldsymbol{y}_{t-1} + \boldsymbol{e}_1 y_t \tag{4.2}$$

where

$$\boldsymbol{S} \triangleq \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \boldsymbol{I}_{K-1} & \mathbf{0} \end{bmatrix}, \quad \boldsymbol{e}_1 \triangleq [1, 0, \ldots, 0]^\top . \tag{4.3}$$

We assume the parameters of SwAR to be stationary over longer segments of time and therefore index them with the slower-evolving switching state index $i = 1, \ldots, N$, related to t as $i = \lceil t/N \rceil$.

Here, $\lceil \cdot \rceil$ denotes the ceiling function that returns the largest integer smaller or equal than its argument, while W is the window length. The above equation makes sure that i is intuitively aligned with segments of length W, i.e. $t \in [1, W]$ corresponds to $i = 1$. To denote the start and end indices of the time segment

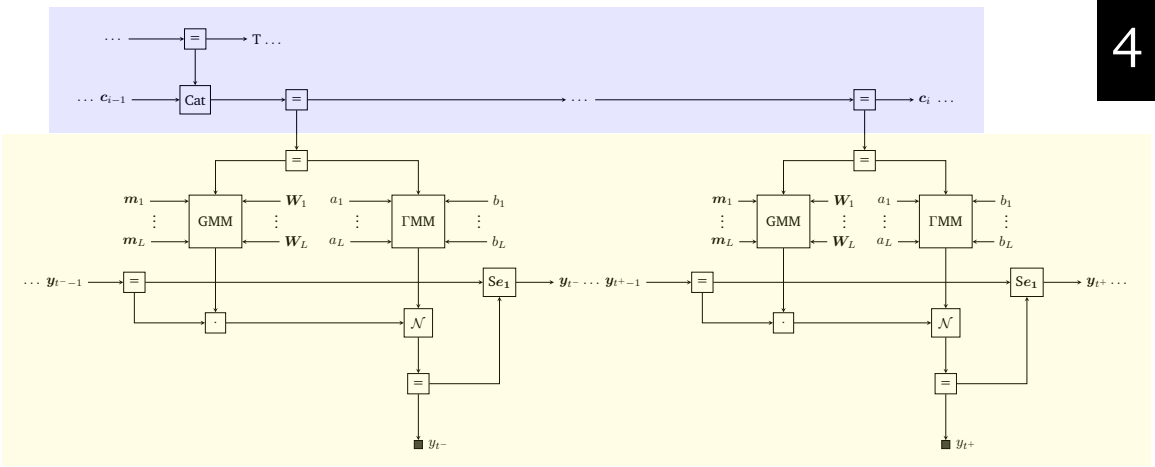


Figure 4.1: A Forney-style factor graph of a single switching state time slice of the switching autoregressive model. Each switching state time segment indexed over k (blue part) is connected to a total of W repetitions of the part of the model indexed over t (yellow part). The GMM and ΓMM nodes denote the Gaussian and Gamma mixture nodes respectively. The Sc node represents (4.2). Propagating messages backward from future time steps results in an inference smoothing algorithm, whereas if we only propagate messages forward in time, a filtering algorithm results.

corresponding to switching state index i, we define $t^- = (i-1)W + 1$ and $t^+ = iW$ as an implicit function of i, respectively. Implicitly we also constrain (4.1) to only be valid for matching time indices, i.e. for $t = t^-, t^- + 1, \ldots, t^+$.

The AR likelihood function of (4.1) is extended with the mixture models

$$\boldsymbol{\theta}_i \sim \prod_{l=1}^{L} \mathcal{N}\left(\boldsymbol{m}_l, \boldsymbol{W}_l^{-1}\right)^{\boldsymbol{c}_i} \quad \gamma_i \sim \prod_{l=1}^{L} \Gamma\left(a_l, b_l\right)^{\boldsymbol{c}_i} \tag{4.4}$$

to form a SwAR model with N switching states or contexts. Here $\Gamma(a, b)$ denotes the Gamma distribution with shape and rate parameters a and b, respectively. The variable $\boldsymbol{c}_i = [c_{1i}, \ldots, c_{Ni}]^\top$ denotes a 1-of-N binary vector with elements $c_{li} \in \{0, 1\}$, constrained by $\sum_l c_{li} = 1$. The switching behavior is achieved by modeling the temporal dynamics as

$$\boldsymbol{c}_i \sim \mathrm{Cat}(\mathrm{T}\boldsymbol{c}_{i-1}), \tag{4.5}$$

where $\mathrm{Cat}(\boldsymbol{\pi})$ denotes a categorical distribution with event probabilities $\boldsymbol{\pi}$. We model the individual columns of the transition matrix T by a Dirichlet distribution $\mathrm{Dir}(\boldsymbol{\zeta})$ as

$$\mathrm{T}_{1:N,j} \sim \mathrm{Dir}(\boldsymbol{\zeta}_j), \tag{4.6}$$

where $\boldsymbol{\zeta}_j$ denotes the vector of concentration parameters corresponding to the j^{th} column of T. The switching state is initialized by a categorical distribution as

$$\boldsymbol{c}_0 \sim \mathrm{Cat}(\boldsymbol{\pi}_0) = \prod_{l=1}^{L} \pi_{l0}^{c_{l0}} \quad \text{such that} \quad \sum_{l=1}^{L} \pi_{l0} = 1, \tag{4.7}$$

where the individual event probabilities can be chosen as $\pi_{l0} = 1/L$ if the initial switching state is unknown. Additionally, we assign prior probability distributions to the hyperparameters of the SwAR model in (4.4):

$$\begin{aligned} \boldsymbol{m}_l &\sim \mathcal{N}(\boldsymbol{\mu}_{l0}, \boldsymbol{\Lambda}_{l0}^{-1}) \quad & \boldsymbol{W}_l &\sim \mathcal{W}(\boldsymbol{V}_0, n_0) \\ a_l &\sim \Gamma(\alpha_{l0}^{(a)}, \beta_{l0}^{(a)}) \quad & b_l &\sim \Gamma(\alpha_{l0}^{(b)}, \beta_{l0}^{(b)}) \end{aligned} \tag{4.8}$$

with $\mathcal{W}(\cdot, \cdot)$ denoting the Wishart distribution.

The SwAR model described by (4.1)-(4.8) can be represented by a Forney-style Factor Graph (FFG) as depicted in Figure 4.1. An FFG is an *undirected* graph where nodes represent factors of a global function and edges represent variables [93]. In an FFG, an edge is connected to a node if and only if the factor corresponding to the node is a function of the variable corresponding to the edge. If the variable is shared between more than 2 factors, we can make use of equality nodes of type $\delta(x - x')\delta(x - x'')$ that constrain the beliefs over two "copy variables" x' and x'' to be equal to the belief over x [69]. In an FFG, factors are drawn as square open nodes and observations or fixed variables are represented by small black squares, whose factors can be regarded as Dirac delta functions centered on the observed value. For a detailed explanation of the FFG formalism, we refer to [94, 48, 43].

4.3 Problem statement

Given a SwAR model and a collection of observations $\boldsymbol{y}$, we are interested in tracking the marginal distributions of the model's latent variables. Computation of these posterior distributions requires the integration and summation of the model specified by (4.1)-(4.6) with respect to all nuisance variables. These computations do not yield any analytical solutions and therefore lead to an intractable probabilistic inference. This paper addresses the problem of computing approximate marginal distributions in the SwAR model.

4.4 Inference

This section describes how probabilistic inference can be realized in the SwAR model.

4.4.1 Variational message passing

The factorized structure of the SwAR model allows for the distributed calculation of the posterior distributions of its variables through a set of smaller local computations called messages. Intractability in these computations prevents us from performing exact message passing-based inference, also known as belief propagation [97] or the sum-product algorithm [96]. Consequently we result to variational message passing (VMP) [68, 77].

To illustrate this, consider the probabilistic model $p(\boldsymbol{y}, \boldsymbol{z})$, with observations $\boldsymbol{y}$ and latent variables $\boldsymbol{z}$. As the computation of the exact posterior $p(\boldsymbol{z}|\boldsymbol{y})$ is intractable, we resort to variational inference, where we approximate the true posterior distribution by the tractable approximate posterior distribution $q(\boldsymbol{z}) \approx p(\boldsymbol{z}|\boldsymbol{y})$. The probabilistic inference then concerns the minimization of the variational free energy (VFE) functional

$$F[q] = D_{\mathrm{KL}}[q(\boldsymbol{z}) \| p(\boldsymbol{z}|\boldsymbol{y})] - \ln p(\boldsymbol{y}), \tag{4.9}$$

where D_{KL} is the Kullback-Leibler divergence. To enable efficient optimization of the VFE for the SwAR model we assume an additional factorization on $q(\boldsymbol{z})$,

$$q(\boldsymbol{z}) = \prod_a q_a(\boldsymbol{z_a}) \,, \tag{4.10}$$

where $\boldsymbol{z_a}$ refers to a set of node-bound local variables (one or many) such that $\cup_a \boldsymbol{z_a} = \boldsymbol{z}$. To enable efficient optimization of the VFE, the approximate posterior distribution is factorized under a mean-field assumption as

$$q(\boldsymbol{z}) = \prod_i q_i(z_i). \tag{4.11}$$

VMP concerns the iterative updating of marginals as $q_j(z_j) \propto \vec{\nu}(z_j) \cdot \overleftarrow{\nu}(z_j)$, where $\vec{\nu}(z_j)$ and $\overleftarrow{\nu}(z_j)$ are forward and backward variational messages on edge z_j. The outgoing variational message $\vec{\nu}(z_j)$ on edge z_j from a factor $f(\boldsymbol{z})$, with incoming marginals $q_i(z_i)$ for $i \neq j$, can be derived as [77]

$$\vec{\nu}(z_j) \propto \exp \int \prod_{i \neq j} q_i(z_i) f(\boldsymbol{z}) \mathrm{d}\boldsymbol{z}_{\setminus j}. \tag{4.12}$$

The approximate marginals $q_i(z_i)$ and variational messages $\vec{\nu}(z_j)$ and $\overleftarrow{\nu}(z_j)$ are iteratively updated until the VFE converges.

4.4.2 Expectation maximization

As a further specification of the VMP procedure, we can constrain the form of approximate marginals to $q_j(z_j) = \delta(z_j - \hat{z}_j)$. By selecting $\hat{z}_j$ through the optimization problem

$$\hat{z}_j = \underset{z_j}{\operatorname{argmax}} (\vec{\nu}(z_j) \overleftarrow{\nu}(z_j)) \,, \tag{4.13}$$

we perform a local expectation-maximization procedure through message passing [71, 43]. This constraint is enforced for the variables α_l in the Gamma mixture node [111].

4.4.3 Inference in the switching autoregressive model

Inference in the SwAR model of (4.1)-(4.8) is performed through a hybrid message passing scheme that includes both sum-product and variational messages. By enforcing different variational constraints on the approximate posterior distributions of the variables in the model, we can obtain different local inference procedures [43]. The graph in Figure 4.1 submits to a combination of sum-product message passing, (structured) VMP, and expectation maximization. Around all deterministic nodes, sum-product message passing is performed. Expectation maximization is performed on the edges corresponding to the variables α_l and all other variables submit to (structured) VMP. The message passing update rules for all nodes have already been derived in previous works. Update rules corresponding to the mixture nodes of (4.4) can be found in [25, Table A2] for the Gaussian mixture node and [109, Table I] for the Gamma mixture node. [25, Table A5] summarizes the update rules for the nodes corresponding to the switching state transition of (4.5). The update rules corresponding to the Gaussian factor in (4.1) are summarized in [25, Table A1].

4.5 Experiments

All experiments[1] have been implemented in the Julia programming language [101]. We used the following computer configuration: *Operating system*: macOS Big Sur, *Processor*: 2,7 GHz Quad-Core Intel Core i7, *RAM*: 16GB.

4.5.1 Verification experiments

To verify the proposed inference method, we synthesized data from 100 SwAR generative models with the likelihood in (4.1) with AR order $K = 2$ and $L = 2$ switching states. To ensure the stationarity of the generated processes, we resample unstable process configurations. An example of a generated SwAR signal is shown in Figure 4.2. We used uninformative priors for the transition matrix T and

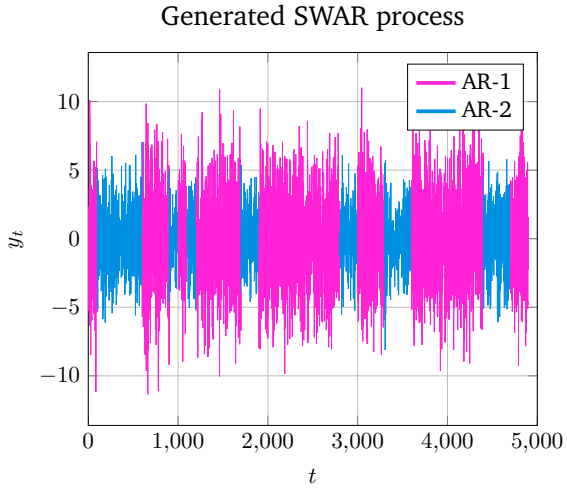


Figure 4.2: Example of a generated SwAR signal. The constituent AR processes have been denoted by different colors.

initial switching state $\boldsymbol{c}_0$. As for the rest of the model parameters, we used informative priors, i.e., the means of the prior distributions are centered at an ϵ-area ($\epsilon > 0$, $\epsilon^2 \approx 0$) of the means of the corresponding generative distributions. We motivate the usage of informative priors by the non-convexity of the mean-field assumption of our approximate posterior distribution around mixture nodes. This induces multiple solutions for our inference task [103, Ch. 5]. Following the problem definition task in Section 4.3, we seek to obtain the quantities $q(\boldsymbol{\theta}_i|\boldsymbol{y})$, $q(\gamma_i|\boldsymbol{y})$,

[1] All experiments are available at `https://github.com/biaslab/swar`.

$q(\boldsymbol{c}_i|\boldsymbol{y})$, $q(\mathrm{T}|\boldsymbol{y})$ and $q(\boldsymbol{m}_l|\boldsymbol{y})$, $q(\boldsymbol{W}_l|\boldsymbol{y})$, $q(a_l|\boldsymbol{y})$, $q(b_l|\boldsymbol{y})$ for every $l = 1, \ldots, L$. The notation $q(\cdot|\boldsymbol{y})$ refers to the marginal distribution after all observations $\boldsymbol{y}$.

Additionally, we want to verify the convergence of the proposed methodology by monitoring the VFE. The inference results are presented in Figures 4.3 and 4.4.

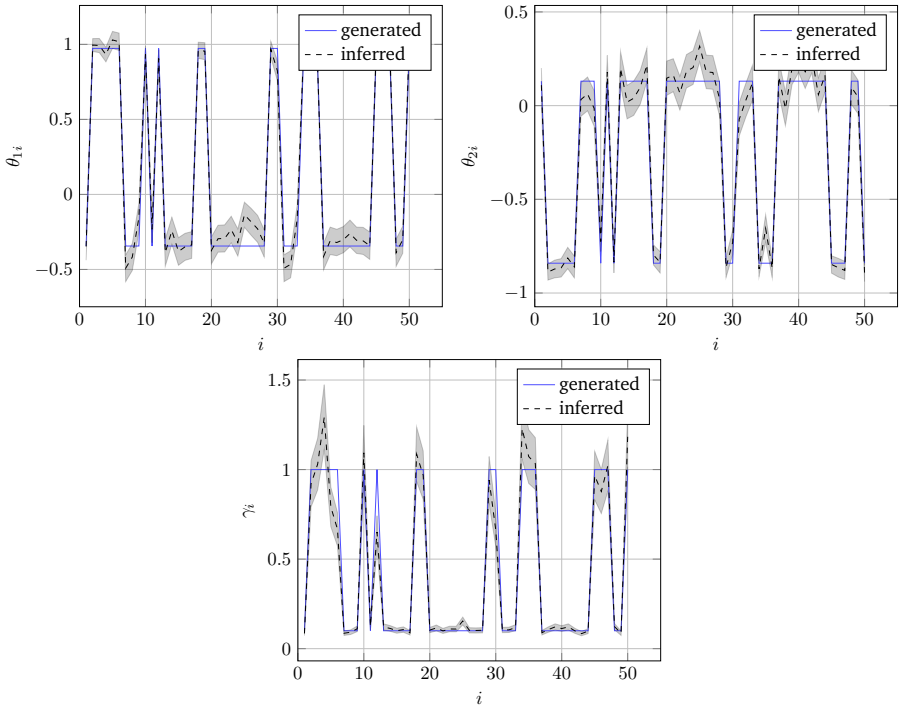


Figure 4.3: Inference results on the synthetic dataset. The dashed lines correspond to the expected values of the posterior estimates. The shaded regions correspond to the inferred standard deviation of the approximate posterior distributions around the estimated mean. The solid blue lines correspond to the true underlying values of the latent parameters in the generative processes. (Top) Inference results for the AR coefficients obtained from the joint marginal distribution $q(\boldsymbol{\theta}_i|\boldsymbol{y})$. (Bottom) Inferred approximate posterior distributions of the precision variables $q(\gamma_i|\boldsymbol{y})$.

We evaluate the performance of the inference for the switching states by computing a categorical accuracy metric, defined as

$$acc = \frac{tp + tn}{R \cdot M}, \tag{4.14}$$

where tp, and tn are the number of true positive and true negative values, respectively. R corresponds to the number of total synthetic data sets, which in this ex-

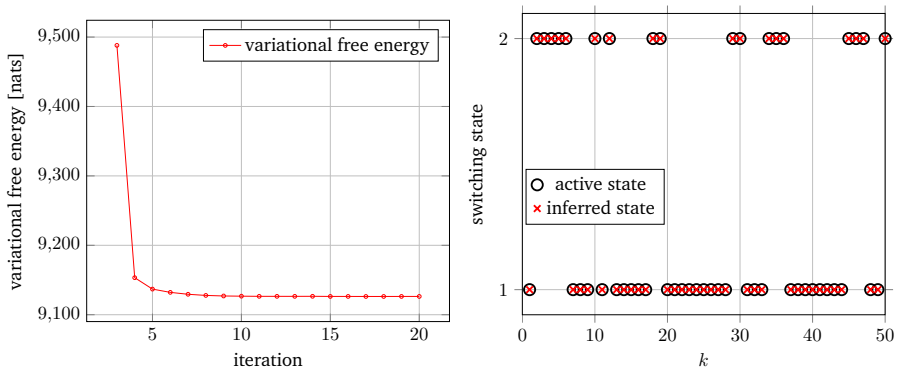


Figure 4.4: Inference results on the synthetic dataset. (Left) Evolution of the variational free energy averaged over all generated data sets. (Right) True and inferred evolution of the switching state per frame. Each frame consists of $W = 100$ data points. Circles denote the active switching states that were used to generate the frame. Crosses denote the mode of the inferred switching states.

periment is set to $R = 100$. In this experiment, we achieved a categorical accuracy of $acc = 0.84$.

4

4.5.2 Validation experiments

To validate the proposed inference procedure, we used 8 seconds of an audio signal, composed of the concatenation of sounds from two different acoustic environments: a train station and a bar. Specifically, we have ≈ 2.6 sec of train sound, followed by ≈ 2.6 sec of bar noise, ending with another train station noise of ≈ 2.6 sec. The sampling frequency was 8 kHz and the audio file is available at `https://github.com/biaslab/swar/data/`. The task is to identify the states of each window or to classify which acoustic environment is present in the window. In our experiment, we set the maximum window size to 15000 samples (or 1.875 seconds). In this way, our signal breaks into $M = 5$ windows, where the 5th window contains 4000 samples. The choice of 15000 reflects our beliefs about the temporal structure of the signal. In other words, we assume that the switches in the acoustic signal happen at the seconds-level, not at the milliseconds level. We used informative priors for the AR coefficients and precision parameters of the SwAR model. These priors were obtained from performing parameter estimation in the autoregressive model [111]. We have little prior information about the initial state of the audio signal. Thus, we assigned vague (uninformative) priors for the initial state $\boldsymbol{c}_0$ and transition matrix T. We present the inference result in Figure 4.5. Although some frames contain overlapping acoustic signals due to current segmentation, good classification results

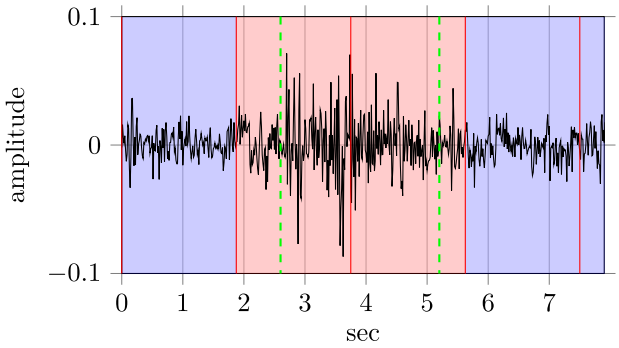


Figure 4.5: Inference results for the audio signal. The acoustic signal is represented by a solid black line. Windows of 15000 samples are separated by red solid lines. The green vertical lines correspond to the locations where the underlying acoustic signal changes. The first frame was identified as a train sound (blue region). The two frames in the middle signify a bar sound (red region). The last two frames were classified as train sound.

were achieved through the automated message passing-based inference procedure.

4

4.6 Discussion and Conclusion

We have introduced a SwAR model that includes efficient joint variational tracking of states, parameters, and variational free energy. In this work, we have demonstrated just one way of approximating the posterior distribution of α_l. In particular, we employed a local expectation-maximization procedure to estimate the α_l parameter. Although this approach delivers reasonable estimates, it is not suited for online inference scenarios. For these scenarios, one could resort to the moment matching procedure as proposed in [109].

This paper introduced the SwAR model composed of a Gaussian and Gamma mixture model. Owing to the modularity of the factor graph approach, this model can be easily extended, and its inference algorithm can be automatically generated based on efforts from previous works. The correctness of the proposed message passing-based inference has been verified on multiple datasets synthesized from the SwAR model. Finally, we demonstrated the convergence of the inference procedure through the minimization of VFE. The proposed model can be easily extended to a latent SwAR model using the update rules of [111], where instead of directly observing y_t we observe a noisy variable $z_t \sim \mathcal{N}\left(y_t, \tau^{-1}\right)$. In future work, we aim to use the SwAR model as a module in more complex hierarchical systems.

Chapter 5

AIDA: An Active Inference-based Design Agent for Audio Processing Algorithms

This chapter is based on the original work referenced below. Contributions are split evenly among the first three authors. The first author made the most significant contribution to creating the generative model for the acoustic environment and developing the demonstrator. The original idea, simulations, and text have been established in close collaboration between the first three authors. Notations have been adjusted to reflect conventions throughout the dissertation.

> **Albert Podusenko**, **Bart van Erp**, **Magnus Koudahl**, Bert de Vries, *AIDA: An Active Inference-Based Design Agent for Audio Processing Algorithms*, Special issue on Advances in Speech Enhancement using Audio Signal Processing Techniques, Frontiers in Signal Processing, 2022

Abstract

In this paper, we present AIDA, an active inference-based agent that iteratively designs a personalized audio processing algorithm through situated interactions with a human client. The target application of AIDA is to propose on-the-spot the most interesting alternative values for the tuning parameters of a hearing aid (HA) algorithm whenever a HA client is not satisfied with their HA performance. AIDA interprets searching for the "most interesting alternative" as an issue of optimal (acoustic) context-aware Bayesian trial design. In computational terms, AIDA is realized as an active inference-based agent with an Expected Free Energy criterion for trial design. This type of architecture is inspired by neuro-economic models on

efficient (Bayesian) trial design in brains and implies that AIDA comprises generative probabilistic models for acoustic signals and user responses. We propose a novel generative model for acoustic signals as a sum of time-varying autoregressive filters and a user response model based on a Gaussian Process Classifier. The full AIDA agent has been implemented in a factor graph for the generative model, and all tasks (parameter learning, acoustic context classification, trial design, etc.) are realized by a variational message passing on the factor graph. All verification and validation experiments and demonstrations are freely accessible at our GitHub repository.

5.1 Introduction

Hearing aids (HA) are often equipped with specialized noise reduction algorithms. These algorithms are developed by teams of engineers who aim to create a single optimal algorithm that suits any user in any situation. Taking a one-size-fits-all approach to HA algorithm design leads to two problems prevalent throughout today's hearing aid industry. First, modeling all possible acoustic environments is simply infeasible. The daily lives of HA users are varied, and the different environments they traverse even more so. Given differing acoustic environments, a single static HA algorithm cannot possibly account for all eventualities - even without taking into account the particular constraints imposed by the HA itself, such as limited computational power and allowed processing delays [112]. Secondly, hearing loss is highly personal and can differ significantly between users. Each HA user consequently requires their own, individually tuned HA algorithm that compensates for their unique hearing loss profile [113, 114, 115] and satisfies their personal preferences for parameter settings [116]. Considering that HAs nowadays often consist of multiple interconnected digital signal processing units with many integrated parameters, the task of personalizing the algorithm requires exploring a high-dimensional search space of parameters, which often do not yield a clear physical interpretation. The current most widespread approach to personalization requires the HA user to physically travel to an audiologist who manually tunes a subset of all HA parameters. This is a burdensome activity that is not guaranteed to yield an improved listening experience for the HA user.

From these two problems, it becomes clear that we need to move towards a new approach for hearing aid algorithm design that empowers the user. Ideally, users should be in control of their own HA algorithms and should be able to change and update them at will instead of having to rely on teams of engineers that operate with long design cycles, separated from the users' living experiences.

The question then becomes, how do we move the HA algorithm design away from engineers and into the hands of the user? While a naive implementation that allows for tuning HA parameters with sliders on, for example, a smartphone

is trivial to develop, even a small number of adjustable parameters gives rise to a large, high-dimensional search space that the HA user needs to learn to navigate. This puts a large burden on the user, essentially asking them to be their own trained audiologist. Clearly, this is not a trivial task, and this approach is only feasible for a small set of parameters, which carry a clear physical interpretation. Instead, we wish to support the user with an agent that intelligently proposes new parameter trials. In this setting, the user is only tasked to cast (positive or negative) appraisals of the current HA settings. Based on these appraisals, the agent will autonomously traverse the search space with the goal of proposing satisfying parameter values for that user under the current environmental conditions in as few trials as possible.

Designing an intelligent agent that learns to efficiently navigate a parameter space is not trivial. In the solution approach in this paper, we rely on a probabilistic modeling approach inspired by the free energy principle (FEP) [117]. The FEP is a framework originally designed to explain the kinds of computations that biological, intelligent agents (such as the human brain) might be performing. Recent years have seen the FEP applied to the design of synthetic agents as well [118, 119, 120, 121]. A hallmark feature of FEP-based agents is that they exhibit a dynamic trade-off between exploration and exploitation [122, 123, 124], which is a highly desirable property when learning to navigate an HA parameter space. Concretely, the FEP proposes that intelligent agents should be modeled as probabilistic models. These types of models do not only yield point estimates of variables but also capture uncertainty through modeling full posterior probability distributions. Furthermore, user appraisals and actions can be naturally incorporated by simply extending the probabilistic model. Taking a model-based approach also allows for fewer parameters than alternative data-driven solutions, as we can incorporate field-specific knowledge, making it more suitable for computationally constrained hearing aid devices. The novelty of our approach is rooted in the fact that the entire proposed system is framed as a probabilistic generative model in which we can perform (active) inference through (expected) free energy minimization.

In this paper we present AIDA[1], an active inference-based design agent for the situated development of context-dependent audio processing algorithms, which provides the user with her own controllable audio processing algorithm. This approach embodies an FEP-based agent that operates in conjunction with an acoustic model and actively learns optimal context-dependent tuning parameter settings. After formally specifying the problem and solution approach in Section 5.2 we make the following contributions:

1. We develop a modular probabilistic model that embodies situated, (acoustic) scene-dependent, and personalized design of its corresponding hearing aid algorithm in Section 5.3.1.

[1] Aida is a girl's name of Arabic origin, meaning "happy". We use this name as an abbreviation for an "Active Inference-based Design Agent" that aims to make an end-user "happy".

2. We develop an expected free energy-based agent (AIDA) in Section 5.3.2, whose proposals for tuning parameter settings are well-balanced in terms of seeking more information about the user's preferences (explorative agent behavior) versus seeking to optimize the user's satisfaction levels by taking advantage of previously learned preferences (exploitative agent behavior).

3. Inference in the acoustic model and AIDA is elaborated upon in Section 5.4 and their operations are individually verified through representative experiments in Section 5.5. Furthermore, all elements are jointly validated through a demonstrator application in Section 5.5.4.

We have intentionally postponed a more thorough review of related work to Section 5.7 as we deem it more relevant after the introduction of our solution approach. Finally, Section 5.6 discusses the novelty and limitations of our approach and Section 5.8 concludes this paper.

5.2 Problem statement and proposed solution approach

5.2.1 Automated hearing aid tuning by optimization

In this paper, we consider the problem of choosing values for the tuning parameters $\boldsymbol{u}$ of a hearing aid algorithm that processes an acoustic input signal $\boldsymbol{x}$ to output signal $\boldsymbol{y}$. In Figure 5.1, we sketch an automated optimization-based approach to this problem. Assume that we have access to a generic "signal quality" model which rates the quality of a HA output signal $\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{u})$, as a function of the HA input $\boldsymbol{x}$ and parameters $\boldsymbol{u}$, by a rating $r(\boldsymbol{x}, \boldsymbol{u}) \triangleq r(\boldsymbol{y})$. If we run this system on a representative set of input signals $\boldsymbol{x} \in \mathcal{X}$, then the tuning problem reduces to the optimization task

$$\boldsymbol{u}^* = \underset{\boldsymbol{u}}{\operatorname{argmax}} \left(\sum_{\boldsymbol{x} \in \mathcal{X}} r(\boldsymbol{x}, \boldsymbol{u}) \right) . \tag{5.1}$$

Unfortunately, in commercial practice, this optimization approach does not always result in satisfactory HA performance, because of two reasons. First, the signal quality models in the literature have been trained on large databases of preference ratings from many users and therefore only model the average HA client rather than any specific client [125, 126, 127, 128, 129, 130]. Secondly, the optimization approach averages over a large set of different input signals, so it will not deal with acoustic context-dependent client preferences. By acoustic context, we consider signal properties that depend on environmental conditions such as being inside, outside, in a car or at the mall. Generally, client preferences for HA tuning parameters are both highly *personal* and *context-dependent*. Therefore, there is a need to develop a *personalized*, *context-sensitive* controller for tuning HA parameters $\boldsymbol{u}$.

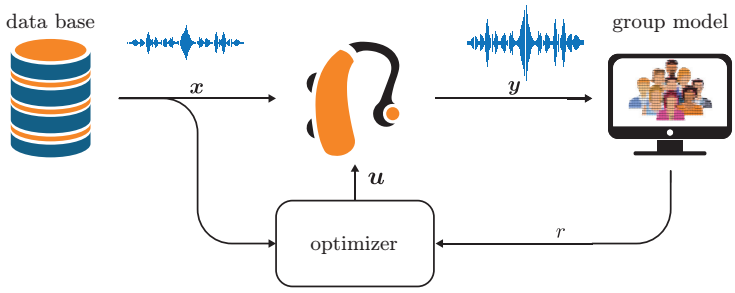


Figure 5.1: A schematic overview of the conventional approach to hearing aid algorithm tuning. Here the parameters of the hearing aid $\boldsymbol{u}$ are optimized with respect to some generic user rating model $r(\boldsymbol{y})$ for a large data base $\mathcal{X}$ of input data $\boldsymbol{x}$.

5.2.2 Situated hearing aid tuning with the user in-the-loop

In this paper, we will develop a personalized, context-aware design agent, based on the architecture shown in Figure 5.2. In contrast to Figure 5.1, the outside world (rather than a database) produces an input signal $\boldsymbol{x}$ under situated conditions that is processed by a hearing aid algorithm to produce an output signal $\boldsymbol{y}$. A particular human hearing aid client listens to the signal $\boldsymbol{y}$ and is invited to cast at any time binary appraisals $r \in \{0, 1\}$ about the current performance of the hearing aid algorithm, where 1 and 0 correspond to the user being satisfied and unsatisfied, respectively. Context-aware trials for HA tuning parameters are provided by AIDA. Rather than an offline design procedure, the whole system designs continually under *situated* conditions. The HA device itself houses a custom hearing aid algorithm, based on state inference in a generative acoustic model. The acoustic model contains two sub-models: 1) a source dynamics model and 2) a context dynamics model.

