

# A Factor Graph Approach to Active Inference

***Citation for published version (APA):***

Koudahl, M. T. (2024). *A Factor Graph Approach to Active Inference*. [Phd Thesis 1 (Research TU/e / Graduation TU/e), Electrical Engineering]. Eindhoven University of Technology.

***Document status and date:***

Published: 23/01/2024

***Document Version:***

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

***Please check the document version of this publication:***

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

***General rights***

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

[www.tue.nl/taverne](http://www.tue.nl/taverne)

***Take down policy***

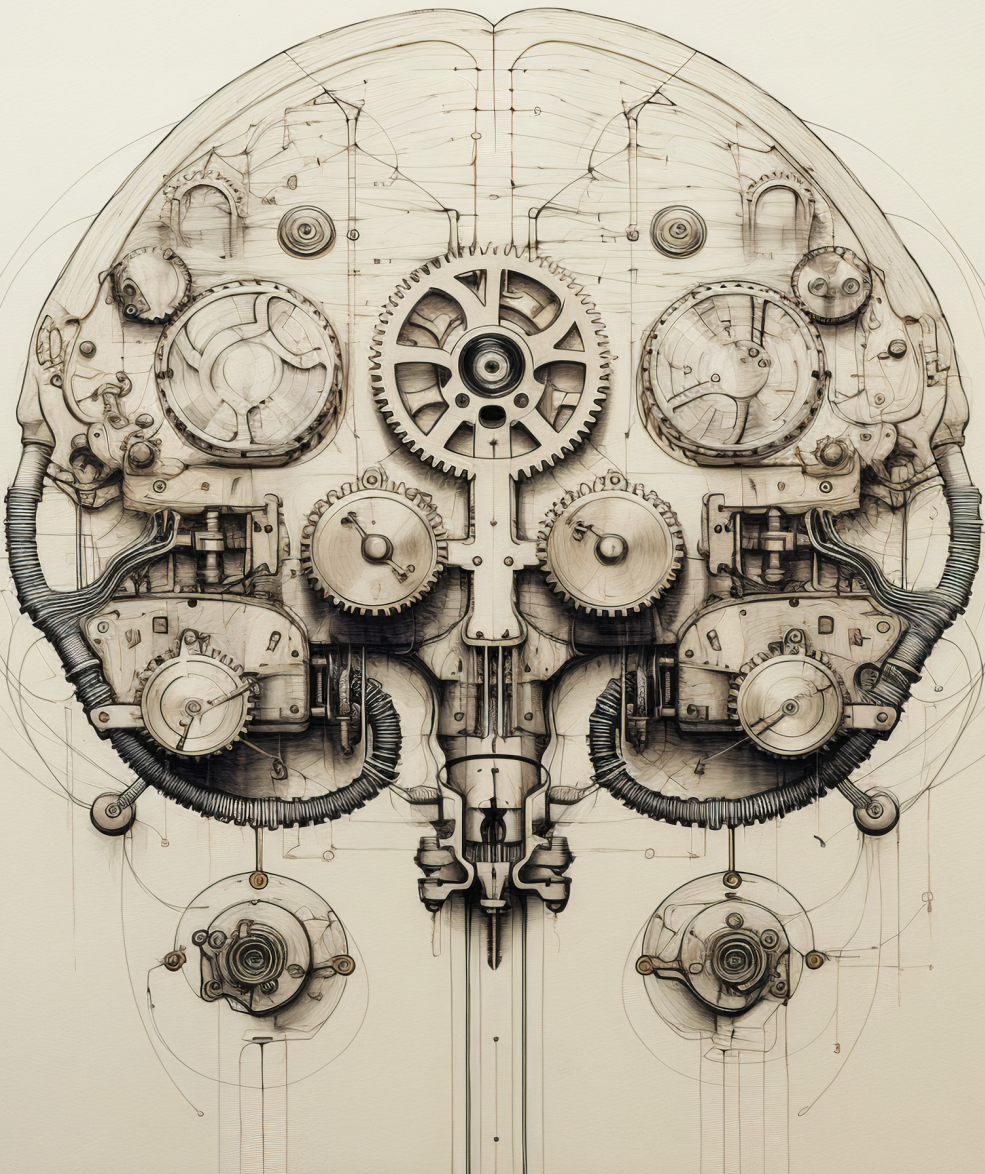
If you believe that this document breaches copyright please contact us at:

[openaccess@tue.nl](mailto:openaccess@tue.nl)

providing details and we will investigate your claim.

# A Factor Graph Approach to Active Inference

Magnus Tønder Koudahl





# A Factor Graph Approach to Active Inference

Magnus Tønder Koudahl

Copyright © 2023 by Magnus Tønder Koudahl. All Rights Reserved.  
No parts of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the prior written permission of the author.

A Factor Graph Approach to Active Inference by Magnus Tønder Koudahl.

A catalogue record is available from the Eindhoven University of Technology Library

ISBN 978-90-386-5923-7

Keywords: Active Inference, Epistemics, Free Energy, Factor Graphs, Message Passing

The work in this dissertation was partially funded by GN Hearing and by the NWO Perspective programme Efficient Deep Learning.  
Printed by Ridderprint

# A Factor Graph Approach to Active Inference

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus prof.dr. S.K. Lenaerts, voor een commissie aangewezen door het College voor Promoties, in het openbaar te verdedigen op **dinsdag 23 januari 2024 om 16:00 uur**

door

Magnus Tønder Koudahl

geboren te Kopenhagen, Denemarken

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

voorzitter: prof.dr.ir. P.G.M Baltus  
1e promotor: prof.dr.ir. A. de Vries  
co-promotor: dr. T.W. van de Laar  
leden: prof.dr. R. Moran (Kings College London, UK)  
prof.dr.ir. J.P.M.G. Linnartz  
dr. S. Thill (Radboud Universiteit, Nijmegen)  
dr. A. Özçelikkale (Uppsala Universitet, Sweden)

# Abstract

The brain is the most sophisticated and complex information-processing system that we know of. Consequently, mimicking the brain's capabilities is in many ways the ultimate goal of machine learning (ML). Yet the dominant ways in which we design modern ML systems are skewed towards tools and algorithms that are first and foremost designed to be efficiently implementable and scalable on GPUs. Predictably, even though much of the initial inspiration for ML came from the brain, most ML systems end up functioning quite differently from brains. This presents a bit of a conundrum: On the one hand, we know that brains can do all the things we are trying to make our ML systems do. On the other, we are building our ML systems using very different approaches to how we believe the brain works.

The main idea explored in this dissertation is that if we wish to make artificial systems that mimic the brain, a good starting point is arguably to mimic the brain. However to turn this rather vague statement into actionable research questions, we need to operationalise what "mimicking the brain" actually means. To this end, we rely on the framework of Active Inference (AIF). AIF is a theory from the field of computational neuroscience that proposes that the brain is inherently generative in nature. From the point of view of AIF, the brain entails a generative model of its environment and is constantly trying to make sure that it can accurately predict the state of the world around it.

Crucially for our purposes, AIF can be formulated using equations which makes it amenable to be run *in silico*. This provides our link between the world of neuroscience and that of ML. If we can translate the theoretical neuroscience of AIF to a form that can be simulated on computers, can we then design new



ML systems that inherit the desirable qualities of the brain posited by AIF?

To accomplish this, we need to turn AIF into a practically viable engineering tool. This goal forms the central focus of the dissertation. We approach the problem by first casting AIF as message passing (MP) on a factor graph representation of a generative model. Once in this form, we are much more free in the problems that can be addressed using AIF. We demonstrate this by extending AIF to work with generative models already popular for practical applications such as linear Gaussian dynamical systems and Gaussian process classifiers. Once we have demonstrated that AIF can indeed be applied to practical engineering problems, we propose a new interpretation of AIF that is specifically targeted at engineering applications. This takes the form of a new type of factor graph which provides accurate notation for AIF and associated MP update rules that together provide a scalable, distributed algorithm for AIF. In doing so, we provide a path forward for designing practical tools that can make AIF viable for engineering as well as being a neuroscientific theory.

# Contents

<b>Abstract</b>	<b>v</b>
<b>List of Symbols</b>	<b>1</b>
<b>1 Introduction</b>	<b>7</b>
1.1 Motivation . . . . .	7
1.2 The Free Energy Principle . . . . .	10
1.2.1 Variational Free Energy Minimisation . . . . .	11
1.3 Forney-style Factor Graphs . . . . .	14
1.4 Inference on Graphs and Bethe Free Energy . . . . .	15
1.4.1 Belief Propagation . . . . .	16
1.4.2 Variational and Hybrid Message Passing . . . . .	18
1.5 Active Inference . . . . .	19
1.5.1 Expected Free Energy . . . . .	21
1.6 Research Questions . . . . .	24
1.7 Summary of Contributions . . . . .	26
1.8 Accountability, Outline and Suggested Reading Orders . . . . .	27
<b>2 A Message Passing Perspective on Planning under Active Inference</b>	<b>31</b>
2.1 Introduction . . . . .	32
2.2 Generative Model and Inference . . . . .	32
2.3 Expected Free Energy . . . . .	34
2.4 Sophisticated Inference . . . . .	38
2.5 Advantages of the Message Passing Perspective . . . . .	39

2.6	Conclusions . . . . .	40
<b>3</b>	<b>On Epistemics in Expected Free Energy for Linear Gaussian State Space Models</b>	<b>41</b>
3.1	Introduction . . . . .	42
3.2	Exploration and Exploitation . . . . .	44
3.3	Generative Model . . . . .	45
3.4	Perception as Bayesian Filtering . . . . .	48
3.5	Action Selection under Active Inference . . . . .	54
3.5.1	Computing G - Expected Free Energy . . . . .	55
3.5.2	Mutual Information Computation . . . . .	60
3.5.3	Pure Exploration as a function of Additive Control Signals . . . . .	60
3.5.4	Pure Exploration as a function of Multiplicative Control Signals . . . . .	62
3.5.5	Instrumental Value and Expected Free Energy . . . . .	63
3.6	Experiments . . . . .	65
3.6.1	Pure Epistemics for Additive Controls . . . . .	65
3.6.2	Pure Epistemics for Multiplicative Controls . . . . .	66
3.6.3	Lack of Epistemics for Expected Free Energy . . . . .	67
3.7	Discussion . . . . .	70
3.8	Conclusions . . . . .	71
<b>4</b>	<b>AIDA: An Active Inference-based Design Agent</b>	<b>73</b>
4.1	Introduction . . . . .	74
4.2	Problem Statement and Proposed Solution Approach . . . . .	77
4.2.1	Automated Hearing Aid Tuning by Optimisation . . . . .	77
4.2.2	Situated Hearing Aid Tuning with the User in-the-loop . . . . .	77
4.3	Model Specification . . . . .	81
4.3.1	Acoustic Model . . . . .	82
4.3.2	AIDA's User Response Model . . . . .	87
4.4	Solving Tasks by Probabilistic Inference . . . . .	88
4.4.1	Inference for Context Classification . . . . .	88
4.4.2	Inference for Trial Design of HA Tuning Parameters . . . . .	90
4.4.3	Inference for Executing the Hearing Aid Algorithm . . . . .	92
4.5	Experimental Verification and Validation . . . . .	93
4.5.1	Context Classification Verification . . . . .	94
4.5.2	Trial Design Verification . . . . .	95
4.5.3	HA Algorithm Execution Verification . . . . .	102
4.5.4	Validation Experiments . . . . .	105

---

4.6	Related Work . . . . .	107
4.7	Discussion . . . . .	109
4.8	Conclusions . . . . .	111
<b>5</b>	<b>Realising Synthetic Active Inference Agents</b>	<b>113</b>
5.1	Introduction . . . . .	114
5.2	The Lagrangian Approach to Message Passing . . . . .	115
5.2.1	Bethe Free Energy and Forney-style Factor Graphs . . . . .	116
5.3	Defining Epistemic Objectives . . . . .	120
5.3.1	Constructing a Local Epistemic Objective . . . . .	121
5.3.2	Generalised Free Energy . . . . .	122
5.4	LAIF - Lagrangian Active Inference . . . . .	124
5.5	Constrained Forney-style Factor Graphs . . . . .	126
5.5.1	Factorisation Constraints . . . . .	128
5.5.2	Form Constraints . . . . .	130
5.5.3	$\delta$ -constraints and Data Points . . . . .	131
5.5.4	Moment Matching Constraints . . . . .	132
5.5.5	P-substitution on CFFGs . . . . .	133
5.5.6	CFFG Compression . . . . .	134
5.6	General GFE-Based Message Updates . . . . .	140
5.6.1	Local Lagrangian . . . . .	142
5.6.2	Local Stationary Solutions . . . . .	142
5.6.3	Message Updates . . . . .	144
5.6.4	Convergence Considerations . . . . .	145
5.7	Application to a Discrete-Variable Model . . . . .	146
5.7.1	Goal-Observation Submodel . . . . .	146
5.7.2	GFE-Based Message Updates . . . . .	147
5.8	Classical AIF and the Original GFE Algorithm as Special Cases of LAIF . . . . .	147
5.8.1	Reconstructing the Original GFE Method . . . . .	151
5.9	LAIF for Policy Inference . . . . .	153
5.9.1	Model Specification . . . . .	153
5.10	Conclusions . . . . .	160
<b>6</b>	<b>Conclusions and Future Outlook</b>	<b>163</b>
6.1	Future Outlook . . . . .	167

<b>A</b>	<b>Appendices for Chapter 3</b>	<b>169</b>
A.1	Perception as Bayesian filtering . . . . .	169
A.2	Linearly Related Gaussian variables . . . . .	170
A.3	Mutual Information Bound . . . . .	172
A.4	Mutual information derivation . . . . .	173
<b>B</b>	<b>Appendices for Chapter 4</b>	<b>177</b>
B.1	Probabilistic Model Overview . . . . .	177
B.1.1	Acoustic Model . . . . .	177
B.1.2	AIDA's User Response Model . . . . .	178
B.2	Inference Realisation . . . . .	179
B.2.1	Realisation of Inference for Context Classification . . . . .	179
B.2.2	Realisation of Inference for Trial Design . . . . .	181
<b>C</b>	<b>Appendices for Chapter 5</b>	<b>185</b>
C.1	Expected Free Energy . . . . .	185
C.2	VFE and GFE . . . . .	186
C.3	Proofs of Local Stationary Solutions in Section 5.6.2 . . . . .	188
C.3.1	Proof of Lemma 1 . . . . .	188
C.3.2	Proof of Lemma 2 . . . . .	188
C.3.3	Proof of Lemma 3 . . . . .	189
C.4	Proofs of Message Update Expressions in Section 5.6.3 . . . . .	190
C.4.1	Proof of Theorem 1 . . . . .	190
C.4.2	Proof of Theorem 2 . . . . .	191
C.4.3	Proof of Corollary 1 . . . . .	192
C.5	Derivations of Message Updates in Figure 5.21 . . . . .	192
C.5.1	Intermediate Results . . . . .	192
C.5.2	Average Energy . . . . .	194
C.5.3	Message One . . . . .	194
C.5.4	Direct Result for Message Two . . . . .	195
C.5.5	Indirect Result for Message Two . . . . .	195
C.5.6	Direct Result for Message Three . . . . .	196
C.6	The Transition Mixture Node . . . . .	196
	<b>Acknowledgments</b>	<b>217</b>
	<b>Biography</b>	<b>221</b>
	<b>List of Publications</b>	<b>223</b>



## Notational Conventions


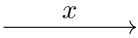
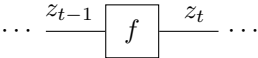
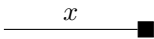
Notation	Chapter	Explanation
$a$		A generic variable or scalar.
$\hat{a}$		A variable $a$ constrained to follow a $\delta$ -distribution centered at the value $\hat{a}$ .
$\mathbf{a}$		Vector or vector valued variable.
$\mathbf{a}$		Sequence or set of variables, ex $\mathbf{a} = (a_1, a_2 \dots a_T)$ .
$\mathbf{a}_{t:T}$		Sequence of variables indexed by $t$ up to $T$ .
$\mathbf{A}$		The matrix $A$ .
$\mathbf{A}(z)$		The matrix $A$ as a function of $z$ .
$x_t$		Observation variable at time $t$ .
$z_t$		Latent state variable at time $t$ .
$u_t$		Action or control variable at time $t$ .
$c_t$	Chapter 5	Goal prior parameters at time $t$ .
$o_t$	Chapter 4	Context variable at time $t$ .
$y_t$	Chapter 4	Hearing aid output signal at time $t$ .
$r_t$	Chapter 4	User response at time $t$ .
$n_t$	Chapter 4	Noise at time $t$ .
$\mathbf{e}_i$	Chapter 4	Cartesian unit vector with 1 in the $i$ 'th position and 0's everywhere else.
$\theta$		Parameter variable.
$\phi$		Parameter variable.
$\mathcal{G}$		A generic graph.
$\mathbf{I}$		Identity matrix of appropriate dimensionality.
$\mathbf{0}$		Appropriately sized vector of 0's.
$Z$		Normalisation constant.
$T$		Planning horizon of policy.
$\text{diag}(\cdot)$		Returns the diagonal elements of a matrix as a column vector.
$\Phi(\cdot)$	Chapter 4	Cumulative Gaussian distribution function.
$\mu(\cdot)$		A belief propagation or GFE-based message.
$\nu(\cdot)$		A variational (VMP) message.
$m_l(\cdot)$	Chapter 4	Mean function of the $l$ 'th Gaussian Process.
$K_l(\cdot, \cdot)$	Chapter 4	Kernel function of the $l$ 'th Gaussian Process.
$f(\cdot)$		Generic function.
$s_a$	Chapter 5	Variables pertaining to the node $a$ on a (C)FFG.
$q_a(s_a)$	Chapter 5	Variational distribution pertaining to the node $a$ on a (C)FFG.
$s_i$	Chapter 5	Variable pertaining to the edge $i$ on a (C)FFG.
$q_i(s_i)$	Chapter 5	Variational distribution pertaining to the edge $i$ on a (C)FFG.
$\mathcal{Q}$		Family of variational distributions, $q \in \mathcal{Q}$ .
$d_i$	Chapter 5	Degree of the $i$ 'th edge on a (C)FFG.
$f_a(s_a)$	Chapter 5	Generic factor function of $s_a$ .

## Probability Distributions, Free Energies and Information-theoretic Quantities

Notation	Explanation
$Cat(x   z)$	Categorical distribution over $x$ parameterised by $z$ .
$\mathcal{N}(\mathbf{x}   \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\mathbf{x}$ with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ .
$Dir(\mathbf{x}   \boldsymbol{\alpha})$	Dirichlet distribution over $\mathbf{x}$ parameterised by concentration parameters $\boldsymbol{\alpha}$ .
$Ber(x   p)$	Bernoulli distribution over $x$ parameterised by $p$ .
$\Gamma(x   \alpha, \beta)$	Gamma distribution over $x$ parameterised by shape $\alpha$ and rate $\beta$ .
$\delta(x - \hat{x})$	Dirac or Kronecker delta centered at $\hat{x}$ .
$F$	A variational or Bethe free energy functional.
$F[q; u]$	A variational or Bethe free energy functional as a functional of $q$ and a function of $u$ .
$G[q; u_t]$	Expected Free Energy as a functional of $q$ and a function of $u_t$ .
$G(u_t)$	Expected Free Energy as a function $u_t$ and given that $q = q^*, \arg \min_{q \in \mathcal{Q}} G[q, u_t]$ .
$\mathcal{G}[q; u_t]$	Generalised Free Energy as a functional of $q$ and a function of $u_t$ .
$H[x]$	Entropy of the distribution $p(x)$ .
$H[x   z]$	Conditional entropy of $p(x   z)$ given $p(z)$ .
$KL[p(x)    q(x)]$	Kullback-Leibler divergence between $p(x)$ and $q(x)$ .
$I[x, z]$	Mutual information between $p(x)$ and $p(z)$ .



## Forney-style Factor Graph Notation

Notation	Explanation
	A factor node with node function $f$ .
	An edge corresponding to the variable $x$ . Arrowheads only serve to disambiguate between "forwards" and "backwards" directions of messages along the edge and do not imply that the underlying graph is directed.
	An ellipsis ( $\dots$ ) indicates that the graph is extended by similar, repeating sections.
	An edge representing a variable that has been clamped to a fixed value.

# Acronyms

<b>EFE</b>	expected free energy
<b>BFE</b>	Bethe free energy
<b>CBFE</b>	constrained Bethe free energy
<b>VFE</b>	variational free energy
<b>GFE</b>	generalised free energy
<b>AIF</b>	Active Inference
<b>ML</b>	machine learning
<b>FEP</b>	Free Energy Principle
<b>NESS</b>	non-equilibrium steady state
<b>FFG</b>	Forney-style factor graph
<b>CFFG</b>	constrained Forney-style factor graph
<b>BP</b>	belief propagation
<b>VMP</b>	variational message passing
<b>MP</b>	message passing
<b>EP</b>	expectation propagation
<b>SI</b>	sophisticated inference
<b>LAIF</b>	Lagrangian Active Inference
<b>POMDP</b>	partially observed Markov decision process
<b>AR</b>	auto-regressive
<b>TVAR</b>	time-varying auto-regressive
<b>KL</b>	Kullback-Leibler divergence
<b>HA</b>	hearing aid

(Continued)

<b>GP</b>	Gaussian process
<b>GPC</b>	Gaussian process classifier
<b>MI</b>	mutual information
<b>LGDS</b>	linear Gaussian dynamical system

# Chapter 1

## Introduction

*"Since all models are wrong, the scientist must be alert to what is importantly wrong. It is inappropriate to be concerned about mice when there are tigers abroad."*

– George Box, 1976

### 1.1 Motivation

One of the few systems we know of that is (arguably) intelligent, is the human brain. Every minute of every day we juggle multiple, complicated tasks without a second thought and with a very high level of mastery. Causal reasoning, parsing disparate sensory streams, controlling complicated locomotion, and monitoring and maintaining homeostatic bounds are all handled effortlessly to the point where they are taken for granted. At higher levels of abstraction, we can navigate complicated interpersonal relationships, make long-term complicated plans and execute them over timescales of multiple years, and perform impressive feats of abstract reasoning. We can reason by analogy and generalise across tasks in a remarkably data-efficient manner. When presented with a novel task, we are able to draw on past experiences and quickly perform a series of experiments to gather information about the new task efficiently. As an example, think back to the last time you lost your way in an unfamiliar place.