Inference in the acoustic model is based on the observed signal $\boldsymbol{x}$ and yields the output $\boldsymbol{y}$ and context c. Based on this context signal c and previous user appraisals r, AIDA will actively propose new parameters trials $\boldsymbol{u}$ with the goal of making the user happy. Technically, the objective is that AIDA expects to receive fewer negative appraisals in the future, relative to not making parameter adaptations, see Section 5.3.2 for details.

The design of AIDA is non-trivial. For instance, since there is a priori no personalized model of HA ratings for any particular user, AIDA will have to build such a model on-the-fly from the context c and user appraisals r. Since the system operates under situated conditions, we want to impose as little burden on the end user as possible. As a result, most users will only once in a while cast an appraisal and this complicates the learning of a personalized HA rating model.

To make this desire for very lightweight interactions concrete, we now sketch how we envision a typical interaction between AIDA and a HA client. Assume

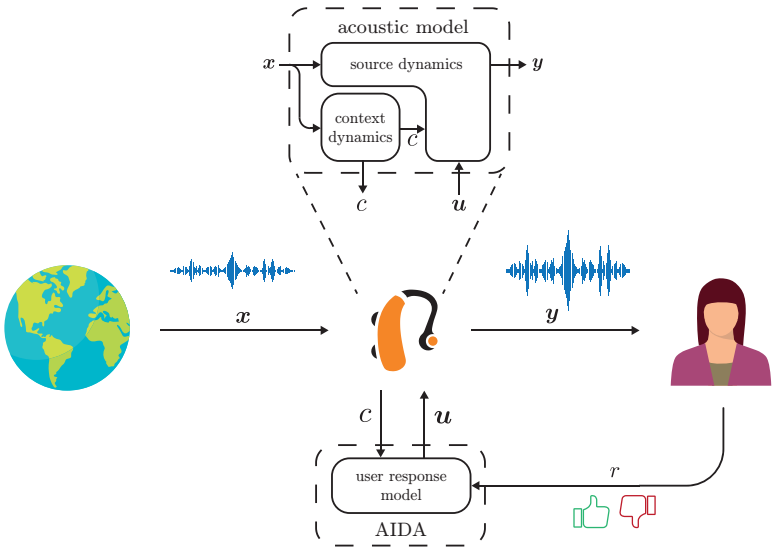


Figure 5.2: A schematic overview of the proposed situated HA design loop containing AIDA. An incoming signal $\boldsymbol{x}$ enters the hearing aid and is used to infer the context of the user c. Based on this context and previous user appraisals, AIDA proposes a new set of parameters $\boldsymbol{u}$ for the hearing aid algorithm. Based on the input signal, the proposed parameters, and the current context, the output $\boldsymbol{y}$ of the hearing aid is determined, which is used together with the context in the hearing aid algorithm. The parameters $\boldsymbol{u}$ are actively optimized by AIDA, based on the inferred context c from the input signal $\boldsymbol{x}$ and appraisals r from the user in the loop. All individual subsystems represent parts of a probabilistic generative model as described in Section 5.3, where the corresponding algorithms follows from performing probabilistic inference in these models as described in Section 5.4.

that the HA client is in a conversation with a friend at a restaurant. The signal of interest, in this case, is the friend's speech signal, while the interfering signal is an environmental babble noise signal. The HA algorithm tries to separate the input signal $\boldsymbol{x}$ into its constituent speech and noise source components, then applies gains $\boldsymbol{u}$ to each source component and sums these weighted source signals to produce output $\boldsymbol{y}$. If the HA client is happy with the performance of her HA, she will not cast any appraisals. After all, she is in the middle of a conversation and has no imperative to change the HA behavior. However, if she cannot understand her conversation partner, the client may covertly tap her watch or make another gesture to indicate that she is not happy with her current HA settings. In response, AIDA, which may be implemented as a smartwatch application, will reply instantaneously by sending a tuning parameter update $\boldsymbol{u}$ to the hearing aid algorithm in

an effort to fix the client's current hearing problem. Since the client's preferences are context-dependent, AIDA needs to incorporate information about the acoustic context from HA input $\boldsymbol{x}$. As an example, the HA user might leave the restaurant for a walk outside. Walking outside presents a different type of background noise and consequently requires different parameter settings.

Crucially, we would like HA clients to be able to tune their hearing aids without interruption of any ongoing activities. Therefore, we will not demand that the client has to focus visual attention on interacting with a smartphone app. At most, we want the client to apply a tap or make a simple gesture that does not draw any attention away from the ongoing conversation. A second criterion is that we do not want the conversation partner to notice that the client interacts with the agent. The client may actually be in a situation (e.g., a business meeting) where it is not appropriate to demonstrate that her priorities have shifted to tuning her hearing aids. In other words, the interactions must be very lightweight and covert. A third criterion is that we want the agent to learn from as few appraisals as possible. Note that, if the HA has 10 tuning parameters and 5 interesting values (very low, low, middle, high, very high) per parameter, then there are 5^{10} (about 10 million) parameter settings. We do not want the client to get engaged in an endless loop of disapproving new HA proposals, as this will lead to frustration and distraction from the ongoing conversation. Clearly, this means that each update of the HA parameters cannot be selected randomly: we want the agent to propose the most interesting values for the tuning parameters based on all observed past information and certain goal criteria for future HA behavior. In Section 5.4.2, we will quantify what *most interesting* means in this context.

In short, the goal of this paper is to design an intelligent agent that supports the user-driven situated design of a personalized audio processing algorithm through a very lightweight interaction protocol.

In order to accomplish this task, we will draw inspiration from the way how human brains to design algorithms (e.g., for speech and object recognition, riding a bike, etc.) solely through environmental interactions. Specifically, we base the design of AIDA on the Active Inference (AIF) framework. Originating from the field of computational neuroscience, AIF proposes to view the brain as a prediction engine that models sensory inputs. Formally, AIF accomplishes this by specifying a probabilistic generative model of incoming data. Performing approximate Bayesian inference in this model by minimizing free energy then constitutes a unified procedure for both data processing and learning. To select tuning parameter trials, an AIF agent predicts the *expected* free energy in the near future, given a particular choice of parameter settings. AIF provides a single, unified method for designing all components of AIDA. The design of a HA system that is controlled by an AIF-based design agent involves solving the following tasks:

1. Classification of acoustic context

2. Selecting acoustic context-dependent trials for the HA tuning parameters.

3. Execution of the HA signal processing algorithm (that is controlled by the trial parameters).

Task 1 (context classification) involves determining the most probable current acoustic environment. Based on a dynamic context model (described in Section 5.3.1), we infer the most probable acoustic environment as described in Section 5.4.1.

Task 2 (trial design) encompasses proposing alternative settings for the HA tuning parameters. Sections 5.3.2 and 5.4.2 describe the user response model and execution of AIDA's trial selection procedure based on expected free energy minimization, respectively.

Finally, task 3 (hearing aid algorithm execution) concerns performing variational free energy minimization with respect to the state variables in a generative probabilistic model for the acoustic signal. In Section 5.3.1 we describe the generative acoustic model underlying the HA algorithm and Section 5.4.3 describes the inferred HA algorithm itself.

Crucially, in the AIF framework, all three tasks can be accomplished by variational free energy minimization in a generative probabilistic model for observations. Since we can automate variational free energy minimization by a probabilistic programming language, the only remaining task for the human designer is to specify the generative models. The following section describes the model specification.

5.3 Model specification

In this section, we present the generative model of the AIDA-controlled HA system, as illustrated in Figure 5.2. In Section 5.3.1, we describe a generative model for the HA input and output signals $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively. In this model, the hearing aid algorithm follows through performing probabilistic inference, as will be discussed in Section 5.4. Part of the hearing aid algorithm is a mechanism for inferring the current acoustic context. In Section 5.3.2 we introduce a model for agent AIDA that is used to infer new parameter trials.

Throughout this section, we will make use of factor graphs for the visualization of probabilistic models. In this paper we focus on Forney-style factor graphs (FFG), as introduced in [93] with notational conventions adopted from [94]. FFGs represent factorized functions by undirected graphs, whose nodes represent the individual factors of the global function. The nodes are connected by edges representing the mutual arguments of the factors.

5.3.1 Acoustic model

Our acoustic model of the observed signal and hearing aid output consists of a model of the source dynamics of the underlying signals and a model for the context dynamics.

Model of source dynamics

We assume that the observed acoustic signal $\boldsymbol{x}$ consists of a speech signal (or more generally, a target signal that the HA client wants to focus on) and an additive noise signal (that the HA client is not interested in), as

$$x_t = s_t + n_t \tag{5.2}$$

where $x_t \in \mathbb{R}$ represents the observed signal at time t, i.e. the input to the HA. The speech and noise signals are represented by $s_t \in \mathbb{R}$ and $n_t \in \mathbb{R}$, respectively. At this point, the source dynamics of s_n and n_t need to be further specified. Here we choose to model the speech signal by a time-varying autoregressive model and the noise signal by a context-dependent autoregressive model. The remainder of this subsection will elaborate on both these source models and will further specify how the hearing aid output is generated. An FFG visualization of the described acoustic model is depicted in Figure 5.3.

Historically, autoregressive (AR) models have been widely used to represent speech signals [105, 83]. As the dynamics of the vocal tract exhibit non-stationary behavior, speech is usually segmented into individual frames that are assumed to be quasi-stationary. Unfortunately, the signal is often segmented without any prior information about the phonetic structure of the speech signal. Therefore the quasi-stationarity assumption is likely to be violated and time-varying dynamics are more likely to occur in the segmented frames [131]. To address this issue, we can use a time-varying prior for the coefficients of the AR model, leading to a time-varying AR (TVAR) model [41]

$$\boldsymbol{\theta}_t \sim \mathcal{N}\left(\boldsymbol{\theta}_{t-1},\ \omega \mathrm{I}_K\right) \tag{5.3a}$$

$$\boldsymbol{s}_t \sim \mathcal{N}\left(A(\boldsymbol{\theta}_t)\boldsymbol{s}_{t-1},\ V\left(\gamma\right)\right) \tag{5.3b}$$

where $\boldsymbol{\theta}_t = [\theta_{1t}, \theta_{2t}, ..., \theta_{Kt}]^\intercal \in \mathbb{R}^K$, $\boldsymbol{s}_t = [s_t, s_{t-1}, ..., s_{t-K+1}]^\intercal \in \mathbb{R}^K$ are the coefficients and states of an K^{th} order TVAR model for speech signal $s_t = \boldsymbol{e}_1^\intercal \boldsymbol{s}_t$. We use $\mathcal{N}(\boldsymbol{m}, \boldsymbol{V})$ to denote a Gaussian distribution with mean $\boldsymbol{m}$ and covariance matrix $\boldsymbol{V}$. In this model, the AR coefficients $\boldsymbol{\theta}_t$ are represented by a Gaussian random walk with process noise covariance $\omega \mathrm{I}_M$, with I_K denoting the identity matrix of size $(K \times K)$, scaled by $\omega \in \mathbb{R}_{>0}$. $\gamma \in \mathbb{R}_{>0}$ represents the process noise precision matrix of the AR process. Here, we have adopted the state-space formulation of TVAR models as in [88], where $V(\gamma) = (1/\gamma)\boldsymbol{e}_1\boldsymbol{e}_1^\intercal$ creates a covariance matrix with

a single non-zero entry. We use $\boldsymbol{e}_i$ to denote an appropriately sized Cartesian standard unit vector that represents a column vector of zeros where only the i^{th} entry is 1. $A(\boldsymbol{\theta})$ denotes the companion matrix of size $(K \times K)$, defined as

$$A(\boldsymbol{\theta}) = \begin{bmatrix} \boldsymbol{\theta}^\intercal & \\ \mathrm{I}_{K-1} & \mathbf{0} \end{bmatrix}. \tag{5.4}$$

Multiplication of a state vector by this companion matrix, such as $A(\boldsymbol{\theta}_t)\boldsymbol{s}_{t-1}$, basically performs two operations: an inner product $\boldsymbol{\theta}_t^\intercal \boldsymbol{s}_{t-1}$ and a shift of $\boldsymbol{s}_{t-1}$ by one time step to the past.

The acoustic model also encompasses a model for background noise, such as the sounds at a bar or train station. Many of these background sounds can be well represented by colored noise [132], which in turn can be modeled by a low-order AR model [107, 106]

$$\boldsymbol{n}_t \sim \mathcal{N}\left(A(\boldsymbol{\varrho}_i)\boldsymbol{n}_{t-1},\ V\left(\tau_i\right)\right), \qquad \text{for } t = t^-, t^- + 1, \ldots, t^+ \tag{5.5}$$

where $\boldsymbol{\varrho}_i = [\varrho_{1k}, \varrho_{2k}, ..., \varrho_{Nk}]^\intercal \in \mathbb{R}^N$, $\boldsymbol{n}_t = [n_t, n_{t-1}, ..., n_{t-N+1}]^\intercal \in \mathbb{R}^N$ are the coefficients and states of an AR model of order $N \in \mathbb{N}^+$ for noise signal $n_t = \boldsymbol{e}_1^\intercal \boldsymbol{n}_t$. $\tau_i \in \mathbb{R}_{>0}$ denotes the process noise precision of the AR process. In contrast to the speech model, we assume the processes $\boldsymbol{\varrho}_i$ and τ_i to be stationary when the user is in a specific acoustic environment or context. To make clear that contextual states change much slower than raw acoustic data signals, we index the slower parameters at time index k, which is related to index t by

$$k = \left\lceil \frac{t}{W} \right\rceil. \tag{5.6}$$

Here, $\lceil \cdot \rceil$ denotes the ceiling function that returns the largest integer smaller or equal than its argument, while W is the window length. The above equation makes sure that k is intuitively aligned with segments of length W, i.e. $t \in [1, W]$ corresponds to $k = 1$. To denote the start and end indices of the time segment corresponding to context index i, we define $t^- = (i-1)W + 1$ and $t^+ = iW$ as an implicit function of i, respectively. The context can be assumed to be stationary within a longer period of time compared to the speech signal. However, abrupt changes in the dynamics of background noise may occasionally occur. For example, if the user moves from a train station to a bar, the parameters of the AR model that are attributed to the train station will now inadequately describe the background noise of the new environment. To deal with these changing acoustic environments, we introduce context-dependent priors for the background noise, using a Gaussian

and Gamma mixture model:

$$\boldsymbol{\varrho}_i \sim \prod_{l=1}^{L} \mathcal{N}\left(\boldsymbol{m}_l, \boldsymbol{V}_l\right)^{\boldsymbol{c}_i} \tag{5.7a}$$

$$\tau_i \sim \prod_{l=1}^{L} \Gamma\left(a_l, b_l\right)^{\boldsymbol{c}_i} \tag{5.7b}$$

The context at time index i, denoted by $\boldsymbol{c}_i$, comprises a 1-of-L binary vector with elements $c_{li} \in \{0, 1\}$, which are constrained by $\sum_i c_{li} = 1$. $\Gamma(a, b)$ represents a Gamma distribution with shape and rate parameters a and b, respectively. The hyperparameters $\boldsymbol{m}_l$, $\boldsymbol{V}_l$, a_l and b_l define the characteristics of the different background noise environments.

Now that an acoustic model of the environment has been formally specified, we will extend this model with the goal of obtaining a HA algorithm. The principal goal of a HA algorithm is to improve the audibility and intelligibility of acoustic signals. Audibility can be improved by amplifying the received input signal. Intelligibility can be improved by increasing the Signal-to-Noise Ratio (SNR) of the received signal. Assuming that we can infer the constituent source signals s_t and n_t from the received signal x_t, the desired HA output signal can be modeled by

$$y_t = u_{sk} s_t + u_{nk} n_t, \qquad \text{for } t = t^-, t^- + 1, \ldots, t^+ \tag{5.8}$$

where $\boldsymbol{u}_i = [u_{sk}, u_{nk}]^\intercal \in [0, 1]^2$ represents a vector of 2 tuning parameters or source-specific gains for the speech and background noise signal, respectively. In this expression, the output of the hearing aid is modeled by a weighted sum of the constituent source signals. The gains control the amplification of the extracted speech and noise signals individually and thus allow the user to perform source-specific filtering, also known as soundscaping [133]. Because of imperfections during inference of the source signals (see Section 5.4), the gains simultaneously reflect a trade-off between residual noise and speech distortion.

Finding good values for the gains $\boldsymbol{u}$ can be a difficult task because the preferred parameter settings may depend on the specific listener and on the acoustic context.

Next, we describe the acoustic context model that will allow AIDA to make context-dependent parameter proposals.

Model of context dynamics

As HA clients move through different acoustic background settings, such as being in a car, doing groceries, watching TV at home, etc.) the preferred parameter settings for HA algorithms tend to vary. The context signal allows distinguishing between these different acoustic environments.

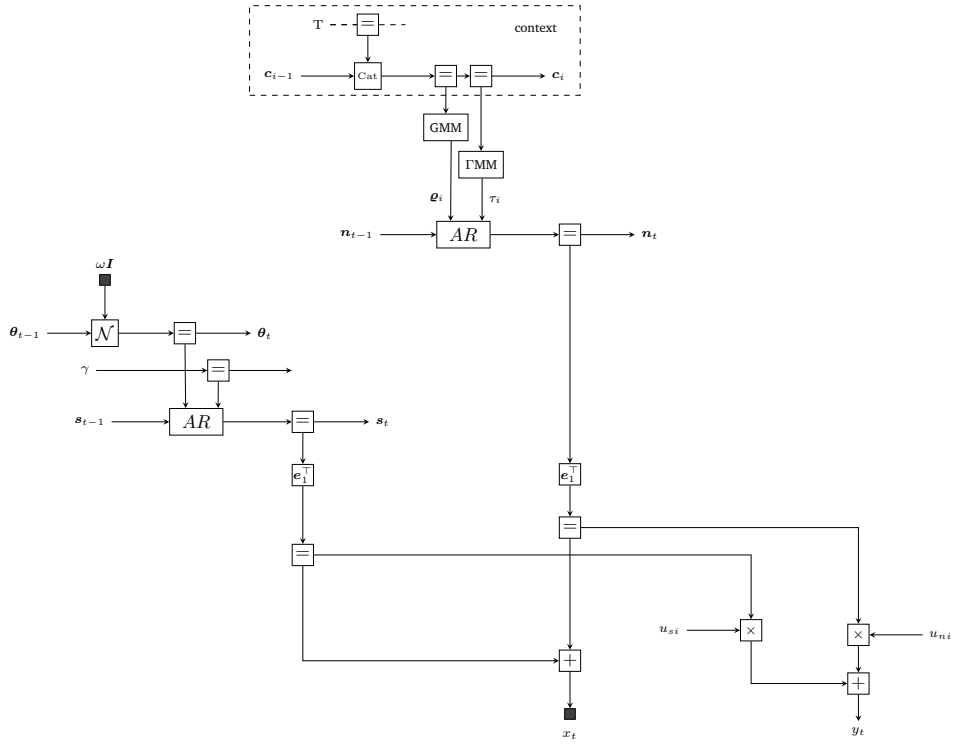


Figure 5.3: A Forney-style factor graph representation of the acoustic source signals model as specified by (5.3)-(5.11) at time index t. The observation x_t is specified as the sum of a latent speech signal s_t and a latent noise signal n_t. The speech signal is modeled by a time-varying autoregressive process, where its coefficients $\boldsymbol{\theta}_t$ are modeled by a Gaussian random walk. The noise signal is a context-dependent autoregressive process, modeled by Gaussian (GMM) and Gamma mixture models (ΓMM) for the parameters $\boldsymbol{\varrho}_i$ and τ_i, respectively. The selection variable of these mixture models represents the context $\boldsymbol{c}_i$. The model for the context dynamics is enclosed by the dashed box. The composite AR factor node represents the autoregressive transition dynamics specified by (5.3b). The output of the hearing aid y_t is modeled as the weighted sum of the extracted speech and noise signals.

The hidden context state variable $\boldsymbol{c}_i$ at time index k is a 1-of-L encoded binary vector with elements $c_{li} \in \{0, 1\}$, which are constrained by $\sum_l c_{li} = 1$. This context is responsible for the operations of the noise model in (5.7). Context transitions are supported by a dynamic model

$$\boldsymbol{c}_i \sim \mathrm{Cat}(\mathrm{T}\boldsymbol{c}_{i-1}), \tag{5.9}$$

where the elements of transition matrix T, are defined as $\mathrm{T}_{ij} = p(c_{ik} = 1 \mid c_{j,i-1} = 1)$, which are constrained by $\mathrm{T}_{ij} \in [0, 1]$ and $\sum_{j=1}^{L} \mathrm{T}_{ij} = 1$. We model the individual columns of T by a Dirichlet distribution as

$$\mathrm{T}_{1:L,j} \sim \mathrm{Dir}(\boldsymbol{\zeta}_j), \tag{5.10}$$

where $\boldsymbol{\zeta}_j$ denotes the vector of concentration parameters corresponding to the j^{th} column of T. The context state is initialized by a categorical distribution as

$$\boldsymbol{c}_0 \sim \mathrm{Cat}(\boldsymbol{\pi}) = \prod_{l=1}^{L} \pi_l^{c_{l0}} \quad \text{such that} \quad \sum_{l=1}^{L} \pi_l = 1, \tag{5.11}$$

where the vector $\boldsymbol{\pi} = [\pi_1, \pi_2, \ldots, \pi_L]^\intercal$ contains the event probabilities, whose elements can be chosen as $\pi_l = 1/L$ if the initial context is unknown. An FFG representation of the context dynamics model is shown in the dashed box in Figure 5.3.

5.3.2 AIDA's user response model

The goal of AIDA is to continually provide the most "interesting" settings for the HA tuning parameters $\boldsymbol{u}_i$, where interesting has been quantitatively interpreted by minimization of Expected Free Energy. But how does AIDA know what the client wants? In order to learn the client's preferences, she is invited to cast at any time her appraisal $r_i \in \{\varnothing, 0, 1\}$ of current HA performance. To keep the user interface very light, we will assume that appraisals are binary, encoded by $r_i = 0$ for disapproval and $r_i = 1$ indicating a positive experience. If a user does not cast an appraisal, we will just record a missing value, i.e., $r_i = \varnothing$. The subscript k for r_i indicates that we record appraisals at the same rate as the context dynamics.

If a client submits a negative appraisal $r_i = 0$, AIDA interprets this as an expression that the client is not happy with the current HA settings $\boldsymbol{u}_i$ in the current acoustic context $\boldsymbol{c}_i$ (and vice versa for positive appraisals). To *learn* client preferences from these appraisals, AIDA holds a context-dependent generative model to *predict* user appraisals and updates this model after observing actual appraisals. In this paper, we opt for a Gaussian Process Classifier (GPC) model as the generative model for binary user appraisals. A Gaussian Process (GP) is a very flexible probabilistic model and GPCs have successfully been applied to preference learning in a variety of tasks before [134, 135, 136]. For an in-depth discussion on GPs, we refer the reader to [137]. Specifically, the context-dependent user response model is defined as

$$\upsilon_i(\cdot) \sim \prod_{l=1}^{L} \mathrm{GP}(\mathcal{M}_l(\cdot), \mathcal{K}_l(\cdot,\cdot))^{\boldsymbol{c}_i} \tag{5.12a}$$

$$r_i \sim \mathrm{Ber}(\Phi(\upsilon_i(\boldsymbol{u}_i)))\,. \qquad \text{if } r_i \in \{0, 1\} \tag{5.12b}$$

In (5.12a), $v_i(\cdot)$ is a latent function drawn from a mixture of GPs with mean functions $\mathcal{M}_l(\cdot)$ and kernels $\mathcal{K}_l(\cdot,\cdot)$. Evaluating $v_i(\cdot)$ at the point $\boldsymbol{u}_i$ provides an estimate of user preferences. Without loss of generality, we can set $\mathcal{M}_l(\cdot) = 0$. Since $\boldsymbol{c}_i$ is one-hot encoded, raising to the power $\boldsymbol{c}_i$ serves to select the GP that corresponds to the active context. $\Phi(\cdot)$ denotes the Gaussian cumulative distribution function, defined as $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} \exp\left(-t^2/2\right) \mathrm{d}t$. This map in (5.12b) casts $v_i(\boldsymbol{u}_i)$ to a Bernoulli-distributed variable r_i.

5.4 Solving tasks by probabilistic inference

This section elaborates on solving the three tasks of Section 5.2.2: 1) context classification, 2) trial design and 3) hearing aid algorithm execution. All tasks can be solved through probabilistic inference in the generative model specified by (5.2)-(5.12b) in Section 5.3. In this section, the inference goals are formally specified based on the previously proposed generative model.

For the realization of the inference tasks, we will use variational message passing in a factor graph representation of the generative model. Message passing-based inference is highly efficient, modular, and scales well to large inference tasks [48, 72]. With message passing, inference tasks in the generative model reduce to automatable procedures revolving around local computations on the factor graphs.

5.4.1 Inference for context classification

The acoustic context $\boldsymbol{c}_i$ describes the dynamics of the background noise model through (5.5) and (5.7). For determining the current environment of the user, the goal is to infer the current context based on the preceding observations. Technically we are interested in determining the marginal distribution $p(\boldsymbol{c}_i \mid \boldsymbol{x}_{1:t^+})$, where the index range over t of $\boldsymbol{x}$ takes into account the relation between t and k as defined in (5.6). In our online setting, we wish to calculate this marginal distribution iteratively by solving

$$\underbrace{p(\boldsymbol{c}_i \mid \boldsymbol{x}_{1:t^+})}_{\text{posterior}} \propto \int \underbrace{p(z_{t^-:t^+}, \Psi_i, \boldsymbol{x}_{t^-:t^+} \mid z_{t^--1}, \boldsymbol{c}_i)}_{\text{observation model}} \underbrace{p(\boldsymbol{c}_i, \mathrm{T} \mid \boldsymbol{c}_{i-1})}_{\text{context dynamics}} \cdot \underbrace{p(\boldsymbol{c}_{i-1}, z_{t^--1} \mid \boldsymbol{x}_{1:t^--1})}_{\text{prior}} \, \mathrm{d}z_{t^--1:t^+} \, \mathrm{d}\Psi_i \, \mathrm{d}\boldsymbol{c}_{i-1} \, \mathrm{dT}. \tag{5.13}$$

The observation model is fully specified by the model specification in Section 5.3, similarly as the context dynamics. The prior distribution is a joint result of the iterative execution of both (5.13) and (5.18), where the latter refers to the HA algorithm

execution from Section 5.4.3. The calculation of this marginal distribution renders intractable and therefore exact inference of the context is not possible. This is a result of 1) the intractability resulting from the autoregressive model as described in the previous subsection and of 2) the intractability that is a result of performing message passing with mixture models. In (5.7) the model structure contains a Normal and Gamma mixture model for the AR-coefficients and process noise precision parameter, respectively. Exact inference with these mixture models quickly leads to intractable inference through message passing, especially when multiple background noise models are involved. Therefore, we need to resort to a variational approximation where the output messages of these mixture models are constrained to be within the exponential family.

Although variational inference with the mixture models is feasible [47, 25, 109], it is prone to converge to local minima of the Bethe free energy (BFE) for more complicated models. The variational messages originating from the mixture models are constrained to either Normal or Gamma distributions, possibly losing important multi-modal information, and as a result, they can lead to a suboptimal inference of the context variable. Because the context is vital for the above-underdetermined source separation stage, we wish to limit the amount of (variational) approximations during context inference. At the cost of increased computational complexity, we will remove the variational approximation around the mixture models and instead expand the mixture components into distinct models. As a result, each distinct model now contains one of the mixture components for a given context and now results in exact messages originating from the priors of $\boldsymbol{\xi}_i$ and τ_i. Therefore we only need to resort to a variational approximation for the autoregressive node. By expanding the mixture models into distinct models to reduce the number of variational approximations, the calculation of the posterior distribution of the context $p(\boldsymbol{c}_i \mid \boldsymbol{x}_{1:t^+})$ reduces to an approximate Bayesian model comparison problem, similarly as described in [133]. Appendix C.3.1 gives a more in-depth description on how we use Bayesian model comparison for solving the inference task in (5.13).

5.4.2 Inference for trial design of HA tuning parameters

The goal of proposing alternative HA tuning parameter settings (task 3) is to receive positive user responses in the future. Free energy minimization over desired future user responses can be achieved through a procedure called Expected Free Energy (EFE) minimization [122, 138].

EFE as a trial selection criterion induces a natural trade-off between explorative (information-seeking) and exploitative (reward-seeking) behavior. In the context of situated HA personalization, this is desirable because soliciting user feedback can be burdensome and invasive, as described in Section 5.2.2. From the agent's point of view, this means that striking a balance between gathering information about user preferences and satisfying learned preferences is vital. The EFE provides a way to

tackle this trade-off, inspired by neuro-scientific evidence that brains operate under a similar protocol [122, 139]. The EFE is defined as [122]

$$G_{\boldsymbol{u}}[q] = \mathbb{E}_{q(r,\upsilon|\boldsymbol{u})}\left[\ln \frac{q(\upsilon \mid \boldsymbol{u})}{p(r,\upsilon \mid \boldsymbol{u})}\right], \tag{5.14}$$

where the subscript indicates that the EFE is a function of a trial $\boldsymbol{u}$. The EFE can be decomposed into [122]

$$G_{\boldsymbol{u}}[q] \approx -\underbrace{\mathbb{E}_{q(r|\boldsymbol{u})}\Big[\ln p(r)\Big]}_{\text{Utility drive}} - \underbrace{\mathbb{E}_{q(r,\upsilon|\boldsymbol{u})}\left[\ln \frac{q(\upsilon \mid \boldsymbol{u}, r)}{q(\upsilon \mid \boldsymbol{u})}\right]}_{\text{Information gain}}, \tag{5.15}$$

which contains an information gain term and a utility-driven term. Minimization of the EFE reduces to maximization of both these terms. Maximization of the utility drive pushes the agent towards matching predicted user responses $q(r \mid \boldsymbol{u})$ with a goal prior over *desired* user responses $p(r)$. This goal prior allows the encoding of beliefs about future observations that we wish to observe. Setting the goal prior to match positive user responses then drives the agent towards parameter settings that it believes make the user happy in the future. The information gain term in (5.15) drives agents that optimize the EFE to seek out responses that are maximally informative about latent states υ.

To select the next set of gains $\boldsymbol{u}$ to propose to the user, we need to find

$$\boldsymbol{u}^* = \underset{\boldsymbol{u}}{\operatorname{argmin}}\left(\min_q G_{\boldsymbol{u}}[q]\right). \tag{5.16}$$

Intuitively, one can think of (5.16) as a two-step procedure with an inner and an outer loop. The inner loop finds the approximate posterior q using (approximate) Bayesian inference, conditioned on a particular action parameter $\boldsymbol{u}$. The outer loop evaluates the resulting EFE as a function of $\boldsymbol{u}$ and proposes a new set of gains to bring the EFE down. For our experiments, we consider a candidate grid of possible gains. For each candidate, we compute the resulting EFE and then select the lowest scoring proposal as the next set of gains to be presented to the user.

The probabilistic model used for AIDA is a mixture of GPC. For simplicity, we will restrict inference to the GP corresponding to the MAP estimate of $\boldsymbol{c}_i$. Between trials, the corresponding GP needs to be updated to adapt to the new data gathered from the user. Specifically, we are interested in finding the posterior over the latent user preference function

$$p(\upsilon^* \mid \boldsymbol{u}_{1:k}, r_{1:k-1}) = \int p(\upsilon^* \mid \boldsymbol{u}_{1:k-1}, \boldsymbol{u}_i, \upsilon) p(\upsilon \mid \boldsymbol{u}_{1:k-1}, r_{1:k-1}) \mathrm{d}\upsilon\,. \tag{5.17}$$

where we assume AIDA has access to a dataset consisting of previous queries $\boldsymbol{u}_{1:k-1}$ and appraisals $r_{1:k-1}$ and we are querying the model at $\boldsymbol{u}_i$. While this inference

task in the GPC is intractable, there exist a number of techniques for approximate inference, such as variational Bayesian methods, Expectation Propagation, and the Laplace approximation [137]. Appendix C.3.2 describes the exact details of the inference realization of the inference tasks of AIDA.

5.4.3 Inference for executing the hearing aid algorithm

The main goal of the proposed hearing aid algorithm is to improve audibility and intelligibility by re-weighing inferred source signals in the HA output signal. In our model of the observed signal in (5.2)-(5.7) we are interested in iteratively inferring the marginal distribution over the latent speech and noise signals $p(\boldsymbol{s}_t, \boldsymbol{n}_t \mid \boldsymbol{x}_{1:t})$. This inference task is in literature sometimes referred to as informed source separation [140]. Inferring the latent speech and noise signals tries to optimally disentangle these signals from the observed signal based on the sub-models of the speech and noise source.This requires us to compute the posterior distributions associated with the speech and noise signals. To do so, we perform probabilistic inference by means of message passing in the acoustic model of (5.2)-(5.7). The posterior distributions can be calculated in an online manner using sequential Bayesian updating by solving the Chapman-Kolmogorov equation [95]

$$\underbrace{p(z_t, \Psi_i \mid \boldsymbol{x}_{1:t})}_{\text{posterior}} \propto \underbrace{p(x_t \mid z_t)}_{\text{observation}} \int \underbrace{p(z_t \mid z_{t-1}, \Psi_i)}_{\text{state dynamics}} \underbrace{p(z_{t-1}, \Psi_i \mid \boldsymbol{x}_{1:t-1})}_{\text{prior}} \mathrm{d}z_{t-1}, \quad \text{for } t = t^-, t^- + 1, \ldots, t^+ \tag{5.18}$$

where z_t and Ψ_i denote the sets of dynamic states and static parameters $z_t = \{\boldsymbol{\theta}_t, \boldsymbol{s}_t, \boldsymbol{n}_t\}$ and $\Psi_i = \{\gamma, \tau_i, \boldsymbol{\zeta}_i\}$, respectively. Here, the states and parameters correspond to the latent AR and TVAR models of (5.3) and (5.5). Furthermore, we assume that the context does not change, i.e. k is fixed. When the context does change (5.18) will need to be extended by integrating over the varying parameters. Unfortunately, the solution of (5.18) is not analytically tractable. This happens because of 1) the integration over large state spaces, 2) the non-conjugate prior-posterior pairing, and 3) the absence of a closed-form solution for the evidence factor [111]. To circumvent this issue, we resort to a hybrid message passing algorithm that combines structured variational message passing (SVMP) and loopy belief propagation for the minimization of Bethe free energy [43].

Owing to the modularity of the factor graphs, the message passing update rules can be tabulated and only need to be derived once for each of the included factor nodes. The derivations of the sum-product update rules for elementary factor nodes can be found in [48] and the derived structured variational rules for the composite AR node can be found in [111]. The variational updates in the mixture models can be found in [25, 109]. The required approximate marginal distribution of some variable z can be computed by multiplying the incoming and outgoing variational

messages on the edges corresponding to the variables of our interest as $q(z) \propto \vec{\nu}(z) \cdot \overleftarrow{\nu}(z)$.

Based on the inferred posterior distributions of s_t and n_t, these signals can be used for inferring the hearing aid output through (5.8) to produce a personalized output that compromises between residual noise and speech distortion.

5.5 Experimental verification & validation

In this section, we first verify our approach for the three design tasks of Section 5.2.2. Specifically, in Section 5.5.1 we evaluate the context inference approach by reporting the classification performance of correctly classifying the context corresponding to a signal segment. In Section 5.5.2 we evaluate the performance of our intelligent agent that actively proposes hearing aid settings and learns user preferences. The execution of the hearing aid algorithm is verified in Section 5.5.3 by evaluating the source separation performance. To conclude this section, we present a demonstrator for the entire system in Section 5.5.4.

All algorithms have been implemented in the scientific programming language `Julia` [101]. Probabilistic inference in our model is automated using the open source `Julia` package `ReactiveMP`[2]. All of the experiments presented in this section can be found at our AIDA GitHub repository[3].

5.5.1 Context classification verification

To verify that the context is appropriately inferred through Bayesian model selection, we generated synthetic data from the following generative model:

$$\boldsymbol{c}_i \sim \mathrm{Cat}(\mathrm{T}\boldsymbol{c}_{i-1}) \tag{5.19a}$$

with priors

$$\boldsymbol{c}_0 \sim \mathrm{Cat}(\boldsymbol{\pi}) \tag{5.20a}$$

$$\mathrm{T}_{1:L,j} \sim \mathrm{Dir}(\boldsymbol{\zeta}_j), \tag{5.20b}$$

where $\boldsymbol{c}_o$ is chosen to have length $L = 4$. The event probabilities $\boldsymbol{\pi}$ and concentration parameters $\boldsymbol{\zeta}_j$ are defined as $\boldsymbol{\pi} = [0.25, 0.25, 0.25, 0.25]^\intercal$ and $\boldsymbol{\zeta}_j = [1.0, 1.0, 1.0, 1.0]^\intercal$, respectively. We generated a sequence of 1000 frames, each containing 100 samples, such that we have 100 x 1000 data points. Each frame is associated with one of the 4 different contexts. Each context corresponds to an AR model with the parameters presented in Table 5.1.

[2] `ReactiveMP` [73] is available at `https://github.com/biaslab/ReactiveMP.jl`.

[3] The AIDA GitHub repository with all experiments is available at `https://github.com/biaslab/AIDA`.

Table 5.1: The parameters of autoregressive processes that are used for generating a time series with simulated context dynamics.

AR order	ϱ				τ^{-1}
1	-0.308				1.0
2	0.722	-0.673			2.0
3	-0.081	0.079	-0.362		0.5
4	-1.433	-0.174	0.757	0.466	1.0

For verification of the context classification procedure, we wish to identify which model best approximates the observed data. To do that, 4 models with the same specifications as were used to generate the dataset were employed. We used informative priors for the coefficients and precision of AR models. Additionally, we extended our set of models with an AR(5) model with weakly informative priors and a Gaussian i.i.d. model that can be viewed as an AR model of zeroth order, i.e. AR(0). The individual frames containing 100 samples each were processed individually and we computed the Bethe free energy for each of the different models. The Bethe free energy is introduced in Appendix C.1. By approximating the true model evidence using the Bethe free energy as described in Appendix C.3.1, we performed approximate Bayesian model selection by selecting the model with the lowest Bethe free energy. This model then corresponds to the most likely context that we are in. We highlight the obtained inference result in Figure 5.4.

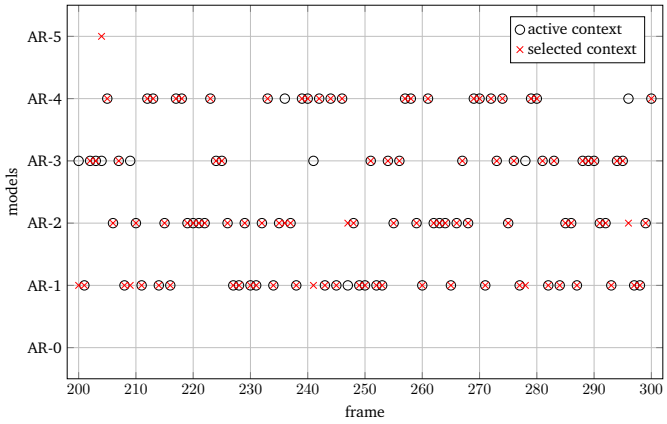


Figure 5.4: True and inferred evolution of contexts from frames 200 to 300. Each frame consists of 100 data points. Circles denote the active contexts that were used to generate the frame. Crosses denote the model that achieves the lowest Bethe free energy for a specific frame.

We evaluate the performance of the context classification procedure using approximate Bayesian model selection by computing the categorical accuracy metric defined as

$$acc = \frac{tp + tn}{N} \tag{5.21}$$

where tp, tn are the number of true positive and true negative values, respectively. N corresponds to the number of total observations, which in this experiment is set to $N = 1000$. In this context classification experiment, we have achieved a categorical accuracy of $acc = 0.94$.

5.5.2 Trial design verification

Evaluating the performance of the intelligent agent is not trivial. Because the agent adaptively trades off exploration and exploitation, accuracy is not an adequate metric. There are reasons for the agent to veer *away* from what it believes is the optimum to obtain more information. As a verification experiment we can investigate how the agent interacts with a simulated user. Our simulated user samples binary appraisals r_i based on the HA parameters $\boldsymbol{u}_i$ as

$$r_i \sim \mathrm{Ber}\left(\frac{2}{1 + \exp\left((\boldsymbol{u}_i - \boldsymbol{u}^*)^T \Lambda_{\text{user}} (\boldsymbol{u}_i - \boldsymbol{u}^*)\right)}\right), \tag{5.22}$$

where $\boldsymbol{u}^*$ denotes the optimal parameter setting, $\boldsymbol{u}_i$ is the set of parameters proposed by AIDA at time k, Λ_{user} is a diagonal weighting matrix that controls how quickly the probability of positive appraisals decays with the squared distance to $\boldsymbol{u}^*$. The constant 2 ensures that when $\boldsymbol{u}_i = \boldsymbol{u}^*$, the probability of positive appraisals is 1 instead of 0.5. For our experiments, we set $\boldsymbol{u}^* = [0.8, 0.2]^\intercal$ and the diagonal elements of Λ_{user} to 0.004. This results in the user preference function $p(r_i = 1 \mid \boldsymbol{u}_i)$ as shown in Figure 5.5. The kernel used for AIDA is a squared exponential kernel given by

$$\mathcal{K}(\boldsymbol{u}, \boldsymbol{u}') = \sigma^2 \exp\left\{-\frac{\|\boldsymbol{u} - \boldsymbol{u}'\|_2^2}{2l^2}\right\}, \tag{5.23}$$

where l and σ are the hyperparameters of this kernel. Intuitively, σ is a static noise parameter and l encodes the smoothness of the kernel function. Both hyperparameters were initialized to $\sigma = l = 0.5$, which is uninformative on the scale of the experiment. We let the agent search for 80 trials and update hyperparameters every 5^{th} trial using conjugate gradient descent as implemented in `Optim.jl` [141]. We constrain both hyperparameters to the domain $[0.1, 1]$ to ensure the stability of the optimization. As we will see, for large parts of each experiment, AIDA only receives negative appraisals. The generative model of AIDA is fundamentally a classifier, and unconstrained optimization can lead to degenerate results when the data set only

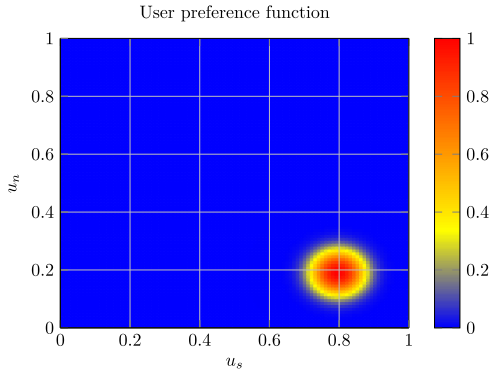


Figure 5.5: Simulated user preference function $p(r_i = 1 \mid \boldsymbol{u}_i)$. The coloring corresponds to the probability of the user giving a positive appraisal for the search space of gains $\boldsymbol{u}_i = [u_{sk}, u_{nk}]^\intercal$.

contains examples of a single class. For all experiments, the first proposal of AIDA was a randomly sampled parameter from the admissible set of parameters because the AIDA has no prior knowledge about the user preference function. This random initial proposal led to distinct behavior for all simulated agents.

We provide two verification experiments for AIDA. First, we will thoroughly examine a single run to investigate how AIDA switches between exploratory and exploitative behavior. Secondly, we examine the aggregate performance of an ensemble of agents to test the average performance. To assess the performance for a single run, we can examine the evolution of the distinct terms in the EFE decomposition of (5.15) over time. We expect that when AIDA is primarily exploring, the utility drive is relatively low while the information gain is relatively high. When AIDA is primarily engaged in the exploitation, we expect the opposite pattern. We show these terms separately in Figure 5.6.

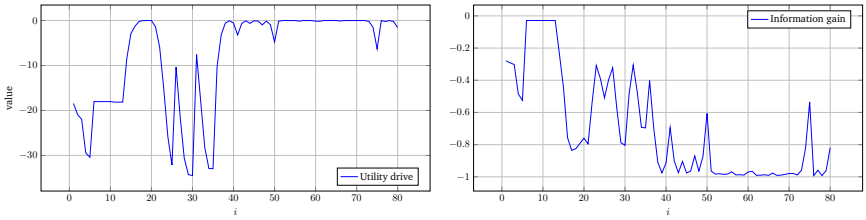


Figure 5.6: Evolution of the utility drive and negative information gain after throughout a single experiment.

Figure 5.6 shows that there are distinct phases to the experiment. At the begin-

ning ($i < 5$) AIDA sees a sharp decrease in utility drive and information gain terms. This indicates saturation of the search space such that no points present good options. This happens early due to uninformative hyperparameter settings in the GPC. After trial 5, these hyperparameters are optimized and the agent no longer thinks it has saturated the search space, which can be explained by the jumps in Figure 5.6 from trial 5 to 6. From trial 6 throughout 15 we observe a relatively high information gain and relatively low utility drive, meaning that the agent is still exploring the search space for parameter settings which yield a positive user appraisal. The agent obtains its first positive appraisal at $i = 16$, as denoted by the jump in utility drive and drop in information gain. This first positive appraisal is followed by a period of oscillations in both terms, where the agent is refining its parameters. Finally, AIDA settles down to predominantly exploitative behavior starting from 41^{st} trial. To examine the first transition, we can visualize the EFE landscape at $i = 5$ and $i = 6$, the upper row of Figure 5.7.

Recall that AIDA is minimizing EFE. Therefore, it is looking for the lowest values corresponding to blue regions and avoiding the high values corresponding to red regions. Between $k = 5$ and $k = 6$ we perform the first hyperparameter update, which drastically changes the EFE landscape. This indicates that initial parameter settings were not informative, as we did not cover the majority of the search space within the first 5 iterations. The yellow regions at $k = 6$ indicates regions corresponding to previous proposals of AIDA that resulted into negative appraisals. We can visualize snapshots of the exploration phase starting from $k = 6$ in a similar manner. The second row of Figure 5.7 displays the EFE landscape at two different time instances during the exploration phase. It shows that over the course of the experiment, AIDA gradually builds a representation over the search space. In trial 16 this takes the form of patterns of connected regions that denote areas that AIDA believes are unlikely to results in positive appraisals.

Once AIDA receives its first positive appraisal at $k = 16$, it switches from exploring the search space to focusing only on the local region. If we examine Figure 5.6, we see that at this time the information gain term is still reasonably high. This indicates a subtle point: once AIDA receives a positive appraisal, it starts with *local* exploration around where the optimum might be located. However, the agent was located near the boundary of the optimum and next receives a negative appraisal. Therefore in trials 18 to 22 AIDA queries points which it deems most informative. At time 23 the position of AIDA in the search space (black dot in the third row of Figure 5.7) returns to the edge of the user preference function in Figure 5.5. This causes AIDA to receive a mixture of positive and negative appraisals in the following trials, leading to the oscillations seen in Figure 5.6. Finally, we can examine the landscape after AIDA has confidently located the optimum and switched to purely exploitative behavior. This happens at $k = 42$ where the utility drive goes to 0 and the information gain concentrates around -1.

The last row of Figure 5.7 shows that once $\boldsymbol{u}^*$ is confidently located, AIDA dis-

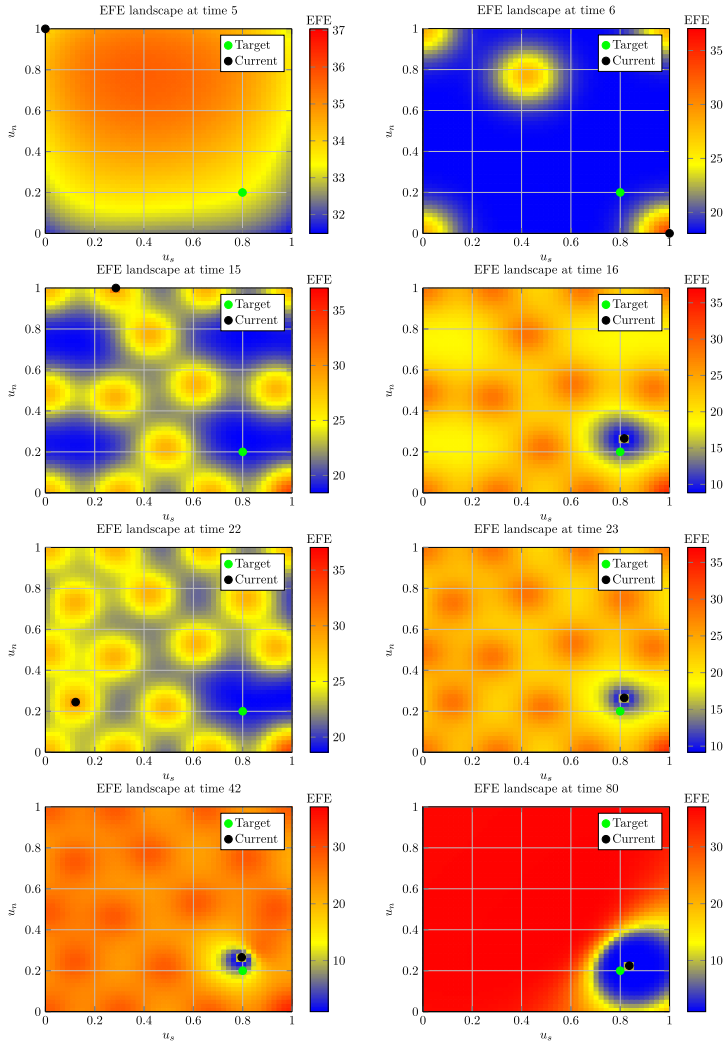


Figure 5.7: Snapshot of EFE landscape at different time points as a function of gains u_s and u_n. The black dot denotes the current parameter settings and the green dot denotes $\boldsymbol{u}^*$.

regards the remainder of the search space in favour of providing good parameter settings. Finally, if the user continues to supply data to AIDA, it will gradually extend the potential region of samples around the optimum. This indicates that if a user keeps requesting updated parameters, AIDA will once again perform local

exploration around the optimum. This further indicates that AIDA accommodates gradual retraining as user's hearing loss profile changes over time.

Having thoroughly examined an example run and investigated the types of behavior produced by AIDA, we can now turn our attention to aggregate performance over an ensemble of agents. To that end we repeat the experiment 80 times with identical hyperparameters, but with different initial proposals. The metric we are most interested in is how quickly AIDA is able to locate the optimum and produce a positive appraisal.

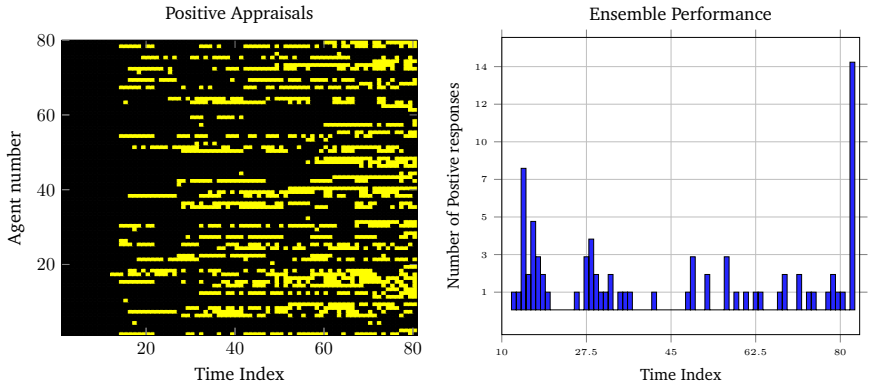


Figure 5.8: (Left) Heatmap showing ensemble performance over 80 agents. Positive and negative responses are indicated with yellow and black squares, respectively. (Right) Histogram showing time indices where the agents receive their first positive response. The right most column indicates agents that failed to obtain a positive appraisal. In total, 66/80 agents solve the task, corresponding to a success rate of 82.5%.

Figure 5.8 shows a heatmap of when each agent obtains positive responses. Positive responses are indicated by yellow squares and negative responses by black squares. Each row contains results for a single AIDA-agent and each column indicates a time step of the experiment. Consistent with the results for a single agent, we see that each experiment starts with a period of exploration. A large number of rows also show a yellow square within the first 35 trials, indicating that the optimum was found. Interestingly, no agents receive only positive responses, even after locating the optimum. This follows from AIDA actively trading off exploration and exploitation. When exploring, AIDA can select parameters that are suboptimal with respect to eliciting positive user responses, to gather more information. Figure 5.8 also shows a histogram indicating when each agent obtains its first positive appraisal. The very right column shows agents that failed to locate the optimum within the designated number of trials. In total, 66/80 agents correctly solve the task, corresponding to a success rate of 82.5%. Disregarding unsuccessful runs, on

average, AIDA obtains a positive response in 37.8 trials with a median of 29.5 trials.

5.5.3 Hearing aid algorithm execution verification

To verify the proposed inference methodology for the hearing aid algorithm execution, we synthesized data by sampling from the following generative model:

$$\boldsymbol{\theta}_t \sim \mathcal{N}\left(\boldsymbol{\theta}_{t-1},\ \omega \mathrm{I}_K\right) \tag{5.24a}$$
$$\boldsymbol{s}_t \sim \mathcal{N}\left(A(\boldsymbol{\theta}_t)\boldsymbol{s}_{t-1},\ V\left(\gamma\right)\right) \tag{5.24b}$$
$$\boldsymbol{n}_t \sim \mathcal{N}\left(A(\boldsymbol{\varrho})\boldsymbol{n}_{t-1},\ V\left(\tau\right)\right) \tag{5.24c}$$
$$x_t = s_t + n_t, \tag{5.24d}$$

with priors

$$\boldsymbol{\theta}_0 \sim \mathcal{N}(\mathbf{0}, \omega \mathrm{I}_M) \tag{5.25a}$$
$$\boldsymbol{\varrho} \sim \mathcal{N}(\mathbf{0}, \mathrm{I}_N) \tag{5.25b}$$
$$\gamma \sim \Gamma(1.0, 1e-4) \tag{5.25c}$$
$$\tau \sim \Gamma(1.0, 1.0) \tag{5.25d}$$
$$\omega = 1e-4 \tag{5.25e}$$

where K and R are the orders of TVAR and AR models, respectively, and where $K \geq R$ holds, as we assume that the noise signal can be modeled by a lower AR order in comparison to the speech signal. We use an uninformative prior for the output of the hearing aid y_t as in Figure 5.3 to prevent interactions from that part of the graph. We generated 1000 distinct time series of length 100. For each generated time series, the (TV)AR orders K and R were sampled from the discrete domains $[4, 8]$ and $[1, 4]$, respectively. We resampled the priors that initially resulted into unstable TVAR and AR processes.

The generated time series were used in the following experiment. We first created a probabilistic model with the same specifications as the generative model in (5.24). However, we used non-informative priors for the states and parameters of the model that corresponds to the TVAR process in (5.24b). To ensure the identifiability of the separated sources, we used weakly informative priors for the parameters of the AR process in (5.24c). Specifically, the mean of the prior for $\boldsymbol{\zeta}$ was centered around the real AR coefficients that were used in the data generation process. The goals of the experiment are 1) to verify that the proposed inference procedure recovers the hidden states $\boldsymbol{\theta}_t$, s_t and n_t for each generated dataset and 2) to verify convergence of the BFE as convergence is not guaranteed, because our graph contains loops [142]. For a typical case, the inference results for the hidden states s_t and n_t are shown in the top row of Figure 5.9. The bottom row of Figure 5.9 shows the tracking of the time-varying coefficients $\boldsymbol{\theta}_t$. This plot does not show

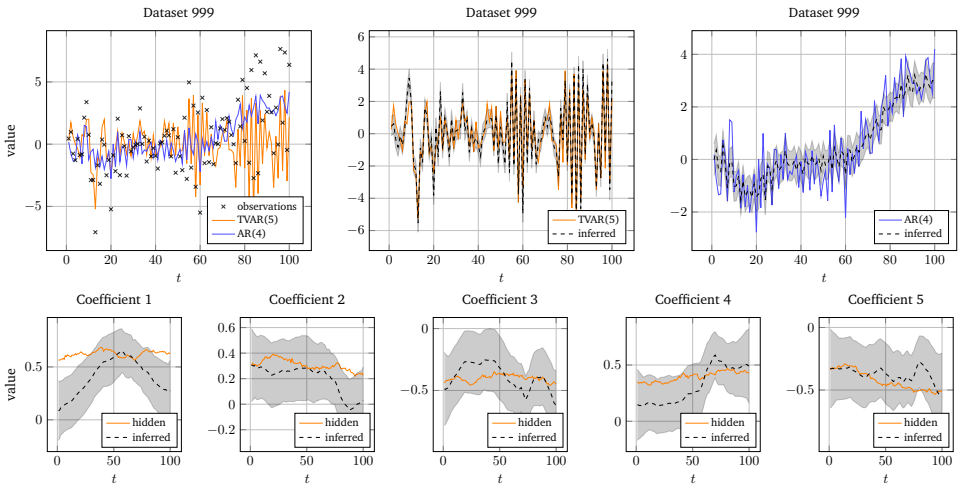


Figure 5.9: (Top) Inference results for the hidden states s_t and n_t of coupled (TV)AR process on dataset 999. (left) The generated observed signal x_t with underlying generated signals s_t and n_t. (center) The latent signal s_t and its corresponding posterior approximation. (right) The latent signal n_t and its corresponding posterior approximation. The dashed lines corresponds to the mean of the posterior estimates. The transparent regions represent the corresponding remaining uncertainty as plus-minus one standard deviation from the mean. (Bottom) Inference results for the coefficients $\boldsymbol{\theta}_t$ of dataset 999. The solid lines correspond to the true latent AR coefficients. The dashed lines correspond to the mean of the posterior estimates of the coefficients and the transparent regions correspond to plus-minus one standard deviation from the mean of the estimated coefficients.

the correlation between the inferred coefficients, whereas this actually contains vital information for modeling an acoustic signal. Namely, the coefficients together specify a set of poles, which influence the characteristics of the frequency spectrum of the signal. An interesting example is depicted in Figure 5.10. We can see that the inference results for the latent states s_t and n_t are swapped with respect to the true underlying signals. This behavior is undesirable in standard algorithms when the output of the HA is produced based on hard-coded gains. However, the presence of our intelligent agent can still find the optimal gains for this situation. The automation of the hearing aid algorithm and intelligent agent will relieve this burden on HA clients. As can be seen from Figure 5.11, the Bethe free energy averaged over all generated time series monotonically decreases. Note that even though the proposed hybrid message passing algorithm results in a stationary solution, it does not provide convergence guarantees.

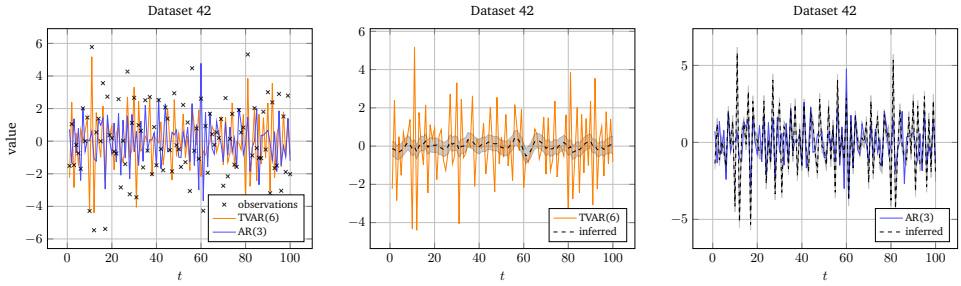


Figure 5.10: Inference results for the hidden states s_t and n_t of coupled (TV)AR process on dataset 42. In this particular case it can be noted that the inferred states are swapped with respect to the true underlying signals. However, the accompanying intelligent agent is able to cope with these kinds of situations, such that the HA clients do not experience any problems as a result.

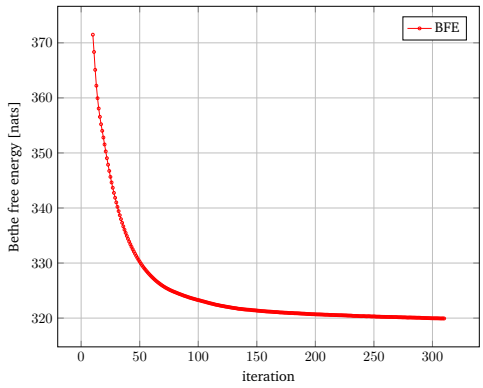


Figure 5.11: Evolution of the Bethe free energy for the coupled autoregressive model averaged over all generated time series. The iteration index specifies the number of marginal updates for all edges in the graph.

5.5.4 Validation experiments

For the validation of the proposed HA algorithm and AIDA, we created an interactive web application[4] to demonstrate the the joint system. Figure 5.12 shows the interface of the demonstrator.

[4] A web application of AIDA is available at `https://github.com/biaslab/AIDA-app/`.

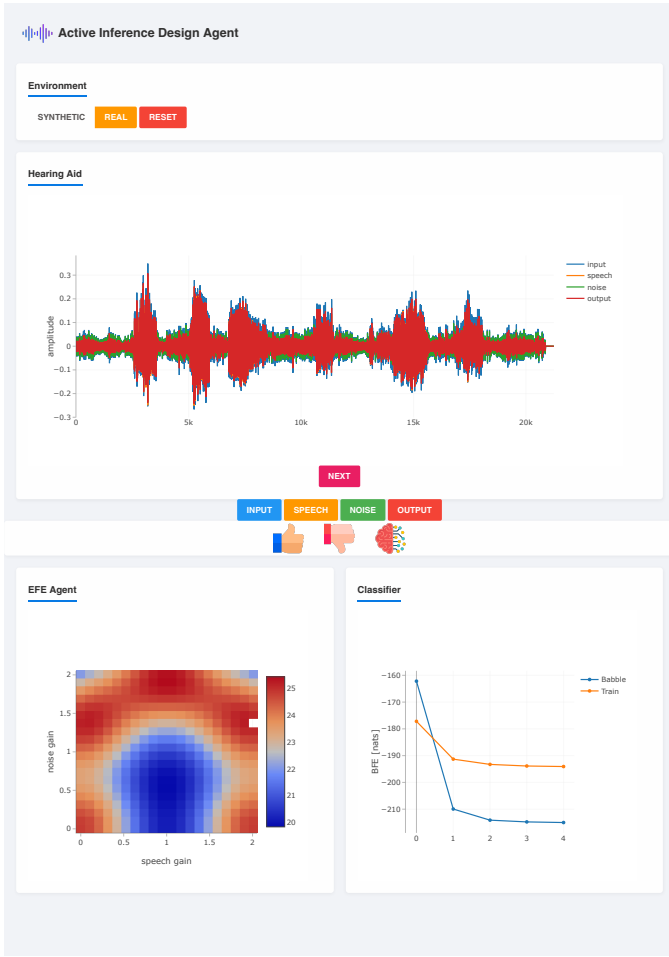


Figure 5.12: Screenshot of the interactive web application of AIDA. The dashboard consists of four distinct cells. The top cell *Environment* allows the user to change the interfering noise signal from a generated noise signal (synthetic) to a real noise signal. Furthermore it contains a reset button for resetting the application. The *Hearing Aid* cell provides an interactive plot of the input, separated speech, separated noise, and generated output waveform signals. Each waveform can be played when the corresponding button is pressed. The *NEXT* button loads a new audio file for evaluation. The *thumbs-up* and *thumbs-down* buttons correspond to providing AIDA with positive and negative appraisals, respectively. The *brain* button starts optimization of the parameters of GPC. The *EFE Agent* cell reflects the agent's beliefs about optimal parameters for the user as an EFE heatmap. The *Classifier* cell shows the Bethe free energy (BFE) score for the different models, corresponding to the different contexts. For the real noise signal, the algorithm automatically determines whether we are surrounded by babble noise, or by noise from a train station.

The user listens to the output of the hearing aid algorithm by pressing the "output" button. The buttons "speech" and "noise" correspond to the beliefs of AIDA about the constituent signals of the HA input. Note that in reality the user does not have access to this information and can only listen to HA output. After listening to the output signal, the user is invited to assess the performance of the current HA setting. The user can send positive and negative appraisals by pressing the thumb up or thumb down buttons respectively. Once the appraisal is sent, AIDA updates its beliefs about the parameters' space and provides new settings for the HA algorithm to make the user happy. As AIDA models user appraisals using a GPC, we provide an additional button that forces AIDA to optimize the parameters of GPC. This could be useful when AIDA has already collected some feedback from the user that contains both positive and negative appraisals.