Most likely you were able to come up with a strategy for finding your way back to familiar ground whether that was through looking for landmarks, making use of the sun or relying on a learned sense of direction. The point being, that brains are very efficient at gathering information and can quickly solve novel tasks from very little data.

Modern deep machine learning (ML) systems share very few of these characteristics. While ML has undoubtedly produced impressive results in all the domains mentioned above, the models used generally require very large datasets and do not necessarily generalise well outside of the specific task they were trained on. Even recent large language models only start to show some cross-domain capability once they have consumed quite literally the entire internet and the training process has run for several days or weeks. All of this is very unlike the way our everyday brains seem to work. This presents an interesting question: How to make our ML systems behave more like brains?

Active Inference (AIF) is one avenue down which we find an answer to this puzzle. AIF is a corollary of the Free Energy Principle (FEP) that provides a theoretical framework describing the kinds of computations that a brain *might* do. The core thesis is that a brain entails a generative model of the world it inhabits and its core task is making sure that model is fit for purpose. In essence, the claim is that we are in the business of making sense of the world around us by constantly trying to predict what is going to happen next. A good model is one that makes accurate predictions.

The brain accomplishes this task by minimising a variational free energy (VFE) functional<sup>1</sup>. The VFE is an objective that scores both the accuracy of the model in predicting how the world is going to evolve as well as how complex the model is. The ideal model is as simple as possible but not simpler — after all, it should still make accurate predictions.

Brains are also housed in bodies, meaning we have ways of interacting with the world around us. This means we are equipped with two distinct ways of ensuring our model makes accurate predictions. The first is to simply change our model when faced with evidence that it is insufficient. In everyday terms, we know this as perception when it happens fast and learning when it happens more slowly. The second way is to acknowledge that we are more than passive observers of events around us and act on the world to make it conform to the predictions of our model. If we change the data so it agrees with our model, it's almost a tautology to say the model is a good fit for the data.

As an example, we can consider a person on her way to an important ap-

---

<sup>1</sup>We are using VFE as a noun referring to a specific functional, to be defined in Section 1.2.1.

pointment. With a route planned out, it is reasonable to assume her generative model predicts that she will make her appointment on time. However, she now learns that her bike has suffered a flat tire! Flat tires are not predicted by her model and hence constitute an error that must be resolved through one of the two pathways outlined.

First, we can examine how to resolve the situation by updating the model. Updating the model in this case means changing the prediction. Faced with a flat tire, her prediction that she is going to be on time for her appointment is no longer accurate. She can resolve this prediction error by accepting that she will be late and update her model accordingly.

However, she can also choose to resolve the prediction error by acting on the world, for instance by calling a taxi. In this case, the prediction that she will arrive on time is unaltered. Instead, she intervenes on the world to make it conform to her predictions, regardless of the flat tire. A central feature of AIF is that it allows us to place this second pathway front and centre and use it to design systems that *act* on their environment in intelligent ways.

The previous example brings us to the central focus of this dissertation. At a fundamental level, our machines are still very different from brains, yet many of the qualities of the brain are highly desirable for machine learning systems. Therefore, in order to bring the benefits of AIF to machine learning, we need to find ways to translate the theoretical neuroscience of AIF to a form that is amenable to running in software instead of wetware. We need to make the theory *practical* so that it can move out of the ivory tower and into our ordinary lives. AIF needs to become an engineering tool as much as a neuroscientific theory. This task forms the core of this dissertation and can be stated as the following, overarching research theme:

**Theme:** *Can we develop a practical toolset for describing and constructing artificial Active Inference agents?*

So how can we go about making our ML systems function more like the person in the above scenario? To properly set the stage for answering these questions, we need to delve deeper into the details of the FEP and AIF. The following sections will introduce the necessary background material before moving on to the concrete research questions that form the foundation for the work in this dissertation.

## 1.2 The Free Energy Principle

1 To introduce the Free Energy Principle, we can start with an observation. Namely that most things in nature, if left alone, tend to disappear. Mountains erode, stars collapse and living things perish. All of this can be thought of as the thing in question - stars, mountains, life - settling into some kind of steady state equilibrium with its environment. Once at this steady state, nothing further happens. Once a star has collapsed in on itself, it will not spontaneously expand and start shining bright again.

This phenomenon is a consequence of all things being made up of "stuff" that needs to be in a very particular configuration for us to say that thing exists. A mountain is only a mountain as long as all the "stuff" that makes it up is stacked up high. If exposed to the elements for extended periods of time, the mountain will wear down and all the "stuff" that used to be a mountain will become sand, rocks, silicon chips, wedding rings, iron supplements, and a myriad of other things.

When we consider biological systems from this point of view, we are faced with a conundrum. If you look straight down, you will see a biological system that, like the mountains and the stars, is made out of "stuff". Yet, in order for you to persist, all of this "stuff" has to stay in a very particular configuration that is consistent with you being you. This configuration is also very, very far from being at an equilibrium. In other words, rather than settling into an equilibrium, you persist as a non-equilibrium steady state (NESS) in stalwart defiance of nature's inherent tendency to break things down. You accomplish this through interacting with the world around you, seeking out food and shelter so as to be able to keep all of your "stuff" in a reasonable configuration.

From the point of view of the brain as a single, biological system, we can start to carve up the world according to these ideas. On the one side, we have the brain itself - the internal states of the system. Its objective is to maintain its NESS and by extension stay alive. On the other side, we have the world at large with everything in it - the external states. In between the two we have a boundary - called a Markov blanket - made up of two types of channels. One type we can call "sensory" in nature. They comprise all the pathways through which the world can influence our system, for example, our eyes and ears and the touch receptors in our skin. The other type, which we can call "active", comprises all the pathways through which our system can influence the world, such as our hands and feet. We illustrate this partitioning of the world in Fig. 1.1.

In Fig. 1.1 we have added arrows that indicate which states can influence each other. For example, the arrow Active  $\rightarrow$  External states indicates that active

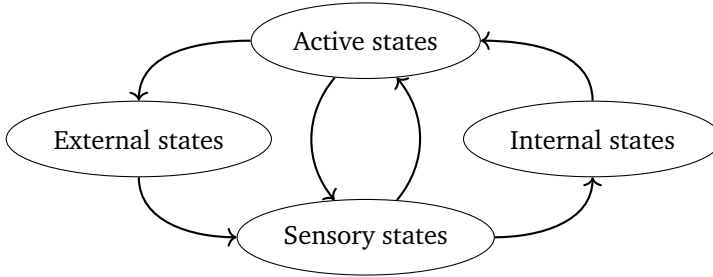


Figure 1.1: The world partitioned according to the FEP, focusing on a single system.

states can influence external states but not vice versa.

With this partition in hand, we are now in a position to start formalising the task of a system trying to maintain a NESS under the FEP. Concretely, the FEP casts this idea of maintaining a NESS as a process of free energy minimisation.

### 1.2.1 Variational Free Energy Minimisation

"Free energy" in this context refers primarily to the variational free energy (VFE), a construct originating in statistical physics that has since found applications within several adjacent fields such as neuroscience, machine learning, and signal processing. To construct a VFE to minimise, we first need a model which, for our purposes, is simply a positively valued function of some variables  $f(\mathbf{s})$ . The second ingredient we require is a probability distribution  $q(\mathbf{s})$  which we will refer to as the *variational* distribution. With these two in hand, we can define the VFE as

$$F[q] = \int q(\mathbf{s}) \log \frac{q(\mathbf{s})}{f(\mathbf{s})} d\mathbf{s}. \quad (1.1)$$

Throughout this section, we will work with continuous variables. For discrete variables, integration can be replaced with summation. VFE minimisation is then concerned with the problem of finding

$$q^*(\mathbf{s}) = \arg \min_{q \in \mathcal{Q}} F[q] \quad (1.2)$$



with  $\mathcal{Q}$  constraining the set of available candidates for  $q$ . The optimal solution is given by

$$q^*(s) = p(s) = \frac{f(s)}{Z} \quad (1.3)$$

where  $Z = \int f(s)ds$  is a normalisation constant and  $p(s)$  is the exact posterior. We use the term *posterior* to refer to the  $q^*(s)$  we obtain after minimising some free energy functional. Unfortunately, calculating the normalisation constant  $Z$  in (1.3) is often difficult or even intractable. To remedy this issue, we can use  $\mathcal{Q}$  to restrict the family of  $q$ 's we consider in order to make the problem tractable again at the cost of obtaining an approximate solution instead of the exact one. In this case, the solution becomes

$$F[q^*] = \int q^*(s) \log \frac{q^*(s)}{f(s)} ds = \text{KL}[q^*(s) || p(s)] - \log Z \geq -\log Z. \quad (1.4)$$

where we are left with a Kullback-Leibler divergence (KL) between the exact posterior  $p(s)$  and our best approximation  $q^*(s)$ . Practically, this KL-term scores the divergence between our approximation and the optimal solution. Since the KL is non-negative, we see that Eq. (1.4) provides an upper bound to  $-\log Z$ . The term  $-\log Z$  is then known as the negative log evidence or the *surprisal*.

To see how this procedure ties back to the ideas presented at the beginning of Section 1.2, we can assume that the system we consider entails a model of external states that is also consistent with it remaining at NESS and that the variational distribution  $q$  is encoded by the systems internal states. One example could be a model that consistently predicts a body temperature of 37 degrees Celsius. In this case, VFE minimisation will ensure that as data arrives through sensory states, the system can continually update its (approximate) posterior over the state of the outside world in a manner that is also consistent with remaining at NESS. Or in other words, if the system can keep  $q$  (its internal states) close to  $f$  (a model of the world that is consistent with remaining at NESS, we can understand "staying alive" as a process of (variational) free energy minimisation. If your model predicts a body temperature of 37 degrees and you start to feel cold, you put on a jacket to resolve the error. If the system makes accurate predictions and predicts that it will stay alive, staying alive becomes a self-fulfilling prophecy.

To get an idea of how this practically manifests, we can consider an example. The goal of this exercise is to show how we can get Bayes rule as a special case of

free energy minimisation when we update our beliefs following an observation. Let us start with the model

$$p(x, z) = p(x|z)p(z), \quad (1.5)$$

where  $x$  is some observation and  $z$  is a latent variable. For instance,  $z$  could be encoding body temperature and  $x$  the signals coming from cold receptors in your skin. With this model, the corresponding VFE is

$$F[q] = \iint q(x, z) \log \frac{q(x, z)}{p(x, z)} dx dz. \quad (1.6)$$

Now we write  $q(x, z) = q(z | x)q(x)$ , which gives us

$$F[q] = \iint q(z | x)q(x) \log \frac{q(z | x)q(x)}{p(x, z)} dx dz. \quad (1.7)$$

Let us now assume that we have gathered an observation  $\hat{x}$  and wish to incorporate this information into our model. To continue with our example, this could represent a sensation of feeling cold. To incorporate this new piece of information, we constrain  $q(x)$  to be equal to our observation  $\hat{x}$  by setting  $q(x) = \delta(x - \hat{x})$  where  $\delta(\cdot)$  denotes the Dirac- $\delta$  function. Plugging this into the expression for the VFE in Eq. (1.6), we find

$$F[q] = \iint q(z | x)\delta(x - \hat{x}) \log \frac{q(z | x)\delta(x - \hat{x})}{p(x, z)} dx dz. \quad (1.8)$$

Now we can make use of the sifting property of the Dirac- $\delta$  to obtain

$$F[q] = \int q(z | \hat{x}) \log \frac{q(z | \hat{x})}{p(\hat{x}, z)} dz \quad (1.9a)$$

$$= \int q(z | \hat{x}) \log \frac{q(z | \hat{x})}{p(z | \hat{x})} dz - \underbrace{\log p(\hat{x})}_{\text{Surprisal}}. \quad (1.9b)$$

From Eq. (1.9b) we see that the solution that minimises our free energy is  $q^*(z | \hat{x}) = p(z | \hat{x})$ . This solution corresponds exactly to applying Bayes rule to update our beliefs when receiving an observation

$$q^*(z | \hat{x}) = p(z | \hat{x}) = \underbrace{\frac{p(\hat{x} | z)p(z)}{p(\hat{x})}}_{\text{Bayes rule}}. \quad (1.10)$$

Having Bayes rule as a special case means we can subsume Bayesian inference under the heading of free energy minimisation. For this reason, we will also refer to free energy minimisation as *inference* moving forwards.

### 1.3 Forney-style Factor Graphs

As we just saw with the body temperature example, most of the time a model is not just a model. The exact architecture of the model in question matters when we want our theory to apply to the real world. When we need to know the structure of a particular model, it is practically useful to have a concise, graphical notation available for accurately specifying models. For the remainder of this dissertation, we will express models using the language of graphs to accomplish this goal. More specifically, we will employ Forney-style factor graphs (FFGs) as introduced in [25] with notational conventions adopted from [69, 92]. FFGs visualize factorised functions as undirected graphs, where nodes represent individual factors of the global function  $f(\mathbf{s})$  and edges represent variables. A node is connected to an edge if and only if the corresponding variable is an argument of that factor.

Formally, an FFG is a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  with nodes  $\mathcal{V}$  and edges  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ . An FFG can represent a model of the factorised function

$$f(\mathbf{s}) = \prod_{a \in \mathcal{V}} f_a(\mathbf{s}_a), \quad (1.11)$$

where  $f_a(\mathbf{s}_a)$  denotes the set of factors indexed by  $a \in \mathcal{V}$  and  $\mathbf{s}$  denotes the set of variables (indexed by  $i \in \mathcal{E}$  when necessary). The variable  $\mathbf{s}_a$  represents the set of all neighbouring variables of factor  $f_a$ . In an FFG, a node can be connected to an arbitrary number of edges, but edges are constrained to have a maximum degree of two. When a variable is an argument of more than two factors, this constraint can be satisfied by introducing equality factors. An equality factor is defined as  $f_{=}(s, s', s'') = \delta(s - s')\delta(s - s'')$  with  $\delta(\cdot)$  being either the Dirac or Kronecker  $\delta$ -function. Here the variables  $s'$  and  $s''$  are copies of

$s$ , whose posterior distributions are constrained to be identical by the equality factor. For a more extensive overview of FFGs, we refer the interested reader to [69, 70]. As an example, consider the factorised function

$$f(s_1, s_2, s_3, s_4, s_5) = f_a(s_1)f_b(s_1, s_2, s_3)f_c(s_3, s_4, s_5)f_d(s_4). \quad (1.12)$$

The FFG representation of this function is visualised in Fig. 1.2.

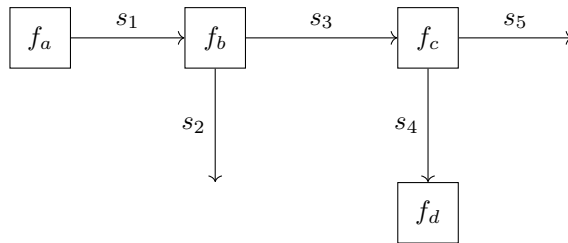


Figure 1.2: An FFG representation of the model given by Eq. (1.12). The edges are arbitrarily labelled using arrowheads to distinguish between forward and backward messages on the graph.

Sometimes, like in Fig. 1.2, we will use arrowheads to denote a "forwards" and "backwards" direction on an edge in an FFG. This does not mean the graph is directed. Instead, it is a pedagogical tool that allows us to talk about forwards and backwards *messages* flowing on an FFG. To establish what a "message" is, we need to marry the idea of free energy minimisation with that of model specification using graphs.

## 1.4 Inference on Graphs and Bethe Free Energy

Having introduced FFGs as a way to represent models, we can now start to work with free energy minimisation on graphs. This approach will take us to message passing (MP) algorithms which are the main tool used throughout this dissertation. The first step is to introduce an approximation to the VFE so that it distributes over the FFG. We can accomplish this by appealing to the Bethe approximation [114, 116] which constrains our variational distribution to factorise as

$$q(\mathbf{s}) = \prod_{a \in \mathcal{V}} q_a(\mathbf{s}_a) \prod_{i \in \mathcal{E}} q_i(s_i)^{1-d_i} \quad (1.13)$$

where  $d_i$  is the degree of the  $i$ 'th edge.  $q_a(\mathbf{s}_a)$  denotes a local (to the  $a$ 'th node) variational distribution, similar for  $q_i(s_i)$  and the  $i$ 'th edge. Under the Bethe approximation, the free energy becomes

$$F[q] = \sum_{a \in \mathcal{V}} \underbrace{\int q_a(\mathbf{s}_a) \log \frac{q_a(\mathbf{s}_a)}{f_a(\mathbf{s}_a)} d\mathbf{s}_a}_{F[q_a]} + \sum_{i \in \mathcal{E}} (1 - d_i) \underbrace{\int q_i(s_i) \log \frac{1}{q_i(s_i)} ds_i}_{H[q_i]}, \quad (1.14)$$

which defines the Bethe free energy (BFE). Eq. (1.14) shows that the BFE distributes over the FFG as a sum of node local free energies  $F[q_a]$  and edge local entropies  $H[q_i]$ . Moving forwards we will omit repeated subscripts unless necessary and let the argument indicate whether we are referring to an edge or node local marginal, for example writing  $q(s_i)$  instead of  $q_i(s_i)$ . The Bethe approximation is exact for tree-structured graphs, in which case the VFE and BFE are equal. For our purposes, the BFE is especially useful since having the free energy distributed over the graph opens the door to distributed free energy minimisation algorithms in the form of MP. From the point of view of MP, the BFE is special since stationary points of the BFE correspond to the celebrated belief propagation (BP) algorithm [62, 84]. In turn, BP is unique among MP algorithms since it provides exact inference on tree-structured graphs.

### 1.4.1 Belief Propagation

Solving Eq. (1.2) for the model in Eq. (1.11) results in the posterior given by Eq. (1.3). However, computing this posterior exactly for any given  $q_i(s_i)$  requires integration over all other variables in the model. Often this leads to a high dimensional integral which can be very difficult to solve.

However, due to conditional independencies in the model - following from the factorisation of  $f$  - we can instead replace the difficult global integration problem with a series of smaller, local computations. This approach leads to the sum-product or belief propagation algorithm [62, 84].

The result of one such local computation is called a *message* and we denote it by  $\mu(\cdot)$ . The BP message  $\tilde{\mu}(s_i)$  [62] flowing out of some node with factor function  $f_a$  and  $s_i \in \mathbf{s}_a$  is defined as

$$\tilde{\mu}(s_i) \propto \int f_a(s_a) \prod_{s_j \in s_a \setminus i} \tilde{\mu}(s_j) ds_{a \setminus i}, \quad (1.15)$$

where the notation  $s_a \setminus i$  denotes the set of variables  $s_a$  excluding the element  $s_i$  and each  $\tilde{\mu}(s_j)$  is an incoming message to the factor node. The arrows optionally indicate whether the message is flowing in the forwards (generative)  $\tilde{\mu}$  or backwards  $\tilde{\mu}$  direction.

Propagating such messages throughout the graph allows us to determine the posterior marginal distributions of some variable  $s_i$  - performing inference - by taking the product of messages colliding on the corresponding edge as

$$p(s_i) \propto \tilde{\mu}(s_i) \cdot \tilde{\mu}(s_i). \quad (1.16)$$

As an example, suppose we wish to calculate the posterior marginal of  $s_3$  in the model given by Eq. (1.12). This distribution can be calculated (up to a scaling constant) as

$$\begin{aligned} p(s_3) &= \int f(\mathbf{s}) d\mathbf{s}_{\setminus 3} \\ &\propto \underbrace{\iint \overbrace{f_a(s_1)}^{\tilde{\mu}(s_1)} f_b(s_1, s_2, s_3) ds_1 ds_2}_{\tilde{\mu}(s_3)} \cdot \underbrace{\iint \overbrace{f_d(s_4)}^{\tilde{\mu}(s_4)} f_c(s_3, s_4, s_5) ds_4 ds_5}_{\tilde{\mu}(s_3)}}. \end{aligned} \quad (1.17)$$

Here we see that the posterior marginal of  $s_3$  can be calculated as the product of two terms that each summarise a different part of the model. We can visualise this procedure using the FFG of Eq. (1.12) by drawing the necessary messages on the FFG as shown in Fig. 1.3.

In this example, the prior distributions over  $s_1$  and  $s_4$  are treated as messages themselves, meaning that  $\tilde{\mu}(s_1) = f_a(s_1)$  and  $\tilde{\mu}(s_4) = f_d(s_4)$ . Additionally, the edges corresponding to the variables  $s_2$  and  $s_5$  are not terminated - they only have a degree of 1. In this situation, messages in the backwards direction are taken to be uninformative,  $\tilde{\mu}(s_2) = \tilde{\mu}(s_5) = 1$ . Since posterior marginals are found by multiplication of messages, this ensures that the relevant posteriors are fully determined by the respective forwards messages such that  $p(s_2) \propto$

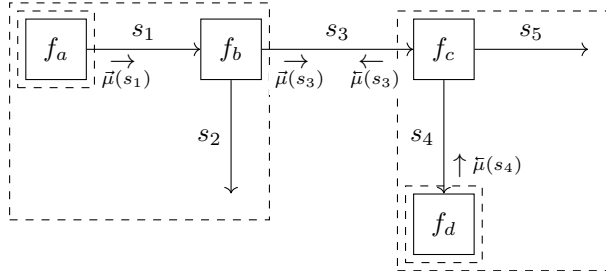


Figure 1.3: An FFG representation of the model in Eq. (1.12). The indicated messages can be interpreted as summaries of the dashed boxes, used for solving Eq. (1.17). Arrows indicate the flow of messages.

$\vec{\mu}(s_2)\vec{\mu}(s_2) = \vec{\mu}(s_2) \cdot 1 = \vec{\mu}(s_2)$  and similar for  $s_5$ . We will refer to edges of degree 1 as *half-edges* or *dangling edges* moving forwards. Examples of dangling edges can be seen in Fig. 1.3 for the variables  $s_2$  and  $s_5$ .

## 1.4.2 Variational and Hybrid Message Passing

Belief propagation represents the ideal case of doing inference because we can obtain the exact solution. Unfortunately, BP is not applicable outside of all but the simplest models. When BP is not an option, we can instead resort to variational inference and the corresponding MP algorithm: Variational message passing (VMP) [23, 112].