The demonstrator works in two environments: synthetic and real. The synthetic environment allows the user to listen to a spoken sentence with two artificial noise sources, i.e. either interference from a sinusoidal wave or a drilling machine. In the synthetic environment the hearing aid algorithm exploits the knowledge about acoustic contexts, i.e, it uses informative priors for the AR model that corresponds to noise. The real environment uses the data from NOIZEUS speech corpus[5]. In particular, the real environment consists of 30 sentences pronounced in two different noise environments. Here the user is either experiencing surrounding noise at a train station or babble noise. In the real environment, the HA algorithm uses weakly informative priors for the background noise which influences the performance of the HA algorithm. Both the HA algorithm and AIDA determine the acoustic context based on the Bethe free energy score, which is also shown in the demonstrator. The context with the lower Bethe free energy score corresponds to the selected acoustic context.

5.6 Discussion

We have introduced a design agent that is capable of tuning the context-dependent parameters of a hearing aid algorithm by incorporating user feedback. Throughout the paper, we have made several design choices whose implications we shortly review in this section.

The audio model introduced in Section 5.3.1 describes the dynamics of the speech signal perturbed by colored noise. Despite the fact that the proposed inference algorithm allows for the decomposition of such signals into speech and noise components, there are a few limitations that must be highlighted. First, the identifiability of the coupled AR model depends on the selected priors. Non-informative priors can lead to poor source estimation [143, 144]. To tackle the identifiability

[5]The NOIZEUS database is available at `https://ecs.utdallas.edu/loizou/speech/noizeus/`.

issue, we use informative context-dependent priors. In other words, for each context, we use a different set of priors that better describe the dynamics of the acoustic signal in that context. Secondly, throughout our experiments, we used fixed orders of TVAR and AR models. In reality, we do not have prior information about the actual order of the underlying signals. Therefore, to continuously update our models of the underlying sources we need to perform active order selection, which can be realized using Bayesian model reduction [145, 146]. Thirdly, our model assumes that the hearing aid device only has access to a monaural input, which means that the observed signal originates from a single microphone. As a result, we do not use any spatial information about an acoustic signal that could have been obtained using multiple microphones. This assumption is mostly influenced by our desire to focus on the concept of designing a novel class of hearing aid algorithms rather than building a real-world HA engine. Fortunately, the proposed framework allows for the easy substitution of source models with more versatile models that might be better suited for speech. For instance, one can use several microphones, as commonly done in beamforming [147], or use a frequency decomposition for improving the source separation performance [148, 149, 150]. Inevitably, a more complex model will also likely result in a higher computational burden. Hence, the implementation of this algorithm on an embedded device remains a challenge.

The power of the agent comes from the choice of the objective function. Since the objective is independent of the generative model, a straightforward approach to improving the agent is to adapt the generative model. In particular, a GPC is a nonparametric model with very few assumptions on the underlying function. Placing constraints on the preference function, such as was done in [151, 18], is likely to improve the data efficiency of the agent. Arguably, a core move of [151, 18] is to acknowledge that user preferences are likely to be peaked around one or a few optima. Even if the true preference function has multiple modes, assuming a single peak for the agent is safe since it only needs to locate one of the modes to provide good parameter settings. Making this assumption allows the authors to work with a parametric model over user preferences. Working with a less flexible model predictably leads to higher data efficiency, which can aid the performance of the agent. Given that the target demographic for AIDA consists of HA users, it is of paramount importance that the agent is able to learn an adequate representation of user preferences in as few trials as possible to avoid inconveniencing the user.

During model specification in Section 5.3.2, we make some assumptions on the control variable $\boldsymbol{u}_i$ and user appraisals r_i. First, we set the domain of the elements of the control variable $\boldsymbol{u}_i$ to $[0, 1]$. Note that this is an arbitrary constraint that we use for illustrative purposes. The domain can be easily rescaled without loss of generality. For example, in our demonstrator, we use the default domain of $\boldsymbol{u}_i \in [0, 2]^2$. Secondly, we opt for binary user appraisals, i.e. $r_i \in \{\varnothing, 0, 1\}$. This design choice follows from the requirement of allowing users to communicate covertly to AIDA. Binary user appraisal can more easily be linked to for example

covert wrist movements when wearing a smartwatch to update the control variables. With continuous user appraisals, e.g. $r_i \in [0, 1]$, or pairwise comparison tests the convergence of AIDA can be greatly improved as these appraisals yield more information per appraisal. However, providing AIDA with these appraisals requires more attention, which is undesirable in certain circumstances, for example during business meetings.

Real-world testing of AIDA has not been included in our work. The performance evaluation with human HA clients is not straightforward. To evaluate the performance of AIDA, we need to conduct a randomized controlled trial (RCT), where HA clients should be randomly assigned to either an experimental group or a control group. While the current intelligent AIDA agent can interact with users in real time, the source separation framework is currently limiting the actual real-time performance. Under the current model assumptions, i.e., two autoregressive filters under a variational approximation, we obtain a pretty good source separation performance at the cost of computational complexity. Hence, the complete framework is not suitable for the proper RCT setting. Nonetheless, we provide a demo that simulates AIDA and can be tested freely. In future work, we shall focus on specifying source models that exhibit cheap computations allowing us to run the source separation algorithms in real-time.

5.7 Related work

The problem of hearing aid personalization has been explored in various works. In [114]the HA parameters are tuned according to pairwise user assessment tests, during which the user's perception is encoded using Gaussian processes. The intractable posterior distribution corresponding to the user's perception is then computed using a Laplace approximation with Expected Improvement as the acquisition function used to select the next set of gains. Our agent improves upon [114] in two concrete ways. Firstly, AIDA places a lower cognitive load on the user by not requiring pairwise comparisons. This means the user does not need to keep in her memory what the HA sounded like at the previous trial but only needs to consider the current HA output. AIDA accomplishes this without requiring more trials for training. In fact, since AIDA does not require pre-training but can be trained fully online under in-situ conditions, AIDA requires fewer data to locate optimal gains. Secondly, AIDA can be trained and retrained in a continual learning fashion. In case the user's preferences change over time, for instance, by a change in the hearing loss profile, AIDA can smoothly accommodate the user as long as she continues to provide the agent with feedback. Using EFE as an acquisition function means the agent will engage in local exploration once the optimum is located, leading the agent to naturally learn shifts in the user's preferences by balancing exploration and exploitation. In [115], personalization of the hearing aid compression algo-

rithm is framed in terms of deep reinforcement learning. On the contrary, in our work, we take inspiration from the active inference framework where agents act to maximize model evidence of their underlying generative model. Importantly, this does not require us to explicitly specify a loss function that drives exploitative and epistemic behavior. In the recent work of [151], the hearing aid preference learning algorithm is implemented through sequential Bayesian optimization with pairwise comparisons. Their hearing aid system comprises two subsystems representing a user with their preferences and the agent that guides the learning process. However, [151] focuses only on an exploration through maximizing information gain with a parametric model. The EFE additionally adds a goal-directed term that ensures the agent will stay near the optimum once located, even if other parameter settings provide more information. Extending the model of [151] to employ the full EFE is an exciting potential direction for future work. Finally neither [114] nor [151] takes context dependence into account.

[152] introduces Active Listening (AL), which performs speech recognition based on the principles of active inference. In [152], they regard listening as an active process that is largely influenced by lexical, speaker, and prosodic information. [152] distinguishes itself from conventional audio processing algorithms because it explicitly includes the process of word boundary selection before word classification and recognition, and they regard this as an active process. Word boundaries are selected from a group of candidate word boundaries, based on Bayesian model selection, by choosing the word boundary that optimizes the VFE during classification. In the future, we see the potential of incorporating the AL approach into AIDA. The active inference is successfully applied in the work [32] that studies to model selective attention in a cocktail party listening setup.

The audio processing components of AIDA essentially perform informed source separation [140], where sources are separated based on prior knowledge. Even though blind source separation approaches [153, 154] always use some degree of prior information, we do not focus on this direction and instead, we actively try to model the underlying sources based on variations of auto-regressive processes. For audio processing applications source separation has often been performed in the log-power domain [148, 149, 150]. However, the interaction of the signals in this domain is no longer linear. The intractability that results from performing exact inference in this model is often resolved by simplifying the interaction function [155, 156]. Although this approach has shown to be successful in the past, its performance is limited because of the negligence of phase information.

5.8 Conclusions

This paper has presented AIDA, an active inference design agent for novel situation-aware personalized hearing aid algorithms. AIDA and the corresponding hearing

aid algorithm are based on probabilistic generative models that model the user and the underlying speech and context-dependent background noise signals of the observed acoustic signal, respectively. Through probabilistic inference by means of message passing, we perform informed source separation in this model and use the separated signals to perform source-specific filtering. AIDA then learns personalized source-specific gains through user interaction, depending on the environment that the user is in. Users can give a binary appraisal, after which the agent will make an improved proposal based on expected free energy minimization for encouraging both exploitative and epistemic behavior. AIDA's operations are context-dependent and use the context from the hearing aid algorithm, which is based on Bayesian model selection. Experimental results show that hybrid message passing is capable of finding the hidden states of the coupled AR model that are associated with the speech and noise components. Moreover, Bayesian model selection has been tested for the context inference problem where each source is modeled by the AR process. The experiments on preference learning showed the potential of applying expected free energy minimization for finding the optimal settings of the hearing aid algorithm. Although real-world implementations still present challenges, this novel class of audio processing algorithms has the potential to change the leading approach to hearing aid algorithm design. Future plans encompass developing AIDA towards real-time applications.

Chapter 6

Discussion and Conclusions

6.1 Contributions

This thesis has explored and built a methodology for message passing-based inference in hierarchical autoregressive models. The central research question this dissertation endeavored to answer was:

> *How can Bayesian inference be realized for hierarchical autoregressive models for signal processing applications?*

Chapter 2 presented Forney-style factor graphs (FFGs) as an efficient framework for Bayesian inference in state-space models (Section 2.4). We have introduced a hierarchical model - the TVAR model that could be used in different signal modeling scenarios. Bayesian inference of FFGs is achieved by the message passing algorithm, which has linear complexity with respect to added nodes. We demonstrated how TVAR model could reduce to simpler AR-like models by manipulating AR nodes (Section 2.6). Additionally, we explored the process of online model selection for this class of models.

Intending to add regime-switching behavior to HAR models, chapter 3 focused on a mixture of Gamma distributions model (ΓMM). The ΓMM was cast to an FFG that rests upon composite ΓM nodes, see Fig. 3.1. To deal with the intractability of the posterior distribution of the shape parameter of ΓM, we introduced two MP-based approximation techniques. This preliminary step opened the doors for making a fully-Bayesian switching autoregressive model (SwAR) that tracks changing states in the acoustic environment (Chapter 4).

Finally, this dissertation has presented an active inference-based design agent (AIDA) to develop context-dependent audio processing algorithms (Chapter 5). The modularity of the FFG framework allowed us to fuse the models of previous chapters into a single generative state-space model representing the acoustic environment of AIDA that consists of coupled TVAR and SwAR models. We have presented

a GP-based user preference model that enabled AIDA to set personalized gains depending on the acoustic environment that the user is in. Moreover, Bayesian model selection has been employed for the acoustic context inference problem, where the AR process models each acoustic source. Ultimately, we framed the results into an open-source demonstrator (Figure 5.12).

All derived message passing updates and free-energy computation rules have been implemented in the open-source Julia package `ReactiveMP.jl` that is designed with a focus on efficient and scalable Bayesian inference.

6.2 Strength and Limitations

In this section, we analyze the strengths and limitations of our contributions. To answer this dissertation's central question, we recognized that the construction of hierarchies in autoregressive models could be implemented by extending the mean or variance of the probability distribution for the AR model. Unfortunately, the construction of such hierarchies leads to intractable inference. Hence some questions about approximate inference solutions for hierarchical autoregressive models arose. At first, we focused on a subclass of HAR models, namely TVAR models. The first research question was formulated as follows:

Q1. How can approximate Bayesian inference be implemented for time-varying autoregressive models?

To answer this question, we first showed why we are interested in calculating the model evidence as a performance metric for the TVAR model. Indeed, as shown in Chapter 2, the model evidence is fundamental to assessing the performance of any probabilistic model. We also showed how state inference can be formulated as a prediction-correction process. Unable to compute the model evidence for the TVAR model, we translated the problem to the minimization of VFE. We then cast the TVAR model into the language of Forney-style factor graphs. The FFG framework allowed us to cast the inference problem into the problem of deriving the message update rules for the AR node. We showed how to employ the hybrid message passing algorithm for TVAR models based on a combination of BP and structured VMP. Additionally, we illustrated how to evaluate the VFE for TVAR models by decomposing VFE into the sum of local average energy and entropy terms. Finally, we showed the applicability of the proposed methods for temperature modeling and speech enhancement problems. The main strength of the advocated solution can be summarized as follows:

- Closed-form variational update rules for the AR node;
- Closed-form FE computation rules for the AR node;

- A modular FFG representation of TVAR models. The modular nature of the FFG framework supports flexible re-use of the AR node in different models;

The closed-form update rules are important since we are interested to implement these models on wearable devices, so low power consumption is essential.

One of the limitations of this work is the omission of multivariate autoregressive (MAR) models [70]. This class of models is actively used in finance and neuroscience applications. One of the reasons for omitting MAR is its low applicability to the problem of speech signal processing. As opposed to AR models, MAR models use matrices instead of a vector of AR coefficients. In the context of Bayesian inference, we would need to insert matrix normal distributions for these matrices. However, working with such distributions creates difficulties associated with tractability and speed of inference. Still, creating a MAR node may be of interest in future research endeavors.

This work has also explored two factorizations of the variational posterior distribution of the AR node, namely, a naive mean-field and a structured mean-field factorization between past and current states. Technically, we could consider alternative factorizations and compare their effect on the inference. Furthermore, it would be interesting to compare the inference results of our algorithm with other algorithms, based on sampling or Stochastic Variational Inference (SVI) [157, 158]. However, since we are interested in real-time inference in our application, we elected not to pursue sampling-based inference methods as they are considered accurate but computationally very expensive.

This chapter has not rigorously explored the model selection procedure for TVAR models. In section 2.6, we favored models based on lower values of the minimized VFE. However, VFE is an unnormalized value, meaning that a "blind" comparison of its value for entirely different models and inference algorithms is theoretically unfounded [45, 159]. In our case, we were justified in doing so in view of the fact that inference for competing models can be seen as manipulating the variational constraints rather than the models. That is why we have obtained interpretable empirical results in the model selection experiments.

Our next concrete research question focused on the inference in the ΓM model:

Q2. How can Bayesian inference be implemented for tracking hidden states and parameters in a Gamma mixture model?

As mentioned in Chapter 1, the answer to this question is crucial from the point of view of building SwAR models. Chapter 3 answers **Q2** by introducing the ΓM node and providing two MP-based inference schemes: VMP-EM and VMP-MM. While VMP-EM enjoys faster inference, unlike VMP-MM, it does not provide a "full" posterior distribution for the shape parameter of ΓMM. This means that VMP-EM is not suitable for tracking problems. We showed how combinations of ΓM nodes could be used to process univariate and multivariate observations with positive support. The main strength of our solution can be summarized as follows:

- Development of the ΓM node that can be used as a plug-in module in various hierarchical models;

- Closed-form variational update rules for ΓM node;

This work also has a few shortcomings. First, the performance of the inference has not been compared to methods based solely on sampling. Second, we did not consider alternative approximation techniques of the messages towards the rate parameters. In future studies, it would be interesting to use Gaussian quadrature-based methods [95, Chapter 6] which deterministically select sample points and approximate the integral (update message) as the weighted average. Owing to the deterministic nature of the methods, they can provide faster inference. Perhaps the main disadvantage of this chapter is the lack of rigorous procedures for evaluating the number of mixtures. For example, to choose the optimal number of mixture components, we computed the values of mixing coefficients with different numbers of mixture components. Mixing coefficients that converge to 0 suggest that the corresponding mixture component is not contributing. Theoretically, the number of mixtures can be determined using a Dirichlet process [160] as a prior probability distribution.

The development of the ΓM node allowed us to tackle the next concrete research question easily:

Q3. How can approximate Bayesian inference be implemented in switching autoregressive models?

To answer **Q3** we constructed an FFG representation of the SwAR model using ΓM, GM nodes, and a hidden Markov model (HMM). The inference of the states and parameters in the SwAR model was subject to the local computation of rules of the basis nodes. The strength of this work lies in the modular design of the SwAR model that allows for its integration into more complex hierarchical models. We have successfully applied the SwAR model for acoustic scene classification.

As a disadvantage, chapter 4 did not explore alternative ways of modeling the switching precision of the SwAR. For instance, the precision of the SwAR could follow the Switching Hierarchical Gaussian filter (SHGF) proposed by us in [161]. A comparison of SwAR models based on ΓMM and SHGF would be an exciting direction for future research. Finally, this work has not explored the applicability of the proposed inference method to real-time filtering problems when new observations become available in sequential order.

"Armed" with automated inference procedures in TVAR and SwAR models, we moved on to the final concrete research question of this thesis:

Q4. How can hierarchical autoregressive models support the development of novel personalized hearing aid algorithms?

To answer **Q4**, we introduced in chapter 5 an acoustic environment for the hearing aid agent based on a HAR model composed of both the TVAR and SwAR models. The TVAR model describes a clean speech signal, while the SwAR model characterizes the switching dynamics of the acoustic scenes (noise). The modular FFG approach allowed us to easily couple these two models and carry out inference by using previously derived message passing update rules. The proposed inference algorithm provides strong performance in terms of source separation.

However, the limiting factor of our solution is still the inability to perform real-time inference on a standard laptop. The coupling of two autoregressive models (TVAR and SwAR) through a deterministic addition node creates an unbreakable loop. To still be able to run inference, we used a loopy BP algorithm. We showed empirically that our algorithm converges within more than 200 iterations. Unfortunately, using so many iterations is unacceptable for real-time signal processing. We see one potential remedy that would decrease the computational burden. Instead of a linear mapping between states, one can find a nonlinear state transformation using neural networks such as normalizing flows [162, 163]. In this case, the learning part will be decoupled from inference, yielding a substantially faster signal processing procedure.

Our answer to **Q4** takes inspiration from the Active Inference (AIF) framework, which constitutes a unified variational Bayesian procedure for data processing, learning, and inference of actions. We have introduced an AIF-based agent named AIDA, and presented three tasks that AIDA aims to solve: (1) classification of acoustic context, (2) trial design, and (3) execution of the signal processing algorithm. Although we formulated and worked our all three tasks separately as free-energy minimization tasks, we have yet to fuse them into a single optimization problem. For instance, context classification and signal processing is achieved by minimizing BFE through message passing, while trial design minimizes *Expected* FE of future states, which is a task that we have not yet realized by MP in an FFG. Moreover, to learn the user's preferences, we used a GP classifier, which is a non-parametric model that we did not yet condense into an FFG node. The representation of GPs in FFGs is an exciting perspective for future research.

In summary, the answers to questions **Q1**-**Q4** provide a recipe for the realization of Bayesian inference in hierarchical autoregressive models for audio signal processing applications, hence yielding the answer to the main question of this dissertation. It was shown how HAR models could be built by combining various modules in factor graphs. As a product, we presented the add-on software to the existing message passing frameworks, e.g. `ReactiveMP.jl` and `ForneyLab.jl`. In Chapter 5, it was shown how HAR models could help design audio processing applications. To share our results, we have created an open-source demonstrator of AIDA.

6.3 Outlook

Even though this dissertation focuses on hierarchical autoregressive models, the contribution of the work goes beyond that. The somewhat veiled goal of this work is to show how one can create large, complex systems by manipulating simple independent modules. Indeed, we have started the journey by simply creating an autoregressive node, then a mixture node, and then we "suddenly" emerged on the coupled AR model with time-varying and switching priors (Chapter 5). To ensure the compatibility of the constituent nodes, this dissertation committed to the message passing on Forney-Style factor graphs. However, the same principles for building complicated models can apply to other factor graph frameworks.

In Appendices A.1-B.1, we show in detail how to derive messages and marginals for the autoregressive and Gamma mixture nodes. The reader will notice that, in its essence, auto-regression is nothing more than a linear transformation of Gaussian states. The transition matrix has been parameterized by a multivariate normal distribution. Hence, the analogous derivations can be carried out to similar linear transformations. The same goes for the derivations of the Gamma mixture node. The reader can follow the node formalization and update rules derivations and apply these manipulations to other kinds of mixture nodes, such as a Beta mixture node.

The derivation of update equations for complex nodes is a complex effort that requires patience and attention. Due to the existence of tools that automate stochastic variational inference (SVI) and sampling, it is tempting to skip this manual work. However, we should note that both SVI and sampling-based inference are very resource-intensive and less accurate than their analytical counterparts. Therefore, if we want to adhere to situated, real-time processing, perform online model selection and reach scalable Bayesian inference on resource-constrained devices, we cannot avoid some algebraic acrobatics that leads to closed-form update rules.

Most of this work uses `ReactiveMP.jl` and contributes to this toolbox. In addition to being written in the high-performance Julia programming language, `ReactiveMP.jl` uses a very efficient schedule-free inference engine. Unlike other toolboxes, such as `ForneyLab.jl` [164] or `Infer.NET` [165], `ReactiveMP.jl` does not create any sequential schedule for message updates. A fixed schedule requires traversing the whole factor graph corresponding to the probabilistic model. In real-world applications, the model may be subject to structural adaptation, e.g., it may turn out that some factor nodes are no longer needed or the model needs to be extended. Recomputing the schedule will result in additional computational overhead. `ReactiveMP.jl` reacts to changes in data sources, executing the computations dynamically. As a result, a pre-computed schedule becomes irrelevant.

This dissertation does not include some works devoted to nonlinear autoregressive models in factor graphs [166]. This powerful set of models is used in various fields such as control theory and signal processing [167]. However, it is worth not-

ing that, from the perspective of the factor graph framework, creating nonlinear autoregressive nodes will not be much different from creating a regular AR node. The main difference will be in calculating the approximation of the message update rules.

Appendix A

Message Passing-based Inference in Time-Varying Autoregressive Models

A.1 AR node

Figure A.1 represents a composite AR node.

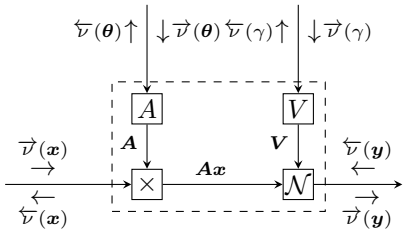


Figure A.1: Autoregressive (AR) node.

The corresponding node function of Figure A.1 $f(\boldsymbol{y}\ \boldsymbol{x}, \boldsymbol{\theta}, \gamma)$:

$$f(\boldsymbol{y}\ \boldsymbol{x}, \boldsymbol{\theta}, \gamma) = \mathcal{N}\left(\boldsymbol{y} \mid \boldsymbol{A}\boldsymbol{x}, \boldsymbol{V}\right)$$

where

$$\boldsymbol{A} = A(\boldsymbol{\theta}) \qquad \boldsymbol{V} = V(\gamma) = \begin{bmatrix} \gamma^{-1} & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & \vdots \\ 0 & 0 & 0 & \dots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \end{bmatrix}$$

.

A.2 Structural Variational Message Passing

The message update rule (2.24) implies a mean-field factorization, meaning that all variables represented by edges around the factor node f are independent. In this paper, we impose a structural dependence between states. To illustrate how structured VMP works, let us consider the example depicted in Figure A.2.

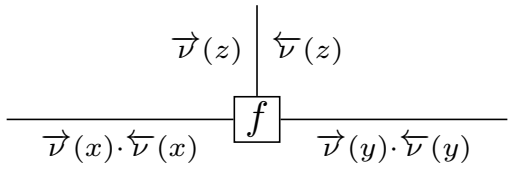


Figure A.2: A node $f(x, y, z)$ representing an arbitrary joint distribution. Arrows above the messages $\nu(\cdot)$ indicate the direction (incoming or outgoing).

Suppose that we constrain the joint posterior (A.1) as

$$q(x, y, z) = q(x, y)q(z) \tag{A.1}$$

The message passing algorithm for updating the marginal posteriors $q^*(x, y)$ and $q^*(z)$ can now be executed as follows: (1) compute outgoing messages $\vec{\nu}(y)$, $\overleftarrow{\nu}(x)$:

$$\vec{\nu}(y) \propto \int \vec{\nu}(x) \exp\left(\int q(z) \log\left[f(x, y, z) \right] \mathrm{d}z \right) \mathrm{d}x \tag{A.2a}$$

$$\overleftarrow{\nu}(x) \propto \int \overleftarrow{\nu}(y) \exp\left(\int q(z) \log\left[f(x, y, z) \right] \mathrm{d}z \right) \mathrm{d}y \tag{A.2b}$$

(2) update the joint posterior $q^*(x, y)$:

$$q^*(x, y) \propto \vec{\nu}(x) \exp\left(\int q(z) \log f(x, y, z) \mathrm{d}z \right) \overleftarrow{\nu}(y) \tag{A.3}$$

A

(3) compute the outgoing message $\vec{\nu}(z)$:

$$\vec{\nu}(z) \propto \exp\left(\int q^*(x,y)\log f(x,y,z)\mathrm{d}x\mathrm{d}y\right), \tag{A.4}$$

(4) update posterior $q^*(z)$:

$$q^*(z) \propto \vec{\nu}(z)\overleftarrow{\nu}(z) \tag{A.5}$$

Every marginal update rule (equations (2.25), (A.3), (A.5)) corresponds to a coordinate descent step on the variational free energy, and therefore the free energy is guaranteed to converge to a local minimum.

A.3 Auxiliary node function

Before obtaining the update messages for AR node, we need to evaluate the auxiliary node function $\tilde{f}(\boldsymbol{x},\boldsymbol{y}) \propto \exp\left\{\mathbb{E}_{q(\gamma)q(\boldsymbol{\theta})}\log\left[f(\boldsymbol{y}\ \boldsymbol{x},\boldsymbol{\theta},\gamma)\right]\right\}$. We also need to address the issue of invertability of the covariance matrix V. To tackle this problem, we assume $\epsilon > 0$, $\epsilon^2 \approx 0$ which allows us to introduce matrix $\boldsymbol{W} = \boldsymbol{V}^{-1}$ ($\boldsymbol{W}\boldsymbol{V} = \boldsymbol{V}\boldsymbol{W} = \boldsymbol{I}$).

$$\boldsymbol{V} = \begin{bmatrix} \gamma^{-1} & 0 & 0 & \dots & 0 \\ 0 & \epsilon & 0 & \dots & \vdots \\ 0 & 0 & \epsilon & \dots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \end{bmatrix}$$

$$\begin{aligned}
\log\tilde{f}(\boldsymbol{x},\boldsymbol{y}) &= \mathbb{E}_{q(\gamma)q(\boldsymbol{\theta})}\log f(\boldsymbol{y}\ \boldsymbol{x},\boldsymbol{\theta},\gamma) + \text{const} \\
&= \frac{1}{2}\mathbb{E}_{q(\gamma)}\left[\log|\boldsymbol{W}|\right] - \frac{1}{2}\mathbb{E}_{q(\gamma)q(\boldsymbol{\theta})}\left[(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x})^\top\boldsymbol{W}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x})\right] + \text{const} \\
&= -\frac{1}{2}\mathbb{E}_{q(\gamma)q(\boldsymbol{\theta})}\left[\mathrm{tr}\left(\boldsymbol{W}\left(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right)\left(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right)^\top\right)\right] + \text{const} \\
&= -\frac{1}{2}\mathrm{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\,\mathbb{E}_{q(\boldsymbol{\theta})}\left[\left(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right)\left(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right)^\top\right]\right) + \text{const} \\
&= -\frac{1}{2}\mathrm{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\left(\boldsymbol{y}\boldsymbol{y}^\top - \boldsymbol{m}_{\boldsymbol{A}}\boldsymbol{x}\boldsymbol{y}^\top - \boldsymbol{y}\boldsymbol{x}^\top\boldsymbol{m}_{\boldsymbol{A}} + \mathbb{E}_{q(\boldsymbol{\theta})}\left[\boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{A}^\top\right]\right)\right) \\
&\quad + \text{const}
\end{aligned}$$

We work out the expectation term inside the trace separately. To do this, we notice, that the product $\boldsymbol{A}\boldsymbol{x}$ can be separated in the shifting operator $\boldsymbol{S}\boldsymbol{x}$ and the inner vector product $\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}$ in the following way:

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{S}\boldsymbol{x} + \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta} = \underbrace{(\boldsymbol{S} + \boldsymbol{e}_1\boldsymbol{\theta}^\top)\boldsymbol{x}}_{\boldsymbol{A}\boldsymbol{x}} \tag{A.6}$$

where

$$\boldsymbol{S} = \begin{bmatrix} \mathbf{0}^\top & \\ \boldsymbol{I}_{K-1} & \mathbf{0} \end{bmatrix} \quad \boldsymbol{e}_1 = (1, 0, \ldots, 0)^\top$$

$$\begin{aligned}
\mathbb{E}_{q(\boldsymbol{\theta})}\left[\boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{A}^\top\right] &= \mathbb{E}_{q(\boldsymbol{\theta})}\left[\left(\boldsymbol{S}\boldsymbol{x} + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{\theta}\right)\left(\boldsymbol{S}\boldsymbol{x} + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{\theta}\right)^\top\right] \\
&= \mathbb{E}_{q(\boldsymbol{\theta})}\left[\boldsymbol{S}\boldsymbol{x}(\boldsymbol{S}\boldsymbol{x})^\top + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{\theta}(\boldsymbol{S}\boldsymbol{x})^\top + \boldsymbol{S}\boldsymbol{x}(\boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{\theta})^\top + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e_1}^\top\right] \\
&= \boldsymbol{S}\boldsymbol{x}(\boldsymbol{S}\boldsymbol{x})^\top + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{m_\theta}(\boldsymbol{S}\boldsymbol{x})^\top + \boldsymbol{S}\boldsymbol{x}(\boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{m_\theta})^\top + \boldsymbol{e_1}\boldsymbol{x}^\top\left[\boldsymbol{V_\theta} + \boldsymbol{m_\theta}\boldsymbol{m_\theta}^\top\right]\boldsymbol{x}\boldsymbol{e_1}^\top \\
&= \left(\boldsymbol{S}\boldsymbol{x} + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{m_\theta}\right)\left(\boldsymbol{S}\boldsymbol{x} + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{m_\theta}\right)^\top + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{V_\theta}\boldsymbol{x}\boldsymbol{e_1}^\top \\
&= \boldsymbol{m_A}\boldsymbol{x}(\boldsymbol{m_A}\boldsymbol{x})^\top + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{V_\theta}\boldsymbol{x}\boldsymbol{e_1}^\top
\end{aligned}$$

Hence

$$\begin{aligned}
&\log \tilde{f}(\boldsymbol{y}, \boldsymbol{x}) \\
&\quad = -\frac{1}{2}\operatorname{tr}\left(\boldsymbol{m_W}\left[\boldsymbol{y}\boldsymbol{y}^\top - \boldsymbol{m_A}\boldsymbol{x}\boldsymbol{y}^\top - \boldsymbol{y}\boldsymbol{x}^\top\boldsymbol{m_A} + \boldsymbol{m_A}\boldsymbol{x}(\boldsymbol{m_A}\boldsymbol{x})^\top + \boldsymbol{e_1}\boldsymbol{x}^\top\boldsymbol{V_\theta}\boldsymbol{x}\boldsymbol{e_1}^\top\right]\right) \\
&\qquad + \text{const} \\
&\quad = -\frac{1}{2}\left(\boldsymbol{y}^\top\boldsymbol{m_W}\boldsymbol{y} - \boldsymbol{y}^\top\boldsymbol{m_W}\boldsymbol{m_A}\boldsymbol{x} - (\boldsymbol{m_A}\boldsymbol{x})^\top\boldsymbol{m_W}\boldsymbol{y} + (\boldsymbol{m_A}\boldsymbol{x})^\top\boldsymbol{m_W}\boldsymbol{m_A}\boldsymbol{x}\right) \\
&\qquad - \frac{m_\gamma}{2}\boldsymbol{x}^\top\boldsymbol{V_\theta}\boldsymbol{x} + \text{const} \\
&\quad = -\frac{1}{2}(\boldsymbol{y} - \boldsymbol{m_A}\boldsymbol{x})^\top\boldsymbol{m_W}(\boldsymbol{y} - \boldsymbol{m_A}\boldsymbol{x}) - \frac{m_\gamma}{2}\boldsymbol{x}^\top\boldsymbol{V_\theta}\boldsymbol{x} + \text{const}
\end{aligned}$$