To derive the VMP algorithm, we can once again start from the problem statement given by Eq. (1.2). If we now employ the Bethe approximation and add appropriate constraints on  $\mathcal{Q}$ , we can obtain a constrained Bethe free energy (CBFE). We can then use variational calculus to solve for stationary points of the CBFE [116] to obtain a new MP algorithm. Predictably, changing the problem also changes the solution, and hence we are now led to messages of the form

$$\nu(s_i) \propto \exp \left\{ \int q(\mathbf{s}_{a \setminus i}) \log f_a(\mathbf{s}_a) d\mathbf{s}_{a \setminus i} \right\} \quad (1.18)$$

where  $q(\mathbf{s}_{a \setminus i})$  denotes the set of posterior marginals around the node  $a$  excluding  $q(s_i)$ . To distinguish between VMP and BP messages, we use  $\nu(\cdot)$  for VMP messages instead of  $\mu(\cdot)$ . With messages in hand, we can then update posterior marginals over the variables in our model sequentially as

$$q_i(s_i) \propto \vec{v}(s_i) \cdot \bar{v}(s_i), \quad (1.19)$$

which corresponds to finding stationary points of the CBFE [116]. This procedure of iteratively calculating messages and posterior marginals is then repeated until convergence.

As an example, we can once again consider the task of inferring  $q(s_3)$  in Eq. (1.12). This requires two messages, one from each adjacent factor node. Each message in turn depends on the set of marginals adjacent to that factor node, as illustrated in Fig. 1.4. Notice that compared to Fig. 1.3, there are far fewer messages flowing towards  $s_3$ . Instead, we now indicate marginals as they are involved in computing the messages through Eq. (1.18).

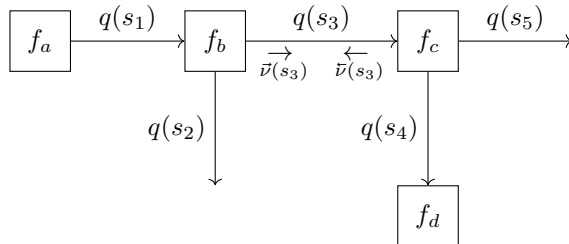


Figure 1.4: An FFG representation of the model in Eq. (1.12), with indicated VMP messages and relevant marginals used for inferring  $q(s_3)$ .

Here we wish to note that under the Bethe approximation, the constraint set defining  $\mathcal{Q}$  also factorises. In turn, this means that hybrid MP algorithms that combine BP, VMP and potentially other MP schemes are possible. [116] shows how to recover both the BP and VMP algorithms from CBFE optimisation alongside a number of other MP schemes such as expectation propagation (EP) [75] and Laplace propagation [104]. The full derivation of these algorithms goes beyond the scope of this background chapter. Instead, we refer interested readers to [116] for an extensive overview of hybrid MP algorithms and their relation to constrained BFE optimisation.

## 1.5 Active Inference

So far we have covered how to perform inference by free energy minimisation and, through the FEP, how this relates to systems persisting against the ravages



of nature. We have seen different ways of updating beliefs through MP and expressing these on FFGs. In the context of the FEP, we are now well equipped to examine systems that take in data and update their beliefs about the world accordingly.

However, thinking back to our original scenario in Section 1.1, we are still missing a key ingredient: Action. Taking in data and changing our beliefs only deals with one of the two pathways we introduced. In the context of our unfortunate biker on her way to a meeting, the methods outlined so far would mainly allow her to accept that she will be late for her meeting. This is clearly not satisfactory for our goal of designing ML systems that function more like human brains. We need to endow our system with the ability to take *action* upon the world in order to make it consistent with the system's predictions and by extension maintain a NESS. This takes us from the realm of just performing inference to the world of Active Inference (AIF). AIF is what will allow our system to pick up the phone and call a cab when faced with a flat tire. Systems with the ability to act upon the world possess agency and accordingly, we refer to them as *agents* moving forwards.

AIF agents are always embedded in an environment and separated from it by a Markov blanket as indicated in Fig. 1.1. What is not shown in Fig. 1.1 is the reciprocal flow of information between the agent and its environment.

Assuming a world that moves in discrete time steps<sup>2</sup>, at every time step the agent will receive an observation from the environment and emit an action. The environment then responds with a new observation and the cycle begins anew. This reciprocal exchange is called the *action-perception loop* [22, 28, 34], illustrated in Fig. 1.5.

The perception part of the loop can be handled through the VFE-minimising machinery we have already covered. Any time our agent receives a new observation, it can update its beliefs through VFE minimisation. However, when it comes to selecting which action to emit, we encounter a problem. To perform VFE minimisation, we would require a new piece of information - an observation<sup>3</sup>. However, when selecting an appropriate action to emit, our agent has to anticipate what the results of said action would be. This requires reasoning about the future and, rather unhelpfully, the future is by definition not observed since it has not happened yet. For our agent to emit sensible actions to the

---

<sup>2</sup>There exist continuous time formulations of AIF written in the form of (stochastic) differential equations as well, see for instance [28, 61]. In this dissertation we will focus solely on the discrete-time case

<sup>3</sup>In the absence of new information, our beliefs should not change. This is known as the Principle of Minimum Updating [14]

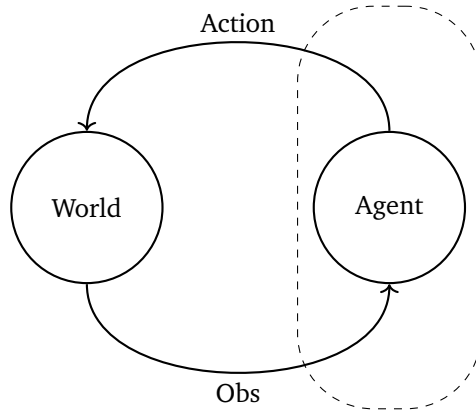


Figure 1.5: A schematic illustration of the action-perception loop. At every time step, the agent receives an observation from the world it inhabits and emits an appropriate action. The dashed box indicates the Markov blanket separating the agent from the world.

world, we need to endow it with the ability to reason forwards in time. In colloquial terms, the agent needs to be able to ask "*What would happen if I did that?*" or, more appropriately for a free energy minimising agent, "*What would my free energy be if I did that?*". In other words, we need a quantity that measures what the agent expects its free energy to be, given a particular course of action.

### 1.5.1 Expected Free Energy

For AIF agents, the solution to this apparent conundrum comes in the form of the expected free energy (EFE). The EFE is an AIF specific construct originally introduced by [33]. To write down the EFE, we first need to commit to a particular form of generative model. Given that we wish to model the future (up to some known planning horizon  $T$ ), the model in question needs to include a notion of time. It also needs to include observations (so that we can model what we expect they might be) and actions (so we can reason about the consequences of different sequences of actions). A generic formulation that satisfies these desiderata is that of a state space model conditioned on a sequence of actions. We can write this model as

$$p(\mathbf{x}, \mathbf{z} \mid \mathbf{u}) = p(z_t) \prod_{k=t+1}^T p(x_k \mid z_k) p(z_k \mid z_{k-1}, u_k) \quad (1.20)$$

where  $\mathbf{x} = x_{t+1:T}$  denotes a sequence of observations,  $\mathbf{z} = z_{t:T}$  denotes a sequence of latent states and  $\mathbf{u} = u_{t+1:T}$  denotes a sequence of actions. Throughout this dissertation, we will also refer to a sequence of actions as a *policy*. To construct the EFE, we can now write what the form of the VFE would have been, in case we had observations  $\hat{\mathbf{x}}$  available.

$$F[q; \mathbf{u}] = \int q(\mathbf{z} \mid \mathbf{u}) \log \frac{q(\mathbf{z} \mid \mathbf{u})}{p(\mathbf{x}, \mathbf{z} \mid \mathbf{u})} d\mathbf{z}. \quad (1.21)$$

Since we do not have an observation  $\hat{\mathbf{x}}$  available to us, Eq. (1.21) does not evaluate to a scalar objective that we can optimise. To remedy this, we can take the expectation of Eq. (1.21) under our predictive distribution over  $\mathbf{x}$ . Taking the expectation of the free energy gives us the expected free energy.

$$G[q; \mathbf{u}] = \underbrace{\iint q(\mathbf{x} \mid \mathbf{z}) \overbrace{q(\mathbf{z} \mid \mathbf{u}) \log \frac{q(\mathbf{z} \mid \mathbf{u})}{p(\mathbf{x}, \mathbf{z} \mid \mathbf{u})}}^{F[q; \mathbf{u}] \text{ if } \hat{\mathbf{x}} \text{ had been observed}} d\mathbf{z} d\mathbf{x}}_{\text{Expected } F[q; \mathbf{u}] \text{ when } \hat{\mathbf{x}} \text{ is not observed}}. \quad (1.22)$$

The EFE has several desirable properties that we wish to incorporate into our ML systems. The main feature we are interested in is that selecting policies based on EFE-minimisation endows AIF agents with an epistemic, exploratory drive to gather information about the world they inhabit. To see how this comes about, we can decompose the EFE in Eq. (1.22) as

$$\begin{aligned}
G[q; \mathbf{u}] &= \iint q(\mathbf{x}, \mathbf{z} | \mathbf{u}) \log \frac{q(\mathbf{z} | \mathbf{u})}{p(\mathbf{x}, \mathbf{z} | \mathbf{u})} d\mathbf{x}d\mathbf{z} \\
&= \iint q(\mathbf{x}, \mathbf{z} | \mathbf{u}) \log \frac{q(\mathbf{z} | \mathbf{u})}{p(\mathbf{z} | \mathbf{x}, \mathbf{u}) \underbrace{p(\mathbf{x})}_{\text{Goal prior}}} d\mathbf{x}d\mathbf{z} \\
&= \iint q(\mathbf{x}, \mathbf{z} | \mathbf{u}) \log \left[ \frac{q(\mathbf{z} | \mathbf{u})q(\mathbf{z} | \mathbf{x}, \mathbf{u})}{p(\mathbf{z} | \mathbf{x}, \mathbf{u})q(\mathbf{z} | \mathbf{x}, \mathbf{u})p(\mathbf{x})} \right] d\mathbf{x}d\mathbf{z} \\
&= \iint q(\mathbf{x}, \mathbf{z} | \mathbf{u}) \log \left[ \frac{q(\mathbf{z} | \mathbf{u})}{q(\mathbf{z} | \mathbf{x}, \mathbf{u})} + \frac{q(\mathbf{z} | \mathbf{x}, \mathbf{u})}{p(\mathbf{z} | \mathbf{x}, \mathbf{u})} + \frac{1}{p(\mathbf{x})} \right] d\mathbf{x}d\mathbf{z} \\
&= \iint q(\mathbf{x}, \mathbf{z} | \mathbf{u}) \log \left[ \frac{q(\mathbf{z} | \mathbf{u})q(\mathbf{x} | \mathbf{u})}{q(\mathbf{z}, \mathbf{x} | \mathbf{u})} + \frac{q(\mathbf{z} | \mathbf{x}, \mathbf{u})}{p(\mathbf{z} | \mathbf{x}, \mathbf{u})} + \frac{1}{p(\mathbf{x})} \right] d\mathbf{x}d\mathbf{z} \\
&= \underbrace{\mathbb{I}[\mathbf{x}, \mathbf{z}]}_{\text{Mutual Information}} + \underbrace{\mathbb{E}_{q(\mathbf{x}|\mathbf{u})}[\text{KL}[q(\mathbf{z} | \mathbf{x}, \mathbf{u}) || p(\mathbf{z} | \mathbf{x}, \mathbf{u})]]}_{\text{Expected divergence}} + \underbrace{\mathbb{E}_{q(\mathbf{x}|\mathbf{u})} \left[ \log \frac{1}{p(\mathbf{x})} \right]}_{\text{Cross entropy}}. \tag{1.23}
\end{aligned}$$

What Eq. (1.23) shows, is that the EFE can be decomposed into a mutual information term that drives exploration, an expected KL-divergence<sup>4</sup> and a cross-entropy loss which is a popular objective function for ML systems. The cross-entropy in question is between the predicted observations given a particular policy  $q(\mathbf{x} | \mathbf{u})$  and a *goal prior*  $p(\mathbf{x})$ . Intuitively, it tries to match the predicted observations with desired the observation specified by the goal prior. From the point of view of the FEP, the goal prior defines observations that are consistent with remaining at NESS. On the other hand, from the point of view of the engineer designing artificial agents, the goal prior provides a means to encode tasks for AIF agents by specifying a set of desired observations instead of a cost function. To relate this idea back to our body temperature example,  $p(\mathbf{x})$  could specify a desired observation of having a body temperature of 37 degrees.  $q(\mathbf{x} | \mathbf{u})$  would then encode what we would predict ("I'll feel warm") given a particular sequence of actions (- "If I put on a jacket").

The EFE is the subject of many of the upcoming analyses, so we defer a deeper investigation to later chapters, in particular Chapters 2 and 3 and their accompanying appendices. For now, we simply wish to emphasise that EFE,

<sup>4</sup>For many practical applications, this term is often assumed small enough to be negligible and is therefore left out, see for instance [15, 33, 36, 76, 82]

as an objective functional, endows AIF agents with both a goal-directed drive (a cross-entropy loss) as well as an epistemic, information-seeking drive (the mutual information term).

## 1

## 1.6 Research Questions

At this point, we have an idea of how the theory behind AIF works and what we need to do on the way to our goal of developing a practical toolset for artificial AIF. Practically, this means we need to overcome a series of obstacles. These form our concrete research questions, the first of which is

**Q1:** *How can factor graphs be used to accurately describe the model assumptions and inference processes of an Active Inference agent?*

If we wish to make tools for AIF using FFGs, we need to first define exactly how the two are related. This turns out to be a surprisingly tricky task since the theory is undergoing active development and has seen several different incarnations throughout its lifespan. While all of these are centered around the themes covered in Section 1.2, several minor differences influence the details of what a practical implementation should look like. This is especially true for EFE computation and its role in planning out future policies. Additionally, given our wish to use MP, we need to accurately state the problem using FFGs. Answering this question for discrete-time AIF is the focus of Chapter 2.

Having established how to interpret AIF using FFGs in Chapter 2, the next obstacle we need to overcome relates to the capabilities of AIF. Outside of work relying heavily on deep learning, AIF has been closely tied to a particular choice of generative model - a discrete partially observed Markov decision process (POMDP). Yet many tasks that we might wish to address as engineers do not easily conform to a discrete state space representation and are much easier to approach using continuous variables. One example could be controlling an autonomous vehicle or drone which necessitates navigating in continuous coordinate spaces. Solving this problem becomes our second research question which we can state as

**Q2:** *Is a linear Gaussian dynamical system a useful generative model for an Active Inference agent?*

The linear Gaussian dynamical system (LGDS) is ubiquitous within engineering and constitutes a simple, widely applicable model that we can use to

expand the scope of AIF to include continuously varied state spaces. We tackle this research question in Chapter 3.

In the end, the true litmus test for whether we can develop practical engineering tools for AIF will always be practical engineering. In the words of Richard Feynman - "*What I cannot create, I do not understand*". Only when AIF can easily be used to develop applications and bring useful features to a concrete engineering problem, will we have truly *practical* tools. This need for a practical engineering problem brings us to our third research question

**Q3:** *How can an artificial Active Inference agent meaningfully support situated personalisation of hearing aid algorithms?*

We tackle this research question in Chapter 4. Chapter 4 details the design of AIDA, the Active Inference Design Agent. AIDA is a system designed for tuning hearing aid (HA) devices based on user feedback obtained *in situ*. Tuning HAs is currently a time-consuming and highly personalised endeavour that involves expert audiologists manually adjusting parameters. This is a difficult task at the best of times and further complicated by the fact that very few natural auditory environments resemble an audiologist's office. As a consequence, the parameter settings that are appropriate in the audiologist's office might be sub-optimal when for instance attending a concert or visiting a restaurant. In an ideal world, HA parameters would instead be adjusted on the fly as the HA user goes about her day, always applying highly personalised, context-dependent audio corrections. AIDA aims to bring this ideal world about by allowing users to train their own personalised HA algorithm through online feedback on HA performance. The component of AIDA responsible for selecting the best HA parameters for the user given the current auditory context, is powered by AIF.

As Chapter 4 demonstrates, bringing AIF to engineering is a laborious endeavour. In particular, every time we wish to utilise AIF with a new generative model, we need to derive all the relevant quantities by hand. Having to do manual derivations for every model takes time, is prone to human error, and leads to unnecessarily long design cycles. In order to have a practical engineering tool, we need to make AIF easy to apply, even for a non-expert. To accomplish this, we need unambiguous, reusable descriptions that can be combined to form full-fledged applications.

A framework that already possesses these qualities is that of FFGs. FFGs allow engineers to build complex, interwoven systems out of simple building blocks simply by writing down the FFG representation of their problem and employing the associated message passing algorithm. For our tooling, we desire a similar level of flexibility which motivates our final research question

**Q4:** *Can free energy minimisation in artificial Active Inference agents be formulated as an automatable optimisation problem by message passing on a factor graph?*

To answer this question, in Chapter 5 we develop a new constrained Forney-style factor graph (CFFG) notation that allows for writing down not just the generative model, but also the exact variational inference problem to be solved. This includes constraints on the variational distribution such as factorisation and form constraints. Accurately specifying both the generative model and the set of constraints in turn uniquely determines the MP algorithm that should be applied to obtain a solution.

However, AIF also relies on several customised versions of the free energy — such as the EFE— which have so far not been expressible using a graphical notation. Using CFFGs we demonstrate how this can be accomplished by succinctly restating the core algorithms used for prior work on AIF.

Finally, we derive a version of the custom objective functionals used for AIF that is local to a single node on a CFFG. With our new functional in hand, we proceed to derive custom MP updates for finding stationary points of said functional, using Lagrangian optimisation. By implication, we are then able to apply AIF to arbitrary, free-form graphical models provided they can be specified using CFFGs. Notably our new MP updates interface with existing MP algorithms, meaning we can leverage the power and flexibility of prior work while retaining the desirable features of AIF.

## 1.7 Summary of Contributions

In summary, we make the following concrete contributions

- To address **Q1**, we summarise prior work on AIF using FFGs in Chapter 2, particularly focused on action selection and planning. The other core piece of practical AIF modelling is state estimation which we review in Chapter 3 in the context of Bayesian filtering. Together, perception and action allow us to handle the entire action-perception loop which is the foundation we need for practical AIF modelling.
- In Chapter 3 we explore **Q2** by deriving the necessary updates for AIF, using a LGDS as our generative model. We particularly focus on the epistemic, information-seeking types of behaviour that are characteristic of AIF agents and discover that they are largely absent in LGDSs.

- To answer **Q3** we develop AIDA in Chapter 4. AIDA is a fully Bayesian system that performs both audio processing and *in situ* personalisation of a HA device based on user feedback.
- Chapters 2, 3 and 4 all use variations of the same methodology which follows from the FFG-based analysis conducted as part of addressing **Q1**. In Chapter 5 we address **Q4** and fully develop this methodology by introducing CFFGs and deriving an accompanying MP update that allows for writing Lagrangian Active Inference (LAIF) purely as a MP algorithm.

## 1.8 Accountability, Outline and Suggested Reading Orders

As any scientist knows, science is a collaborative effort. The days of the lone genius are (with a few rare exceptions) in the past - and in either case I would not claim to be either a genius or a loner by any stretch of the imagination. Instead, the work presented in this dissertation was conducted as a series of team efforts in which I played a central part.

This means that while I take full responsibility for every word of this dissertation, it is important to acknowledge that it necessarily includes the work of collaborators to produce a coherent, readable document. To clearly give credit where credit is due, when the content of a chapter includes significant contributions from collaborators it is prefaced with a small attribution as to who was the *primus motor* for each part of the work presented.

The remainder of the dissertation is structured as follows. In Chapter 2 we describe planning under AIF as message passing on a FFG. This perspective forms the basis for the analyses following in Chapter 3 where we extend AIF to work with LGDSs. In Chapter 4 we tackle an industry problem by developing a fully Bayesian audio processing agent based on AIF. The core contribution here is in a preference learning module that applies AIF to a Gaussian process classifier (GPC). Finally, in Chapter 5 we develop LAIF as a fully general version of AIF that applies to arbitrary graphical models. To accurately write down the AIF optimisation problem using a graphical syntax, we develop the CFFG notation in Chapter 5. CFFGs are a general-purpose graphical syntax for writing constrained free energy optimisation problems on factorised models. Conclusions and perspectives on future work can be found in Chapter 6.

Each chapter is written to be a self-contained unit, meaning some points are repeated between chapters. The dissertation can therefore be read in any



order. While the most comprehensive picture is obtained by reading the entire dissertation back to back, we recognise that readers will pick up the document for different purposes. To that end, we have designed three alternate reading orders, tailored for readers with particular foci.

1

Readers who are new to AIF and intend on using the dissertation as a principled introduction to the field should follow the [Review Track](#). The [Review Track](#) starts by establishing a foundation through Chapter 2, proceeds through Chapter 3 with a particular focus on Sections 3.4 and 3.5 and ends with Section 5.8 of Chapter 5. For the notation used in Section 5.8, it is recommended to take a slight detour through Section 5.5 first.

If one is mostly interested in applications of AIF, it is recommended to follow the [Applications Track](#) instead. The applications track starts by covering the entirety of Chapter 3 for LGDS models, following up with Chapter 4 for the most involved practical application in the dissertation. Finally one can end with the experiments on direct policy inference at the end of Chapter 5, specifically Sections 5.9 and onwards. Again, Section 5.5 is recommended to familiarise oneself with the notation used in 5.9

Finally, the reader whose interest is mainly piqued by the development of tools for AIF should follow the [Theory & Tools Track](#). [Theory & Tools](#) starts with Chapter 2 as an alternate way of viewing EFE computation using FFGs. Coupling the FFG view with the action selection mechanisms in Chapter 3, Section 3.5 constitutes the core method used for most of the dissertation. Finally, the theoretician will find the most relevant content in Chapter 5 which should be read in full. Additionally, we recommend that a reader on the [Theory & Tools Track](#) also visits the appendices liberally as they often contain derivations that were too unwieldy to be part of the main text. The suggested reading orders are illustrated schematically in Fig. 1.6.

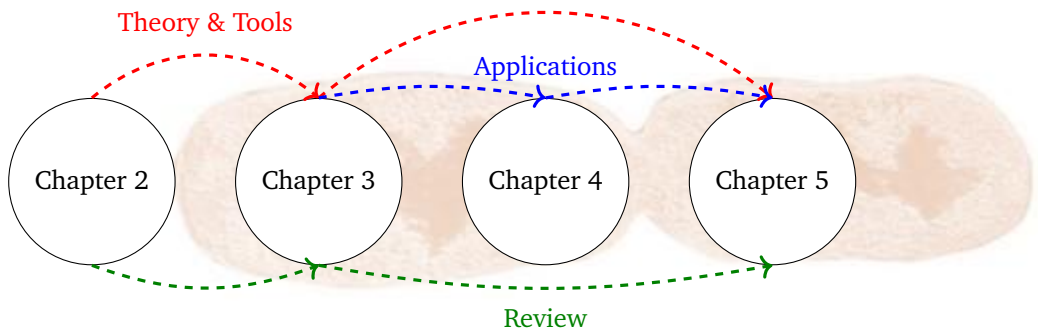


Figure 1.6: Suggested reading orders.



# Chapter 2

## A Message Passing Perspective on Planning under Active Inference

*"It's a dangerous business, Frodo, going out your door. You step onto the road, and if you don't keep your feet, there's no knowing where you might be swept off to."*

*– J.R.R. Tolkien on planning, 1954*

This chapter presents a message passing (MP) interpretation of planning under Active Inference (AIF). Specifically, we show how the AIF planning procedure can be broken into a (partial) message passing sweep over a graph, followed by local computations of a cost functional, the Expected free energy (EFE). Using Forney-style factor graphs (FFGs) we then proceed to show how one can derive novel planning schemes by local changes to the underlying graph and message passing schedule. We illustrate this by first isolating the “sophisticated” aspect of Sophisticated inference (SI) [35] and then proposing a novel planning algorithm by combining the sophisticated update mechanism with a different message passing schedule. Our main contribution is a modular view of planning under AIF that can serve as a framework for understanding existing algorithms, deriving new ones, and extending the class of models amenable to

AIF. Approaching AIF from a MP perspective also shows how it can be efficiently implemented using off-the-shelf probabilistic programming software, broadening the class of models available to researchers and practitioners. This chapter is based on the original work referenced below and foreshadows the methodology we will use in Chapters 4 and 5.

**Koudahl, M.,** Buckley, C.L., de Vries, B. (2023). *A Message Passing Perspective on Planning Under Active Inference*. In: , et al. *Active Inference. IWAJ 2022. Communications in Computer and Information Science*, vol 1721. Springer, Cham.

2

## 2.1 Introduction

AIF is a common modeling framework for studying decision-making and, in recent years, also for designing synthetic agents. A key facet that sets AIF apart from other approaches is the choice of planning objective. AIF uses the EFE, which is a cost functional that promises a balanced trade-off between exploration and exploitation.

In this paper, we present a particular interpretation of EFE-based planning under AIF using known message passing-based inference methods on a FFG. We show that the standard EFE planning algorithm is equivalent to performing a forward pass using standard belief propagation (BP) messages, followed by a separate computation phase based on the resulting marginals.

By explicitly writing planning under AIF as message passing on a graph, we can clearly delineate the practical steps used for EFE computation. Doing so means we can isolate parts of more complicated schemes such as the sophisticated aspect of SI [35] and the backwards influence from future time steps hinted at by [81]. It also allows us to propose new algorithms as variations based on the common underlying theme and indicates a method for implementing AIF in a broader class of models using efficient inference software.

## 2.2 Generative Model and Inference

Planning under AIF centers around a generative model of the future. The generative model traditionally used is a discrete partially observed Markov decision process (POMDP) [33, 35, 44]. We let  $\mathbf{x}_t$  denote an observation,  $\mathbf{z}_t$  a latent state, and  $\hat{u}_t$  a fixed control at time step  $t$ . Now we can write the model after

having observed  $\mathbf{x}_t$  and conditioned on a fixed sequence of actions  $\hat{\mathbf{u}}_{1:T}$  - which we will refer to as a *policy* moving forwards - as

$$\begin{aligned}
 & p(\underbrace{\mathbf{x}_{t+1:T}, \mathbf{z}_{t:T}}_{\text{Future}} \mid \underbrace{\hat{\mathbf{u}}_{t+1:T}}_{\text{Policy}}, \underbrace{\mathbf{x}_{1:t}, \hat{\mathbf{u}}_{1:t}}_{\text{Past}}) \\
 &= \underbrace{p(\mathbf{z}_t \mid \mathbf{x}_{1:t}, \hat{\mathbf{u}}_{1:t})}_{\text{State Prior}} \prod_{k=t+1}^T \underbrace{p(\mathbf{x}_k \mid \mathbf{z}_k)}_{\text{Likelihood}} \underbrace{p(\mathbf{z}_k \mid \mathbf{z}_{k-1}, \hat{\mathbf{u}}_k)}_{\text{State Transition}}
 \end{aligned} \tag{2.1}$$

where

$$p(\mathbf{z}_t \mid \mathbf{x}_{1:t}, \hat{\mathbf{u}}_{1:t}) = \text{Cat}(\mathbf{z}_t \mid \mathbf{d}) \tag{2.2a}$$

$$p(\mathbf{z}_k \mid \mathbf{z}_{k-1}, \hat{\mathbf{u}}_k) = \text{Cat}(\mathbf{z}_k \mid \mathbf{B}_{u_k} \mathbf{z}_{k-1}) \tag{2.2b}$$

$$p(\mathbf{x}_k \mid \mathbf{z}_k) = \text{Cat}(\mathbf{x}_k \mid \mathbf{A} \mathbf{z}_k). \tag{2.2c}$$

Here  $p(\mathbf{z}_t \mid \mathbf{x}_{1:t}, \hat{\mathbf{u}}_{1:t})$  represents the Bayesian filtering solution over observed time steps  $1 : t$ , which we summarise in the parameter vector  $\mathbf{d}$ . Both  $\mathbf{z}_k$ ,  $\mathbf{z}_t$ , and  $\mathbf{x}_k$  are discrete variables following Categorical distributions, as for instance described in [10, Ch. 2.]. Note that Eq. (2.1) extends into the future until some known horizon  $T > t$  and is conditioned on a policy over the full trajectory  $\hat{\mathbf{u}}_{1:T}$ . We use  $\mathbf{B}_{u_k}$  to denote the transition matrix  $\mathbf{B}$  corresponding to the action  $\hat{u}_k$ . Planning under AIF relies on comparing choices of  $\mathbf{B}_{u_k}$  which we emphasise by this notation.

We can visualise Eq. (2.2) using FFG formalism. In an FFG, a node represents a factor (function of variables) and an edge represents a variable. An edge connects to a node if and only if the corresponding variable is an argument of that factor. The FFG of the model described by Eq. (2.2) is shown in Fig. 2.1. In Fig. 2.1, the T-nodes denote a discrete state transition (multiplication of a variable with categorical distribution by a transition matrix). For pedagogical purposes, we have also labelled T-nodes with the matching equation in Eq. (2.2) and indicated the forward direction of the graph by arrowheads on edges. When a variable is fixed (for example when we condition on a policy), we indicate this by a small, black square.

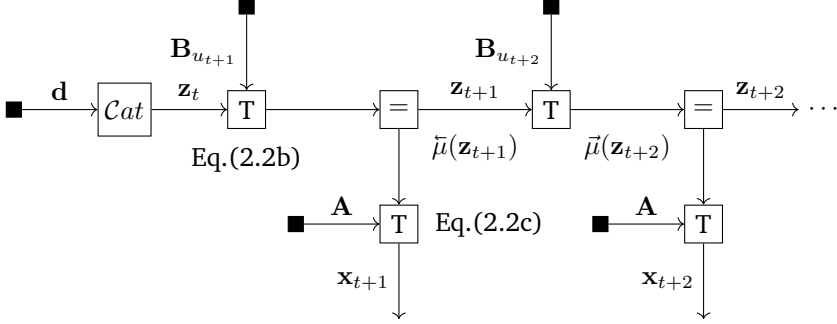


Figure 2.1: FFG of discrete POMDP as used for planning in standard AIF models.

To perform inference in this model, we can utilise BP [35, 70]. BP proceeds by passing messages along the edges of a graph towards variables that we wish to infer. We can illustrate this on our FFG by drawing arrows that outline the messages we wish to pass, see Fig. 2.2. When two messages collide they are multiplied (and normalized) to obtain a posterior marginal for the variable on an edge. For the model given by Eq. (2.2), all variables are discrete and related by discrete state transitions. We therefore only need the forward  $\vec{\mu}(\cdot)$  and backward  $\bar{\mu}(\cdot)$  BP-messages around a T-node. For the variables  $\mathbf{z}_k$ ,  $\mathbf{z}_{k-1}$  and the transition matrix  $\mathbf{B}_{u_k}$ , the messages are

$$\vec{\mu}(\mathbf{z}_k) = \text{Cat}\left(\mathbf{z}_k \mid \frac{1}{Z} \mathbf{B}_{u_k} \mathbf{z}_{k-1}\right), \quad \bar{\mu}(\mathbf{z}_{k-1}) = \text{Cat}\left(\mathbf{z}_{k-1} \mid \frac{1}{Z} \mathbf{B}_{u_k}^T \mathbf{z}_k\right), \quad (2.3)$$

where we slightly abuse notation by having  $\mathbf{z}_k$  denote the parameter vector of the incoming message on the edge  $\mathbf{z}_k$  (instead of the random variable  $\mathbf{z}_k$ ), similar for  $\mathbf{z}_{k-1}$  and the edge  $\mathbf{z}_{k-1}$ .  $Z$  is a normalisation constant that we can ignore when the columns of the transition matrix are normalised, which we assume going forward. To illustrate, in Fig. 2.1 we have indicated the messages flowing out of a T-node towards the variables  $\mathbf{z}_{t+1}$  and  $\mathbf{z}_{t+2}$ .

## 2.3 Expected Free Energy

Planning under AIF involves first computing the EFE for each time step given a policy and then summing the results over time steps. This procedure is repeated for a set of admissible policies. Based on the sum-total EFEs for each

policy, one then constructs a categorical distribution over possible policies and sample a course of action from there [22, 33, 60]. We explicitly appeal to a recursive formulation of EFE computation here as put forward in [35]. While the exposition given here will be in terms of the discrete POMDP described in Section 2.2, similar steps can be performed for other generative models, see for instance [60] for an example using linear Gaussian dynamical systems (LGDSs). The EFE for a single time step  $k$  is defined as

$$G(\hat{u}_k) = \sum_{\mathbf{x}_k} \sum_{\mathbf{z}_k} p(\mathbf{x}_k | \mathbf{z}_k) q(\mathbf{z}_k | \hat{u}_k) \log \frac{q(\mathbf{z}_k | \hat{u}_k)}{p(\mathbf{x}_k, \mathbf{z}_k | \hat{u}_k, \mathbf{x}_{1:t})}. \quad (2.4)$$

Notably, EFE only depends on *prior* time steps and not on the full trajectory. For computational purposes, the EFE for a single time step is often rewritten as<sup>1</sup>

$$G(\hat{u}_k) = \underbrace{\sum_{\mathbf{z}_k} q(\mathbf{z}_k | \hat{u}_k) \text{H}[\mathbf{x}_k | \mathbf{z}_k]}_{\text{Ambiguity}} + \underbrace{\text{KL}[q(\mathbf{x}_k | \hat{u}_k) || p(\mathbf{x}_k)]}_{\text{Risk}} \quad (2.5)$$

Here, we wish to note that all required quantities are written in terms of  $\mathbf{x}_k$ , the observations. This means everything we need to compute Eq. (2.5) is available around the likelihood node. With slight abuse of notation we find  $q(\mathbf{z}_k | \hat{u}_k)$  and  $q(\mathbf{x}_k | \hat{u}_k)$  by using the forward message Eq. (2.3) as

$$\mathbf{z}_k = \mathbf{B}_{\hat{u}_k} \mathbf{z}_{k-1} \quad (2.6a)$$

$$\mathbf{x}_k = \mathbf{A} \mathbf{z}_k. \quad (2.6b)$$

Now, we are ready to compute Eq. (2.5). Following [22, Eq. D.2-3] we can evaluate Eq. (2.5) for the model 2.2 as

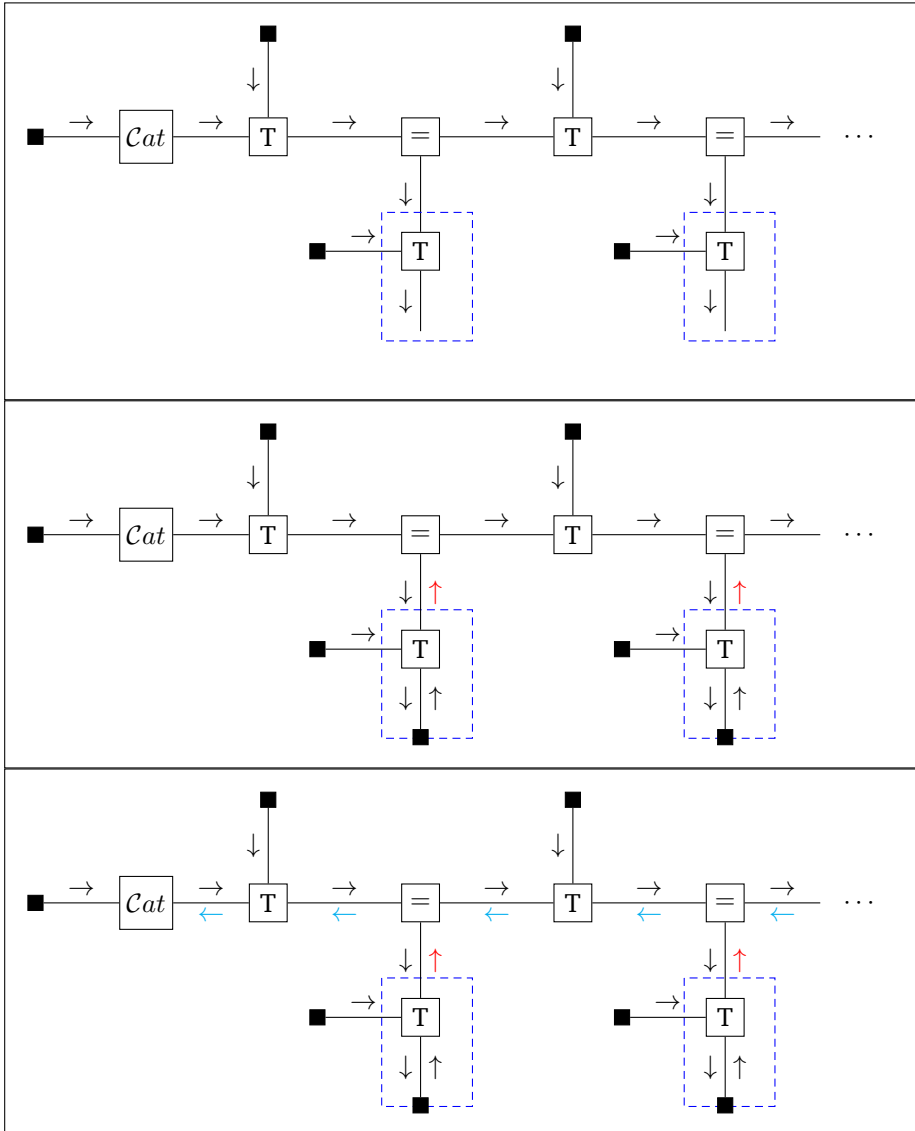
$$G(\hat{u}_k) = - \underbrace{\text{diag}(\mathbf{A}^T \log \mathbf{A})^T}_{\text{Ambiguity}} \mathbf{z}_k + \underbrace{\mathbf{x}_k^T (\log \mathbf{x}_k - \log \mathbf{c}_k)}_{\text{Risk}} \quad (2.7)$$

where we have slightly adapted the original notation to be consistent with our exposition. Here  $\mathbf{c}_k$  refers to the parameter vector of the goal prior  $p(\mathbf{x}_k)$

<sup>1</sup>The equality is only correct when we can do exact inference, see [60] or Chapter 3 for details



and the  $\text{diag}(\cdot)$  operator takes as argument a matrix and returns its diagonal entries as a column vector. In Eq. (2.7), the RHS is implicitly a function of  $\hat{u}_k$  through the choice of  $\mathbf{B}_{\hat{u}_k}$  in Eq. (2.6a). We see that the quantities used in Eq. (2.7) can be obtained by applying a forward MP sweep on the generative model. We can visualise this on the FFG as shown in the top panel of Fig. 2.2. The [boxed areas](#) indicate where we obtain the quantities required for Eq. (2.7). As we can see, EFE computation corresponds exactly to a forward MP sweep followed by a secondary computation around the likelihood nodes.



2

Figure 2.2: Comparison of MP schedules (shown with arrows) for standard EFE planning (panel 1), SI (panel 2, including backward messages) and SI + smoothing (panel 3, including a smoothing pass), shown on the FFG of the generative model for a discrete POMDP Eq. (2.2). The boxed areas contain all the quantities needed for calculating the EFE by Eq. (2.5). T-nodes indicate discrete state transitions, *Cat* nodes a categorical distribution and = nodes an equality constraint. Small, black squares are used for variables with fixed values and  $\dots$  indicate that the graph extends forward until an arbitrary planning horizon  $T$ .

## 2.4 Sophisticated Inference

Having established the MP view of EFE computation, we can use it to examine algorithms for AIF planning. A recent development is SI [35] which better accounts for future belief updates compared to the standard approach [33, 60]. There are several moving parts to the SI algorithm, such as the branching/pruning of the policy search tree and recursive EFE evaluation, which we will not cover here. We limit ourselves only to the innovation that lends the sophisticated aspect and show how it can be interpreted as adding an additional, fixed node to the FFG and passing an additional message.

To do so, we investigate what occurs when we fix a node on the graph. Formally, fixing a node means adding a constraint to the optimisation problem in the form of a  $\delta$ -function that forces the variable on that edge to take on a particular value [116]. For the SI algorithm, we fix the half-edges that denote  $\mathbf{x}_k$  as if they had been observed, see Fig. 2.2, panel 2. This is equivalent to enforcing the constraint [116]

$$q(\mathbf{x}_k) = \delta(\mathbf{x}_k - \hat{\mathbf{x}}_k^i) \quad (2.8)$$

where  $\hat{\mathbf{x}}_k^i$  is a one-hot encoded vector with 1 in the  $i$ 'th position and zeros everywhere else.  $\delta(\cdot)$  is the Kronecker  $\delta$ -function which only evaluates to 1 if  $\hat{\mathbf{x}}_k^i = \mathbf{x}_k$ .

The index  $i$  represents a branching point of the algorithm and is evaluated for all indices of  $\mathbf{z}_k$  that exceed a threshold, resulting in a forward search tree [35] that branches for different choices of  $i$ . The details of the branching procedure and subsequent tree search are beyond the present exposition and we refer interested readers to [35] for algorithmic details. The exposition given here corresponds to a single path through the search tree for a fixed policy.

The next step is to pass a **backward message** towards  $\mathbf{z}_k$ . To pass the **backward message** through the likelihood node, we use Eq. (2.3) and the fact that the clamped node is one-hot encoded to obtain a message towards  $\mathbf{z}_k$  given by

$$\tilde{\mu}(\mathbf{z}_k) = \text{Cat}(\mathbf{z}_k \mid \mathbf{A}_{*i}) \quad (2.9)$$

where  $\mathbf{A}_{*i}$  indicates the  $i$ 'th column of  $\mathbf{A}$ . In practice, this procedure is equivalent to performing a filtering step given that  $\hat{\mathbf{x}}_k^i$  was observed.

We can visualise the MP schedule used for the fixed policy SI algorithm on the FFG as shown in Fig. 2.2, panel 2 which makes the sophisticated aspect readily apparent: By passing the **backward message**, Eq. (2.9),  $\mathbf{z}_k$  now incorporates

information from the simulated observation  $\hat{\mathbf{x}}_k^i$ .  $\hat{\mathbf{x}}_k^i$  is a simulated observation since it is selected by the SI algorithm rather than generated by the agents environment.

In the case of a single pass through the search tree, the recursive EFE formulation given by [35] reduces to the standard EFE expression given by Eq. (2.7). A subtle note here is that Eq. (2.7) is still evaluated for the downward *message* given by Eq. (2.6b) rather than the *marginal* given by the product of colliding messages on the edge  $\mathbf{x}_k$ . In other words, information from the simulated observation at a particular time step is not considered when calculating the EFE of that time step.

Interestingly, the simulated observations that give SI its sophisticated properties bear similarity to alternative approaches to epistemics using constrained Bethe free energy (CBFE) instead of EFE [66]. In [66], the authors were able to induce exploration while only relying on standard MP procedures and point-mass constraints (that can be interpreted as a different way of selecting observations) instead of the EFE.

## 2.5 Advantages of the Message Passing Perspective

Viewing EFE computation from a MP perspective provides several advantages of which we will highlight three. First, it allows us to step back and work with update equations in the abstract which in turn opens the avenue for working with EFE in a broader class of models. The forwards sweep relies on off-the-shelf MP equations which can be automated in software. Any efficient MP toolbox, for example, [3], can therefore be used for performing inference. Finalising an EFE implementation then only requires solving Eq. (2.5) around the likelihood nodes. As an example, [60] used a similar approach to derive the EFE update equations for linear Gaussian models, making EFE available in continuous state spaces. Second, taking the MP perspective allows a unified perspective on different planning algorithms proposed under AIF. We have demonstrated this by showing how a core aspect (the sophistication) of SI can be interpreted as fixing a node on the FFG and subsequently passing a **backward message**. Third, by casting the planning problem as MP we can extend upon current work and derive new EFE-based planning algorithms by manipulating the underlying FFG and MP schedule. As an example, we showed an extension to the standard algorithm by incorporating a **smoothing pass** alongside the forwards pass. This algorithm requires no updates to the EFE computation in Eq. (2.5) itself, uses known MP rules as implemented in ex. [3] and can be combined with the

SI **backward message**. We show this algorithm on the FFG in Fig 2.2, panel 3. The **smoothing pass** is closely related to the generalised free energy (GFE) introduced in [81]. In [81], the authors also incorporate a **smoothing pass**, however, they rely on custom update rules that are not immediately expressible using known message passing schemes. We show how the GFE update rules are amenable to an MP interpretation in Chapter 5.