We can write the auxiliary node function as

$$\tilde{f}(\boldsymbol{x}, \boldsymbol{y}) \propto \mathcal{N}(\boldsymbol{y}|\boldsymbol{m_A}\boldsymbol{x}, \boldsymbol{m_W}^{-1})\mathcal{N}(\boldsymbol{x}|\mathbf{0}, (m_\gamma\boldsymbol{V_\theta})^{-1}) \tag{A.7}$$

A.4 Update of message to y

Owing Eq. (A.7),

$$\begin{aligned}
\vec{\nu}(\boldsymbol{y}) &\propto \int \vec{\nu}(\boldsymbol{x})\tilde{f}(\boldsymbol{x}, \boldsymbol{y})\mathrm{d}\boldsymbol{x} \\
&\propto \int \mathcal{N}(\boldsymbol{x}|\boldsymbol{m_x}, \boldsymbol{V_x})\mathcal{N}(\boldsymbol{y}|\boldsymbol{m_A}\boldsymbol{x}, \boldsymbol{m_W}^{-1})\mathcal{N}(\boldsymbol{x}|\mathbf{0}, (m_\gamma\boldsymbol{V_\theta})^{-1})\mathrm{d}\boldsymbol{x} \\
&\propto \int \mathcal{N}(\boldsymbol{x}|\boldsymbol{\Lambda}^{-1}\boldsymbol{z}, \boldsymbol{\Lambda}^{-1})\mathcal{N}(\boldsymbol{y}|\boldsymbol{m_A}\boldsymbol{x}, \boldsymbol{m_W}^{-1})\mathrm{d}\boldsymbol{x}
\end{aligned}$$

where

$$\begin{aligned}\boldsymbol{\Lambda} &= \boldsymbol{V}_{\boldsymbol{x}}^{-1} + m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}} \\ \boldsymbol{z} &= \boldsymbol{V}_{\boldsymbol{x}}^{-1} \boldsymbol{m}_{\boldsymbol{x}}\end{aligned}$$

In this way, the message $\vec{\nu}(\boldsymbol{y})$

$$\boxed{\vec{\nu}(\boldsymbol{y}) \propto \mathcal{N}\left(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{A}}(\boldsymbol{V}_{\boldsymbol{x}}^{-1} + m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1} \boldsymbol{V}_{\boldsymbol{x}}^{-1} \boldsymbol{m}_{\boldsymbol{x}}, \boldsymbol{m}_{\boldsymbol{A}}(\boldsymbol{V}_{\boldsymbol{x}}^{-1} + m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1} \boldsymbol{m}_{\boldsymbol{A}}^\top + \boldsymbol{m}_{\boldsymbol{V}}\right)}$$

A.5 Update of message to x

Owing Eq. (A.7),

$$\begin{aligned}\overleftarrow{\nu}(\boldsymbol{x}) &\propto \int \overleftarrow{\nu}(\boldsymbol{y}) \tilde{f}(\boldsymbol{x}, \boldsymbol{y}) \mathrm{d}\boldsymbol{y} \\ &\propto \int \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{y}}, V_{\boldsymbol{y}}) \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x}, \boldsymbol{m}_{\boldsymbol{W}}^{-1}) \mathcal{N}(\boldsymbol{x} | \boldsymbol{0}, (m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1}) \mathrm{d}\boldsymbol{y}\end{aligned}$$

Let us consider the log of $\mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x}, \boldsymbol{m}_{\boldsymbol{W}}^{-1})$:

$$\begin{aligned}\log\left[\mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x}, \boldsymbol{m}_{\boldsymbol{W}}^{-1})\right] &= (\boldsymbol{y} - \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x})^\top \boldsymbol{m}_{\boldsymbol{W}} (\boldsymbol{y} - \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x}) + \text{const} \\ &= (-\boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{y} + \boldsymbol{x})^\top \boldsymbol{m}_{\boldsymbol{A}}^\top \boldsymbol{m}_{\boldsymbol{W}} \boldsymbol{m}_{\boldsymbol{A}} (-\boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{y} + \boldsymbol{x}) + \text{const}\end{aligned}$$

Which yields,

$$\mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x}, \boldsymbol{m}_{\boldsymbol{W}}^{-1}) \propto \mathcal{N}(\boldsymbol{x} | \boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{y}, (\boldsymbol{m}_{\boldsymbol{A}}^\top \boldsymbol{m}_{\boldsymbol{W}} \boldsymbol{m}_{\boldsymbol{A}})^{-1}) \tag{A.8}$$

Therefore,

$$\begin{aligned}\overleftarrow{\nu}(\boldsymbol{x}) &\propto \int \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{y}}, V_{\boldsymbol{y}}) \mathcal{N}(\boldsymbol{x} | \boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{y}, (\boldsymbol{m}_{\boldsymbol{A}}^\top \boldsymbol{m}_{\boldsymbol{W}} \boldsymbol{m}_{\boldsymbol{A}})^{-1}) \mathcal{N}(\boldsymbol{x} | \boldsymbol{0}, (m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1}) \mathrm{d}\boldsymbol{y} \\ &\propto \mathcal{N}(\boldsymbol{x} | \boldsymbol{0}, (m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1}) \int \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{y}}, V_{\boldsymbol{y}}) \mathcal{N}(\boldsymbol{x} | \boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{y}, (\boldsymbol{m}_{\boldsymbol{A}}^\top \boldsymbol{m}_{\boldsymbol{W}} \boldsymbol{m}_{\boldsymbol{A}})^{-1}) \mathrm{d}\boldsymbol{y} \\ &\propto \mathcal{N}(\boldsymbol{x} | \boldsymbol{0}, (m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1}) \mathcal{N}(\boldsymbol{x} | \boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{m}_{\boldsymbol{y}}, \boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{V}_{\boldsymbol{y}} \boldsymbol{m}_{\boldsymbol{A}}^{-\top} + (\boldsymbol{m}_{\boldsymbol{A}}^\top \boldsymbol{m}_{\boldsymbol{W}} \boldsymbol{m}_{\boldsymbol{A}})^{-1}) \\ &\propto \mathcal{N}(\boldsymbol{x} | \boldsymbol{0}, (m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1}) \mathcal{N}(\boldsymbol{x} | \boldsymbol{m}_{\boldsymbol{A}}^{-1} \boldsymbol{m}_{\boldsymbol{y}}, \boldsymbol{m}_{\boldsymbol{A}}^{-1} (\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{V}}) \boldsymbol{m}_{\boldsymbol{A}}^{-\top})\end{aligned}$$

Multiplication of two gaussians yields

$$\boxed{\overleftarrow{\nu}(\boldsymbol{x}) \propto \mathcal{N}\left(\boldsymbol{x} | \boldsymbol{\Lambda}^{-1} \boldsymbol{z}, \boldsymbol{\Lambda}^{-1}\right)}$$

where

$$\begin{aligned}\boldsymbol{\Lambda} &= \boldsymbol{m}_{\boldsymbol{A}}^\top \left(\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{V}}\right)^{-1} \boldsymbol{m}_{\boldsymbol{A}} + m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}} \\ \boldsymbol{z} &= \boldsymbol{m}_{\boldsymbol{A}}^\top \left(\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{V}}\right)^{-1} \boldsymbol{m}_{\boldsymbol{y}}\end{aligned}$$

A.6 Update of message to $\boldsymbol{\theta}$

The outgoing variational message to $\boldsymbol{\theta}$ is defined as

$$\overleftarrow{\nu}(\boldsymbol{\theta}) \propto \exp\left\{\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\gamma)} \log f(\boldsymbol{y}\ \boldsymbol{x}, \boldsymbol{\theta}, \gamma)\right\}$$

Instead of working out $\overleftarrow{\nu}(\boldsymbol{\theta})$, we will work with corresponding $\log$ message

$$\begin{aligned}
\log \overleftarrow{\nu}(\boldsymbol{\theta}) &= \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\gamma)} \left[\log |\boldsymbol{W}|^{\frac{1}{2}} - \frac{1}{2}\left((\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x})^\top \boldsymbol{W}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x})\right)\right] + \text{const} \\
&= -\frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\, \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[\boldsymbol{y}\boldsymbol{y}^\top - \boldsymbol{A}\boldsymbol{x}\boldsymbol{y}^\top - \boldsymbol{y}(\boldsymbol{A}\boldsymbol{x})^\top + \boldsymbol{A}\boldsymbol{x}(\boldsymbol{A}\boldsymbol{x})^\top\right]\right) + \text{const} \\
&= -\frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\, \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[-\boldsymbol{A}\boldsymbol{x}\boldsymbol{y}^\top - \boldsymbol{y}(\boldsymbol{A}\boldsymbol{x})^\top + \boldsymbol{A}\boldsymbol{x}(\boldsymbol{A}\boldsymbol{x})^\top\right]\right) + \text{const} \\
&= -\frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\, \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[-(\boldsymbol{S}\boldsymbol{x}+\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta})\boldsymbol{y}^\top - \boldsymbol{y}(\boldsymbol{S}\boldsymbol{x}+\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta})^\top\right]\right) \\
&\quad - \frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\, \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}(\boldsymbol{S}\boldsymbol{x}+\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta})(\boldsymbol{S}\boldsymbol{x}+\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta})^\top\right) + \text{const}
\end{aligned}$$

To proceed further, we recall one useful property

$$\boldsymbol{S}^\top \boldsymbol{\Sigma} \boldsymbol{e}_1 = \boldsymbol{0} \quad \boldsymbol{e}_1^\top \boldsymbol{\Sigma} \boldsymbol{S} = \boldsymbol{0}^\top$$

where $\boldsymbol{\Sigma}$ is an arbitrary diagonal matrix. Now, let us work out the following term

$$\begin{aligned}
&\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}(\boldsymbol{S}\boldsymbol{x}+\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta})(\boldsymbol{S}\boldsymbol{x}+\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta})^\top\right) \\
&\quad = \operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\left[\boldsymbol{S}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{S}\boldsymbol{x}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e}_1^\top + \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e}_1^\top\right]\right) \\
&\quad = \left[(\boldsymbol{S}\boldsymbol{x})^\top\boldsymbol{m}_{\boldsymbol{W}}\boldsymbol{S}\boldsymbol{x} + \underbrace{\boldsymbol{e}_1^\top\boldsymbol{m}_{\boldsymbol{W}}\boldsymbol{S}\boldsymbol{x}\boldsymbol{\theta}^\top\boldsymbol{x}}_{\boldsymbol{0}^\top} + \underbrace{\boldsymbol{S}^\top\boldsymbol{m}_{\boldsymbol{W}}\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{x}^\top}_{\boldsymbol{0}} + \boldsymbol{e}_1^\top\boldsymbol{m}_{\boldsymbol{W}}\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{x}\right] \\
&\quad = \operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\left[\boldsymbol{S}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e}_1^\top\right]\right)
\end{aligned}$$

Therefore,

$$\begin{aligned}
\log \overleftarrow{\nu}(\boldsymbol{\theta}) &= -\frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\, \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[-\boldsymbol{S}\boldsymbol{x}\boldsymbol{y}^\top - \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{y}^\top - \boldsymbol{y}\boldsymbol{x}^\top\boldsymbol{S}^\top - \boldsymbol{y}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e}_1^\top\right]\right) \\
&\quad - \frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\, \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[\boldsymbol{S}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e}_1^\top\right]\right) + \text{const}
\end{aligned}$$

We move terms which do not depend on $\boldsymbol{\theta}$ to the const, hence

$$\begin{aligned}
\log \breve{\nu}(\boldsymbol{\theta}) &= -\frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[-\boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{y}^\top - \boldsymbol{y}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e}_1^\top + \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{e}_1^\top\right]\right) + \text{const} \\
&= -\frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\left[-\boldsymbol{e}_1\boldsymbol{\theta}^\top(V_{\boldsymbol{x},\boldsymbol{y}^\top} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{y}}^\top) - (V_{\boldsymbol{x},\boldsymbol{y}^\top} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{y}}^\top)\boldsymbol{\theta}\boldsymbol{e}_1^\top\right]\right) \\
&\quad - \frac{1}{2}\operatorname{tr}\left(\boldsymbol{m}_{\boldsymbol{W}}\left[\boldsymbol{e}_1\left(\operatorname{tr}(\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{V}_{\boldsymbol{x}}) + \boldsymbol{m}_{\boldsymbol{x}}^\top\boldsymbol{\theta}\boldsymbol{\theta}^\top\boldsymbol{m}_{\boldsymbol{x}}\right)\boldsymbol{e}_1^\top\right]\right) + \text{const} \\
&= -\frac{1}{2}\left[-\underbrace{\boldsymbol{e}_1^\top\boldsymbol{m}_{\boldsymbol{W}}(V_{\boldsymbol{x},\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{y}}\boldsymbol{m}_{\boldsymbol{x}}^\top)}_{\boldsymbol{z}^\top}\boldsymbol{\theta} - \boldsymbol{\theta}^\top\underbrace{(V_{\boldsymbol{x},\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{y}}^\top)\boldsymbol{m}_{\boldsymbol{W}}\boldsymbol{e}_1}_{\boldsymbol{z}}\right] \\
&\quad - \frac{1}{2}\left[\boldsymbol{\theta}^\top\underbrace{m_\gamma(\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{x}}^\top)}_{\boldsymbol{\Lambda}}\boldsymbol{\theta}\right] + \text{const} \\
&= -\frac{1}{2}\left[\boldsymbol{\theta}^\top\boldsymbol{\Lambda}\boldsymbol{\theta} - \boldsymbol{z}^\top\boldsymbol{\theta} - \boldsymbol{\theta}^\top\boldsymbol{z}\right] + \text{const}
\end{aligned}$$

Hence,

$$\boxed{\breve{\nu}(\boldsymbol{\theta}) \propto \mathcal{N}(\boldsymbol{\Lambda}^{-1}\boldsymbol{z}, \boldsymbol{\Lambda}^{-1})}$$

where

$$\begin{aligned}
\boldsymbol{\Lambda} &= m_\gamma(\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{x}}^\top) \\
\boldsymbol{z} &= (\boldsymbol{V}_{\boldsymbol{x}\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{y}}^\top)\boldsymbol{e}_1 m_\gamma
\end{aligned}$$

A.7 Update of message to γ

$$\begin{aligned}
\log \breve{\nu}(\gamma) &= \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\log f(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\theta},\gamma) + \text{const} \\
&= \mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\log|\boldsymbol{W}|^{\frac{1}{2}} - \frac{1}{2}\left((\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x})^\top\boldsymbol{W}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x})\right)\right] + \text{const} \\
&= \log|\boldsymbol{W}|^{\frac{1}{2}} - \frac{1}{2}\operatorname{tr}\left(\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{y}\boldsymbol{y}^\top - \boldsymbol{A}\boldsymbol{x}\boldsymbol{y}^\top + \boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{A}^\top - \boldsymbol{y}\boldsymbol{x}^\top\boldsymbol{A}^\top\right]\right) \\
&\quad + \text{const}
\end{aligned}$$

First of all, let us work out the term $\log |\boldsymbol{W}|^{\frac{1}{2}}$

$$\begin{aligned}\log |\boldsymbol{W}|^{\frac{1}{2}} &= \frac{1}{2}\log \begin{vmatrix} \gamma & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{\epsilon} & 0 & \dots & \vdots \\ 0 & 0 & \frac{1}{\epsilon} & \dots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \end{vmatrix} \\ &= \frac{1}{2}\log\gamma + \frac{1}{2}(1-K)\log(\epsilon) = \log\gamma^{\frac{1}{2}} + \text{const}\end{aligned}$$

We split the expression under the expectation into four terms:
I: $\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{y}\boldsymbol{y}^\top\right]$
II: $\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{A}\boldsymbol{x}\boldsymbol{y}^\top\right]$
III: $\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{y}\boldsymbol{x}^\top\boldsymbol{A}^\top\right]$ and
IV: $\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{A}^\top\right]$

Term I:

$$\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{y}\boldsymbol{y}^\top\right] = \boldsymbol{W}\left(\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{y}}\boldsymbol{m}_{\boldsymbol{y}}^\top\right)$$

Recalling Eq. (A.6), term II:

$$\begin{aligned}\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{A}\boldsymbol{x}\boldsymbol{y}^\top\right] &= \boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left((\boldsymbol{S}+\boldsymbol{e}_1\boldsymbol{\theta}^\top)\boldsymbol{x}\boldsymbol{y}^\top\right) \\ &= \boldsymbol{W}\left(\boldsymbol{m}_{\boldsymbol{A}}(V_{\boldsymbol{x}\boldsymbol{y}^\top} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{y}}^\top)\right)\end{aligned}$$

Term III:

$$\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{y}\boldsymbol{x}^\top\boldsymbol{A}^\top\right] = \boldsymbol{W}\left((V_{\boldsymbol{y}\boldsymbol{x}^\top} + \boldsymbol{m}_{\boldsymbol{y}}\boldsymbol{m}_{\boldsymbol{x}}^\top)\boldsymbol{m}_{\boldsymbol{A}}^\top\right)$$

Term IV:

$$\begin{aligned}\boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{A}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{A}^\top\right] &= \boldsymbol{W}\,\mathbb{E}\left[(\boldsymbol{S}+\boldsymbol{e}_1\boldsymbol{\theta}^\top)\boldsymbol{x}\boldsymbol{x}^\top(\boldsymbol{S}+\boldsymbol{e}_1\boldsymbol{\theta}^\top)^\top\right] \\ &= \boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})}\left[\boldsymbol{S}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{e}_1\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{S}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{e}_1^\top + \boldsymbol{e}_1\boldsymbol{\theta}^\top\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{\theta}\boldsymbol{e}_1^\top\right] \\ &= \boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[\boldsymbol{S}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{e}_1\boldsymbol{m}_{\boldsymbol{\theta}}^\top\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{S}^\top + \boldsymbol{S}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{m}_{\boldsymbol{\theta}}\boldsymbol{e}_1^\top\right] \\ &\quad + \boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[\boldsymbol{e}_1(\boldsymbol{x}^\top\boldsymbol{V}_{\boldsymbol{\theta}}\boldsymbol{x} + \boldsymbol{m}_{\boldsymbol{\theta}}^\top\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{m}_{\boldsymbol{\theta}})\boldsymbol{e}_1^\top\right] \\ &= \boldsymbol{W}\,\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})}\left[\boldsymbol{m}_{\boldsymbol{A}}\boldsymbol{x}\boldsymbol{x}^\top\boldsymbol{m}_{\boldsymbol{A}}^\top + \boldsymbol{e}_1\boldsymbol{x}^\top\boldsymbol{V}_{\boldsymbol{\theta}}\boldsymbol{x}\boldsymbol{e}_1^\top\right] \\ &= \boldsymbol{W}\left[\boldsymbol{m}_{\boldsymbol{A}}(\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{x}}^\top)\boldsymbol{m}_{\boldsymbol{A}}^\top + \boldsymbol{e}_1(\text{tr}(\boldsymbol{V}_{\boldsymbol{\theta}}\boldsymbol{V}_{\boldsymbol{x}}) + \boldsymbol{m}_{\boldsymbol{x}}^\top\boldsymbol{V}_{\boldsymbol{\theta}}\boldsymbol{m}_{\boldsymbol{x}})\boldsymbol{e}_1^\top\right]\end{aligned}$$

As the resulting message should depend solely on γ we need to get rid of all terms which incorporate matrix $\boldsymbol{W}$. We notice that

$$\operatorname{tr}(\boldsymbol{W}\boldsymbol{\Sigma}) = \operatorname{tr}\left(\boldsymbol{\Sigma} \cdot \begin{pmatrix} \gamma & 0 & 0 & \dots & 0 \\ 0 & \frac{1}{\epsilon} & 0 & \dots & \vdots \\ 0 & 0 & \frac{1}{\epsilon} & \dots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \end{pmatrix}\right) = \boldsymbol{e}_1^\top \gamma \boldsymbol{\Sigma} \boldsymbol{e}_1 + \text{const}$$

where $\boldsymbol{\Sigma}$ is an arbitrary matrix of the same dimensionality as the matrix $\boldsymbol{W}$. In this way

$$\begin{aligned} \log \tilde{\nu}(\gamma) = \log \gamma^{\frac{1}{2}} &- \frac{\gamma}{2} \boldsymbol{e}_1^\top \left[\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{y}} \boldsymbol{m}_{\boldsymbol{y}}^\top - 2 \boldsymbol{m}_{\boldsymbol{A}} (V_{\boldsymbol{x}\boldsymbol{y}^\top} + \boldsymbol{m}_{\boldsymbol{x}} \boldsymbol{m}_{\boldsymbol{y}}^\top) \right] \boldsymbol{e}_1 \\ &- \frac{\gamma}{2} \boldsymbol{e}_1^\top \left[\boldsymbol{m}_{\boldsymbol{A}} (\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}} \boldsymbol{m}_{\boldsymbol{x}}^\top) \boldsymbol{m}_{\boldsymbol{A}}^\top + \operatorname{tr}(\boldsymbol{V}_{\boldsymbol{\theta}} \boldsymbol{V}_{\boldsymbol{x}}) + \boldsymbol{m}_{\boldsymbol{x}}^\top \boldsymbol{V}_{\boldsymbol{\theta}} \boldsymbol{m}_{\boldsymbol{x}}) \right] \boldsymbol{e}_1 \end{aligned}$$

After exponentiating $\log \tilde{\nu}(\gamma)$ it yields the gamma distribution:

$$\tilde{\nu}(\gamma) \propto \gamma^{\frac{1}{2}} \exp\left\{ -\frac{\gamma}{2} b \right\}$$

or

$$\boxed{\tilde{\nu}(\gamma) \propto \Gamma\left(\frac{3}{2}, \frac{b}{2}\right)}$$

where

$$\begin{aligned} b = &\left(\boldsymbol{V}_{\boldsymbol{y}} + \boldsymbol{m}_{\boldsymbol{y}} \boldsymbol{m}_{\boldsymbol{y}}^\top \right) - 2 \left(\boldsymbol{m}_{\boldsymbol{A}} (V_{\boldsymbol{x}\boldsymbol{y}^\top} + \boldsymbol{m}_{\boldsymbol{x}} \boldsymbol{m}_{\boldsymbol{y}}^\top) \right) \\ &+ \left(\boldsymbol{m}_{\boldsymbol{A}} (\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}} \boldsymbol{m}_{\boldsymbol{x}}^\top) \boldsymbol{m}_{\boldsymbol{A}}^\top \right) + \operatorname{tr}(\boldsymbol{V}_{\boldsymbol{\theta}} \left(\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}} \boldsymbol{m}_{\boldsymbol{x}}^\top \right)) \end{aligned}$$

A.8 Derivation of $q(\boldsymbol{x}, \boldsymbol{y})$

The joint variational distribution is given by

$$\begin{aligned} q(\boldsymbol{x}, \boldsymbol{y}) &\propto \vec{\nu}(\boldsymbol{x}) \tilde{f}(\boldsymbol{x}, \boldsymbol{y}) \tilde{\nu}(\boldsymbol{y}) \\ &= \mathcal{N}(\boldsymbol{x} | \boldsymbol{m}_{\boldsymbol{x}}, \boldsymbol{V}_{\boldsymbol{x}}) \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x}, \boldsymbol{m}_{\boldsymbol{W}}^{-1}) \mathcal{N}(\boldsymbol{x} | \boldsymbol{0}, (m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}})^{-1}) \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{y}}, \boldsymbol{V}_{\boldsymbol{y}}) \\ &= \mathcal{N}\left(\boldsymbol{x} | \boldsymbol{\Lambda}^{-1} \boldsymbol{z}, \boldsymbol{\Lambda}^{-1}\right) \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{y}}, \boldsymbol{V}_{\boldsymbol{y}}) \mathcal{N}(\boldsymbol{y} | \boldsymbol{m}_{\boldsymbol{A}} \boldsymbol{x}, \boldsymbol{m}_{\boldsymbol{W}}^{-1}) \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{\Lambda} &= \boldsymbol{V}_{\boldsymbol{x}}^{-1} + m_\gamma \boldsymbol{V}_{\boldsymbol{\theta}} \\ \boldsymbol{z} &= \boldsymbol{V}_{\boldsymbol{x}}^{-1} \boldsymbol{m}_{\boldsymbol{x}} \end{aligned}$$

$$q(\boldsymbol{x}, \boldsymbol{y}) \propto \mathcal{N}\left(\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{x}\end{bmatrix} \middle| \begin{bmatrix}\boldsymbol{m_y}\\ \boldsymbol{\Lambda}^{-1}\boldsymbol{z}\end{bmatrix}, \begin{bmatrix}\boldsymbol{V_y}^{-1} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Lambda}\end{bmatrix}^{-1}\right) \mathcal{N}(\boldsymbol{y}|\boldsymbol{m_A}\boldsymbol{x}, \boldsymbol{m_W}^{-1})$$

Let us rearrange the terms in the Gaussian $\mathcal{N}(\boldsymbol{y}|\boldsymbol{m_A}\boldsymbol{x}, \boldsymbol{m_W}^{-1})$

$$\begin{aligned}\mathcal{N}(\boldsymbol{y}|\boldsymbol{m_A}\boldsymbol{x}, \boldsymbol{m_W}^{-1}) &\propto \exp\left(-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{m_A}\boldsymbol{x})^\top \boldsymbol{m_W}(\boldsymbol{y}-\boldsymbol{m_A}\boldsymbol{x})\right)\\ &\propto \exp\left(-\frac{1}{2}\left[\boldsymbol{y}^\top\boldsymbol{m_W}\boldsymbol{y} - \boldsymbol{y}^\top\boldsymbol{m_W}\boldsymbol{m_A}\boldsymbol{x} + \boldsymbol{x}^\top\boldsymbol{m_A}^\top\boldsymbol{m_W}\boldsymbol{m_A}\boldsymbol{x} - \boldsymbol{x}^\top\boldsymbol{m_A}^\top\boldsymbol{m_W}\boldsymbol{y}\right]\right)\\ &\propto \mathcal{N}\left(\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{x}\end{bmatrix} \middle| \begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}, \begin{bmatrix}\boldsymbol{m_W} & -\boldsymbol{m_W}\boldsymbol{m_A}\\ -\boldsymbol{m_A}^\top\boldsymbol{m_W} & \boldsymbol{m_A}^\top\boldsymbol{m_W}\boldsymbol{m_A}\end{bmatrix}^{-1}\right)\end{aligned}$$

$$q(\boldsymbol{x}, \boldsymbol{y}) \propto \mathcal{N}\left(\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{x}\end{bmatrix} \middle| \begin{bmatrix}\boldsymbol{m_y}\\ \boldsymbol{\Lambda}^{-1}\boldsymbol{z}\end{bmatrix}, \begin{bmatrix}\boldsymbol{V_y}^{-1} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Lambda}\end{bmatrix}^{-1}\right) \tag{A.9a}$$

$$\cdot\, \mathcal{N}\left(\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{x}\end{bmatrix} \middle| \begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}, \begin{bmatrix}\boldsymbol{m_W} & -\boldsymbol{m_W}\boldsymbol{m_A}\\ -\boldsymbol{m_A}^\top\boldsymbol{m_W} & \boldsymbol{m_A}^\top\boldsymbol{m_W}\boldsymbol{m_A}\end{bmatrix}^{-1}\right) \tag{A.9b}$$

The final expression for the joint marginal is

$$\boxed{q(\boldsymbol{x}, \boldsymbol{y}) \propto \mathcal{N}\left(\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{x}\end{bmatrix} \middle| \hat{\boldsymbol{W}}^{-1}\begin{bmatrix}\boldsymbol{V_y}^{-1}\boldsymbol{m_y}\\ \boldsymbol{V_x}^{-1}\boldsymbol{m_x}\end{bmatrix}, \hat{\boldsymbol{W}}^{-1}\right)}$$

where

$$\hat{\boldsymbol{W}} = \begin{bmatrix}\boldsymbol{m_W} + \boldsymbol{V_y}^{-1} & -\boldsymbol{m_W}\boldsymbol{m_A}\\ -\boldsymbol{m_A}^\top\boldsymbol{m_W} & \boldsymbol{m_A}^\top\boldsymbol{m_W}\boldsymbol{m_A} + \boldsymbol{\Lambda}\end{bmatrix}$$

The precision matrix $\hat{\boldsymbol{W}}$, to put it mildly, is quite far from a nice shape as it contains "unpleasant" matrix $\boldsymbol{m_W}$ with ϵ^{-1} on the diagonal. Let us workout the covariance matrix $\hat{\boldsymbol{V}} = \hat{\boldsymbol{W}}^{-1}$. To do this, we recall two important matrix identities:

$$(\boldsymbol{A}+\boldsymbol{B})^{-1} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}(\boldsymbol{B}^{-1}+\boldsymbol{A}^{-1})^{-1}\boldsymbol{A}^{-1} \tag{A.10}$$

and

$$\begin{aligned}&\begin{bmatrix}\boldsymbol{A} & \boldsymbol{B}\\ \boldsymbol{C} & \boldsymbol{D}\end{bmatrix}^{-1}\\ &\quad= \begin{bmatrix}(\boldsymbol{A}-\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1} & -(\boldsymbol{A}-\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1}\boldsymbol{B}\boldsymbol{D}^{-1}\\ -\boldsymbol{D}^{-1}\boldsymbol{C}(\boldsymbol{A}-\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1} & \boldsymbol{D}^{-1}+\boldsymbol{D}^{-1}\boldsymbol{C}(\boldsymbol{A}-\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1}\boldsymbol{B}\boldsymbol{D}^{-1}\end{bmatrix}\end{aligned}$$

A

Let us denote the block elements of $\hat{\boldsymbol{W}}$ as follows:

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{m_W} + \boldsymbol{V_y}^{-1} \qquad \boldsymbol{B} = -\boldsymbol{m_W}\boldsymbol{m_A} \\ \boldsymbol{C} &= -\boldsymbol{m_A}^\top \boldsymbol{m_W} \qquad \boldsymbol{D} \underbrace{\boldsymbol{m_A}^\top \boldsymbol{m_W} \boldsymbol{m_A} + \boldsymbol{V_x}^{-1} + m_\gamma \boldsymbol{V_\theta}}_{\boldsymbol{D}^*} \end{aligned}$$

In this way,

$$(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1} = (\underbrace{\boldsymbol{m_W} + \boldsymbol{V_y}^{-1}}_{\boldsymbol{A}} - \boldsymbol{m_W}\boldsymbol{m_A}\boldsymbol{D}^{-*}\boldsymbol{m_A}^\top \boldsymbol{m_W})^{-1}$$

$$\boldsymbol{D} = \boldsymbol{A}^{-1} - \boldsymbol{A}^{-1}(\boldsymbol{A}^{-1} - (\boldsymbol{m_W}\boldsymbol{m_A}\boldsymbol{D}^{-*}\boldsymbol{m_A}^\top \boldsymbol{m_W})^{-1})^{-1}\boldsymbol{A}^{-1}$$

Let us work out the auxiliary terms

$$\begin{aligned} \boldsymbol{A}^{-1} = (\boldsymbol{m_W} + \boldsymbol{V_y}^{-1})^{-1} &= \boldsymbol{V_y} - \boldsymbol{V_y}(\boldsymbol{m_V} + \boldsymbol{V_y})^{-1}\boldsymbol{V_y} \\ &= \underbrace{\boldsymbol{m_V} - \boldsymbol{m_V}(\boldsymbol{V_y} + \boldsymbol{m_V})^{-1}\boldsymbol{m_V}}_{\boldsymbol{BD}} \end{aligned}$$

$$\begin{aligned} &(\boldsymbol{m_W}\boldsymbol{m_A}\boldsymbol{D}^{-*}\boldsymbol{m_A}^\top \boldsymbol{m_W})^{-1} = \boldsymbol{m_V}\boldsymbol{m_A}^{-\top} D^* \boldsymbol{m_A}^{-1}\boldsymbol{m_V} \\ &= \boldsymbol{m_V}\boldsymbol{m_A}^{-\top}(\boldsymbol{m_A}^\top \boldsymbol{m_W}\boldsymbol{m_A} + \boldsymbol{V_x}^{-1} + m_\gamma \boldsymbol{V_\theta})\boldsymbol{m_A}^{-1}\boldsymbol{m_V} \\ &= \underbrace{\boldsymbol{m_V} + \boldsymbol{m_V}\boldsymbol{m_A}^{-\top}(\boldsymbol{V_x}^{-1} + m_\gamma \boldsymbol{V_\theta})\boldsymbol{m_A}^{-1}\boldsymbol{m_V}}_{\boldsymbol{F}} \end{aligned}$$

Hence,

$$(\boldsymbol{A} - \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1} = \boldsymbol{E} - \boldsymbol{E}(\boldsymbol{F} + \boldsymbol{E})^{-1}\boldsymbol{E}$$

Next, let us consider $\boldsymbol{D}^{-1}$, $\boldsymbol{D}^{-1}\boldsymbol{C}$ and $\boldsymbol{B}\boldsymbol{D}^{-1}$:

$$\begin{aligned} \boldsymbol{D}^{-1} = \boldsymbol{D}^{-*} &= \left(\boldsymbol{m_A}^\top \boldsymbol{m_W}\boldsymbol{m_A} + (\boldsymbol{V_x}^{-1} + m_\gamma \boldsymbol{V_\theta})\right)^{-1} \\ &= \boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top} \\ &\quad - \boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top}\left[\boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top} + (\boldsymbol{V_x}^{-1} + m_\gamma \boldsymbol{V_\theta})^{-1}\right]^{-1}\boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top} \end{aligned}$$

$$\begin{aligned} \boldsymbol{D}^{-1}\boldsymbol{C} &= \boldsymbol{D}^{-1}(-\boldsymbol{m_A}^\top \boldsymbol{m_W}) \\ &= -\boldsymbol{m_A}^{-1} + \boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top}\left[\boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top} + (\boldsymbol{V_x}^{-1} + m_\gamma \boldsymbol{V_\theta})^{-1}\right]^{-1}\boldsymbol{m_A}^{-1} \end{aligned}$$

$$\begin{aligned} \boldsymbol{B}\boldsymbol{D}^{-1} &= (-\boldsymbol{m_W}\boldsymbol{m_A})\boldsymbol{D}^{-1} \\ &= -\boldsymbol{m_A}^{-\top} + \boldsymbol{m_A}^{-\top}\left[\boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top} + (\boldsymbol{V_x}^{-1} + m_\gamma \boldsymbol{V_\theta})^{-1}\right]^{-1}\boldsymbol{m_A}^{-1}\boldsymbol{m_V}\boldsymbol{m_A}^{-\top} \end{aligned}$$

Although the resulting expressions do not have a nice form, we got rid of "unpleasant" matrix $\boldsymbol{m_W}$.