## 2.6 Conclusions

The MP perspective on EFE-based planning is not new and has been touched upon in for instance [15, 16, 35, 36, 60, 67]. Our contribution is to formalise this notion by explicitly writing out the necessary steps for planning under AIF in terms of the required messages, and to demonstrate that taking the MP perspective can yield new insights and potentially even new algorithms. An immediate advantage of the MP perspective is that it becomes easy to understand which computations are necessary for a particular planning algorithm, which procedures may be combined to design new planning algorithms, and how to isolate differences between planning algorithms. Additionally, we have only investigated the simplest version of the EFE. Since [33], numerous extensions have been proposed that for instance augment the EFE with additional epistemic terms [100] or express goals in terms of  $p(\mathbf{z}_k)$  rather than  $p(\mathbf{x}_k)$  [22]. Interpreting these more recent developments in terms of MP is an interesting area for future study. Finally, we have focused on the case where an explicit generative model is available, as is common for AIF studies, and have deliberately eschewed discussions of deep AIF such as [72, 110, 115] When parameterising the generative model using deep neural networks, one generally loses the ability to utilise closed-form MP updates. Instead, deep AIF often relies on sampling-based methods to approximate the messages, trading off interpretability and speed for increased flexibility.

# Chapter 3

## On Epistemics in Expected Free Energy for Linear Gaussian State Space Models

*"The truth always turns out to be simpler than you thought."*

– Richard Feynman, 1985

Active Inference (AIF) is a framework that can be used both to describe information processing in naturally intelligent systems, such as the human brain, and to design synthetic intelligent systems (agents). In this chapter we show that expected free energy (EFE) minimisation, a core feature of the framework, does not lead to purposeful explorative behaviour in linear Gaussian dynamical systems (LGDSs).

We provide a simple proof that, due to the specific construction used for the EFE, the terms responsible for the exploratory (epistemic) drive become constant in the case of linear Gaussian systems. This renders AIF equivalent to KL control. From a theoretical point of view this is an interesting result since it is generally assumed that EFE minimisation will *always* introduce an exploratory drive in AIF agents.

While the full EFE objective does not lead to exploration in LGDSs, the principles of its construction can still be used to design objectives that include an

epistemic drive. We provide an in-depth analysis of the mechanics behind the epistemic drive of AIF agents and show how to design objectives for LGDSs that do include an epistemic drive. Concretely, we show that focusing solely on epistemics and dispensing with goal-directed terms leads to a form of maximum entropy exploration that is heavily dependent on the type of control signals driving the system. Additive controls do not permit such exploration. From a practical point of view this is an important result since LGDSs with additive controls are an extensively used model class, encompassing for instance Linear Quadratic Gaussian controllers. On the other hand, LGDSs driven by multiplicative controls such as switching transition matrices do permit an exploratory drive.

This chapter is based on the original work referenced below. The approach we take relies on the Forney-style factor graph (FFG) specification of AIF introduced in Chapter 2. I conceived of the original idea and performed all analyses and experiments. Derivations were performed by me in close cooperation with W. M. Kouw.

**Koudahl, M. T., Kouw, W. M., & de Vries, B. (2021).** *On Epistemics in Expected Free Energy for Linear Gaussian State Space Models.* *Entropy*, 23(12), 1565. MDPI AG.

### 3.1 Introduction

AIF is a mathematical description of information processing in intelligent systems. In brief, it states that agents, originally biological but in later years also artificial, act to minimise their surprise by seeking out stimuli and states that are compatible with their model of the world. AIF is an attractive framework for designing artificial agents since AIF agents possess a well-balanced drive towards both explorative (epistemic) and exploitative (pragmatic, goal-driven) behaviour. These characteristics follow from choosing the EFE as the objective function for planning.

In this chapter, we explicitly derive the equations for applying AIF in LGDSs with the EFE objective. In doing so we uncover a novel result showing that, in the case of linear models, the epistemic term of the EFE objective becomes constant. This means that any application of EFE in LGDS will not lead to exploration and the resulting agents will engage in purely goal-driven behaviour. The proof is given in Section 3.5.5.

The remainder of the chapter is structured as an in-depth analysis of the AIF framework and the mechanisms driving its claims to epistemic behaviour. We isolate the epistemic term of the EFE and identify it as a (bound on) mutual

information (MI). We then show that, when considering epistemics in isolation instead of the full EFE construct, it is still possible to generate an epistemic drive using the machinery of AIF. Isolating epistemics corresponds to a special case of EFE, where priors on future observations are left unspecified [34, 98]. We analyze the behaviour of the resulting epistemic drive and show that, for the case of additive controls, the epistemic drive is independent of state transitions and only depends on the prior variance associated with the belief over the control signal. On the other hand, LGDSs driven by multiplicative control signals do exhibit a dependence between state transitions and the epistemic drive.

Prior work on AIF in LGDS such as [5, 6, 12, 29, 67], have focused mostly on the goal-directed components of the AIF framework. The results, while impressive, largely do not address questions of epistemics and exploration. This means that in cases where AIF is applied to LGDSs, EFE and the resulting desirable exploratory drive have so far not been thoroughly investigated. Our results show that, provided the model in question can be cast as a LGDS, incorporating EFE does not lead to meaningful exploration.

The present chapter makes the following contributions:

- We derive the filtering and planning equations for AIF using EFE in LGDSs, Sections 3.4 and 3.5.
- We consider the epistemic term of EFE in isolation and show that in the case of additive controls actions become decoupled from state transitions when computing the epistemic term of EFE, Section 3.5.3. Therefore, we do not find meaningful exploration in this case.
- We show that in the case of multiplicative controls, meaningful exploratory behaviour re-emerges when isolating the epistemic term of EFE, Section 3.5.4.
- We prove that when considering the full EFE construct, parts of the instrumental and epistemic value terms cancel each other out. This renders the epistemic value constant. In turn, the EFE functional becomes equivalent to KL control plus an additive constant, Section 3.5.5.
- We provide simulations that corroborate our claims. We first demonstrate the differences in exploration when considering purely epistemic agents using both additive and multiplicative control signals. Finally, we show that LGDS agents optimising the full EFE do not exhibit epistemic drives under any circumstances, Section 3.6.



The core message is thus that translating AIF to the linear Gaussian case presents unique challenges, specifically because the exploration/exploitation trade-off that follows from EFE minimisation does not manifest.<sup>1</sup>

## 3.2 Exploration and Exploitation

In this section, we aim to introduce the concepts of exploration and exploitation on intuitive grounds before commencing with our formal analysis. Exploitation refers to goal-directed behaviour. An agent that engages in exploitation performs actions that are aimed at optimising some measure of preferences which we will refer to as "Instrumental value". As an example, consider minimisation of mean squared error, cross-entropy or a similar cost function. Exploration, on the other hand, refers to behaviour directed at collecting information about the environment in which the agent is embedded. An agent that engages in exploration performs actions that are aimed at acquiring further information about its environment. We will refer to any metric that quantifies the value of gathering information as "Epistemic value". Optimising epistemic value biases the agent towards actions that gather information. We will refer to this bias in action selection as an "Epistemic drive". There are many candidates for the epistemic value term. We will briefly consider two that are particularly relevant for the present analysis. This will not be a formal comparison but an intuitive introduction to the qualitative differences in behaviour that can be expected from agents that employ different epistemic value terms.

First, we can consider agents that aim to maximise entropy (uncertainty). For such an agent, the epistemic drive biases it towards seeking out areas of state space where uncertainty is high. By repeatedly visiting uncertain areas of state space, the agent collects observations in said areas which in turn reduces uncertainty. As an example, we can consider an agent trying to navigate an arena. The agent is equipped with a sensor and the arena is subject to strong winds that induce sensor noise by pushing the agent around. In this case, maximising entropy drives the agent to seek out parts of the arena where the winds (and corresponding sensor noise) are high. This means the agent collects information primarily in areas where more observations are needed, due to increased sensor noise.

Second, we can consider an agent that aims to maximise MI, also known as Information Gain. We provide a formal treatment of MI in Appendix A.4.

---

<sup>1</sup>Code is available at [github.com/biaslab/efe\\_lgds](https://github.com/biaslab/efe_lgds)

Intuitively, MI scores the reduction in uncertainty that the agent expects given a particular observation. In the present example, an agent that optimises MI might correctly identify that although windy areas are noisy, collecting information in those areas is unlikely to reduce uncertainty because the winds will remain high. Instead, the agent will prefer to move towards areas that have less wind, to obtain more accurate measurements. This is the approach taken by AIF agents when optimising EFE [33, 97].

Optimising both instrumental and epistemic value terms by selecting actions necessarily entails a trade-off between short-term gains (exploitation) and gathering information in order to perform better in the future (exploration). Having agents that can optimally balance this trade-off is therefore desirable because it allows for autonomous systems that can learn to navigate novel environments to achieve desired goals. A core feature of the EFE is that it presents a single objective functional that encompasses both instrumental and epistemic value terms [33, 97].

In order to formally unpack how AIF manifests both instrumental and epistemic value terms, we now need to detail the LGDS model that specifies our agent before deriving the equations for computing the EFE objective.

### 3.3 Generative Model

AIF is fundamentally a model-based approach [33, 41, 49]. As such, the core part of an agent is given by a generative model. Given a generative model, the agent engages in a perception-action loop with its environment. In practice, this means the agent will, at any time step, absorb a new observation and emit a new action. The first step is always perception, followed by action selection and emission.

Letting  $\mathbf{x} \in \mathbb{R}^d$  denote observations,  $\mathbf{z} \in \mathbb{R}^n$  a latent state vector and  $\mathbf{u}$  actions (we will use "actions" and "controls" interchangeably to refer to  $\mathbf{u}$  throughout), the generative model for an agent at a single time step, indicated by subscripts  $t$ , has the form <sup>2</sup>

$$p(\mathbf{x}_t, \mathbf{z}_t \mid \mathbf{u}_t, \mathbf{z}_{t-1}) = \underbrace{p(\mathbf{x}_t \mid \mathbf{z}_t)}_{\text{Likelihood}} \underbrace{p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_t)}_{\text{State transition}}. \quad (3.1)$$

<sup>2</sup>This form can be extended, for example by including parameters  $\theta$ .

If applied recursively, this model corresponds to a discrete-time state space model. A common approach when designing AIF agents is to work directly with a policy defined as a *particular* sequence of actions  $\mathbf{u}_{t+1:T}$  [21, 33, 35, 81] where the subscript denotes discrete time steps ranging from the next time step  $t + 1$  to some known planning horizon  $T$ . In Eq. (3.1) we indicate this by explicitly conditioning on  $\mathbf{u}_t$ . This sequence of actions is then considered either as an explicit vector of control signals [26, 33, 35, 107] or amortised for instance by neural networks [72, 108–110]. Proceeding in this way leads to a particular scheme for action selection which we will detail in Section 3.5.

In this chapter, we consider the case of LGDS with multiplicative or additive controls. To clarify the distinction between additive and multiplicative controls, we define "multiplicative controls" as state transitions of the form

$$p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_t) = \mathcal{N}(\mathbf{z}_t \mid \mathbf{B}(\mathbf{u}_t)\mathbf{z}_{t-1}, \boldsymbol{\Sigma}_z), \tag{3.2}$$

where  $\mathbf{z}_t \in \mathbb{R}^n$  is a latent state vector,  $\mathbf{u}_t$  is a discrete control signal and  $\mathbf{B}(\mathbf{u}_t)$  is the transition matrix. We consider the case where the control signal functions as a selector variable. Formally we define a vector of candidate transition matrices  $[\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_S]$  and let

$$\mathbf{B}(\mathbf{u}_t) = \prod_{s=1}^S \mathbf{B}_s^{u_{ts}}. \tag{3.3}$$

Here  $\mathbf{u}_t$  is a one-hot encoded vector  $\mathbf{u}_t = [u_{t1}, \dots, u_{tS}]$  that takes values in  $u_{ts} \in \{0, 1\}$  and where  $\sum_{s=1}^S u_{ts} = 1$ . Each  $\mathbf{B}_s$  is raised to the power given by  $u_{ts}$ , which means that only the selected  $\mathbf{B}_s$  will be active. The control signal therefore influences state transitions by selecting the transition matrix  $\mathbf{B}$  directly.

We can visualise this model using the FFG formalism [25]. In an FFG, each edge represents a variable and each node a factor. An edge is connected to a node if and only if the corresponding variable is an argument of that factor. Each edge connects at most two nodes. When a variable is an argument of more than two factors, we can circumvent the two-node-per-edge limit by linking edges together through equality factors. This effectively creates an auxiliary variable (a new edge) for which the posterior beliefs are constrained to be equal to the beliefs for the original variable. The new edge can also be attached to two factors, so by adding equality factors we can use the same variable as an argument

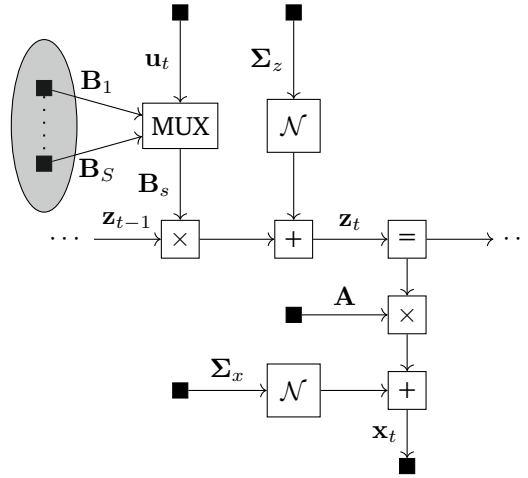


Figure 3.1: FFG of the generative model of an agent with multiplicative control signals.

of multiple factors. Observed variables and clamped parameters are denoted by a small black square and selected actions by a small black diamond. The selection mechanism described by Eq. (3.3) is denoted by the multiplexer (MUX) node. Instead of cluttering the graph by drawing the full set of  $[\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_S]$  candidate transition matrices as separate nodes, we denote them by a shaded circle. The circle contains  $S$  nodes and their corresponding outgoing edges all connect to the MUX node. For a further introduction to FFGs, see [69, 70]. The FFG of the multiplicative model can be seen in Fig. 3.1.

For comparison, we now consider the case of additive controls. For the additive case, the generative model is again given by Eq. (3.1). However exact model specification is a little more involved. We consider transition models of the form

$$p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_t) = \mathcal{N}(\mathbf{z}_t \mid \mathbf{B}\mathbf{z}_{t-1} + \mathbf{b}(\mathbf{u}_t), \Sigma_z), \quad (3.4)$$

where  $\mathbf{B} \in \mathbb{R}^{n \times n}$  is a known transition matrix and  $\mathbf{b}(\mathbf{u}_t) \in \mathbb{R}^n$  is a vector function that adds to the latent state. To rigorously compare the multiplicative and additive control cases,  $\mathbf{u}_t$  must remain a categorical selector variable. To that end, we introduce an auxiliary variable  $\mathbf{b}_t$ . The purpose of  $\mathbf{b}_t$  is to allow  $\mathbf{u}_t$

to function as a categorical selector variable. Instead of selecting between transition matrices  $\mathbf{B}_s$ ,  $\mathbf{u}_t$  now selects the parameters  $\Theta_s = \{\boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s\}$  of a Gaussian input signal. Formally, we will write the generative model as

$$p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_t) = \int p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{b}_t) p(\mathbf{b}_t | \mathbf{u}_t) d\mathbf{b}_t \tag{3.5a}$$

$$= \int \mathcal{N}(\mathbf{z}_t | \mathbf{B}\mathbf{z}_{t-1} + \mathbf{b}_t, \boldsymbol{\Sigma}_z) \prod_{s=1}^S \mathcal{N}(\mathbf{b}_t | \Theta_s)^{u_{ts}} d\mathbf{b}_t. \tag{3.5b}$$

where we recognise a similar selection mechanism of Eq. (3.3) in the second term of Eq. (3.5b). Selecting an action means fixing  $\mathbf{u}_t = \hat{\mathbf{u}}_t$  which leads to selecting one of the candidate Gaussian distributions. With only a single Gaussian surviving, integration over  $\mathbf{b}_t$  becomes straightforward and yields

$$p(\mathbf{z}_t | \mathbf{z}_{t-1}, \hat{\mathbf{u}}_t) = \mathcal{N}(\mathbf{z}_t | \mathbf{B}\mathbf{z}_{t-1} + \hat{\boldsymbol{\mu}}_{u_t}, \boldsymbol{\Sigma}_z + \hat{\boldsymbol{\Sigma}}_{u_t}), \tag{3.6}$$

where  $\hat{\Theta}_s = \{\hat{\boldsymbol{\mu}}_{u_t}, \hat{\boldsymbol{\Sigma}}_{u_t}\}$  represent the parameters of the Gaussian distribution selected by  $\hat{\mathbf{u}}_t$ .

The factor graph of the additive model is shown in Fig. 3.2 where the MUX node now selects between  $\Theta_{1:s}$ . In both the multiplicative and additive settings, we employ a likelihood term of the form:

$$p(\mathbf{x}_t | \mathbf{z}_t) = \mathcal{N}(\mathbf{x}_t | \mathbf{A}\mathbf{z}_t, \boldsymbol{\Sigma}_x), \tag{3.7}$$

where  $\mathbf{A} \in \mathbb{R}^{d \times n}$  is a known emission matrix and  $\boldsymbol{\Sigma}_x$  represents measurement or observation noise. Having established the relevant model structures, we now examine the perception/action loop starting with perception.

### 3.4 Perception as Bayesian Filtering

The perception part of the action/perception loop involves performing inference about observed data and can be cast as a Bayesian filtering problem. This part of the process describes the agent inferring the hidden state of its environment based on the sequence of actions taken so far, the resulting sequence of states visited, and the accompanying observations.

We can write the resulting inference problem in an intuitive way as a prediction-correction process:

$$\underbrace{p(z_t | \mathbf{x}_{1:t})}_{\text{posterior}} = \underbrace{\frac{p(x_t | z_t)}{p(x_t | \mathbf{x}_{1:t-1})}}_{\substack{\text{correction} \\ \text{based on } x_t}} \times \underbrace{p(z_t | \mathbf{x}_{1:t-1})}_{\substack{\text{prediction of } z_t \\ \text{based on } \mathbf{x}_{1:t-1}}}. \quad (3.8)$$

The above shows how the inference over states can be accomplished recursively (due to the model obeying the Markov property) by first computing a prediction for the next hidden state  $z_t$  to generate a prior belief which is then updated in a correction step based on the observed data point  $x_t$ .

The Bayesian filtering problem is generic. To see how it translates to our case, we can expand the prior predictive in terms of our generative model

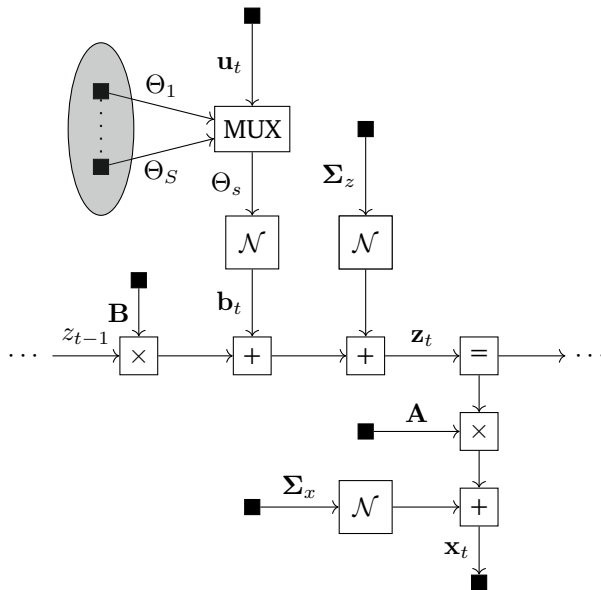


Figure 3.2: FFG of the generative model of an agent with additive controls.

3

$$\underbrace{p(\mathbf{z}_t \mid \mathbf{x}_{1:t})}_{\text{state posterior}} = \underbrace{\frac{p(\mathbf{x}_t \mid \mathbf{z}_t)}{p(\mathbf{x}_t \mid \mathbf{x}_{1:t-1})}}_{\text{evidence}} \underbrace{\int \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_t)}_{\text{state transition}} \underbrace{\delta(\mathbf{u}_t - \hat{\mathbf{u}}_t)}_{\text{control signal}} \underbrace{p(\mathbf{z}_{t-1} \mid \mathbf{x}_{1:t-1})}_{\text{state prior}} d\mathbf{z}_{t-1} d\mathbf{u}_t, \tag{3.9}$$

We use  $\delta(\mathbf{u}_t - \hat{\mathbf{u}}_t)$  where  $\delta$  is the Dirac- $\delta$  to fix the value of  $\mathbf{u}_t$  to the selected action  $\hat{\mathbf{u}}_t$  for that particular time step. The particular value chosen for  $\hat{\mathbf{u}}_t$  is the result of the action selection procedure described in Section 3.5. The evidence term is given by

$$p(\mathbf{x}_t \mid \mathbf{x}_{1:t-1}) = \int p(\mathbf{x}_t \mid \mathbf{z}_t) \left( \int \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_t) \delta(\mathbf{u}_t - \hat{\mathbf{u}}_t) p(\mathbf{z}_{t-1} \mid \mathbf{x}_{1:t-1}) d\mathbf{z}_{t-1} d\mathbf{u}_t \right) d\mathbf{z}_t. \tag{3.10}$$

For the LGDS models considered in this chapter, filtering can be performed using the Kalman filtering equations. We will first work this out explicitly in the multiplicative case and then in the additive. To show how to perform filtering in the multiplicative model, we start by assuming that the agent has selected an action  $\mathbf{u}_t = \hat{\mathbf{u}}_t$  by the procedure described in Section 3.5. We can then calculate the prior predictive distribution of Eq. (3.9) according to our model specification Eq. (3.1) as

$$p(\mathbf{z}_t | \mathbf{x}_{1:t-1}) = \iint p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{u}_t) \delta(\mathbf{u}_t - \hat{\mathbf{u}}_t) p(\mathbf{z}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{z}_{t-1} d\mathbf{u}_t \quad (3.11a)$$

$$= \iint \underbrace{\mathcal{N}(\mathbf{z}_t | \mathbf{B}(\mathbf{u}_t)\mathbf{z}_{t-1}, \boldsymbol{\Sigma}_z)}_{\text{State transition}} \times \underbrace{\delta(\mathbf{u}_t - \hat{\mathbf{u}}_t)}_{\text{Selected control signal}} \underbrace{\mathcal{N}(\mathbf{z}_{t-1} | \boldsymbol{\mu}_{z_{t-1}}, \boldsymbol{\Sigma}_{z_{t-1}})}_{\text{State prior}} d\mathbf{z}_{t-1} d\mathbf{u}_t \quad (3.11b)$$

$$= \int \mathcal{N}(\mathbf{z}_t | \mathbf{B}(\hat{\mathbf{u}}_t)\mathbf{z}_{t-1}, \boldsymbol{\Sigma}_z) \mathcal{N}(\mathbf{z}_{t-1} | \boldsymbol{\mu}_{z_{t-1}}, \boldsymbol{\Sigma}_{z_{t-1}}) d\mathbf{z}_{t-1} \quad (3.11c)$$

$$= \mathcal{N}(\mathbf{z}_t | \underbrace{\hat{\mathbf{B}}_t \boldsymbol{\mu}_{z_{t-1}}}_{\boldsymbol{\mu}_{z_t}^-}, \underbrace{\hat{\mathbf{B}}_t \boldsymbol{\Sigma}_{z_{t-1}} \hat{\mathbf{B}}_t^T + \boldsymbol{\Sigma}_z}_{\boldsymbol{\Sigma}_{z_t}^-}), \quad (3.11d)$$

which we recognise as the prediction step of a Kalman filter [99]. We use the superscript  $-$  notation to indicate that the variable in question is not based on the full data set  $\mathbf{x}_{1:t}$  but instead on a smaller data set  $\mathbf{x}_{1:t-1}$ . In moving from Eq. (3.11b) to Eq. (3.11c) we rely on the sifting property of the Dirac- $\delta$  to substitute the selected value for  $\mathbf{u}_t$  in Eq. (3.3). Since  $\mathbf{B}(\mathbf{u}_t)$  is a function of  $\mathbf{u}_t$  and  $\mathbf{u}_t$  is now fixed to  $\hat{\mathbf{u}}_t$ , we can directly substitute the selected parameterisation by setting  $\mathbf{B}(\mathbf{u}_t) = \hat{\mathbf{B}}_t$  where  $\hat{\mathbf{B}}_t$  denotes the parameterisation given by the selected  $\mathbf{B}_s$ . This takes us from Eq. (3.11b) to Eq. (3.11c). Finally, we can rely on standard results for linearly related and jointly Gaussian variables to go from Eq. (3.11c) to Eq. (3.11d), see for example [99, Appendix A.1] for details of this move in the context of Gaussian state space models or Appendix A.2 for an abbreviated version.

For the additive control case, we can calculate the prior predictive distribution in a similar fashion. Starting from the model definition Eq. (3.1), we can write



$$p(\mathbf{z}_t | \mathbf{x}_{1:t-1}) = \iiint p(\mathbf{z}_t | \mathbf{z}_{t-1}, \mathbf{b}_t) p(\mathbf{b}_t | \mathbf{u}_t) \delta(\mathbf{u}_t - \hat{\mathbf{u}}_t) \times p(\mathbf{z}_{t-1} | \mathbf{x}_{1:t-1}) d\mathbf{u}_t d\mathbf{b}_t d\mathbf{z}_{t-1} \quad (3.12a)$$

$$= \iiint \underbrace{\mathcal{N}(\mathbf{z}_t | \mathbf{B}\mathbf{z}_{t-1} + \mathbf{b}_t, \Sigma_z) \prod_{s=1}^S \mathcal{N}(\mathbf{b}_t | \boldsymbol{\mu}_s, \Sigma_s)^{u_{ts}}}_{\text{State transition}} \underbrace{\delta(\mathbf{u}_t - \hat{\mathbf{u}}_t)}_{\text{Selected control signal}} \times \underbrace{\mathcal{N}(\mathbf{z}_{t-1} | \boldsymbol{\mu}_{z_{t-1}}, \Sigma_{z_{t-1}})}_{\text{State prior}} d\mathbf{u}_t d\mathbf{b}_t d\mathbf{z}_{t-1} \quad (3.12b)$$

$$= \iint \mathcal{N}(\mathbf{z}_t | \mathbf{B}\mathbf{z}_{t-1} + \mathbf{b}_t, \Sigma_z) \prod_{s=1}^S \mathcal{N}(\mathbf{b}_t | \boldsymbol{\mu}_s, \Sigma_s)^{\hat{u}_{ts}} \times \mathcal{N}(\mathbf{z}_{t-1} | \boldsymbol{\mu}_{z_{t-1}}, \Sigma_{z_{t-1}}) d\mathbf{b}_t d\mathbf{z}_{t-1} \quad (3.12c)$$

$$= \iint \mathcal{N}(\mathbf{z}_t | \mathbf{B}\mathbf{z}_{t-1} + \mathbf{b}_t, \Sigma_z) \underbrace{\mathcal{N}(\mathbf{b}_t | \hat{\boldsymbol{\mu}}_{u_t}, \hat{\Sigma}_{u_t})}_{\substack{\text{Selected} \\ \text{parameters } \hat{\Theta}_s}} \times \mathcal{N}(\mathbf{z}_{t-1} | \boldsymbol{\mu}_{z_{t-1}}, \Sigma_{z_{t-1}}) d\mathbf{b}_t d\mathbf{z}_{t-1} \quad (3.12d)$$

$$= \int \mathcal{N}(\mathbf{z}_t | \mathbf{B}\mathbf{z}_{t-1} + \hat{\boldsymbol{\mu}}_{u_t}, \Sigma_z + \hat{\Sigma}_{u_t}) \mathcal{N}(\mathbf{z}_{t-1} | \boldsymbol{\mu}_{z_{t-1}}, \Sigma_{z_{t-1}}) d\mathbf{z}_{t-1} \quad (3.12e)$$

$$= \mathcal{N}(\mathbf{z}_t | \underbrace{\mathbf{B}\boldsymbol{\mu}_{z_{t-1}} + \hat{\boldsymbol{\mu}}_{u_t}}_{\boldsymbol{\mu}_{z_t}^-}, \underbrace{\Sigma_z + \hat{\Sigma}_{u_t} + \mathbf{B}\Sigma_{z_{t-1}}\mathbf{B}^T}_{\Sigma_{z_t}^-}). \quad (3.12f)$$

We again denote the selected parameters for the additive control at the  $k$ -th time step as  $\hat{\boldsymbol{\mu}}_{u_k}$  and  $\hat{\Sigma}_{u_k}$ . To proceed from Eq. (3.12b) to Eq. (3.12c) we rely on the sifting property of the Dirac- $\delta$  and substitute the selected value for  $\mathbf{u}_t$ . The move from Eq. (3.12c) to Eq. (3.12d) acknowledges that only the selected parameterisation is active once we substitute  $\mathbf{u}_t = \hat{\mathbf{u}}_t$  as covered in Section 3.3. The final steps from Eq. (3.12d) to Eq. (3.12e) and from Eq. (3.12e) to Eq. (3.12f) uses standard results for multiplication and marginalisation of jointly Gaussian variables for which we again refer to [99, Appendix A.1] and Appendix A.2. In summary, we see that when  $\mathbf{u}_t = \hat{\mathbf{u}}_t$  the model specification

given in Eq. (3.5b) reduces to a standard LGDS and can be updated using the prediction step of the Kalman filtering equations.

Both the additive and multiplicative models use similar likelihood models, meaning they can be updated using the same Kalman correction step. To perform the Kalman correction step, we need to apply Bayes rule

$$\underbrace{p(\mathbf{z}_t | \mathbf{x}_{1:t})}_{\text{posterior}} \underbrace{p(\mathbf{x}_t | \mathbf{x}_{1:t-1})}_{\text{evidence}} = \underbrace{p(\mathbf{x}_t | \mathbf{z}_t)}_{\text{likelihood}} \underbrace{p(\mathbf{z}_t | \mathbf{x}_{1:t-1})}_{\text{prior}}, \quad (3.13)$$

where the factors on the right-hand side (RHS) are given and the terms on the left-hand side are the desired factors. This equation can be solved analytically. First, we evaluate the RHS as

$$p(\mathbf{x}_t | \mathbf{z}_t) p(\mathbf{z}_t | \mathbf{x}_{1:t-1}) = \mathcal{N}(\mathbf{x}_t | \mathbf{A}\mathbf{z}_t, \Sigma_x) \mathcal{N}(\mathbf{z}_t | \boldsymbol{\mu}_{z_t}^-, \Sigma_{z_t}^-) \quad (3.14a)$$

$$= \mathcal{N} \left( \begin{bmatrix} \mathbf{z}_t \\ \mathbf{x}_t \end{bmatrix} \middle| \begin{bmatrix} \boldsymbol{\mu}_{z_t}^- \\ \mathbf{A}\boldsymbol{\mu}_{z_t}^- \end{bmatrix}, \begin{bmatrix} \Sigma_{z_t}^- & \Sigma_{z_t}^- \mathbf{A}^T \\ \mathbf{A}\Sigma_{z_t}^- & \mathbf{A}\Sigma_{z_t}^- \mathbf{A}^T + \Sigma_x \end{bmatrix} \right). \quad (3.14b)$$

Then if, for notational convenience, we rewrite the covariance matrix as

$$\begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix} \triangleq \begin{bmatrix} \Sigma_{z_t}^- & \Sigma_{z_t}^- \mathbf{A}^T \\ \mathbf{A}\Sigma_{z_t}^- & \mathbf{A}\Sigma_{z_t}^- \mathbf{A}^T + \Sigma_x \end{bmatrix}, \quad (3.15)$$

Eq. (3.14b) can be written as the product of the state posterior due to the theorem for decomposing a multi-variate Gaussian into the product of a conditional distribution [99, Appendix A.1],

$$p(\mathbf{z}_t | \mathbf{x}_{1:t}) = \mathcal{N}(\mathbf{z}_t | \boldsymbol{\mu}_{z_t}^- + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{x}_t - \mathbf{A}\boldsymbol{\mu}_{z_t}^-), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}), \quad (3.16)$$

and evidence

$$p(\mathbf{x}_t | \mathbf{x}_{1:t-1}) = \mathcal{N}(\mathbf{x}_t | \mathbf{A}\boldsymbol{\mu}_{z_t}^-, \Sigma_{22}). \quad (3.17)$$

Finally, we can also calculate the conditional distribution  $p(\mathbf{x}_t | \mathbf{z}_t, \mathbf{x}_{1:t-1})$  [99, Appendix A.1]. While this is not required for solving the Bayesian filtering problem, it will prove useful for deriving the epistemic value term in Appendix

A.4 used for the action selection procedure described in Section 3.5. We can find it as

$$p(\mathbf{x}_t \mid \mathbf{z}_t, \mathbf{x}_{1:t-1}) = \mathcal{N}(\mathbf{x}_t \mid \mathbf{A}\boldsymbol{\mu}_{z_t}^- + \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}(\mathbf{z}_t - \boldsymbol{\mu}_{z_t}^-), \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}). \quad (3.18)$$

Having described perception as Bayesian filtering, we now turn our attention to action selection under AIF.

### 3.5 Action Selection under Active Inference

When we are interested in constructing AIF agents, arguably the core task is action selection. Under AIF we solve this task by first computing a prior over future control signals. Technically, we seek to compute

$$p(\mathbf{u}_{t+1:T}) \propto \exp(-G(\mathbf{u}_{t+1:T})), \quad (3.19)$$

i.e., the prior on controls is a softmax function of  $G(\mathbf{u}_{t+1:T})$  [33, Eq. 7]. Here,  $G(\mathbf{u}_{t+1:T})$  denotes the EFE for a policy that extends into the future until a known horizon  $T$ . We further discuss the computation of  $G(\mathbf{u}_{t+1:T})$  in Section 3.5.1. To obtain the control prior at time  $t + 1$ , we can marginalise this distribution as

$$\begin{aligned} p(u_{t+1}) &= \int \cdots \int p(\mathbf{u}_{t+1:T}) du_{t+2} \cdots du_T \\ &\propto \int \cdots \int \exp(-G(\mathbf{u}_{t+1:T})) du_{t+2} \cdots du_T. \end{aligned} \quad (3.20)$$

If we assume independent control priors for each time step, that is, if we assume  $p(\mathbf{u}_{t+1:T}) = \prod_{k=t+1}^T p(u_k)$ , or equivalently,

$$\exp(-G(\mathbf{u}_{t+1:T})) = \prod_{k=t+1}^T \exp(-G(u_k)), \quad (3.21)$$

then the marginalisation Eq. (3.20) evaluates to

$$p(u_{t+1}) \propto \exp(-G(u_{t+1})). \quad (3.22)$$

Since the marginalisation procedure is identical for any other time step, we can deduce that the total EFE for a policy is equal to the exponentiated sum of EFEs at individual time steps. That is

$$p(\mathbf{u}_{t+1:T}) \propto \exp(-G(\mathbf{u}_{t+1:T})) = \prod_{k=t+1}^T \exp(-G(u_k)) = \exp\left(-\sum_{k=t+1}^T G(u_k)\right). \quad (3.23)$$

This suggests a recursive scheme over time steps for computing policy priors, similar to the proposal by [35].

In the AIF literature the executed action is then commonly sampled from  $p(\mathbf{u}_{t+1:T})$  and emitted to the environment [33, 35]. Other action selection approaches such as selecting the MAP estimate (the  $\arg \max$  or mode of the distribution) are also possible. We now turn our attention to how the EFE is computed.

### 3.5.1 Computing G - Expected Free Energy

The EFE is an AIF specific construct that attempts to model what the variational free energy (VFE) would be at a future time step, conditioned on a particular sequence of actions. Of special interest is the decomposition of EFE into an epistemic term and an instrumental (exploitative) term. It is due to this decomposition that AIF claims an adaptive trade-off between exploration and exploitation [33, 35].

Note that we provide the derivation here only for the simplest case. There are extensions to the EFE such as [101] that include additional terms which induce changes in the agent's behaviour. Including these additional terms is not necessary for our core argument so we omit them here and refer interested readers to [22, 101].

To show how we arrive at our formulation for EFE, we first need to introduce variational inference. While the filtering equations in Section 3.4 permit analytical solutions by applying Bayes rule directly, this is often not the case as solving the required integrals can become intractable. In those cases, we can instead

approximate the exact solution  $p(z_t | \mathbf{x}_{1:t})$  by a recognition density  $q(z_t)$ . Formally we accomplish this by minimising the Kullback-Leibler divergence (KL) between the exact solution and our recognition density

$$KL[q(z_t)||p(z_t | \mathbf{x}_{1:t})] = \int q(z_t) \log \frac{q(z_t)}{p(z_t | \mathbf{x}_{1:t})} dz_t \tag{3.24}$$

If we now multiply and divide by  $p(x_t | \mathbf{x}_{1:t-1})$  inside the log-operator,

$$KL[q(z_t)||p(z_t | \mathbf{x}_{1:t})] = \int q(z_t) \left[ \log \frac{q(z_t)}{p(x_t, z_t | \mathbf{x}_{1:t-1})} + \log p(x_t | \mathbf{x}_{1:t-1}) \right] dz_t \tag{3.25}$$

$$= \underbrace{\int q(z_t) \log \frac{q(z_t)}{p(x_t, z_t | \mathbf{x}_{1:t-1})} dz_t}_{\text{VFE } F[q]} + \underbrace{\log p(x_t | \mathbf{x}_{1:t-1})}_{\text{log-evidence}} \tag{3.26}$$

we obtain the VFE  $F[q]$  by noting that  $\log p(x_t | \mathbf{x}_{1:t-1})$  is not dependent on  $z$ . Since the term  $p(x_t, z_t | \mathbf{x}_{1:t-1})$  in the denominator of  $F[q]$  is given by our generative model, we can choose constraints on  $q(z_t)$  to make optimisation of Eq. (3.26) tractable. Minimising  $F[q]$  then constitutes an upper bound on  $-\log p(x_t | \mathbf{x}_{1:t-1})$ , meaning we can optimise Eq. (3.26) to obtain an approximate solution to our original inference problem. We define the optimal recognition density  $q^*$  as the one that minimises  $F[q]$ :

$$q^* = \arg \min_{q \in \mathcal{Q}} F[q] \tag{3.27}$$

For further background on variational inference, we refer interested readers to the seminal works by [11, 116]. Now we are ready to introduce the EFE. To do so, we start by obtaining our best estimate of the time step  $k$  in question by integrating out contributions from past time steps as

$$p(x_k, z_k | u_k) = \int p(x_k, z_k | z_{k-1}, u_k) p(z_{k-1} | \mathbf{x}_{1:t}) dz_{k-1}, \tag{3.28}$$

where we write  $k$  instead of  $t$  to indicate that we are referring to an arbitrary time step within the planning horizon  $t < k \leq T$ .  $p(z_{k-1} | \mathbf{x}_{1:t})$  denotes the posterior state estimate at the previous time step given all available observations. For notational brevity, we suppress the dependency on  $\mathbf{x}_{1:t}$  on the LHS. Unless otherwise noted, all distributions are conditioned on prior observations moving forward.

For LGDSs,  $p(z_{k-1} | \mathbf{x}_{1:t})$  is available from recursive application of the filtering equations described in Section 3.4. For  $k = t + 1$  it is given by Eq. (3.16) and for  $k > t + 1$ ,  $p(z_{k-1} | \mathbf{x}_{1:t})$  is given by Eq. (3.11) in the case of multiplicative controls and Eq. (3.12) in the case of additive controls.<sup>3</sup>

Now we can write out the VFE conditioned on a particular action  $u_k = \hat{u}_k$  and recognition density as  $F[q; u_k]$ . Note that while  $F[q]$  is a *functional* (a function of a function) of  $q$ , we also explicitly include conditioning on action given by the *parameter*  $u_k$ . To differentiate, we separate them with a semicolon when writing  $F[q; u_k]$ .

Given the factorisation in Eq. (3.23), it is sufficient to consider a single time step  $k$  since we can substitute any value for  $k$ . This gives us

$$F[q; u_k] = \int q(z_k | u_k) \log \frac{q(z_k | u_k)}{p(x_k, z_k | u_k)} dz_k. \quad (3.29)$$

However, this expression includes observations  $x_k$  which are not available, since we are working with time steps in the future ( $t < k \leq T$ ), and the future is by definition not observed yet. To alleviate this issue, we can take the expectation of this expression with respect to the data-generating distribution over observations. When the data-generating distribution is available from the generative model, we can equivalently write  $p(x_k | z_k)$  instead of  $q(x_k | z_k)$ . This gives the expression for the EFE at the  $k$ 'th time step:

$$G[q; u_k] = \underbrace{\int \int q(x_k | z_k) \left[ \overbrace{q(z_k | u_k) \log \frac{q(z_k | u_k)}{p(x_k, z_k | u_k)} dz_k}^{F[q; u_k] \text{ if } x_k \text{ was observed}} \right] dx_k}_{\text{Expected } F[q; u_k] \text{ since } x_k \text{ is not yet observed}}. \quad (3.30)$$

<sup>3</sup>When  $p(z_{k-1} | \mathbf{x}_{1:t})$  can not be obtained through application of Bayes rule (providing the exact solution), one can resort to variational inference (providing an approximate solution). In that case, derivations must instead proceed in terms of the approximate posterior  $q(z_{k-1} | \mathbf{x}_{1:t})$ .

As with the VFE in Eq. (3.26), we are interested in the minimum of Eq. (3.30) which once again entails finding  $q^*$ . For clarity of notation, we define the solution as

$$G(u_k) = G[q^*; u_k] = \arg \min_{q \in \mathcal{Q}} G[q; u_k], \tag{3.31}$$

where  $G(u_k)$  is used to compute the policy prior by plugging into Eq. (3.23). Note that  $G(u_k)$  is a scalar value that denotes the expectation of  $F[q^*; u_k]$  under a particular set of constraints on  $q$  and given a specific action  $u_k$ . To get an intuition for  $G(u_k)$  it can be useful to think of the computation as a two-step procedure consisting of an inner and an outer loop. The inner loop performs variational inference and finds  $q^*$  conditioned on an action  $u_k$ . The outer loop then computes the resulting EFE by taking the expectation of  $F[q^*; u_k]$  under the matching data generating distribution. A core property of EFE is that it introduces an epistemic value term into the optimisation. This leads agents that optimise EFE to seek out areas of state space that have high information gain under the current model, allowing for a principled trade-off between exploration and exploitation [97, 100]. To show how this comes about, we can decompose the EFE into a cross-entropy loss and a MI term where the latter quantifies the information gain (in nats or bits) about hidden states  $z_k$  from observing outcomes  $x_k$ . For the following derivation, we will need a bound, the details of which can be found in Appendix A.3. Starting from Eq. (3.30), we can factorise the denominator as  $p(x_k, z_k | u_k) = p(z_k | x_k, u_k)p(x_k)$ , leading to

$$\begin{aligned} G(u_k) &= \iint q(x_k | z_k) \left[ q(z_k | u_k) \log \frac{q(z_k | u_k)}{p(z_k | x_k, u_k)p(x_k)} dz_k \right] dx_k \\ &= \iint q(x_k | z_k) \left[ q(z_k | u_k) \left[ \log \frac{q(z_k | u_k)}{p(z_k | x_k, u_k)} - \log p(x_k) \right] dz_k \right] dx_k. \end{aligned} \tag{3.32}$$

Now we apply the bound from Appendix A.3 to swap  $q$  for  $p$  in the denominator. Making use of this inequality is a standard move across the AIF literature [21, 22, 33, 35, 73]<sup>4</sup>. Instead of applying the bound, another option is to utilise Eq. (3.30) as is, see [43] for an example. We proceed as

<sup>4</sup>The bound becomes exact when we perform exact inference. This is the case in the models we consider here and the discrete partially observed Markov decision process (POMDP) models often employed in AIF research, see for instance [22, 33]

$$\begin{aligned} & \iint q(x_k | z_k) \left[ q(z_k | u_k) \left[ \log \frac{q(z_k | u_k)}{p(z_k | x_k, u_k)} - \log p(x_k) \right] dz_k \right] dx_k \quad (3.33) \\ & \geq \iint q(x_k | z_k) \left[ q(z_k | u_k) \left[ \log \frac{q(z_k | u_k)}{q(z_k | x_k, u_k)} - \log p(x_k) \right] dz_k \right] dx_k . \end{aligned}$$

Finally, we split the integral and integrate over  $z_k$  to obtain

$$\begin{aligned} G(u_k) & \geq \iint q(x_k, z_k | u_k) \log \frac{1}{p(x_k)} dz_k dx_k \\ & \quad - \iint q(x_k, z_k | u_k) \log \frac{q(z_k | x_k, u_k)}{q(z_k | u_k)} dz_k dx_k \quad (3.34a) \end{aligned}$$

$$= \int q(x_k | u_k) \log \frac{1}{p(x_k)} dx_k - \iint q(x_k, z_k | u_k) \log \frac{q(z_k | x_k, u_k)}{q(z_k | u_k)} dz_k dx_k \quad (3.34b)$$

$$\begin{aligned} & = \underbrace{\int q(x_k | u_k) \log \frac{1}{p(x_k)} dx_k}_{\text{cross-entropy}} \\ & \quad - \underbrace{\iint q(x_k, z_k | u_k) \log \frac{q(z_k, x_k | u_k)}{q(z_k | u_k) q(x_k | u_k)} dz_k dx_k}_{\text{Mutual Information, } I[x_k, z_k | u_k]} \quad (3.34c) \end{aligned}$$

In the last line, we multiply and divide by  $q(x_k | u_k)$  to make the MI term explicit. Readers familiar with the broader AIF literature such as [33, 81, 97] might not immediately recognise the form of Eq. (3.34c) as a common decomposition of the EFE. The equivalence between Eq. (3.34c) and the EFE was originally noted in [33] where the move from Eq. (3.34b) to Eq. (3.34c) is done to show the relation between AIF and InfoMax methods. An advantage of writing the EFE as Eq. (3.34c) is that it cleanly shows how the EFE can be viewed as a combination of two well-known and widely established objectives.