A.9 Free energy derivations

In this section, we describe how to compute the variational free energy of AR node $f(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\theta}, \gamma)$. Note that essentially AR node implements the univariate Gaussian $f(y, \boldsymbol{x}, \boldsymbol{\theta}, \gamma) = \mathcal{N}\left(y \mid \boldsymbol{\theta}^\top \boldsymbol{x}, \gamma^{-1}\right)$ (Multivariate formulation is needed for bookkeeping previous states). The free energy functional is defined as

$$\begin{aligned} F[q] &\triangleq U[q] - H[q] \\ U[q] &\triangleq -\mathbb{E}_{q(\boldsymbol{x},y)q(\boldsymbol{\theta})q(\gamma)} \log f \\ H[q] &\triangleq -\mathbb{E}_{q(\boldsymbol{x},y)q(\boldsymbol{\theta})q(\gamma)} \log q \end{aligned}$$

At first, let us work out the entropy term $H[q]$.

$$\begin{aligned} H[q] &= -\mathbb{E}_{q(\boldsymbol{x},y)} \log q(\boldsymbol{x}, y) - \mathbb{E}_{q(\boldsymbol{\theta})} \log q(\boldsymbol{\theta}) - \mathbb{E}_{q(\gamma)} \log q(\gamma) \\ &= \frac{1}{2}\left(\log |2\pi e V_{\boldsymbol{x}y}| + \log |2\pi e \boldsymbol{V}_{\boldsymbol{\theta}}|\right) \\ &\quad - \alpha - \log \beta + \log \Gamma(\alpha) + (1-\alpha)\psi(\alpha) \end{aligned}$$

where $\psi(\alpha)$ denotes digamma function.
Now, let us consider the average energy $U[q]$

$$-\mathbb{E}_{q(\boldsymbol{x},y)q(\boldsymbol{\theta})q(\gamma)}\left[\log \frac{\gamma^{1/2}}{\sqrt{2\pi}} - \frac{\gamma}{2}(y - \boldsymbol{\theta}^\top \boldsymbol{x})^2\right]$$

We split the expression under the expectation into two terms:
I: $-\mathbb{E}_{q(\gamma)}\left[\log \frac{\gamma^{1/2}}{\sqrt{2\pi}}\right]$
II: $-\mathbb{E}_{q(\boldsymbol{x},\boldsymbol{y})q(\boldsymbol{\theta})q(\gamma)}\left[-\frac{\gamma}{2}(y - \boldsymbol{\theta}^\top \boldsymbol{x})^2\right]$
Term I:

$$-\mathbb{E}_{q(\gamma)}\left[\log \frac{\gamma^{1/2}}{\sqrt{2\pi}}\right] = -\mathbb{E}_{q(\gamma)}\left[\frac{1}{2}\log\gamma - \frac{1}{2}\log 2\pi\right] = -\frac{1}{2}\left[\psi(\alpha) - \log\beta\right] + \frac{1}{2}\log 2\pi$$

Term II:

$$\begin{aligned} -\mathbb{E}_{q(\boldsymbol{x},y)q(\boldsymbol{\theta})q(\gamma)}\left[-\frac{\gamma}{2}(y - \boldsymbol{\theta}^\top \boldsymbol{x})^2\right] &= \frac{m_\gamma}{2}\mathbb{E}_{q(\boldsymbol{x},y)q(\boldsymbol{\theta})}\left[(y - \boldsymbol{\theta}^\top \boldsymbol{x})^2\right] \\ &= \frac{m_\gamma}{2}\mathbb{E}_{q(\boldsymbol{x},y)q(\boldsymbol{\theta})}\left[y^2 - 2y\boldsymbol{\theta}^\top \boldsymbol{x} + \boldsymbol{\theta}^\top \boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{\theta}\right] \\ &= \frac{m_\gamma}{2}\left[\sigma_y^2 + m_y^2 - 2\left[V_{y\boldsymbol{x}^\top} + m_y \boldsymbol{m}_{\boldsymbol{x}}^\top\right]\boldsymbol{m}_{\boldsymbol{\theta}} + \mathrm{tr}\left[(\boldsymbol{V}_{\boldsymbol{\theta}} + \boldsymbol{m}_{\boldsymbol{\theta}}\boldsymbol{m}_{\boldsymbol{\theta}}^\top)\boldsymbol{V}_{\boldsymbol{x}}\right]\right] \\ &\quad + \frac{m_\gamma}{2}\left[\boldsymbol{m}_{\boldsymbol{\theta}}^\top(\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{x}}^\top)\boldsymbol{m}_{\boldsymbol{\theta}}\right] \end{aligned}$$

hence

$$U[q] = -\frac{1}{2}\left[\psi(\alpha) - \log\beta\right] + \frac{1}{2}\log 2\pi + \frac{m_\gamma}{2}d$$

where

$$d = \sigma_y^2 + m_y^2 - 2\left[V_{y\boldsymbol{x}^\top} + m_y\boldsymbol{m}_{\boldsymbol{x}}^\top\right]\boldsymbol{m}_{\boldsymbol{\theta}} + \mathrm{tr}\left[(\boldsymbol{V}_{\boldsymbol{\theta}} + \boldsymbol{m}_{\boldsymbol{\theta}}\boldsymbol{m}_{\boldsymbol{\theta}}^\top)\boldsymbol{V}_{\boldsymbol{x}}\right] + \boldsymbol{m}_{\boldsymbol{\theta}}^\top(\boldsymbol{V}_{\boldsymbol{x}} + \boldsymbol{m}_{\boldsymbol{x}}\boldsymbol{m}_{\boldsymbol{x}}^\top)\boldsymbol{m}_{\boldsymbol{\theta}}$$

Appendix B

Message Passing-based Inference in Gamma-Mixture Models

B.1 Gamma Mixture node

The likelihood function of the Gamma mixture node is specified by

$$f(x_t, \boldsymbol{s}_t, \boldsymbol{a}, \boldsymbol{b}) = p(x_t|\boldsymbol{s}_t, \boldsymbol{a}, \boldsymbol{b}) = \prod_{l=1}^{L} \Gamma(x_t|a_l, b_l)^{s_{tl}}, \tag{B.1}$$

where $\Gamma(x_t|a_l, b_l)$ specifies the Gamma distribution for x_t with shape and rate parameters a_l and b_l, respectively. $\boldsymbol{a} \triangleq [a_1, \ldots, a_M]$ and $\boldsymbol{b} \triangleq [b_1, \ldots, b_M]$ are vectors of the parameters of the Gamma distributions such that $a_l, b_l \in \mathbb{R}_{>0}$ for every $m = 1, \ldots, M$. For each observation x_t we have a corresponding latent selector variable $\boldsymbol{s}_t$ comprising a 1-of-M binary vector with elements $s_{tl} \in \{0, 1\}$, which are constrained by $\sum_l s_{tl} = 1$.

We assume a mean-field factorization around the Gamma mixture node as

$$q(x_t, \boldsymbol{s}_t, \boldsymbol{a}, \boldsymbol{b}) = q(x_t)q(\boldsymbol{s}_t)q(\boldsymbol{a})q(\boldsymbol{b}) \tag{B.2}$$

where $q(\boldsymbol{a}) = \prod_{l=1}^{L} q(a_l)$ and $q(\boldsymbol{b}) = \prod_{l=1}^{L} q(b_l)$.

We assume the following functional forms for the approximate posterior marginals:

$$\begin{aligned}
q(x_t) &= \Gamma(x_t \mid \hat{\alpha}_t^{(x)}, \hat{\beta}_t^{(x)}) \quad \hat{\alpha}_t^{(x)},\ \hat{\beta}_t^{(x)} \in \mathbb{R}_{>0} \\
q(\boldsymbol{s}_t) &= \prod_{l=1}^{L} \hat{\pi}_l^{s_{tl}} \text{ such that } \sum_{m=1}^{M} \hat{\pi}_l = 1 \\
q(a_l) &= \delta(a_l - \hat{a}_l) \text{ or } q(a_l) = \Gamma(a_l \mid \hat{\alpha}_l^{(a)}, \hat{\beta}_l^{(a)}) \quad \hat{\alpha}_t^{(a)},\ \hat{\beta}_t^{(a)} \in \mathbb{R}_{>0} \\
q(b_l) &= \Gamma(b_l \mid \hat{\alpha}_l^{(b)}, \hat{\beta}_l^{(b)}) \quad \hat{\alpha}_t^{(b)},\ \hat{\beta}_t^{(b)} \in \mathbb{R}_{>0}
\end{aligned}$$

Here the marginal $q(a_l)$ has two forms, depending on the inference methodology. For expectation-maximization the marginal follows a Dirac delta function and for moment matching the marginal follows a Gamma function. As a result, we will not express the expectations related to this distributions in terms of the corresponding parameters.

B.2 Mathematical identities

In this section we will derive some relatively common expectations to simplify derivations later on. We use $C \in \mathbb{R}$ to denote a constant. Consider the following expectation where $q(x) = \Gamma(x \mid \alpha, \beta)$:

$$\begin{aligned}
\mathrm{E}_{q(x)}\left[\ln \Gamma(x)\right] &= \mathrm{E}_{q(x)}\left[-\ln(x) + \ln \Gamma(x+1)\right] \quad (\Gamma(x+1) = x\Gamma(x)) \\
&\approx -\mathrm{E}_{q(x)}\left[\ln(x)\right] + \mathrm{E}_{q(x)}\left[\ln\left(\sqrt{2\pi x}\left(\frac{x}{e}\right)^x\right)\right] \quad \textit{Stirling's approximation} \\
&= -\mathrm{E}_{q(x)}\left[\ln(x)\right] + \mathrm{E}_{q(x)}\left[\frac{1}{2}\ln(2\pi x) + x(\ln(x) - 1)\right] \\
&= \frac{1}{2}\ln(2\pi) - \frac{1}{2}\mathrm{E}_{q(x)}\left[\ln(x)\right] + \mathrm{E}_{q(x)}\left[x\ln(x)\right] - \mathrm{E}_{q(x)}\left[x\right] \\
&= \frac{1}{2}\ln(2\pi) - \frac{1}{2}(\psi(\alpha) - \ln(\beta)) - \frac{\alpha}{\beta} + \mathrm{E}_{q(x)}\left[x\ln(x)\right]
\end{aligned} \tag{B.3}$$

where Stirling's approximation approximates the Γ-function (especially well for $x \geq 1$). The term $\mathrm{E}_{q(x)}[x\ln(x)]$ can be determined as

$$\begin{aligned}
\mathrm{E}_{q(x)}[x\ln(x)] &= \int_0^\infty q(x) x \ln(x) \mathrm{d}x \\
&= \int_0^\infty \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} x \ln(x) \mathrm{d}x \\
&= \frac{\Gamma(\alpha+1)}{\Gamma(\alpha)\beta} \int_0^\infty \frac{\beta^{(\alpha+1)}}{\Gamma(\alpha)} x^{(\alpha+1)-1} e^{-\beta x} \ln(x) \mathrm{d}x \\
&= \frac{\alpha}{\beta} \mathrm{E}_{x\sim\Gamma(\alpha+1,\beta)}\left[\ln(x)\right] \\
&= \frac{\alpha}{\beta}(\psi(\alpha+1) - \ln(\beta))
\end{aligned} \tag{B.4}$$

Concluding

$$\mathrm{E}_{q(x)}\left[\ln \Gamma(x)\right] = \frac{1}{2}\ln(2\pi) - \frac{1}{2}(\psi(\alpha) - \ln(\beta)) + \frac{\alpha}{\beta}\left(-1 + \psi(\alpha+1) - \ln(\beta)\right) \tag{B.5}$$

Here $\psi(\cdot)$ denotes the digamma function.

B.3 Message $\vec{\nu}(x_t)$

The message $\ln \vec{\nu}(x_t)$ can be determined

$$\begin{aligned}
\ln \vec{\nu}(x_t) &= \mathrm{E}_{q(\boldsymbol{s}_t)q(\boldsymbol{a})q(\boldsymbol{b})}\left[\ln\left(\prod_{l=1}^{L}\Gamma(x_t|a_l,b_l)^{s_{tl}}\right)\right]+C \\
&= \mathrm{E}_{q(\boldsymbol{s}_t)q(\boldsymbol{a})q(\boldsymbol{b})}\left[\sum_{m=1}^{M} s_{tl}\ln(\Gamma(x_t \mid a_l,b_l))\right]+C \\
&= \sum_{m=1}^{M}\mathrm{E}_{q(s_{tl})}\left[s_{tl}\right]\mathrm{E}_{q(a_l)q(b_l)}\left[\ln(\Gamma(x_t \mid a_l,b_l))\right]+C \\
&= \sum_{m=1}^{M}\hat{\pi}_l\mathrm{E}_{q(a_l)q(b_l)}\left[\ln\left(\frac{b_l^{a_l}}{\Gamma(a_l)}x_t^{a_l-1}e^{-b_l x_t}\right)\right]+C \\
&= \sum_{m=1}^{M}\hat{\pi}_l\mathrm{E}_{q(a_l)q(b_l)}\left[-\ln(\Gamma(a_l))+a_l\ln(b_l)+(a_l-1)\ln(x_t)-b_l x_t\right]+C \\
&= \sum_{m=1}^{M}\hat{\pi}_l\left(\mathrm{E}_{q(a_l)q(b_l)}\left[(a_l-1)\ln(x_t)-b_l x_t\right]\right) \\
&= \sum_{m=1}^{M}\hat{\pi}_l\left(\left(\mathrm{E}_{q(a_l)}\left[a_l\right]-1\right)\ln(x_t)-\mathrm{E}_{q(b_l)}\left[b_l\right]x_t\right)+C \\
&= \sum_{m=1}^{M}\hat{\pi}_l\left(\left(\mathrm{E}_{q(a_l)}\left[a_l\right]-1\right)\ln(x_t)-\frac{\hat{\alpha}_l^{(b)}}{\hat{\beta}_l^{(b)}}x_t\right)+C \\
&= \left(\sum_{m=1}^{M}\hat{\pi}_l\mathrm{E}_{q(a_l)}\left[a_l\right]-1\right)\ln(x_t)-\left(\sum_{m=1}^{M}\hat{\pi}_l\frac{\hat{\alpha}_l^{(b)}}{\hat{\beta}_l^{(b)}}\right)x_t+C
\end{aligned} \tag{B.6}$$

From this, the variational message $\vec{\nu}(x_t)$ can be determined as

$$\boxed{\vec{\nu}(x_t) \;\propto \Gamma\left(x_t\middle|\;\sum_{m=1}^{M}\hat{\pi}_l\mathrm{E}_{q(a_l)}\left[a_l\right],\;\sum_{m=1}^{M}\hat{\pi}_l\frac{\hat{\alpha}_l^{(b)}}{\hat{\beta}_l^{(b)}}\right)} \tag{B.7}$$

B.4 Message $\overleftarrow{\nu}(s_t)$

The message $\ln \overleftarrow{\nu}(\boldsymbol{s}_t)$ can be determined as

$$\begin{aligned}
\ln \overleftarrow{\nu}(\boldsymbol{s}_t) &= \mathrm{E}_{q(\boldsymbol{a})q(\boldsymbol{b})q(x_t)} \left[\ln \left(\prod_{l=1}^{L} \Gamma(x_t|a_l, b_l)^{s_{tl}} \right) \right] + C \\
&= \mathrm{E}_{q(\boldsymbol{a})q(\boldsymbol{b})q(x_t)} \left[\sum_{m=1}^{M} s_{tl} \ln \left(\Gamma(x_t|a_l, b_l) \right) \right] + C \\
&= \mathrm{E}_{q(\boldsymbol{a})q(\boldsymbol{b})q(x_t)} \left[\sum_{m=1}^{M} s_{tl} \ln \left(\frac{b_l^{a_l} x_t^{a_l - 1}}{\Gamma(a_l)} \exp\left(-b_l x_t\right) \right) \right] + C \\
&= \sum_{m=1}^{M} s_{tl} \mathrm{E}_{q(\boldsymbol{a})q(\boldsymbol{b})q(x_t)} \left[a_l \ln b_l + (a_l - 1) \ln x_t - \ln \Gamma(a_l) - b_l x_t \right] + C \\
&= \sum_{m=1}^{M} s_{tl} \left(\mathrm{E}_{q(a_l)q(b_l)}[a_l \ln b_l] + \mathrm{E}_{q(a_l)q(x_t)}(a_l - 1) \ln x_t \right) \\
&\quad + \sum_{m=1}^{M} s_{tl} \left(-\mathrm{E}_{q(a_l)} \left[\ln \Gamma(a_l) \right] - \mathrm{E}_{q(x_t)q(b_l)}[b_l x_t] \right) + C \\
&= \sum_{m=1}^{M} s_{tl} \ln \rho_{tl} + C
\end{aligned}$$

From this, the variational message $\overleftarrow{\nu}(\boldsymbol{s}_t)$ can be determined as

$$\boxed{\overleftarrow{\nu}(\boldsymbol{s}_t) \propto \exp \sum_{m=1}^{M} s_{tl} \ln \rho_{tl} = \prod_{m=1}^{M} \rho_{tl}^{s_{tl}}}$$

where

$$\begin{aligned}
\rho_{tl} = &\exp \left\{ \mathrm{E}_{q(a_l)q(b_l)}[a_l \ln b_l] + \mathrm{E}_{q(a_l)q(x_t)}(a_l - 1) \ln x_t \right\} \\
&\cdot \exp \left\{ -\mathrm{E}_{q(a_l)} \left[\ln \Gamma(a_l) \right] + \mathrm{E}_{q(x_t)q(b_l)}[b_l x_t] \right\}
\end{aligned}$$

In order to ensure that the message will be a proper distribution, the event probabilities have to sum to 1. Hence, all event probabilities are normalized and the message becomes:

$$\overleftarrow{\nu}(\boldsymbol{s}_t) = \prod_{m=1}^{M} \left(\frac{\rho_{tl}}{\sum_l \rho_{tl}} \right)^{s_{tl}} \tag{B.8}$$

The individual expectations of ρ_{tl} can be calculated as

$$\mathrm{E}_{q(a_l)q(b_l)}\left[a_l \ln b_l\right] = \mathrm{E}_{q(a_l)}[a_l]\left(\psi(\hat{\alpha}_l^{(b)}) - \ln(\hat{\beta}_l^{(b)})\right) \tag{B.9a}$$

$$\mathrm{E}_{q(a_l)q(x_t)}[(a_l - 1)\ln x_t] = \left(\mathrm{E}_{q(a_l)}[a_l] - 1\right)\left(\psi(\hat{\alpha}_l^{(x)}) - \ln(\hat{\beta}_l^{(x)})\right) \tag{B.9b}$$

$$\mathrm{E}_{q(x_t)q(b_l)}[b_l x_t] = \frac{\hat{\alpha}_l^{(b)}\hat{\alpha}_t^{(x)}}{\hat{\beta}_l^{(b)}\hat{\beta}_t^{(x)}} \tag{B.9c}$$

The expectation $\mathrm{E}_{q(a_l)}[\ln\Gamma(a_l)]$ has been derived in Section B.2.

B.5 Message $\overleftarrow{\nu}(b_l)$

The message $\overleftarrow{\nu}(b_l)$ can be determined as

$$\begin{aligned}
\ln\overleftarrow{\nu}(b_l) &= \mathrm{E}_{\setminus q(b_l)}\left[\ln\left(\prod_{l=1}^{L}\Gamma(x_t|a_l,b_l)^{s_{tl}}\right)\right] + C \\
&= \mathrm{E}_{\setminus q(b_l)}\left[\sum_{m=1}^{M} s_{tl}\ln(\Gamma(x_t \mid a_l,b_l))\right] + C \\
&= \mathrm{E}_{q(\boldsymbol{s}_t)}\left[s_{tl}\right]\mathrm{E}_{q(a_l)q(x_t)}\left[\ln\left(\Gamma(x_t \mid a_l,b_l)\right)\right] + C \\
&= \hat{\pi}_l\mathrm{E}_{q(a_l)q(x_t)}\left[-\ln(\Gamma(a_l)) + a_l\ln(b_l) + (a_l-1)\ln(x_t) - b_l x_t\right] + C \\
&= \hat{\pi}_l\left(\mathrm{E}_{q(a_l)q(x_t)}\left[a_l\ln(b_l) - b_l x_t\right]\right) + C \\
&= \hat{\pi}_l\left(\mathrm{E}_{q(a_l)}\left[a_l\right]\ln(b_l) - b_l\mathrm{E}_{q(x_t)}\left[x_t\right]\right) + C \\
&= \hat{\pi}_l\left(\mathrm{E}_{q(a_l)}\left[a_l\right]\ln(b_l) - \frac{\hat{\alpha}_t^{(x)}}{\hat{\beta}_t^{(x)}}b_l\right) + C \\
&= \left(\hat{\pi}_l\mathrm{E}_{q(a_l)}\left[a_l\right]\right)\ln(b_l) - \left(\hat{\pi}_l\frac{\hat{\alpha}_t^{(x)}}{\hat{\beta}_t^{(x)}}\right)b_l + C
\end{aligned} \tag{B.10}$$

$$\boxed{\overleftarrow{\nu}(b_l) \propto \Gamma\left(b_l \middle| 1 + \hat{\pi}_l\mathrm{E}_{q(a_l)}\left[a_l\right],\ \hat{\pi}_l\frac{\hat{\alpha}_t^{(x)}}{\hat{\beta}_t^{(x)}}\right)} \tag{B.11}$$

B.6 Message $\overleftarrow{\nu}(a_l)$

$$\begin{aligned}
\ln \overleftarrow{\nu}(a_l) &= \mathrm{E}_{\setminus q(a_l)} \ln \left[\prod_{l=1}^{L} \Gamma(x_t|a_l, b_l)^{s_{tl}} \right] + C \\
&= \mathrm{E}_{\setminus q(a_l)} \left[\sum_{m=1}^{M} s_{tl} \ln \left(\frac{b_l^{a_l} x^{a_l - 1}}{\Gamma(a_l)} \exp\left(-b_l x\right) \right) \right] + C \\
&= \mathrm{E}_{\setminus q(a_l)} \left[\sum_{m=1}^{M} s_{tl} \left(a_l \ln(b_l) + (a_l - 1) \ln(x_t) - \ln(\Gamma(a_l)) - b_l x_t \right) \right] + C \\
&= \hat{\pi}_l \left[a_l \mathrm{E}_{q(b_l)}[\ln(b_l)] + a_l \mathrm{E}_{q(x_t)}[\ln(x_t)] - \ln(\Gamma(a_l)) \right] + C \\
&= \hat{\pi}_l \left[a_l \underbrace{\left(\psi(\hat{\alpha}_l^{(b)}) - \ln(\hat{\beta}_l^{(b)}) + \psi(\alpha_t^{(x)}) - \ln(\beta_t^{(x)}) \right)}_{\zeta_{tl}} - \ln(\Gamma(a_l)) \right] + C \\
&= \hat{\pi}_l \left(a_l \zeta_{tl} - \ln \Gamma(a_l) \right) + C
\end{aligned}$$

From this the variational message $\overleftarrow{\nu}(a_l)$ can be determined as

$$\boxed{\overleftarrow{\nu}(a_l) \propto \exp\left(\hat{\pi}_l \left(a_l \zeta_{tl} - \ln \Gamma(a_l) \right) \right)}$$

B.7 Local variational free energy

The local variational free energy of the Gamma mixture node can be computed as follows:

$$F[q] = \underbrace{-\mathrm{E}_{q(x_t)q(\boldsymbol{a})q(\boldsymbol{b})q(\boldsymbol{s}_t)} \Big[\ln(p(x_t|\boldsymbol{s}_t, \boldsymbol{a}, \boldsymbol{b})) \Big]}_{\text{Average energy}} + \underbrace{\mathrm{E}_{q(x_t)q(\boldsymbol{a})q(\boldsymbol{b})q(\boldsymbol{s}_t)} \Big[\ln(q(x_t)q(\boldsymbol{s}_t)q(\boldsymbol{a})q(\boldsymbol{b}) \Big])}_{\text{-Entropy}}$$

Since the entropy of the incoming marginals can easily be computed, let us focus on the average energy term

$$\begin{aligned}
&\mathrm{E}_{q(x_t)q(\boldsymbol{a})q(\boldsymbol{b})q(\boldsymbol{s}_t)} \Big[\ln(p(x_t|\boldsymbol{s}_t, \boldsymbol{a}, \boldsymbol{b})) \Big] = \mathrm{E}_{q(x_t)q(\boldsymbol{a})q(\boldsymbol{b})q(\boldsymbol{s}_t)} \left[\ln \left(\prod_{l=1}^{L} \Gamma(x_t|a_l, b_l)^{s_{tl}} \right) \right] \\
&= \mathrm{E}_{q(x_t)q(\boldsymbol{a})q(\boldsymbol{b})q(\boldsymbol{s}_t)} \left[\sum_{m=1}^{M} s_{tl} \ln \left(\Gamma(x_t|a_l, b_l) \right) \right] \\
&= \sum_{m=1}^{M} \hat{\pi}_l \mathrm{E}_{q(\boldsymbol{a})q(\boldsymbol{b})q(x_t)} \left[a_l \ln b_l + (a_l - 1) \ln(x_t) - \ln(\Gamma(a_l)) - b_l x_t \right]
\end{aligned}$$

B

The required expectations are given in (B.3) and(B.9).

Appendix C

AIDA: An Active Inference-based Design Agent for Audio Processing Algorithms

C.1 Bethe free energy

The Bethe assumption

$$q(\boldsymbol{z}) = \prod_{a \in \mathcal{V}} q_a(\boldsymbol{z}_a) \prod_{i \in \mathcal{E}} q_i(z_i)^{-1}. \tag{C.1}$$

is a useful constraint on the approximate posterior $q(\boldsymbol{z})$, [168].

Here we made use of the fact that all edges in the FFG have a maximum degree of two, which can be strictly enforced by adding uninformative priors $p(z_i) = 1$ to dangling edges. Under the Bethe assumption, the VFE reduces to the Bethe free energy (BFE)

$$F_B[q, f] = -\sum_{a \in \mathcal{V}} \mathbb{E}_{q(\boldsymbol{z}_a)} \left[\ln f_a(\boldsymbol{z}_a)\right] - \sum_{a \in \mathcal{V}} \mathrm{H}[q_a(\boldsymbol{z}_a)] + \sum_{i \in \mathcal{E}} \mathrm{H}[q_i(z_i)], \tag{C.2}$$

which equals the VFE for acyclic graphs (i.e. trees). The BFE decomposes the VFE into a sum of node-local free energies contributions and edge-specific entropies H.

C.1.1 Variational and hybrid message passing

Under the variational approximation we can employ variational inference in the model, which iteratively finds stationary points on the BFE by fixing all approximate posterior distributions besides the one that is being optimized. This inference

procedure can be cast to a message passing paradigm and is called variational message passing [77]. Here the exact message update rule of reduces to the variational message update rule [77]

$$\vec{\nu}(z_i) \propto \exp\left\{\mathbb{E}_{q(\boldsymbol{z}_{a\setminus i})}\left[\ln f_a(\boldsymbol{z}_a)\right]\right\} \tag{C.3}$$

where $\vec{\nu}(z_i)$ denotes the outgoing variational message on edge z_i. The approximate marginal distributions are then iteratively updated as

$$q_i(z_i) \propto \vec{\nu}(z_i) \cdot \overleftarrow{\nu}(z_i). \tag{C.4}$$

The calculations of variational messages and approximate marginal distributions are then iteratively repeated until convergence of the VFE is reached.

In addition to the structure imposed by the Bethe approximation, additional constraints can be enforced. Depending on these local constraints different inference algorithms naturally follow [43]. [43] shows that amongst others the sum-product algorithm [97, 96], variational message passing [77] and expectation propagation [169] can be recovered. By combining different local constraints we can achieve hybrid message passing-based inference in the probabilistic model. We highly recommend the interested reader the work of [43] for an extensive overview of hybrid message passing schemes.

C.2 Probabilistic model overview

This appendix gives a concise overview of the generative model of the acoustic model and AIDA. The prior distributions are uninformative unless stated otherwise in Section 5.5.

C.2.1 Acoustic model

The observed signal x_t is the sum of a speech and noise signal as

$$x_t = s_t + n_t$$

The speech signal $s_t = \boldsymbol{e}_1^\intercal \boldsymbol{s}_t$ is modeled by a time-varying autoregressive process as

$$\boldsymbol{s}_t \sim \mathcal{N}\left(A(\boldsymbol{\theta}_t)\boldsymbol{s}_{t-1},\ V\left(\gamma\right)\right)$$

The autoregressive coefficients of the speech signal are time-varying as

$$\boldsymbol{\theta}_t \sim \mathcal{N}\left(\boldsymbol{\theta}_{t-1},\ \omega \mathrm{I}_M\right)$$

The noise signal $n_t = \boldsymbol{e}_1^\intercal \boldsymbol{n}_t$ is also modeled by an autoregressive process

$$\boldsymbol{n}_t \sim \mathcal{N}\left(A(\boldsymbol{\varrho}_i)\boldsymbol{n}_{t-1},\ V\left(\tau_i\right)\right)$$

where $t = t^-, t^- + 1, \ldots, t^+$

The parameters of the noise model depend on the context

$$\boldsymbol{\varrho}_i \sim \prod_{l=1}^{L} \mathcal{N}\left(\boldsymbol{m}_l, \boldsymbol{V}_l\right)^{\boldsymbol{c}_i}$$
$$\tau_i \sim \prod_{l=1}^{L} \Gamma\left(a_l, b_l\right)^{\boldsymbol{c}_i}$$

The context $\boldsymbol{c}_i$ evolves over a different time scale indexed by k as

$$\boldsymbol{c}_i \sim \mathrm{Cat}(\mathrm{T}\boldsymbol{c}_{i-1})$$

The transition matrix of the context is modeled as

$$\mathrm{T}_{1:L,j} \sim \mathrm{Dir}(\boldsymbol{\alpha}_j)$$

Finally, the output of the hearing aid algorithm y_t is formed as the weighted sum of the speech and noise signals as

$$y_t = u_{sk} s_t + u_{nk} n_t$$

where $t = t^-, t^- + 1, \ldots, t^+$

C.2.2 AIDA's user response model

The user responses are modeled by a Bernoulli distribution containing a Gaussian cumulative probability distribution that enforces the output $\upsilon_i(\boldsymbol{u}_i)$ to the allowed domain for the argument of the Bernoulli distribution

$$r_i \sim \mathrm{Ber}(\Phi(\upsilon_i(\boldsymbol{u}_i))) \qquad \text{if } r_i \in \{0, 1\}$$

$\upsilon_i(\boldsymbol{u}_i)$ encodes our beliefs about the user response function (evaluated at $\boldsymbol{u}_i$), modeled by a mixture of Gaussian processes as

$$\upsilon_i \sim \prod_{l=1}^{L} \mathrm{GP}(\mathrm{m}_l(\cdot), \mathcal{K}_l(\cdot,\cdot))^{c_{li}}$$

whose kernel function is defined as

$$\mathcal{K}(\boldsymbol{u}, \boldsymbol{u}') = \sigma^2 \exp\left\{-\frac{\|\boldsymbol{u}-\boldsymbol{u}'\|_2^2}{2l^2}\right\}$$

where σ denotes noise and l the length scale of the kernel.