From Eq. (3.34c), we see that  $G(u_k)$  decomposes into a (bound on a) cross-entropy term minus an MI term. Maximising MI is a known way to induce exploration (i.e., information gain about hidden states from observations) in agents and has been employed in multiple settings both within the control theory [4, 13] and reinforcement learning literature [8, 43, 72]. The cross-entropy



loss is between a prior  $p(x_k)$  and the posterior distribution  $q(x_k|u_k)$  over future observations. This allows for interpreting  $p(x_k)$  as a target/goal prior [22, 28]. It endows the agent with an instrumental value term that elicits goal-directed behaviour from inferred policies.

Taking this view,  $G(u_k)$  can be adequately viewed as scoring the behaviour resulting from the action  $u_k$  as a balancing act between MI-based explorative and cross-entropy-based exploitative terms. We now examine each of these terms separately to understand how they work in the linear Gaussian case before considering them jointly. We begin by focusing on the MI and how it may drive exploration when considered in isolation.

### 3.5.2 Mutual Information Computation

Epistemic behaviour in AIF agents can be considered to be driven by minimising negative MI, as shown in Eq. (3.34c). MI is in general defined as

$$I[x, z] = \iint p(x, z) \log \frac{p(x, z)}{p(x)p(z)} dx dz = H[z] - H[z | x] = H[x] - H[x | z]. \tag{3.35}$$

Note that we can write Eq. (3.35) in terms of entropies of either  $x$  or  $z$ . We can do this since MI is symmetric in its arguments.

In the LGDS models we consider, we can evaluate the MI component of  $G(\mathbf{u}_k)$  as

$$I[\mathbf{x}_k, \mathbf{z}_k] = \frac{1}{2} \log |\mathbf{I} + \Sigma_x^{-1} \mathbf{A} \Sigma_{z_k}^{-1} \mathbf{A}^T|. \tag{3.36}$$

The detailed derivation of Eq. (3.36) can be found in Section A.4. To facilitate purely epistemic behaviour, AIF agents can optimise this quantity by selecting appropriate control signals. We will therefore use optimisation of MI as the basis from which to investigate purely epistemic behaviour.

### 3.5.3 Pure Exploration as a function of Additive Control Signals

To show the relation between exploration and controls in the additive case, we now need to show how MI depends on the control signal  $\mathbf{u}_k$ . For clarity of notation we will do the derivation for the case  $k > t + 1$ , i.e. we will write in terms

of the prior predictive  $p(\mathbf{z}_{k-1}|\mathbf{x}_{1:k-2}) = \mathcal{N}(\mathbf{z}_{k-1}|\boldsymbol{\mu}_{z_{k-1}}^-, \boldsymbol{\Sigma}_{z_{k-1}}^-)$  obtained from Eq. (3.12) instead of  $p(\mathbf{z}_{k-1}|\mathbf{x}_{1:k-1}) = \mathcal{N}(\mathbf{z}_{k-1}|\boldsymbol{\mu}_{z_{k-1}}, \boldsymbol{\Sigma}_{z_{k-1}})$  from Eq. (3.16). We do so since observations are not available for  $k > t$ , meaning we can not perform a full filtering step for the prior time step  $k - 1$  unless  $k = t + 1$ . For the case  $k = t + 1$ , we can perform filtering for the state prior and can therefore substitute in parameters of  $p(\mathbf{z}_{k-1}|\mathbf{x}_{1:k-1})$  where appropriate. We start the derivation by generating a prediction from our model using Eq. (3.12). We can find the relevant joint distribution at the  $k$ 'th time step by plugging the result into Eq. (3.14b) to obtain

$$p\left(\begin{bmatrix} \mathbf{z}_k \\ \mathbf{x}_k \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mathbf{z}_k \\ \mathbf{x}_k \end{bmatrix} \middle| \begin{bmatrix} \mathbf{B}\boldsymbol{\mu}_{z_{k-1}}^- + \hat{\boldsymbol{\mu}}_{u_k} \\ \mathbf{A}[\mathbf{B}\boldsymbol{\mu}_{z_{k-1}}^- + \hat{\boldsymbol{\mu}}_{u_k}] \end{bmatrix}, \begin{bmatrix} \hat{\boldsymbol{\Sigma}}_{u_k} + \boldsymbol{\Sigma}_z + \mathbf{B}\boldsymbol{\Sigma}_{z_{k-1}}^- \mathbf{B}^T & [\hat{\boldsymbol{\Sigma}}_{u_k} + \boldsymbol{\Sigma}_z + \mathbf{B}\boldsymbol{\Sigma}_{z_{k-1}}^- \mathbf{B}^T] \mathbf{A}^T \\ \mathbf{A}[\hat{\boldsymbol{\Sigma}}_{u_k} + \boldsymbol{\Sigma}_z + \mathbf{B}\boldsymbol{\Sigma}_{z_{k-1}}^- \mathbf{B}^T] & \mathbf{A}[\hat{\boldsymbol{\Sigma}}_{u_k} + \boldsymbol{\Sigma}_z + \mathbf{B}\boldsymbol{\Sigma}_{z_{k-1}}^- \mathbf{B}^T] \mathbf{A}^T + \boldsymbol{\Sigma}_x \end{bmatrix}\right), \quad (3.37)$$

where we see that the control signal contributes an additive term  $\hat{\boldsymbol{\Sigma}}_{u_k}$  which is the variance associated with the selected action. Interestingly, this means that if we let  $\hat{\boldsymbol{\Sigma}}_{u_k}$  go to 0, the covariance matrix becomes identical to the multiplicative case detailed in Section 3.5.4. We plug the marginal over states  $\mathbf{z}_k$  into Eq. (3.36) to get

$$\begin{aligned} \mathbb{I}[\mathbf{x}_k, \mathbf{z}_k] &= \frac{1}{2} \log |\mathbf{I} + \boldsymbol{\Sigma}_x^{-1} \mathbf{A} \underbrace{[\mathbf{B}\boldsymbol{\Sigma}_{z_{k-1}}^- \mathbf{B}^T + \boldsymbol{\Sigma}_z + \hat{\boldsymbol{\Sigma}}_{u_k}]}_{\boldsymbol{\Sigma}_{z_k}} \mathbf{A}^T| \quad (3.38a) \\ &= \frac{1}{2} \log |\mathbf{I} + \underbrace{\boldsymbol{\Sigma}_x^{-1} \mathbf{A} \mathbf{B} \boldsymbol{\Sigma}_{z_{k-1}}^- \mathbf{B}^T \mathbf{A}^T}_{\text{Dynamics dependent}} + \underbrace{\boldsymbol{\Sigma}_x^{-1} \mathbf{A} \boldsymbol{\Sigma}_z \mathbf{A}^T}_{\text{Policy independent}} + \underbrace{\boldsymbol{\Sigma}_x^{-1} \mathbf{A} \hat{\boldsymbol{\Sigma}}_{u_k} \mathbf{A}^T}_{\text{Policy dependent}}|. \quad (3.38b) \end{aligned}$$

We notice that the MI decomposes into three terms. We label the first "Dynamics dependent" since it depends only on the transition matrix  $\mathbf{B}$ , observation noise  $\boldsymbol{\Sigma}_x$  and the prior state variance  $\boldsymbol{\Sigma}_{z_{k-1}}^-$ . The second term is labeled "Policy independent" since it only depends on the observation noise  $\boldsymbol{\Sigma}_x$  and transition noise  $\boldsymbol{\Sigma}_z$ . Note that neither of the first two terms are influenced by the control signal. The last term is the only one to include  $\hat{\boldsymbol{\Sigma}}_{u_k}$  and is therefore "Policy dependent".

Crucially, the policy dependent term only depends on the variance of the selected control signal  $\hat{\Sigma}_{u_k}$  and the observation noise  $\Sigma_x$ . In other words, it is *independent* of the latent state  $\mathbf{z}_k$ . Since both  $\hat{\Sigma}_{u_k}$  and  $\Sigma_x$  are available a priori, we can precompute the effect of a policy on the epistemic value term before receiving any observations. Further, the result is also independent of the trajectory taken by the agent. Therefore in the case of additive controls, maximising MI does not produce targeted exploration. This necessitates the use of a different model structure when epistemic behaviour is desired. A similar result to ours was obtained by [105] for the case of linear dynamics with additive controls.

### 3.5.4 Pure Exploration as a function of Multiplicative Control Signals

To show how epistemic behaviour re-emerges as a function of multiplicative control signals, we now need to show how MI depends on the choice of transition matrix  $\hat{\mathbf{B}}_k$ . We again proceed by generating a prediction from our model using Eq. (3.11). Plugging this into Eq. (3.14b) gives us the joint distribution as

$$p\left(\begin{bmatrix} \mathbf{z}_k \\ \mathbf{x}_k \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \mathbf{z}_k \\ \mathbf{x}_k \end{bmatrix} \middle| \begin{bmatrix} \hat{\mathbf{B}}_k \boldsymbol{\mu}_{z_{k-1}}^- \\ \mathbf{A} \hat{\mathbf{B}}_k \boldsymbol{\mu}_{z_{k-1}}^- \end{bmatrix}, \begin{bmatrix} \Sigma_z + \hat{\mathbf{B}}_k \Sigma_{z_{k-1}}^- \hat{\mathbf{B}}_k^T & [\Sigma_z + \hat{\mathbf{B}}_k \Sigma_{z_{k-1}}^- \hat{\mathbf{B}}_k^T] \mathbf{A}^T \\ \mathbf{A} [\Sigma_z + \hat{\mathbf{B}}_k \Sigma_{z_{k-1}}^- \hat{\mathbf{B}}_k^T] & \mathbf{A} [\Sigma_z + \hat{\mathbf{B}}_k \Sigma_{z_{k-1}}^- \hat{\mathbf{B}}_k^T] \mathbf{A}^T + \Sigma_x \end{bmatrix}\right). \tag{3.39}$$

Plugging the above into (3.36) we find that

$$I[\mathbf{x}_k, \mathbf{z}_k] = \frac{1}{2} \log |\mathbf{I} + \Sigma_x^{-1} \mathbf{A} \underbrace{[\hat{\mathbf{B}}_k \Sigma_{z_{k-1}}^- \hat{\mathbf{B}}_k^T + \Sigma_z]}_{\Sigma_{z_k}} \mathbf{A}^T| \tag{3.40a}$$

$$= \frac{1}{2} \log |\mathbf{I} + \underbrace{\Sigma_x^{-1} \mathbf{A} \hat{\mathbf{B}}_k \Sigma_{z_{k-1}}^- \hat{\mathbf{B}}_k^T \mathbf{A}^T}_{\text{Policy dependent}} + \underbrace{\Sigma_x^{-1} \mathbf{A} \Sigma_z \mathbf{A}^T}_{\text{Policy independent}}|. \tag{3.40b}$$

We see that MI now decomposes into two terms. The first term depends on  $\hat{\mathbf{B}}_k$  and can be controlled by selecting appropriate transition matrices. The second is *independent* of policy as it only involves process ( $\Sigma_z$ ) and observation noise ( $\Sigma_x$ ). Note that similar terms also appear in the additive case. The

difference between the additive and multiplicative cases is that the choice of transition matrix  $\hat{\mathbf{B}}_k$  is now under the control of the agent. To maximise MI, the agent must therefore select the  $\hat{\mathbf{B}}_k$  that maximises the entropy of its latent states  $\mathbf{z}_k$ . Taking this view offers a nice intuitive explanation for the resulting exploratory drive: To gain the most information, we must perform the actions that lead to the most uncertain outcomes as described in Section 3.2. To learn the most, we must sample where we know the least.

### 3.5.5 Instrumental Value and Expected Free Energy

We now turn our attention to the instrumental value term of  $G(u_k)$ , after which we analyse the full EFE construct. Recall from Eq. (3.34c) that the instrumental value term is a cross-entropy of the form

$$\begin{aligned} \int q(x_k|u_k) \log \frac{1}{p(x_k)} dx_k &= \int q(x_k|u_k) \log \frac{q(x_k|u_k)}{p(x_k)q(x_k|u_k)} dx_k \\ &= \text{KL}[q(x_k|u_k) || p(x_k)] + \text{H}[x_k | u_k] . \end{aligned} \quad (3.41)$$

In many cases, it is not trivial to obtain  $q(x_k|u_k)$  due to intractable integrals. However, in the LGDS we are considering, which only involves linear Gaussian relations, it has a tractable expression given by Eq. (3.17). The KL between two Gaussian distributions is given by

$$\begin{aligned} \text{KL}[q(\mathbf{x}_k | \mathbf{u}_k) || p(\mathbf{x}_k)] &= \\ \frac{1}{2} \left( \log \frac{|\boldsymbol{\Sigma}_p|}{|\boldsymbol{\Sigma}_q|} + n + (\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)^T \boldsymbol{\Sigma}_p^{-1} (\boldsymbol{\mu}_q - \boldsymbol{\mu}_p) + \text{tr}(\boldsymbol{\Sigma}_p^{-1} \boldsymbol{\Sigma}_q) \right) . \end{aligned} \quad (3.42)$$

We use subscripts  $\{p, q\}$  to denote whether a term comes from  $p(\mathbf{x}_k)$  or  $q(\mathbf{x}_k|\mathbf{u}_k)$  and use  $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}\}$  for the parameters of the corresponding distribution. We can now consider both terms of Eq. (3.34c) jointly in the case of LGDS. Taking Eq. (3.35) and Eq. (3.41) together and making the conditioning on  $\mathbf{u}_k$  in Eq. (3.35) explicit, we see the full objective comes out as

$$\begin{aligned}
 G(\mathbf{u}_k) &\geq \underbrace{\text{KL}[q(\mathbf{x}_k | \mathbf{u}_k) || p(\mathbf{x}_k)] + \text{H}[\mathbf{x}_k | \mathbf{u}_k]}_{\text{Instrumental value}} - \underbrace{\text{H}[\mathbf{x}_k | \mathbf{u}_k] + \text{H}[\mathbf{x}_k | \mathbf{u}_k, \mathbf{z}_k]}_{\text{Negative MI}} \\
 &= \underbrace{\text{KL}[q(\mathbf{x}_k | \mathbf{u}_k) || p(\mathbf{x}_k)]}_{\text{Risk}} + \underbrace{\text{H}[\mathbf{x}_k | \mathbf{u}_k, \mathbf{z}_k]}_{\text{Ambiguity}},
 \end{aligned}
 \tag{3.43}$$

where we recover the familiar risk and ambiguity terms. In the specific case of LGDS the inequality becomes an equality when we perform exact inference following the equations laid out in Section 3.4. However note that when combining the instrumental and epistemic terms instead of considering them in isolation, we perform a seemingly innocuous cancellation and remove the entropy  $\text{H}[\mathbf{x}_k | \mathbf{u}_k]$  from the equation. Previously  $\text{H}[\mathbf{x}_k | \mathbf{u}_k]$  appeared twice since we considered the epistemic and instrumental terms separately. However when considering the full EFE construct, this is no longer necessary and we are left with just the ambiguity  $\text{H}[\mathbf{x}_k | \mathbf{u}_k, \mathbf{z}_k]$ . Using the entropy expression for a Gaussian distribution, we can write the ambiguity as

$$\text{H}[\mathbf{x}_k | \mathbf{z}_k, \mathbf{u}_k] = \frac{1}{2} (n \log 2\pi + \log |\Sigma_{22} - \Sigma_{21} \Sigma_{11}^{-1} \Sigma_{12}| + n) \tag{3.44}$$

Recalling the form of the joint given in Eq. (3.14b), we can write each block of the covariance matrix out and find

$$\begin{aligned}
 \text{H}[\mathbf{x}_k | \mathbf{z}_k, \mathbf{u}_k] &= \frac{1}{2} \left( n \log 2\pi + \log \left| \underbrace{\mathbf{A} \Sigma_{z_k} \mathbf{A}^T}_{\Sigma_{22}} + \Sigma_x - \underbrace{\mathbf{A} \Sigma_{z_k}}_{\Sigma_{21}} \underbrace{\Sigma_{z_k}^{-1}}_{\Sigma_{11}^{-1}} \underbrace{\Sigma_{z_k} \mathbf{A}^T}_{\Sigma_{12}} \right| + n \right) \\
 &= \frac{1}{2} \left( n \log 2\pi + \log |\mathbf{A} \Sigma_{z_k} \mathbf{A}^T + \Sigma_x - \mathbf{A} \Sigma_{z_k} \mathbf{A}^T| + n \right) \\
 &= \frac{1}{2} (n \log 2\pi + \log |\Sigma_x| + n)
 \end{aligned}
 \tag{3.45}$$

The cancellation that follows from using a cross-entropy term to drive goal-directed behaviour means that we are left with only the conditional entropy to drive exploration. The above derivation shows that this term is constant and only depends on the observation noise variance  $\Sigma_x$ . This proves that EFE

minimisation in LGDS does not lead to exploration. In fact, minimising a KL between a predicted and desired state (the risk term) is the objective of KL control [54] or message passing based simulations of AIF that minimise VFE [67, 102]. We conclude that in the case of LGDS, the EFE objective is equivalent to the objective of KL control plus an additive constant that depends only on the observation noise variance.

## 3.6 Experiments

We investigate the proposed agents in three different settings. First, we investigate pure epistemics in the additive case and show that they do not manifest. Second, we investigate pure epistemics in the multiplicative case and confirm that the agent does indeed perform maximum entropy exploration. Finally, we provide comparable experiments for full EFE and show that it indeed reduces to a KL divergence plus a constant.

### 3.6.1 Pure Epistemics for Additive Controls

In this section, we investigate how the epistemic component of EFE behaves in the additive case. In particular, we investigate the effects of different transitions on the epistemic value assigned to a policy. For this experiment the transition model is given by Eq. (3.4). We define the state prior as

$$p(\mathbf{z}_{t-1} | \mathbf{z}_{1:t-1}) = \mathcal{N}\left(\mathbf{z}_{t-1} \left| \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right.\right), \quad (3.46)$$

and set both transition and observation noise to identity matrices. We allow the agent a single action by setting  $T = t + 1$ , which will be the case for all experiments. Further we define the transition matrix  $\mathbf{B}$ , emission matrix  $\mathbf{A}$  and observation noise  $\Sigma_x$  as

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \Sigma_x = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \quad (3.47)$$

Note that in the additive case, neither matrix has to be time-varying and so we remove the subscripts. We will also use the same parameterization of  $\Sigma_x$

for all experiments. We compare 4 different candidate parameterisations of the control signal

$$\Theta_1 = \left\{ \mu_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right\}, \Theta_2 = \left\{ \mu_2 = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right\}$$

$$\Theta_3 = \left\{ \mu_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \right\}, \Theta_4 = \left\{ \mu_4 = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \Sigma_4 = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \right\}. \tag{3.48}$$

3

We choose  $\Theta_1$  to function as a baseline. For comparison  $\Theta_2$  shares the same covariance matrix but offers a higher displacement of the mean.  $\Theta_3$  shares the mean parameter with  $\Theta_1$  but has higher variance. Finally,  $\Theta_4$  increases both the mean and variance over  $\Theta_1$ . According to Eq. (3.38) varying the mean should not affect the epistemic value since it does not enter into the MI computation. On the other hand, we expect higher variance to affect the policy independent term and lead to increased epistemic value. Consequently, we hypothesise that  $\Theta_1$  and  $\Theta_2$  will lead to identical results in terms of epistemics even though they result in very different posterior states. Following the same line of reasoning, we hypothesise that  $\Theta_3$  and  $\Theta_4$  will lead to identical results. This in turn implies that  $\Theta_1$  and  $\Theta_3$  will lead to different values even though the displacement is the same and that a similar pattern will hold for  $\Theta_2$  and  $\Theta_4$ . Results are shown in Fig. 3.3, rounded to 3 digits.