C.3 Inference realization

This appendix describes in detail how the inference tasks of Sections 5.4.1 and 5.4.2 are realized. The inference task of Section 5.4.3 is performed by automated message passing using the update rules of [111].

C.3.1 Realization of inference for context classification

The inference task for context classification of (5.13) renders intractable as discussed in Section (5.4.1). To circumvent this problem, we will solve this task as a Bayesian model comparison task.

In a Bayesian model comparison task, we are interested in calculating the posterior probability $p(\mathrm{m}_l \mid \boldsymbol{x})$ of some model m_l after observing data $\boldsymbol{x}$.

The posterior model probability $p(\mathrm{m}_l \mid \boldsymbol{x})$ can be calculated using Bayes' rule as

$$p(\mathrm{m}_l \mid \boldsymbol{x}) = \frac{p(\boldsymbol{x} \mid \mathrm{m}_l)p(\mathrm{m}_l)}{\sum_j p(\boldsymbol{x} \mid m_j)p(m_j)}, \tag{C.7}$$

where the denominator represents the weighted model evidence $p(\boldsymbol{x})$, i.e. the model evidence obtained for the individual models $p(\boldsymbol{x} \mid \mathrm{m}_l)$, weighted by their priors $p(\mathrm{m}_l)$.

To formulate our inference task as a Bayesian model comparison task, the distinct models m_l first have to be specified. In order to do so, we first note that we obtain the priors of $\boldsymbol{c}_{i-1}$ and z_{t^--1} in (5.13) separately, and therefore we implicitly assume a factorization of our prior $p(\boldsymbol{c}_{i-1}, z_{t^--1} \mid \boldsymbol{x}_{1:t^--1})$ as

$$p(\boldsymbol{c}_{i-1}, z_{t^--1} \mid \boldsymbol{x}_{1:t^--1}) = p(\boldsymbol{c}_{i-1} \mid \boldsymbol{x}_{t^--1})\; p(z_{t^--1} \mid \boldsymbol{x}_{1:t^--1}). \tag{C.8}$$

As a result (5.13) can be rewritten as

$$p(\boldsymbol{c}_i \mid \boldsymbol{x}_{1:t^+}) \propto \underbrace{\int p(\boldsymbol{c}_i, \mathrm{T} \mid \boldsymbol{c}_{i-1})\, p(\boldsymbol{c}_{i-1} \mid \boldsymbol{x}_{1:t^--1})\, \mathrm{dT}\, \mathrm{d}\boldsymbol{c}_{i-1}}_{\vec{\mu}(\boldsymbol{c}_i)} \cdot \underbrace{\int p(z_{t^-:t^+}, \Psi_i, \boldsymbol{x}_{t^-:t^+} \mid z_{t^--1}, \boldsymbol{c})\, p(z_{t^--1} \mid \boldsymbol{x}_{1:t^--1})\, \mathrm{d}z_{t^--1:t^+}\, \mathrm{d}\Psi_i}_{p(\boldsymbol{x}_{t^-:t^+} \mid \boldsymbol{x}_{1:t^--1}, \boldsymbol{c}_i)}. \tag{C.9}$$

The first term $\vec{\mu}(\boldsymbol{c}_i)$ can be regarded as the forward message towards the context $\boldsymbol{c}_i$ originating from the previous context. It gives us an estimate of the new context solely based on the context dynamics as stipulated by the transition matrix T. The second term $p(\boldsymbol{x}_{t^-:t^+} \mid \boldsymbol{x}_{1:t^--1}, \boldsymbol{c}_i)$ can be regarded as the incremental model evidence under some given context $\boldsymbol{c}_i$. Comparison of (C.9) and (C.7) allows us to formulate our inference problem in (5.13) into a Bayesian model comparison problem by defining

$$p(\mathrm{m}_l) = \vec{\mu}(\boldsymbol{c}_i = \boldsymbol{e}_l), \tag{C.10a}$$

$$p(\boldsymbol{x} \mid \mathrm{m}_l) = p(\boldsymbol{x}_{t^-:t^+} \mid \boldsymbol{x}_{1:t^--1}, \boldsymbol{c}_i = \boldsymbol{e}_l). \tag{C.10b}$$

We can therefore define a model m_l by clamping the context variable in generative model as $\boldsymbol{c}_i = \boldsymbol{e}_l$. This means that each model only has one active component for both the Gaussian and Gamma mixture nodes and therefore the messages originating from these nodes are exact and do not require a variational approximation.

Despite the expansion of the mixture models, the incremental model evidence $p(\boldsymbol{x}_{t^-:t^+} \mid \boldsymbol{x}_{1:t^--1}, \boldsymbol{c}_i = \boldsymbol{e}_l)$ cannot be computed exactly as the autoregressive source models lead to intractable inference. As a result, we approximate the model evidence in (C.10b) using the Bethe free energy, as defined in (C.2) in Section C.1, as

$$p(\boldsymbol{x} \mid \mathrm{m}_l) \approx \exp\{-F_B[q, \mathrm{m}_l]\}, \tag{C.11}$$

where $F_B[q, \mathrm{m}_l]$ denotes the Bethe free energy observed after convergence of the inference algorithm for model m_l. Similarly the calculation of (C.10a) is intractable. Therefore we will approximate the model prior with the variational message towards $\boldsymbol{c}_i$ instead as

$$p(\mathrm{m}_l) \approx \vec{\nu}(\boldsymbol{c}_i = \boldsymbol{e}_l). \tag{C.12}$$

C.3.2 Realization of inference for trial design

Probabilistic inference in AIDA encompasses 2 tasks: 1) optimal proposal selection and 2) updating of the Gaussian process classifier (GPC). Here we specify how these inference tasks are executed in more detail.

Optimal proposal selection

A closed-form expression of the EFE decomposition in (5.15) can be obtained for the GPC as shown in [134].

The first term in the decomposition, the negative utility drive, resembles the cross-entropy loss between our goal prior and posterior marginal. Since user responses are binary, we can evaluate this binary cross-entropy term as [134]

$$-\mathbb{E}_{q(r|\boldsymbol{u})}\left[\ln p(r)\right] = \Phi\left(\frac{m_{|\boldsymbol{u},D}}{\sqrt{\sigma^2_{|\boldsymbol{u},D}+1}}\right)\ln\mathbb{E}_{p(r)}[r] + \left(1-\Phi\left(\frac{m_{|\boldsymbol{u},D}}{\sqrt{\sigma^2_{|\boldsymbol{u},D}+1}}\right)\right)\ln\left(1-\mathbb{E}_{p(r)}[r]\right), \tag{C.13}$$

where $m_{|\boldsymbol{u},D}$ and $\sigma^2_{|\boldsymbol{u},D}$ denote the posterior mean and variance returned by the GPC when queried at the point $\boldsymbol{u}$ given some data set $D = \{\boldsymbol{u}_{1:k-1}, r_{1:k-1}\}$, respectively. More concretely, the GPC returns a Gaussian distribution from which the posterior mean and variance are extracted as $v(\boldsymbol{u}) = \mathcal{N}(m_{|\boldsymbol{u},D}, \sigma^2_{|\boldsymbol{u},D})$. $\Phi(\cdot)$ denotes the standard Gaussian cumulative distribution function. $p(r)$ denotes the Bernoulli goal prior over desired user feedback. h is the binary entropy function and $C = \sqrt{\frac{\pi \ln 2}{2}}$. For brevity, we denote the data set of parameters and matching user responses collected so far as D.

The second term in the decomposition, the (negative) information gain, describes how much information we gain by observing a new user appraisal. This information gain term (IG) can be expressed in a GPC as [134]

$$\mathrm{IG}[r, v \mid D, \boldsymbol{u}] \approx h\left(\Phi\left(\frac{m_{|\boldsymbol{u},D}}{\sqrt{\sigma^2_{|\boldsymbol{u},D}+1}}\right)\right) - \frac{C}{\sqrt{\sigma^2_{|\boldsymbol{u},D}+C^2}}\exp\left(-\frac{m^2_{|\boldsymbol{u},D}}{2\left(\sigma^2_{|\boldsymbol{u},D}+C^2\right)}\right), \tag{C.14}$$

where the constant C is defined as $C = \sqrt{\frac{\pi \ln 2}{2}}$ and where $h(\cdot)$ is defined as $h(p) = -p\ln(p) - (1-p)\ln(1-p)$.

Inference in the Gaussian process classifier

For our experiments, we use Laplace approximation as described in [137, Chapter 3.4] for performing inference in the GPC. The Laplace approximation is a two-step procedure, where we approximate the posterior distribution by a Gaussian distribution. We first find the mode of the exact posterior, which resembles the mean of

the approximated Gaussian distribution. Then we approximate the corresponding precision as the negative Hessian around the mode. Finding the exact posterior $p(\upsilon \mid D)$ amounts to calculating

$$p(\upsilon \mid D) = \frac{p(r_{1:k-1} \mid \upsilon)p(\upsilon \mid \boldsymbol{u}_{1:k-1})}{p(r_{1:k-1} \mid \boldsymbol{u}_{1:k-1})} \propto p(r_{1:k-1} \mid \upsilon)p(\upsilon \mid \boldsymbol{u}_{1:k-1})\,. \tag{C.15a}$$

Taking the logarithm of (C.15a) and differentiating twice with respect to υ gives

$$\nabla_\upsilon \ln p(\upsilon \mid D) = \nabla_\upsilon \ln p(r_{1:k-1} \mid \upsilon) - \mathcal{K}^{-1}\upsilon \tag{C.16a}$$

$$\nabla_\upsilon \nabla_\upsilon \ln p(\upsilon \mid D) = \nabla_\upsilon \nabla_\upsilon \ln p(r_{1:k-1} \mid \upsilon) - \mathcal{K}^{-1} = -W - \mathcal{K}^{-1} \tag{C.16b}$$

where ∇_υ denotes the gradient with respect to υ, $\mathcal{K} = \mathcal{K}(\boldsymbol{u}_{1:k-1}, \boldsymbol{u}_{1:k-1})$ is the kernel matrix over the queries $\boldsymbol{u}_{1:k-1}$ and $W = -\nabla_\upsilon \nabla_\upsilon \ln p(r_{1:k-1} \mid \upsilon)$ is a diagonal matrix since the likelihood factorizes over independent observations. At the mode $\hat{\upsilon}$ (C.16a) equals zero which implies

$$\hat{\upsilon} = \mathcal{K} \nabla_\upsilon \ln p(r_{1:k-1} | \hat{\upsilon})\,. \tag{C.17}$$

Directly solving (C.17) is intractable because of the recursive non-linear relationship. Instead we can estimate $\hat{\upsilon}$ using Newton's method, where we perform iterations with an adaptive step size. We omit the computational and implementation details here and instead refer to [137, Algorithm 3.1]. We determine the step size using a line search as implemented in `Optim.jl` [141]. Having found the mode $\hat{\upsilon}$, we can construct our posterior approximation as

$$p(\upsilon \mid D) \approx \mathcal{N}\left(\hat{\upsilon}, \left(\mathcal{K}^{-1} + W\right)^{-1}\right), \tag{C.18}$$

where W is evaluated at $\upsilon = \hat{\upsilon}$. If we now recall that evaluating a GP at any finite number of points results in a Gaussian, we see that under the Laplace approximation the solution can be obtained using standard results for marginalization of jointly Gaussian variables. We define the shorthand $\mathcal{K}(\boldsymbol{u}_i, \boldsymbol{u}_{1:k-1}) = \mathcal{K}_{1:k}$ and $\mathcal{K}(\boldsymbol{u}_i, \boldsymbol{u}_i) = \mathcal{K}_i$ and find the posterior mean $m_{|\boldsymbol{u},D}$ as [137, p. 44]

$$m_{|\boldsymbol{u},D} = \mathcal{K}_{1:k}^\intercal \mathcal{K}^{-1} \hat{\upsilon} = \mathcal{K}_{1:k}^\intercal \nabla \ln p(r_{1:k-1} \mid \hat{\upsilon})\,. \tag{C.19}$$

The posterior covariance $\sigma^2_{|\boldsymbol{u},D}$ is given by [137, p. 44]

$$\sigma^2_{|\boldsymbol{u},D} = \mathcal{K}_i - \mathcal{K}_{1:k}^\intercal \left(\mathcal{K} + W^{-1}\right)^{-1} \mathcal{K}_{1:k}\,. \tag{C.20}$$

Bibliography

[1] Keith T. Butler, Daniel W. Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. Machine learning for molecular and materials science. *Nature*, 559(7715):547–555, July 2018. Number: 7715 Publisher: Nature Publishing Group. (Cited on 3)

[2] N Schanche, A Collier Cameron, G Hébrard, L Nielsen, A H M J Triaud, J M Almenara, K A Alsubai, D R Anderson, D J Armstrong, S C C Barros, F Bouchy, P Boumis, D J A Brown, F Faedi, K Hay, L Hebb, F Kiefer, L Mancini, P F L Maxted, E Palle, D L Pollacco, D Queloz, B Smalley, S Udry, R West, and P J Wheatley. Machine-learning approaches to exoplanet transit detection and candidate validation in wide-field ground-based surveys. *Monthly Notices of the Royal Astronomical Society*, 483(4):5534–5547, March 2019. (Cited on 3)

[3] Joe G. Greener, Shaun M. Kandathil, Lewis Moffat, and David T. Jones. A guide to machine learning for biologists. *Nature Reviews Molecular Cell Biology*, 23(1):40–55, January 2022. Number: 1 Publisher: Nature Publishing Group. (Cited on 3)

[4] Nongnuch Artrith, Keith T. Butler, François-Xavier Coudert, Seungwu Han, Olexandr Isayev, Anubhav Jain, and Aron Walsh. Best practices in machine learning for chemistry. *Nature Chemistry*, 13(6):505–508, June 2021. Number: 6 Publisher: Nature Publishing Group. (Cited on 3)

[5] Alex Davies, Petar Veličković, Lars Buesing, Sam Blackwell, Daniel Zheng, Nenad Tomašev, Richard Tanburn, Peter Battaglia, Charles Blundell, András Juhász, Marc Lackenby, Geordie Williamson, Demis Hassabis, and Pushmeet Kohli. Advancing mathematics by guiding human intuition with AI. *Nature*, 600(7887):70–74, December 2021. Number: 7887 Publisher: Nature Publishing Group. (Cited on 3)

[6] Pat Pataranutaporn, Valdemar Danry, Joanne Leong, Parinya Punpongsanon, Dan Novy, Pattie Maes, and Misha Sra. AI-generated characters for supporting personalized learning and well-being. *Nature Machine Intelligence*, 3(12):1013–1022, December 2021. Number: 12 Publisher: Nature Publishing Group. (Cited on 3)

[7] Jens Kober and Jan Peters. Reinforcement Learning in Robotics: A Survey. In Jens Kober and Jan Peters, editors, *Learning Motor Skills: From Algorithms to Robot Experiments*, Springer Tracts in Advanced Robotics, pages 9–67. Springer International Publishing, Cham, 2014. (Cited on 3)

[8] Mike Daily, Swarup Medasani, Reinhold Behringer, and Mohan Trivedi. Self-Driving Cars. *Computer*, 50(12):18–23, December 2017. Conference Name: Computer. (Cited on 3)

[9] Rutvik V. Shah, Gillian Grennan, Mariam Zafar-Khan, Fahad Alim, Sujit Dey, Dhakshin Ramanathan, and Jyoti Mishra. Personalized machine learning of depressed mood using wearables. *Translational Psychiatry*, 11(1):1–18, June 2021. Number: 1 Publisher: Nature Publishing Group. (Cited on 3)

[10] Conor K. Corbin, Lillian Sung, Arhana Chattopadhyay, Morteza Noshad, Amy Chang, Stanley Deresinksi, Michael Baiocchi, and Jonathan H. Chen. Personalized antibiograms for machine learning driven antibiotic selection. *Communications Medicine*, 2(1):1–14, April 2022. Number: 1 Publisher: Nature Publishing Group. (Cited on 3)

[11] Jessica Vamathevan, Dominic Clark, Paul Czodrowski, Ian Dunham, Edgardo Ferran, George Lee, Bin Li, Anant Madabhushi, Parantu Shah, Michaela Spitzer, and Shanrong Zhao. Applications of machine learning in drug discovery and development. *Nature Reviews Drug Discovery*, 18(6):463–477, June 2019. Number: 6 Publisher: Nature Publishing Group. (Cited on 3)

[12] Muhammad Aurangzeb Ahmad, Carly Eckert, and Ankur Teredesai. Interpretable Machine Learning in Healthcare. In *Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '18, pages 559–560, New York, NY, USA, August 2018. Association for Computing Machinery. (Cited on 3)

[13] Holger Fröhlich, Rudi Balling, Niko Beerenwinkel, Oliver Kohlbacher, Santosh Kumar, Thomas Lengauer, Marloes H. Maathuis, Yves Moreau, Susan A. Murphy, Teresa M. Przytycka, Michael Rebhan, Hannes Röst, Andreas Schuppert, Matthias Schwab, Rainer Spang, Daniel Stekhoven, Jimeng Sun, Andreas Weber, Daniel Ziemek, and Blaz Zupan. From hype to reality: data science enabling personalized medicine. *BMC Medicine*, 16(1):150, August 2018. (Cited on 3)

[14] Eklas Hossain, Imtiaj Khan, Fuad Un-Noor, Sarder Shazali Sikander, and Md. Samiul Haque Sunny. Application of Big Data and Machine Learning in Smart Grid, and Associated Security Concerns: A Review. *IEEE Access*, 7:13960–13988, 2019. Conference Name: IEEE Access. (Cited on 3)

[15] Marco Cox and Bert de Vries. A Bayesian binary classification approach to pure tone audiometry. *arXiv:1511.08670 [stat]*, March 2016. arXiv: 1511.08670. (Cited on 3)

[16] Ivan Bocharov, Tjalling Tjalkens, and Bert De Vries. Acoustic Scene Classification from Few Examples. In *26th European Signal Processing Conference (EUSIPCO)*, page 5, Rome, Italy, 2018. (Cited on 3)

[17] Anouk van Diepen, Marco Cox, and Bert de Vries. An In-situ Trainable Gesture Classifier. In *Proceedings of the Twenty-Sixth Benelux Conference on Machine Learning*, pages 66–69, Technische Universiteit Eindhoven, June 2017. (Cited on 3)

[18] Marco Cox and Bert de Vries. A parametric approach to Bayesian optimization with pairwise comparisons. In *NIPS Workshop on Bayesian Optimization (BayesOpt 2017)*, pages 1–5, Long Beach, USA, December 2017. (Cited on 3, 100)

[19] Jorunn Solheim, Kari J. Kværner, and Eva-Signe Falkenberg. Daily life consequences of hearing loss in the elderly. *Disability and Rehabilitation*, 33(22-23):2179–2185, January 2011. Publisher: Taylor & Francis _eprint: https://doi.org/10.3109/09638288.2011.563815. (Cited on 4)

[20] Gill Livingston, Jonathan Huntley, Andrew Sommerlad, David Ames, Clive Ballard, Sube Banerjee, Carol Brayne, Alistair Burns, Jiska Cohen-Mansfield, Claudia Cooper, Sergi G. Costafreda, Amit Dias, Nick Fox, Laura N. Gitlin, Robert Howard, Helen C. Kales, Mika Kivimäki, Eric B. Larson, Adesola Ogunniyi, Vasiliki Orgeta, Karen Ritchie, Kenneth Rockwood, Elizabeth L. Sampson, Quincy Samus, Lon S. Schneider, Geir Selbæk, Linda Teri, and Naaheed Mukadam. Dementia prevention, intervention, and care: 2020 report of the Lancet Commission. *The Lancet*, 396(10248):413–446, August 2020. Publisher: Elsevier. (Cited on 4)

[21] Patrick J. Willems. Genetic Causes of Hearing Loss. *New England Journal of Medicine*, 342(15):1101–1109, April 2000. Publisher: Massachusetts Medical Society _eprint: https://doi.org/10.1056/NEJM200004133421506. (Cited on 4)

[22] Colleen G. Le Prell, Daisuke Yamashita, Shujiro B. Minami, Tatsuya Yamasoba, and Josef M. Miller. Mechanisms of noise-induced hearing loss indicate multiple methods of prevention. *Hearing Research*, 226(1):22–43, April 2007. (Cited on 4)

[23] Gitte Keidser and Karima Alamudi. Real-life efficacy and reliability of training a hearing aid. *Ear and Hearing*, 34(5):619–629, September 2013. (Cited on 4)

[24] Mara Mills. Hearing Aids and the History of Electronics Miniaturization. *IEEE Annals of the History of Computing*, 33(2):24–45, February 2011. Conference Name: IEEE Annals of the History of Computing. (Cited on 4)

[25] Thijs van de Laar. *Automated Design of Bayesian Signal Processing Algorithms*. PhD thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 2019. (Cited on 4, 11, 60, 64, 85, 87)

[26] Christian Stilp. Acoustic context effects in speech perception. *WIREs Cognitive Science*, 11(1):e1517, 2020. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcs.1517. (Cited on 4)

[27] Sergey Kochkin. MarkeTrak VIII: Customer satisfaction with hearing aids is slowly increasing. *The Hearing Journal*, 63(1):11–19, 2010. (Cited on 5)

[28] Karl Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127–138, 2010. (Cited on 6)

[29] Pablo Lanillos, Cristian Meo, Corrado Pezzato, Ajith Anil Meera, Mohamed Baioumy, Wataru Ohata, Alexander Tschantz, Beren Millidge, Martijn Wisse, Christopher L. Buckley, and Jun Tani. Active Inference in Robotics and Artificial Agents: Survey and Challenges. December 2021. (Cited on 6)

[30] Alexander Tschantz, Beren Millidge, Anil K. Seth, and Christopher L. Buckley. Reinforcement Learning through Active Inference. *arXiv:2002.12636 [cs, eess, math, stat]*, February 2020. arXiv: 2002.12636. (Cited on 6)

[31] Ryan Smith, Karl Friston, and Christopher Whyte. A Step-by-Step Tutorial on Active Inference and its Application to Empirical Data. Technical report, PsyArXiv, January 2021. (Cited on 6)

[32] Emma Holmes, Thomas Parr, Timothy D. Griffiths, and Karl J. Friston. Active inference, selective attention, and the cocktail party problem. *Neuroscience and Biobehavioral Reviews*, 131:1288–1304, October 2021. (Cited on 6, 102)

[33] Roger C. Conant and W. Ross Ashby. Every good regulator of a system must be a model of that system. *Intl. J. Systems Science*, pages 89–97, 1970. (Cited on 6)

[34] Richard Turner and Maneesh Sahani. Modeling natural sounds with modulation cascade processes. Advances in Neural Information Processing Systems (NIPS), 2008. (Cited on 7)

[35] G. Udny Yule. On a Method of Investigating Periodicities in Disturbed Series, with Special Reference to Wolfer's Sunspot Numbers. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 226:267–298, 1927. Publisher: The Royal Society. (Cited on 7)

[36] Hirotugu Akaike. Autoregressive model fitting for control. *Annals of the Institute of Statistical Mathematics*, 23(1):163–180, December 1971. (Cited on 7)

[37] Mangala S. Joshi, Prashant P. Bartakke, and M.S. Sutaone. Texture representation using autoregressive models. In *2009 International Conference on Advances in Computational Tools for Engineering Applications*, pages 386–390, July 2009. (Cited on 7)

[38] K.E. Baddour and N.C. Beaulieu. Autoregressive modeling for fading channel simulation. *IEEE Transactions on Wireless Communications*, 4(4):1650–1662, July 2005. (Cited on 7)

[39] J. J. Wright, R. R. Kydd, and A. A. Sergejew. Autoregression models of EEG. *Biological Cybernetics*, 62(3):201–210, January 1990. (Cited on 7)

[40] Matt Shannon, Heiga Zen, and William Byrne. Autoregressive Models for Statistical Parametric Speech Synthesis. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(3):587–597, March 2013. (Cited on 7)

[41] Daniel Rudoy, Thomas F. Quatieri, and Patrick J. Wolfe. Time-Varying Autoregressions in Speech: Detection Theory and Applications. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):977–989, May 2011. (Cited on 8, 16, 79)

[42] K. Kostoglou, A. D. Robertson, B. J. MacIntosh, and G. D. Mitsis. A Novel Framework for Estimating Time-Varying Multivariate Autoregressive Models and Application to Cardiovascular Responses to Acute Exercise. *IEEE Transactions on Biomedical Engineering*, 66(11):3257–3266, November 2019. Conference Name: IEEE Transactions on Biomedical Engineering. (Cited on 8, 16)

[43] İsmail Şenöz, Thijs van de Laar, Dmitry Bagaev, and Bert de Vries. Variational Message Passing and Local Constraint Manipulation in Factor Graphs. *Entropy*, 23(7):807, July 2021. Number: 7 Publisher: Multidisciplinary Digital Publishing Institute. (Cited on 9, 62, 64, 87, 136)

[44] Semih Akbayrak, Ivan Bocharov, and Bert de Vries. Extended Variational Message Passing for Automated Approximate Bayesian Inference. *Entropy*, 23(7):815, June 2021. (Cited on 9)

[45] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017. (Cited on 9, 16, 107)

[46] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer Publishing Company, Incorporated, 2010. (Cited on 9)

[47] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006. (Cited on 9, 54, 85)

[48] Hans-Andrea Loeliger, Justin Dauwels, Junli Hu, Sascha Korl, Li Ping, and Frank R. Kschischang. The Factor Graph Approach to Model-Based Signal Processing. *Proceedings of the IEEE*, 95(6):1295–1322, June 2007. (Cited on 10, 17, 23, 45, 62, 84, 87)

[49] Hirotugu Akaike. Fitting autoregressive models for prediction. *Annals of the Institute of Statistical Mathematics*, 21(1):243–247, December 1969. (Cited on 16)

[50] R Charbonnier, M Barlaud, G Alengrin, and J Menez. Results on AR-modelling of nonstationary signals. *Signal Processing*, 12(2):143–151, March 1987. (Cited on 16)

[51] S. M. Tahir, A. Z. Shaameri, and S. H. S. Salleh. Time-varying autoregressive modeling approach for speech segmentation. In *Proceedings of the Sixth International Symposium on Signal Processing and its Applications (Cat.No.01EX467)*, volume 2, pages 715–718 vol.2, August 2001. (Cited on 16)

[52] Y. J. Chu, S. C. Chan, Z. G. Zhang, and K. M. Tsui. A new regularized TVAR-based algorithm for recursive detection of nonstationarity and its application to speech signals. In *2012 IEEE Statistical Signal Processing Workshop (SSP)*, pages 361–364, August 2012. ISSN: 2373-0803. (Cited on 16)

[53] M. J. Paulik, N. Mohankrishnan, and M. Nikiforuk. A time varying vector autoregressive model for signature verification. In *Proceedings of 1994 37th Midwest Symposium on Circuits and Systems*, volume 2, pages 1395–1398 vol.2, August 1994. (Cited on 16)

[54] Kie B. Eom. Analysis of Acoustic Signatures from Moving Vehicles Using Time-Varying Autoregressive Models. *Multidimensional Systems and Signal Processing*, 10(4):357–378, October 1999. (Cited on 16)

[55] Y. I. Abramovich, N. K. Spencer, and M. D. E. Turley. Time-Varying Autoregressive (TVAR) Models for Multiple Radar Observations. *IEEE Transactions on Signal Processing*, 55(4):1298–1311, April 2007. Conference Name: IEEE Transactions on Signal Processing. (Cited on 16)

[56] Z. G. Zhang, Y. S. Hung, and S. C. Chan. Local Polynomial Modeling of Time-Varying Autoregressive Models With Application to Time–Frequency Analysis of Event-Related EEG. *IEEE Transactions on Biomedical Engineering*, 58(3):557–566, March 2011. Conference Name: IEEE Transactions on Biomedical Engineering. (Cited on 16)

[57] Huan Wang, L. Bai, Jianmei Xu, and W. Fei. EEG recognition through Time-varying Vector Autoregressive Model. In *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pages 292–296, August 2015. (Cited on 16)

[58] K. Sharman and B. Friedlander. Time-varying autoregressive modeling of a class of nonstationary signals. In *ICASSP '84. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 9, pages 227–230, March 1984. (Cited on 16)

[59] G. Ravi Shankar Reddy and R. Rao. Non stationary signal prediction using TVAR model. In *2014 International Conference on Communication and Signal Processing*, pages 1692–1697, April 2014. (Cited on 16)

[60] D. Baptista de Souza, E. V. Kuhn, and R. Seara. A Time-Varying Autoregressive Model for Characterizing Nonstationary Processes. *IEEE Signal Processing Letters*, 26(1):134–138, January 2019. (Cited on 16)

[61] Yuanjin Zheng and Zhiping Lin. Time-varying autoregressive system identification using wavelets. In *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.00CH37100)*, volume 1, pages 572–575 vol.1, June 2000. ISSN: 1520-6149. (Cited on 16)

[62] Todd K. Moon and Jacob H. Gunther. Estimation of Autoregressive Parameters from Noisy Observations Using Iterated Covariance Updates. *Entropy*, 22(5):572, May 2020. Number: 5 Publisher: Multidisciplinary Digital Publishing Institute. (Cited on 16)

[63] J. J. Rajan, P. J. W. Rayner, and S. J. Godsill. Bayesian approach to parameter estimation and interpolation of time-varying autoregressive processes using the Gibbs sampler. *IEE Proceedings - Vision, Image and Signal Processing*, 144(4):249–256, August 1997. Publisher: IET Digital Library. (Cited on 16)

[64] Raquel Prado, Gabriel Huerta, and Mike West. Bayesian time-varying autoregressions: Theory, methods and Applications. In *University of Sao Paolo*, page 2000, 2000. (Cited on 16)

[65] Jouchi Nakajima, Munehisa Kasuya, and Toshiaki Watanabe. Bayesian analysis of time-varying parameter vector autoregressive model for the Japanese economy and monetary policy. *Journal of the Japanese and International Economies*, 25(3):225–245, September 2011. (Cited on 16)

[66] David Barber. *Bayesian Reasoning and Machine Learning*. Cambridge University Press, 2012. (Cited on 16, 60)

[67] Zhong Xionghu, Song Shubiao, and Pei Chengming. Time-varying Parameters Estimation based on Kalman Particle Filter with Forgetting Factors. In *EUROCON 2005 - The International Conference on "Computer as a Tool"*, volume 2, pages 1558–1561, November 2005. (Cited on 16)

[68] John Winn and Christopher M. Bishop. Variational Message Passing. *Journal of Machine Learning Research*, 6(23):661–694, 2005. (Cited on 16, 24, 46, 48, 63)

[69] Sascha Korl. *A factor graph approach to signal modelling, system identification and filtering*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 2005. (Cited on 16, 45, 62)

[70] Will D. Penny and Stephen J. Roberts. Bayesian multivariate autoregressive models with structured priors. *IEE Proceedings - Vision, Image and Signal Processing*, 149(1):33–41, February 2002. (Cited on 16, 107)

[71] J. Dauwels, S. Korl, and H.-A. Loeliger. Expectation maximization as message passing. In *International Symposium on Information Theory, 2005. ISIT 2005. Proceedings*, pages 583–586, September 2005. (Cited on 17, 49, 64)

[72] Marco Cox, Thijs van de Laar, and Bert de Vries. A factor graph approach to automated design of Bayesian signal processing algorithms. *International Journal of Approximate Reasoning*, 104:185–204, January 2019. (Cited on 17, 29, 41, 55, 84)

[73] Dmitry Bagaev and Bert de Vries. Reactive Message Passing for Scalable Bayesian Inference. *arXiv:2112.13251 [cs]*, December 2021. arXiv: 2112.13251. (Cited on 17, 29, 88)

[74] Bert de Vries and Karl J. Friston. A Factor Graph Description of Deep Temporal Active Inference. *Frontiers in Computational Neuroscience*, 11, 2017. (Cited on 18)

[75] James L. Beck. Bayesian system identification based on probability logic. *Structural Control and Health Monitoring*, 17(7):825–847, 2010. (Cited on 19)

[76] Dan Zhang, Wenjin Wang, Gerhard Fettweis, and Xiqi Gao. Unifying Message Passing Algorithms Under the Framework of Constrained Bethe Free Energy Minimization. *arXiv:1703.10932 [cs, math]*, March 2017. arXiv: 1703.10932. (Cited on 24, 25)

[77] Justin Dauwels. On Variational Message Passing on Factor Graphs. In *IEEE International Symposium on Information Theory*, pages 2546–2550, Nice, France, June 2007. (Cited on 24, 25, 46, 48, 63, 64, 136)

[78] G. Cui, X. Yu, S. Iommelli, and L. Kong. Exact Distribution for the Product of Two Correlated Gaussian Random Variables. *IEEE Signal Processing Letters*, 23(11):1662–1666, November 2016. Conference Name: IEEE Signal Processing Letters. (Cited on 28)

[79] Wen-Rong Wu and Po-Cheng Chen. Subband Kalman filtering for speech enhancement. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 45(8):1072–1083, August 1998. Conference Name: IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing. (Cited on 35)

[80] Stephen So and Kuldip K. Paliwal. Modulation-domain Kalman filtering for single-channel speech enhancement. *Speech Communication*, 53(6):818–829, July 2011. (Cited on 35)

[81] Soha A. Nossier, Julie Wall, Mansour Moniri, Cornelius Glackin, and Nigel Cannings. An Experimental Analysis of Deep Learning Architectures for Supervised Speech Enhancement. *Electronics*, 10(1):17, January 2021. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute. (Cited on 35)

[82] Yi Hu and Philipos C. Loizou. Subjective comparison and evaluation of speech enhancement algorithms. *Speech communication*, 49(7):588–601, July 2007. (Cited on 35)

[83] K. Paliwal and A. Basu. A speech enhancement method based on Kalman filtering. In *ICASSP '87. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 12, pages 177–180, Dallas, TX, USA, April 1987. (Cited on 36, 60, 79)

[84] Chang Huai You, Susanto Rahardja, and Soo Ngee Koh. Autoregressive Parameter Estimation for Kalman Filtering Speech Enhancement. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 4, pages IV–913–IV–916, April 2007. ISSN: 2379-190X. (Cited on 36)

[85] Y. Grenier. Time-dependent ARMA modeling of nonstationary signals. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 31(4):899–911, August 1983. Conference Name: IEEE Transactions on Acoustics, Speech, and Signal Processing. (Cited on 36)

[86] Kaniav Kamary, Kerrie Mengersen, Christian P. Robert, and Judith Rousseau. Testing hypotheses via a mixture estimation model. *arXiv:1412.2044 [stat]*, December 2014. arXiv: 1412.2044. (Cited on 39)

[87] Karl J. Friston, Vladimir Litvak, Ashwini Oswal, Adeel Razi, Klaas E. Stephan, Bernadette C. M. van Wijk, Gabriel Ziegler, and Peter Zeidman. Bayesian model reduction and empirical Bayes for group (DCM) studies. *NeuroImage*, 128:413–431, March 2016. (Cited on 39)

[88] Albert Podusenko, Wouter M. Kouw, and Bert de Vries. Online Variational Message Passing in Hierarchical Autoregressive Models. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 1337–1342, Los Angeles, CA, USA, June 2020. ISSN: 2157-8117. (Cited on 40, 79)

[89] S.J. Rennie, J.R. Hershey, and P.A. Olsen. Single-Channel Multitalker Speech Recognition. *IEEE Signal Processing Magazine*, 27(6):66–80, November 2010. (Cited on 44)

[90] Chi Liu, Heng-Chao Li, Kun Fu, Fan Zhang, Mihai Datcu, and William J. Emery. Bayesian estimation of generalized Gamma mixture model based on variational EM algorithm. *Pattern Recognition*, 87:269–284, March 2019. (Cited on 44)

[91] Hassen Sallay, Sami Bourouis, and Nizar Bouguila. Online Learning of Finite and Infinite Gamma Mixture Models for COVID-19 Detection in Medical Images. *Computers*, 10(1):6, January 2021. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute. (Cited on 44)

[92] Michael Wiper, David Rios Insua, and Fabrizio Ruggeri. Mixtures of Gamma Distributions with Applications. *Journal of Computational and Graphical Statistics*, 10(3):440–454, 2001. Publisher: [American Statistical Association, Taylor & Francis, Ltd., Institute of Mathematical Statistics, Interface Foundation of America]. (Cited on 44, 51)

[93] G.David Forney. Codes on graphs: normal realizations. *IEEE Transactions on Information Theory*, 47(2):520–548, February 2001. (Cited on 44, 45, 60, 62, 78)

[94] Hans-Andrea Loeliger. An introduction to factor graphs. *Signal Processing Magazine, IEEE*, 21(1):28–41, January 2004. (Cited on 44, 45, 60, 62, 78)

[95] Simo Särkkä. *Bayesian Filtering and Smoothing*. Cambridge University Press, London ; New York, October 2013. (Cited on 46, 50, 87, 108)

[96] Frank R. Kschischang, Brendan J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001. (Cited on 48, 63, 136)

[97] Judea Pearl. Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach. In *Proceedings of the Second AAAI Conference on Artificial Intelligence*, AAAI'82, pages 133–136, Pittsburgh, Pennsylvania, 1982. AAAI Press. (Cited on 48, 63, 136)

[98] Milan Merkle. Logarithmic Convexity and Inequalities for the Gamma Function. *Journal of Mathematical Analysis and Applications*, 203(2):369–380, October 1996. (Cited on 49)

[99] Stephen P. Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004. (Cited on 49)

[100] Art B. Owen. *Monte Carlo theory, methods and examples*. 2013. (Cited on 50)

[101] J. Bezanson, A. Edelman, S. Karpinski, and V. Shah. Julia: A Fresh Approach to Numerical Computing. *SIAM Review*, 59(1):65–98, January 2017. (Cited on 51, 65, 88)

[102] Fernando Antonio Moala, Pedro Luiz Ramos, and Jorge Alberto Achcar. Bayesian Inference for Two-Parameter Gamma Distribution Assuming Different Noninformative Priors. *Revista Colombiana de Estadística*, 36(2):319–336, 2013. (Cited on 51)

[103] Martin J. Wainwright and Michael I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, November 2008. (Cited on 52, 65)

[104] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. (Cited on 54)

[105] O. Kakusho and M. Yanagida. Hierarchical AR model for time varying speech signals. In *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 7, pages 1295–1298, Paris, France, May 1982. (Cited on 60, 79)

[106] J.D. Gibson, B. Koo, and S.D. Gray. Filtering of colored noise for speech enhancement and coding. *IEEE Transactions on Signal Processing*, 39(8):1732–1742, August 1991. (Cited on 60, 80)

[107] S. Gannot, D. Burshtein, and E. Weinstein. Iterative and sequential Kalman filter-based speech enhancement algorithms. *IEEE Transactions on Speech and Audio Processing*, 6(4):373–385, July 1998. (Cited on 60, 80)

[108] Bertrand Mesot and David Barber. A Bayesian Alternative to Gain Adaptation in Autoregressive Hidden Markov Models. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, volume 2, pages II–437–II–440, April 2007. ISSN: 2379-190X. (Cited on 60)

[109] Albert Podusenko, Wouter M. Kouw, and Bert de Vries. Message Passing-Based Inference for Time-Varying Autoregressive Models. *Entropy*, 23(6):683, June 2021. Number: 6 Publisher: Multidisciplinary Digital Publishing Institute. (Cited on 60, 64, 68, 85, 87)

[110] Şenöz, İsmail, Albert Podusenko, Wouter M Kouw, and Bert de Vries. Bayesian joint state and parameter tracking in autoregressive models. In *Learning for Dynamics and Control*, page 9, 2020. (Cited on 61)

[111] Albert Podusenko, Bart van Erp, Dmitry Bagaev, Şenöz, İsmail, and Bert de Vries. Message Passing-Based Inference in the Gamma Mixture Model. In *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Gold Coast, Australia, October 2021. IEEE. (Cited on 64, 67, 68, 87, 138)

[112] James Kates and Kathryn Arehart. Multichannel Dynamic-Range Compression Using Digital Frequency Warping. *EURASIP Journal on Applied Signal Processing*, 18:3003–3014, 2005. (Cited on 72)

[113] Thijs van de Laar and Bert de Vries. A Probabilistic Modeling Approach to Hearing Loss Compensation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(11):2200–2213, November 2016. (Cited on 72)

[114] J.B.B. Nielsen, J. Nielsen, and J. Larsen. Perception-Based Personalization of Hearing Aids Using Gaussian Processes and Active Learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(1):162–173, January 2015. (Cited on 72, 101, 102)

[115] Nasim Alamdari, Edward Lobarinas, and Nasser Kehtarnavaz. Personalization of Hearing Aid Compression by Human-in-the-Loop Deep Reinforcement Learning. *IEEE Access*, 8:203503–203515, 2020. (Cited on 72, 101)

[116] C. Karadagur Ananda Reddy, N. Shankar, G. Shreedhar Bhat, R. Charan, and I. Panahi. An Individualized Super-Gaussian Single Microphone Speech Enhancement for Hearing Aid Users With Smartphone as an Assistive Device. *IEEE Signal Processing Letters*, 24(11):1601–1605, November 2017. (Cited on 72)

[117] Karl Friston, James Kilner, and Lee Harrison. A free energy principle for the brain. *Journal of Physiology, Paris*, 100(1-3):70–87, September 2006. (Cited on 73)

[118] Thijs van de Laar and Bert de Vries. Simulating Active Inference Processes by Message Passing. *Frontiers in Robotics and AI*, 6:20, 2019. (Cited on 73)

[119] Thijs van de Laar, Ayça Özçelikkale, and Henk Wymeersch. Application of the Free Energy Principle to Estimation and Control. *arXiv preprint arXiv:1910.09823*, 2019. (Cited on 73)

[120] Beren Millidge. Deep Active Inference as Variational Policy Gradients. *arXiv:1907.03876 [cs]*, July 2019. arXiv: 1907.03876. (Cited on 73)

[121] Alexander Tschantz, Manuel Baltieri, Anil K. Seth, and Christopher L. Buckley. Scaling active inference. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020. (Cited on 73)

[122] Karl Friston, Francesco Rigoli, Dimitri Ognibene, Christoph Mathys, Thomas Fitzgerald, and Giovanni Pezzulo. Active inference and epistemic value. *Cognitive Neuroscience*, 6(4):187–214, March 2015. (Cited on 73, 85, 86)

[123] Lancelot Da Costa, Thomas Parr, Noor Sajid, Sebastijan Veselic, Victorita Neacsu, and Karl Friston. Active inference on discrete state-spaces: a synthesis. *arXiv:2001.07203 [q-bio]*, January 2020. arXiv: 2001.07203. (Cited on 73)

[124] Karl Friston, Lancelot Da Costa, Danijar Hafner, Casper Hesp, and Thomas Parr. Sophisticated Inference. *Neural Computation*, 33(3):713–763, March 2021. (Cited on 73)

[125] Antony W. Rix, John G. Beerends, Michael P. Hollier, and Andries P. Hekstra. Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs. In *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001. Proceedings.*, volume 2, pages 749–752. IEEE, 2001. (Cited on 74)

[126] James M. Kates and Kathryn H. Arehart. The hearing-aid speech quality index (HASQI). *Journal of the Audio Engineering Society*, 58(5):363–381, 2010. (Cited on 74)

[127] Cees H. Taal, Richard C. Hendriks, Richard Heusdens, and Jesper Jensen. An Algorithm for Intelligibility Prediction of Time–Frequency Weighted Noisy Speech. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(7):2125–2136, September 2011. (Cited on 74)

[128] John G. Beerends, Christian Schmidmer, Jens Berger, Matthias Obermann, Raphael Ullmann, Joachim Pomy, and Michael Keyhl. Perceptual Objective Listening Quality Assessment (POLQA), The Third Generation ITU-T Standard for End-to-End Speech Quality Measurement Part I—Temporal Alignment. *Journal of the Audio Engineering Society*, 61(6):366–384, July 2013. Publisher: Audio Engineering Society. (Cited on 74)

[129] Andrew Hines, Jan Skoglund, Anil C Kokaram, and Naomi Harte. ViSQOL: an objective speech quality model. *EURASIP Journal on Audio, Speech, and Music Processing*, 2015(1):13, December 2015. (Cited on 74)

[130] Michael Chinen, Felicia S. C. Lim, Jan Skoglund, Nikita Gureev, Feargus O'Gorman, and Andrew Hines. ViSQOL v3: An Open Source Production Ready Objective Speech and Audio Metric. *arXiv:2004.09584 [cs, eess]*, April 2020. (Cited on 74)

[131] J. Vermaak, C. Andrieu, A. Doucet, and S.J. Godsill. Particle methods for Bayesian modeling and enhancement of speech signals. *IEEE Transactions on Speech and Audio Processing*, 10(3):173–185, March 2002. (Cited on 79)

[132] D.C. Popescu and I. Zeljkovic. Kalman filtering of colored noise for speech enhancement. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98*, volume 2, pages 997–1000, Seattle, WA, USA, May 1998. ISSN: 1520-6149. (Cited on 80)

[133] Bart van Erp, Albert Podusenko, Tanya Ignatenko, and Bert de Vries. A Bayesian Modeling Approach to Situated Design of Personalized Soundscaping Algorithms. *Applied Sciences*, 11(20):9535, October 2021. Number: 20 Publisher: Multidisciplinary Digital Publishing Institute. (Cited on 81, 85)

[134] Neil Houlsby, Ferenc Huszár, Zoubin Ghahramani, and Máté Lengyel. Bayesian Active Learning for Classification and Preference Learning. *arXiv:1112.5745 [cs, stat]*, December 2011. (Cited on 83, 140)

[135] Wei Chu and Zoubin Ghahramani. Preference learning with Gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, ICML '05, pages 137–144, New York, NY, USA, August 2005. Association for Computing Machinery. (Cited on 83)

[136] Ferenc Huszar. A GP classification approach to preference learning. In *NIPS Workshop on Choice Models and Preference Learning*, page 4, Sierra Nevada, Spain, 2011. (Cited on 83)

[137] Carl Edward Rasmussen and Christopher K. I Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. (Cited on 83, 87, 140, 141)

[138] Noor Sajid, Philip J. Ball, Thomas Parr, and Karl J. Friston. Active Inference: Demystified and Compared. *Neural Computation*, 33(3):674–712, March 2021. (Cited on 85)

[139] Thomas Parr and Karl J. Friston. Uncertainty, epistemics and active inference. *Journal of The Royal Society Interface*, 14(136):20170376, November 2017. (Cited on 86)

[140] Kevin H. Knuth. Informed Source Separation: A Bayesian Tutorial. *arXiv:1311.3001 [cs, stat]*, November 2013. arXiv: 1311.3001. (Cited on 87, 102)

[141] Patrick K Mogensen and Asbjørn N Riseth. Optim: A mathematical optimization package for Julia. *Journal of Open Source Software*, 3(24):615, April 2018. (Cited on 90, 141)

[142] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 467–475. Morgan Kaufmann Publishers Inc., 1999. (Cited on 95)

[143] Tesheng Hsiao. Identification of Time-Varying Autoregressive Systems Using Maximum a Posteriori Estimation. *IEEE Transactions on Signal Processing*, 56(8):3497–3509, August 2008. (Cited on 99)

[144] Frank Kleibergen and Henk Hoek. Bayesian Analysis of ARMA models using Noninformative Priors. *CentER Discussion Paper*, 1995-116:24, 1995. (Cited on 99)

[145] Karl Friston and Will Penny. Post hoc Bayesian model selection. *Neuroimage*, 56(4-2):2089–2099, June 2011. (Cited on 100)

[146] Karl Friston, Thomas Parr, and Peter Zeidman. Bayesian model reduction. *arXiv:1805.07092 [stat]*, May 2018. arXiv: 1805.07092. (Cited on 100)

[147] A. Ozerov and C. Fevotte. Multichannel Nonnegative Matrix Factorization in Convolutive Mixtures for Audio Source Separation. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(3):550–563, March 2010. (Cited on 100)

[148] Steven Rennie, Trausti Kristjansson, Peder Olsen, and Ramesh Gopinath. Dynamic noise adaptation. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pages 1–4, Toulouse, France, 2006. IEEE. (Cited on 100, 102)

[149] S.J. Rennie, J.R. Hershey, and P.A. Olsen. Single-channel speech separation and recognition using loopy belief propagation. In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009*, pages 3845–3848, Taipei, Taiwan, April 2009. (Cited on 100, 102)

[150] Brendan J Frey, Li Deng, Alex Acero, and Trausti Kristjansson. ALGONQUIN: Iterating Laplace's Method to Remove Multiple Types of Acoustic Distortion for Robust Speech Recognition. In *Proceedings of the Eurospeech Conference*, pages 901–904, Aalborg, Denmark, September 2001. (Cited on 100, 102)

[151] Tanya Ignatenko, Kirill Kondrashov, Marco Cox, and Bert de Vries. On Sequential Bayesian Optimization with Pairwise Comparison. *arXiv:2103.13192 [cs, math, stat]*, March 2021. arXiv: 2103.13192. (Cited on 100, 102)

[152] Karl J. Friston, Noor Sajid, David Ricardo Quiroga-Martinez, Thomas Parr, Cathy J. Price, and Emma Holmes. Active listening. *Hearing Research*, 399(Stimulus-specific adaptation, MMN and predicting coding):107998, January 2021. (Cited on 102)

[153] Y. Laufer and S. Gannot. A Bayesian Hierarchical Model for Blind Audio Source Separation. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 276–280, January 2021. ISSN: 2076-1465. (Cited on 102)

[154] S. Xie, L. Yang, J. Yang, G. Zhou, and Y. Xiang. Time-Frequency Approach to Underdetermined Blind Source Separation. *IEEE Transactions on Neural Networks and Learning Systems*, 23(2):306–316, February 2012. (Cited on 102)

[155] John R Hershey, Peder Olsen, and Steven J Rennie. Signal Interaction and the Devil Function. In *Proceedings of the Interspeech 2010*, pages 334–337, Makuhari, Chiba, Japan, 2010. (Cited on 102)

[156] M.H. Radfar, A.H. Banihashemi, R.M. Dansereau, and A. Sayadiyan. Nonlinear minimum mean square error estimator for mixture-maximisation approximation. *Electronics Letters*, 42(12):724–725, June 2006. (Cited on 102)

[157] Matt Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic Variational Inference. *arXiv:1206.7051 [cs, stat]*, June 2012. arXiv: 1206.7051. (Cited on 107)

[158] Semih Akbayrak, İsmail Şenöz, Alp Sarı, and Bert de Vries. Probabilistic programming with stochastic variational message passing. *International Journal of Approximate Reasoning*, 148:235–252, September 2022. (Cited on 107)

[159] Yuling Yao, Aki Vehtari, Daniel Simpson, and Andrew Gelman. Yes, but Did It Work?: Evaluating Variational Inference. In *Proceedings of the 35th International Conference on Machine Learning*, pages 5581–5590. PMLR, July 2018. ISSN: 2640-3498. (Cited on 107)

[160] Kenichi Kurihara, M. Welling, and Y. Teh. Collapsed Variational Dirichlet Process Mixture Models. In *IJCAI*, 2007. (Cited on 108)

[161] İsmail Şenöz, Albert Podusenko, Semih Akbayrak, Christoph Mathys, and Bert de Vries. The Switching Hierarchical Gaussian Filter. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 1373–1378, July 2021. (Cited on 108)

[162] Bart van Erp and Bert de Vries. Hybrid Inference with Invertible Neural Networks in Factor Graphs. In *2022 30th European Signal Processing Conference (EUSIPCO)*, 2022. in press. (Cited on 109)

[163] Bart van Erp and Bert de Vries. Online Single-Microphone Source Separation using Non-Linear Autoregressive Models. In *2022 International Conference on Probabilistic Graphical Models (PGM)*, October 2022. accepted. (Cited on 109)

[164] Marco Cox, Thijs van de Laar, and Bert de Vries. ForneyLab.jl: Fast and flexible automated inference through message passing in Julia. In *International Conference on Probabilistic Programming*, Boston, MA, October 2018. (Cited on 110)

[165] T. Minka, J.M. Winn, J.P. Guiver, Y. Zaykov, D. Fabian, and J. Bronskill. Infer.NET 2.7, 2018. Microsoft Research Cambridge. (Cited on 110)

[166] Albert Podusenko, Semih Akbayrak, İsmail Şenöz, Maarten Schoukens, and Wouter M. Kouw. Message passing-Based System Identification for NARMAX Models. In *2022 IEEE Conference on Decision and Control (CDC)*, December 2022. (Cited on 110)

[167] Stephen A. Billings. Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains | Wiley, 2013. (Cited on 110)

[168] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Bethe free energy, Kikuchi approximations, and belief propagation algorithms. *Advances in neural information processing systems*, 13:24, 2001. (Cited on 135)

[169] Thomas Minka. Divergence Measures and Message Passing. Technical report, Microsoft Research, 2005. (Cited on 136)

List of Publications

Journal Articles:

- **Albert Podusenko**, Bart van Erp, Magnus Koudahl, Bert de Vries, *AIDA: An Active Inference-Based Design Agent for Audio Processing Algorithms*, Special issue on Advances in Speech Enhancement using Audio Signal Processing Techniques, Frontiers in Signal Processing, 2022, 32 pages
- **Albert Podusenko**, Wouter M. Kouw, Bert de Vries, *Message Passing-based Inference for Time-Varying Autoregressive Models*, Special issue on Bayesian Inference in Probabilistic Graphical Models, Entropy, 2021, 34 pages

Conference Articles:

- **Albert Podusenko**, Bart van Erp, Dmitry Bagaev, Ismail Senoz, Bert de Vries, *Message Passing-Based Inference in the Gamma Mixture Model*. The 31st IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2022) - Proceedings
- **Albert Podusenko**, Bart van Erp, Dmitry Bagaev, Ismail Senoz, Bert de Vries, *Message Passing-Based Inference in Switching Autoregressive Models*. The 30th European Signal Processing Conference (EUSIPCO 2022) - Proceedings

Not discussed in detail in the dissertation:

- Dmitry Bagaev, Bart van Erp, **Albert Podusenko**, Bert de Vries, *ReactiveMP.jl: A Julia package for reactive variational Bayesian inference*, Software Impacts, 2022
- **Albert Podusenko**, Semih Akbayrak, Ismail Senoz, Maarten Schoukens and Wouter M. Kouw, *Message Passing-based System Identification for NARMAX Models*, The 61th IEEE Conference on Decision and Control (CDC 2022) - Proceedings
- Wouter M. Kouw, **Albert Podusenko**, Magnus T. Koudahl and Maarten Schoukens, *Variational message passing for online polynomial NARMAX identification*, 2022, The 2022 American Control Conference (ACC)

- Bart van Erp, **Albert Podusenko**, Tanya Ignatenko, and Bert de Vries. *A Bayesian Modeling Approach to Situated Design of Personalized Soundscaping Algorithms*, Applied Sciences, 2021
- Ismail Senoz, **Albert Podusenko**, B de Vries, *The Switching Hierarchical Gaussian Filter", "Online Variational Message Passing in Hierarchical Autoregressive Models*, 2021 IEEE International Symposium on Information Theory (ISIT 2021) - Proceedings
- Ismail Senoz, **Albert Podusenko**, Wouter M. Kouw, B de Vries, *Bayesian joint state and parameter tracking in autoregressive models*, Conference on Learning for Dynamics and Control 2020 (L4DC 2020) - Proceedings
- **Albert Podusenko**, Wouter M. Kouw, Bert de Vries, *Online variational message passing in hierarchical autoregressive models*. 2020 IEEE International Symposium on Information Theory (ISIT 2020) - Proceedings
- **Albert Podusenko**, Wouter M. Kouw, Bert de Vries, *Online Variational Message Passing in Autoregressive Models*. 40th WIC Symposium on Information Theory in the Benelux / 9th Joint WIC IEEE SP Symposium on Information Theory and Signal Processing in the Benelux (SITB 2019) - Proceedings

Acknowledgments

In 2016, upon finishing my bachelor's degree in Applied Mathematics, I didn't feel that I had learned as much as I had hoped. That feeling brought me to Kyoto, Japan, where I continued my studies, taking a Master's Degree in Computer Science. At the end of the master's program, I still wasn't sure I was sufficiently academically equipped for 'real life' and that there was still room for improvement. I sensed that obtaining a PhD degree would complete the academic circle.

In March 2018, I stumbled upon the vacancy at BIASlab, TU/e – "PhD position Bayesian machine learning for signal processing applications." I was fascinated by the project, so I wrote to the head of the laboratory, **Bert de Vries**, asking him if we could talk about the position. I had no great expectations of receiving a reply because I had little experience in Bayesian Inference. However, a very encouraging reply was not long in coming And, after several rounds of interviews, I was invited to Eindhoven.

Upon joining BIASlab in September 2018, I was unsure if I would pass the first-year evaluation. The academic level just seemed too high. As is often the case, I was the only one who doubted myself and fortunately for me my colleagues and Bert were confident that my concerns were unmerited. As it turned out, they were right. Bert created a great laboratory environment which enabled me to deal with all the challenges.

Bert. Over the past four years, you have supported me greatly when I was experiencing either personal or academic difficulties. You gave me an incredible amount of invaluable advice that I will carry with me throughout my life (as well as my newly developed phobia of the colour blue induced by all your comments on Overleaf). You are one of the most interesting people I have ever met. Thank you for providing me with an opportunity to join BIASlab. You have surrounded me with people from whom I learned much and who have become my friends. I am glad that at the end of my PhD I also found a very good friend in you.

I want to thank the members of my committee, **Martijn Wisse**, **Siep Weiland**, **Rik Vullings**, **Sander Stuijk**, and **Wouter Marco Kouw**, for reading this dissertation and providing me with wonderfully constructive feedback.

This thesis would not have been possible without the support of my friends and family. So, I would like to convey my gratitude to all of those people who facilitated

the production of my PhD dissertation.

Ismail, it is hard to imagine that there was a time when we did not know each other. We 'clicked' immediately and became close friends. Thank you for always standing up for me. You are an incredibly talented researcher who, in the best traditions of the genre, does not realize this. I am grateful for having a chance to meet all of your loving family, who welcomed me like one of their own during my multiple visits to Turkey. **Semih**, throughout our PhDs, we shared numerous ups and downs. We (almost) bravely grappled with those difficulties and made it through. Your presence in my life has made me a better researcher and, most notably, a better person. It was an honour for me to attend your and Melike's wedding. **Dmitry**, thank you; our laboratory has taken research to another level. You are an exceptional researcher, coder, and friend. You are always ready to help, even to the detriment of yourself. We have done a great deal together, from research to extreme sports. We even survived an existential threat together, which is not something you share with many people.

Bart, we met during your master's when I was appointed as your supervisor (luckily, you did not really need one!). I am glad you have eventually joined BIASlab as a PhD student. You are a great team member, and it has been a pleasure working with you. **Magnus**, you have always been a professional and reliable colleague. You are one of few in our lab who has dared to delve into Active Inference. I respect it very much. And we did what seemed impossible at one point — we did AIDA. **Martin**, you are a great team player. You are one of the most hard-working people I have ever met. I will never forget our late-evening work, followed by long nights in a jazz bar.

Wouter, you were my lifeline in the first year of my PhD. Perhaps you are the only one who went through the endless stream of my derivations, despite your workload. And yet our the all-nighter hackathon will remain my most vivid memory. **Thijs**, thank you for navigating me through factor graphs and ForneyLab. I enjoyed every conversation we had on numerous matters. It was a pleasure to learn from you and, of course, thank you for the introduction to the conservation of misery.

I would also like to thank other present and former BIASlab members for their excellent team spirit, which motivated me to do a better job, namely: **Tim**, **Mykola**, **Sepideh**, **Chengfeng**, **Hoang**, **Wouter Jr.**, **Patrick**, **Ivan**, **Marco**, and **Alp**.

Thanks to **Jan** for creating a great working environment. I would also like to thank the secretariat of the SPS group, **Anja**, **Carla**, **Emerald**, and **Judith**, for their help on non-technical matters.

During these four years in the Netherlands, I was lucky to be surrounded by wonderful people. They have made my journey more enjoyable. Thank you, **Anton**, **Anastasia**, **Claudia**, **Davide**, **Daniella**, **Despina**, **Ekaterina**, **Gleb**, **Ilya**, **Irene**, **Iliana**, **Nastya**, **Mariia**, **Maurice**, **Maxim**, **Marco**, **Ousmane**, **Omar**, **Pavle**, **Stella**, **Sasha**, **Tanya**, **Len**, **Lisa**, and **Yunus**. I will always cherish the memories of the time we have spent together.

Dear **Jiali**, you have filled the years of my PhD with love and care. You have witnessed all my ups and downs, yet you never let me lose heart. I have learned so much from you. Probably much more than you learned from me. Your life optimism, your kindness, and your faith in me helped me finish this thesis. Thank you for always being my shelter in a storm.

To my buddy, my housemate **Branislav**. Thank you for supporting me during this endeavour. Although we met under bizarre circumstances, we have built a strong friendship. I admire your readiness to help, no matter how formidable it may be for you.

To my friends **Anton** and **Misha**. These two gentlemen pursued a very similar road starting in Gymnasium №1, Nakhodka, to obtaining a PhD in the top universities of Europe. You have helped me on multiple occasions throughout my PhD. I will never forget that. Thank you.

To my Italian friends **Alessandro** and **Lorenzo**, who I met during my master's program in Japan. Thank you for filling my life with joy, fun, and wit during my studies. I am grateful for each of the memories we shared over the past six years.

During the time of my PhD studies, for various external reasons, I did not have the opportunity to visit my friends as often as I would have liked. Nonetheless, I stayed in touch with my good old friends: **Michael**, **Oleg**, **Andrew**, **Kirill**, **Denis**, **Vitaly**, **Makasyava**, **Dipak**, **Minella**, **Boris**, **Lyoha**, **Maks**, **Alexander**, **Suleiman**, **Andrei**, and **Elina**. I am thankful for you guys for just being there these past four years. **Vika**, thank you for believing in me throughout these years, no matter what.

To my former supervisor **Professor Tanev**. Thank you for pushing me towards pursuing a PhD and your unwavering support during this period. You have a fantastic ability to say the right words at the right time, which stopped me from giving up in moments of despair.

Last but not least, I would like to thank my family. This work would not have been possible without their love and constant support. To my lovely parents, my best friends **Oleg** and **Viktoriia**. Even though we are thousands km away from each other, I feel that you are always around. You taught me that learning never ends. I thank you for all the effort you have put into my education. To my brother, **Denis**, the kindest person I know. I have always felt your support wherever you are. Your music has encouraged me throughout my life and PhD was not an exception. To my lovely grandmother, **Lidia**, when I was graduating from high school, you were celebrating that you had lived to see this. To remind you, ten years have passed since that moment. You have always looked out for me and backed all my decisions. This work is dedicated to you.

Albert Podusenko
October, 2022

Biography

Albert Podusenko was born on May 6th, 1994, in Nakhodka, Primorsky krai, Russian Far East. Already in middle school, he developed an immense interest in computer science. With the understanding that there were exciting developments in this field to come, it became apparent that Albert would be curious about exploring it.

In 2016 he received a BSc degree (cum laude) from the faculty of Applied Mathematics and Control Processes at Saint-Petersburg State University. Upon completion and successfully graduating, Albert was awarded a student scholarship by the Ministry of Education, Culture, Sports, Science, and Technology of Japan for a two-year MSc degree program. In 2018 he received an MSc degree from the Department of Information and Computer Science, Doshisha University.

The same year he joined the Bayesian Intelligent Autonomous Systems Lab (BIASlab) at Eindhoven University of Technology, where he started working towards a PhD degree in the Signal Processing Systems group under the supervision of Professor Bert de Vries. His research mainly focused on Bayesian inference techniques for hierarchical dynamical systems. This dissertation includes some of the main results of this PhD research.

Since September 2022, Albert has been working as a researcher in the BIASlab. His research interests include intelligent systems, probabilistic graphical models, and time-series modeling. Besides research, Albert is very enthusiastic about extreme sports such as breakdancing, skydiving, snowboarding, etc.