We observe that as hypothesised, MI is not affected by the state transition ( $\Theta_1$  and  $\Theta_2$  show identical values). We do find an effect of changing the variance which is again independent of the mean ( $\Theta_3$  and  $\Theta_4$  show identical values). This simple experiment confirms our hypotheses given by Eq. (3.38b): Changing the mean of the control signal does not affect the epistemic term. Changing the variance of the control signal does affect the epistemic term. We conclude that when considering purely epistemic value and additive controls, state transitions and exploration are decoupled. Any effect of the control signal on epistemics is only proportional to the variance of the control, can be pre-computed and does not depend on the agent’s trajectory.

### 3.6.2 Pure Epistemics for Multiplicative Controls

For comparison, we now perform an analogous experiment for the case of multiplicative controls. We define all quantities in the same way as the additive

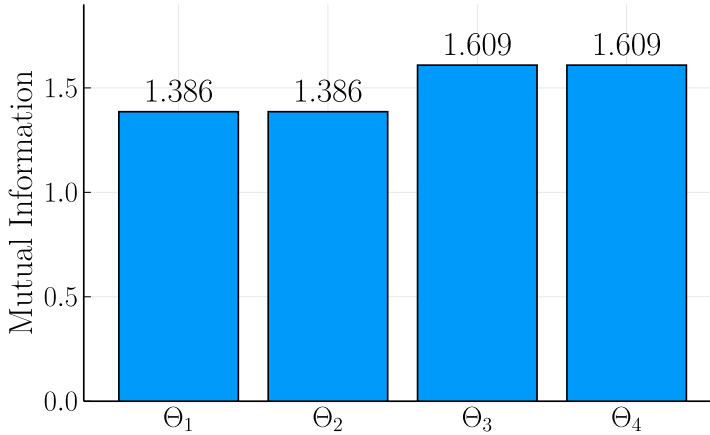


Figure 3.3: Epistemic value for additive control signals given state transitions.

case. The only change we introduce is defining four transition matrices  $\mathbf{B}_{1:4}$  to replace  $\Theta_{1:4}$ . The four candidate transitions we consider are

$$\mathbf{B}_1 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}, \mathbf{B}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{B}_3 = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}, \mathbf{B}_4 = \begin{bmatrix} 100 & 0 \\ 0 & 100 \end{bmatrix}. \quad (3.49)$$

Following Eq. (3.40), we hypothesise that larger transitions should lead to higher MI by virtue of increasing the value of the policy dependent term. We test this hypothesis across four orders of magnitude and show the results in Fig. 3.4.

We observe that, as hypothesised MI increases as a function of the size of the state transition. Larger transitions lead to higher MI although the exact relationship is nonlinear in the size of the transition.

### 3.6.3 Lack of Epistemics for Expected Free Energy

To investigate the behaviour of AIF agents optimising the full EFE construct, we now repeat both the additive and multiplicative experiments but introduce a goal prior  $p(\mathbf{x}_t)$ . We define the state prior and the goal as



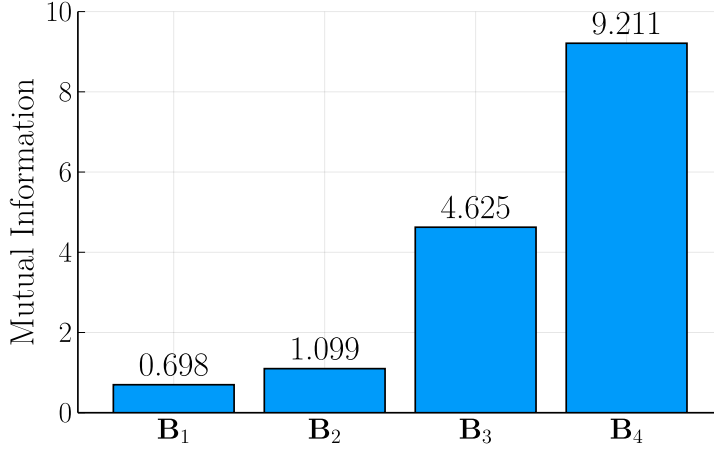


Figure 3.4: Epistemic value for multiplicative control signals given state transitions

$$p(\mathbf{z}_{t-1} | \mathbf{x}_{1:t-1}) = \mathcal{N}\left(\mathbf{z}_{t-1} \left| \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right.\right), \quad p(\mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_t \left| \begin{bmatrix} 3 \\ 3 \end{bmatrix}, \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix} \right.\right). \tag{3.50}$$

Both the multiplicative and the additive agent employ the same emission matrix  $\mathbf{A}$ . For the multiplicative agent, we further define the set of candidate transition matrices  $\mathbf{B}_{1:4}$

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{B}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{B}_2 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, \quad \mathbf{B}_3 = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}, \quad \mathbf{B}_4 = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix}. \tag{3.51}$$

Here we choose  $\mathbf{B}_1$  as the identity matrix to serve as a baseline.  $\mathbf{B}_2$  moves the agent towards the goal but stops short while  $\mathbf{B}_4$  overshoots by the same amount. This means that either transition puts the agent at the same distance from the goal but with different variances and hence different values of the policy dependent term. Finally, we allow  $\mathbf{B}_3$  to move the agent directly to the goal. For the additive case we set the transition matrix  $\mathbf{B} = \mathbf{B}_1$  and consider the set of candidate parameterisations  $\Theta_{1:4}$

$$\Theta_1 = \left\{ \mu_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right\}, \Theta_2 = \left\{ \mu_2 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right\}$$

$$\Theta_3 = \left\{ \mu_3 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \right\}, \Theta_4 = \left\{ \mu_4 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \Sigma_4 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \right\}. \quad (3.52)$$

where we again vary the mean and variance parameters following a similar logic as for the additive experiments. Notably, both  $\Theta_2$  and  $\Theta_4$  take the agent directly to the goal but with different variances. We first examine results in the multiplicative case, shown in Table 3.1.

Table 3.1: EFE for multiplicative controls

Transition	KL	Ambiguity	G	Instrumental	Epistemic
$\mathbf{B}_1$	1.33	2.84	4.17	5.27	-1.10
$\mathbf{B}_2$	0.64	2.84	3.48	5.27	-1.79
$\mathbf{B}_3$	1.37	2.84	4.21	6.60	-2.40
$\mathbf{B}_4$	3.54	2.84	6.38	9.27	-2.89

Here the first column shows the KL between the posterior predictive distribution over observations  $q(\mathbf{x}_t | \mathbf{u}_t)$  and the goal prior  $p(\mathbf{x}_t)$  after the corresponding transition. The second column show the additive constant that corresponds to the ambiguity term. The full EFE is displayed in the third column marked G. Finally, the last two columns display the cross-entropy and negative MI terms as Instrumental and Epistemic value, respectively. From Table 3.1 we see that the lowest KL, and consequently lowest G, is obtained when selecting the  $\mathbf{B}_2$  transition matrix. Recall that  $\mathbf{B}_2$  stopped short of the goal while  $\mathbf{B}_3$  placed the agent directly on top of it. However, because controls are multiplicative,  $\mathbf{B}_3$  also results in substantially larger variance which is penalised in the KL. To show that KL is indeed the only driving factor, we can examine the second column, containing the Ambiguity term. We see that it is constant since the observation noise is constant. In turn, we find that the EFE (third column, G) can be written as the sum of the KL and Ambiguity columns. For completeness, we have also calculated the cross-entropy (Instrumental value) and MI (Epistemic value). Here we observe similar patterns as in the purely exploratory case; larger transitions lead to lower negative MI. This is accurately balanced by the

instrumental term though, highlighting an important point: Our result that EFE does not lead to epistemics is only revealed when we consider a particular way of writing EFE. If we had instead proceeded from the cross-entropy/MI decomposition, the ambiguity constant would not have materialised. We can create a similar table for the additive case, shown in Table 3.2.

Table 3.2: EFE for additive controls

Transition	KL	Ambiguity	G	Instrumental	Epistemic
$\Theta_1$	3.05	2.84	5.88	7.27	-1.39
$\Theta_2$	0.38	2.84	3.22	4.60	-1.39
$\Theta_3$	3.16	2.84	5.99	7.60	-1.61
$\Theta_4$	0.49	2.84	3.33	4.94	-1.61

We observe that the lowest KL and G corresponds to the transition parameterised by  $\Theta_2$  as it takes the agent directly to the goal with small variance. What is interesting about Table 3.2 is the ambiguity column. We obtain the same additive constant as in the multiplicative case which corroborates our results. Even though the dynamics are different and there are substantial differences in both the instrumental and epistemic value terms, the EFE can still be decomposed as a KL and an additive constant that only depends on the observation noise.

### 3.7 Discussion

Viewing EFE from the point of view of mutual information and cross entropy allows for isolating the epistemic and instrumental value terms so they can be investigated separately. This angle was originally taken in [33] and used as a method of relating AIF to other frameworks. Recent work [97, 98] investigates a similar decomposition in the discrete case to highlight how pure exploration and exploitation manifest. Our results as well as [33, 98] all explore how the EFE operates in specific model architectures. Additionally [33, 34, 98] also note the equivalence between the mutual information term and the objective of optimal Bayesian design. While work such as [43, 73, 74] have investigated this link in the general case, deriving the specific equations for a wider class of model architectures promises to be a fruitful area for further research. In those cases, the approach followed in our analysis presents a straightforward way to derive the form of the EFE objective by first decomposing it into a pair of known objective functions and then deriving the expressions separately.

Because EFE can be written in terms of marginal/conditional distributions over the latent states  $z$ , the analysis presented here applies to any model that utilises a linear likelihood. The results do not depend on the transition model, as demonstrated by our experiments showing similar behaviour for EFE minimisation using two different transition models. Our results are consequently equally applicable for a large class of transition models such as auto-regressive models, Gaussian process state space models or deep neural networks without additional adaptation provided the observation model remains linear and Gaussian.

On a similar note, a clear limitation of the present work is the strong reliance on linear observation models. We chose to focus on this case since it allows for an analytical expression of the MI term. However, in general, MI is a difficult quantity to compute and one often has to rely on approximations. When approximations are involved, the present analysis is not necessarily applicable, since the decoupling of control signals and epistemics is only demonstrated for the linear case.

In special cases, one can also approximate the joint covariance matrix instead of the mutual information - this is the case for extended Kalman filters for example. In these cases, the present analysis can still apply. Investigating different methods for handling non-linearities is an interesting area for future work on AIF in Gaussian state space models (both linear and non-linear), that can prove useful for neuroscientists and engineers alike.

## 3.8 Conclusions

In this chapter, we have shown how to apply AIF in linear Gaussian state space models. We have derived the expressions for EFE in the linear Gaussian case and investigated how the epistemic value terms function. In particular, we have shown that in the case of LGDS, EFE reduces to a KL divergence and an additive constant that only depends on observation noise. We therefore conclude that, in the linear Gaussian case, EFE minimisation does not lead to epistemic behaviour.

Additionally, we have provided an analysis of the epistemic value term considered in isolation, since the cancellation that leads to an absence of epistemic drive for the full EFE is not present when the instrumental term is not included. Our analysis showed that using additive control signals renders the epistemic value term independent of state transitions. This in turn means that any contribution to the epistemic value term is only dependent on the variance associated with the control signal. In other words, it is independent of any observations the

agent might receive and any states it may visit, as was previously demonstrated by [105].

Finally, we have shown that utilising multiplicative controls, i.e. selecting from a set of candidate transition matrices, circumvents this problem in the purely epistemic case and provides a meaningful interpretation of controls as inducing epistemic behaviour. The resulting setup is reminiscent of the classical POMDP that is commonly seen in AIF. Future work can investigate this link by applying recent advances for the discrete case such as [35] to continuous state spaces with multiplicative control signals.

# Chapter 4

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

*"To err is human, to really foul things up requires a computer."*

*– Paul Ehrlich, 1986*

In this chapter, we present AIDA, an Active Inference (AIF)-based agent that iteratively designs a personalised 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 realised as an AIF-based agent with an expected free energy (EFE) 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 auto-regressive (AR) filters and a user response model based on a Gaussian process classifier (GPC). The full AIDA system has been implemented using Forney-style factor graphs (FFGs) for the generative model and all tasks (parameter learning, acoustic context classification, trial design, etc.) are realised by variational message passing (VMP) on the factor graph. All verification and validation experiments and demonstrations are freely accessible at our GitHub repository.

The following chapter is based on the original work referenced below. Contributions are split evenly among the first three authors with the original idea, simulations, and text being created in close collaboration.

I focused primarily on designing the preference learning agent and extending AIF to work with GPCs. This is another example of extending AIF to a new class of models in order to solve a practical problem in industry.

**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

## 4.1 Introduction

HAs are often equipped with specialised 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 that are 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 [55]. 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 [2, 65, 78] and satisfies their personal preferences for parameter settings [91]. Considering that HAs nowadays often consist of multiple interconnected digital signal processing units with many integrated parameters, the task of personalising the algorithm

requires exploring a high-dimensional search space of parameters, which often do not permit a clear physical interpretation. The current most widespread approach to personalisation 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 to HA 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' lived experiences.

The question then becomes, how do we move 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 asked 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) [30]. 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 artificial agents as well [64, 67, 72, 109]. A hallmark feature of FEP-based agents is that they exhibit a dynamic trade-off between exploration and exploitation [22, 33, 35], which is a highly desirable property when learning to navigate a 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 domain-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 minimisation.

In this paper we present AIDA<sup>1</sup>, 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 4.2 we make the following contributions:

1. We develop a modular probabilistic model that embodies situated, (acoustic) scene-dependent, and personalised design of its corresponding HA algorithm in Section 4.3.1.
2. We develop an EFE-based agent (AIDA) in Section 4.3.2, whose proposals for tuning parameter settings are well-balanced in terms of seeking information about the user's preferences (explorative agent behaviour) versus seeking to optimise the user's satisfaction levels by taking advantage of previously learned preferences (exploitative agent behaviour).
3. Inference in the acoustic model and AIDA is elaborated upon in Section 4.4 and their operations are individually verified through representative experiments in Section 4.5. Furthermore, all elements are jointly validated through a demonstrator application in Section 4.5.4.

We have intentionally postponed a more thorough review of related work to Section 4.6 as we deem it more relevant after the introduction of our solution approach. Finally, Section 4.7 discusses the novelty and limitations of our approach and Section 4.8 concludes this paper.

---

<sup>1</sup>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".

## 4.2 Problem Statement and Proposed Solution Approach

### 4.2.1 Automated Hearing Aid Tuning by Optimisation

In this chapter, we consider the problem of choosing values for the tuning parameters  $\mathbf{u}$  of a HA algorithm that processes an acoustic input signal  $x$  to output signal  $y$ . In Fig. 4.1, we sketch an automated optimisation-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  $y = f(x, \mathbf{u})$ , as a function of the HA input  $x$  and parameters  $\mathbf{u}$ , by a rating  $r(x, \mathbf{u}) \triangleq r(y)$ . If we run this system on a representative set of input signals  $x \in \mathcal{X}$ , then the tuning problem reduces to the optimisation task

$$\mathbf{u}^* = \arg \max_{\mathbf{u}} \sum_{x \in \mathcal{X}} r(x, \mathbf{u}). \quad (4.1)$$

Unfortunately, in commercial practice, this optimisation 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 [7, 17, 46, 56, 95, 106]. Secondly, the optimisation 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 *personalised, context-sensitive* controller for tuning HA parameters  $\mathbf{u}$ .

### 4.2.2 Situated Hearing Aid Tuning with the User in-the-loop

In this paper, we develop a personalised, context-aware design agent, based on the architecture shown in Fig. 4.2. In contrast to Fig. 4.1, the outside world (rather than a database) produces an input signal  $x$  under situated conditions that is then processed by a HA algorithm to produce an output signal  $y$ . A particular human HA client listens to the signal  $y$  and is invited to cast binary appraisals  $r \in \{0, 1\}$  at any time about the current performance of the HA

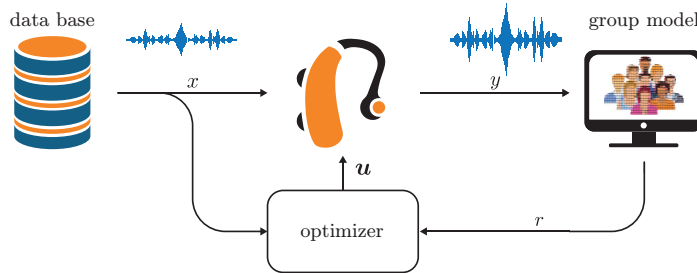


Figure 4.1: A schematic overview of the conventional approach to HA algorithm tuning. Here the parameters of the HA  $u$  are optimised with respect to some generic user rating model  $r(y)$  for a large database  $\mathcal{X}$  of input data  $x$ .

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 HA 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  $x$  and yields the output  $y$  and context  $o$ . Based on the inferred context  $o$  and previous user appraisals  $r$ , AIDA will actively propose new trial parameters  $u$  with the goal of making the user happy, operationalised as providing positive feedback. Technically, the objective of AIDA is to expect fewer negative appraisals in the future, relative to not making parameter adaptations, see Section 4.3.2 for details.

Designing AIDA itself is not a trivial task. For instance, since there is a priori no personalised model of HA ratings for a specific user, AIDA will have to build such a model on-the-fly from context  $o$  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 occasionally cast an appraisal which complicates the learning of a personalised HA rating model.

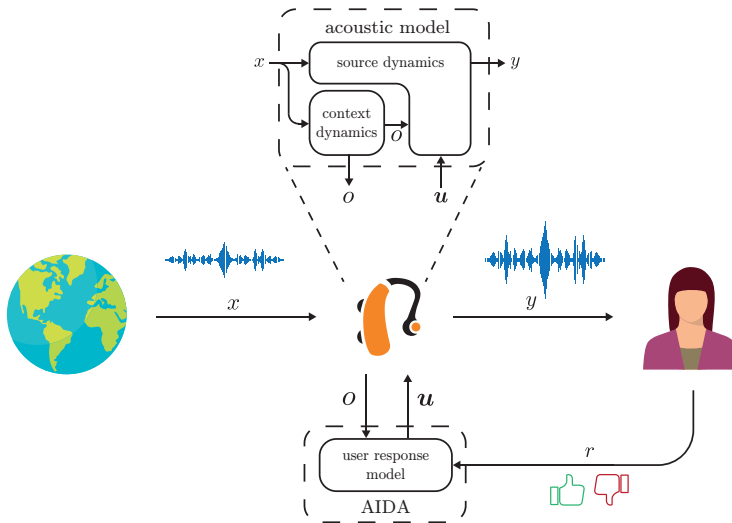


Figure 4.2: A schematic overview of the proposed situated HA design loop containing AIDA. An incoming signal  $x$  enters the HA and is used to infer the present context  $o$ . Based on the context and previous user appraisals, AIDA proposes a new set of parameters  $u$  for the HA algorithm. Based on the input signal, the proposed parameters, and the current context, the output  $y$  of the HA is determined. The parameters  $u$  are actively optimised by AIDA, based on the inferred context  $o$  and appraisals  $r$ . All individual subsystems represent parts of a probabilistic generative model as described in Section 4.3, where the corresponding algorithms follow from performing probabilistic inference in these models as described in Section 4.4.

As an example of the kinds of lightweight interactions we aim for, we now sketch how we envision a typical interaction between AIDA and a HA client. Assume that the HA client is engaged in conversation with a friend at a restaurant. In this case, the signal of interest is the friend’s speech signal while the background environment produces a troublesome babble noise signal. The HA algorithm first separates the input signal  $x$  into its constituent speech and noise source components, then applies gains  $u$  to each source component and finally sums the weighted source components to produce the output signal  $y$ . If the HA client is satisfied with the performance of her HA, she will not cast any appraisals - she is in the middle of a conversation and has no imperative to change the behaviour of her HA. However, if she cannot properly hear 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 for example be implemented as a smartwatch application, will reply by sending a tuning parameter update  $u$  to the HA 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  $x$ . As an example, the HA user might leave the restaurant for a walk outside, presenting a different type of background noise and consequently requiring different parameter settings.

Crucially, we would like HA clients to be able to tune their HA aids without interrupting any ongoing activities. Therefore, we will not demand that the client focus visual attention on interacting with AIDA, for instance through a smartphone app. At most, we want the client to apply a tap or make a simple gesture that does not force attention away from ongoing activities.

A second criterion is that we do not want a potential conversation partner to notice that the client is interacting with the agent. The client may for example be in a situation (for instance a business meeting) where it is not appropriate to demonstrate that her priorities have shifted to tuning her HAs. In other words, the interaction must be very lightweight and covert.

A third criterion is that we want the agent to learn from as few appraisals as possible. If a HA has 10 tuning parameters each taking one of 5 possible values (very low, low, middle, high, very high), then there are  $5^{10}$  (about 10 million) possible 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, discomfort, and distraction. 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 behaviour. In Section 4.4.2, we will quantify exactly 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 personalised audio processing algorithm through a very lightweight interaction protocol.

To accomplish this task, we will draw inspiration from the way human brains 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 AIF framework. Originating from the field of computational neuroscience, AIF proposes a view of 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 minimising free energy then constitutes a unified procedure for both data processing and learning. To select actions, in our case choosing tuning parameter trials for a client, an AIF agent predicts the *expected* free energy of the near future, given a particular choice of parameter settings. In this way, 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 4.3.1), we infer the most probable acoustic environment as described in Section 4.4.1.

Task 2 (trial design) encompasses proposing alternative settings for the HA tuning parameters. Sections 4.3.2 and 4.4.2 describe the user response model and execution of AIDA's trial selection procedure based on expected free energy minimisation, respectively.

Finally, task 3 (HA algorithm execution) concerns performing variational free energy minimisation with respect to the state variables in a generative probabilistic model for the acoustic signal. In Section 4.3.1 we describe the acoustic generative model underlying the HA algorithm and Section 4.4.3 describes the corresponding inference procedure.

Crucially, in the AIF framework, all three tasks can be accomplished by variational free energy (VFE) minimisation in a generative probabilistic model for observations. Since we can automate VFE minimisation by probabilistic programming, the only remaining task for the human designer is to specify the requisite generative models. The next section describes the model specification used for the various components of AIDA.

## 4.3 Model Specification

In this section, we present the generative model of the AIDA-controlled HA system, as illustrated in Fig. 4.2. In Section 4.3.1, we describe a generative

model for the HA input and output signals  $x$  and  $y$  respectively. In this model, the actual HA algorithm follows from performing probabilistic inference, as will be discussed in Section 4.4. In Section 4.3.2 we introduce a model for the agent AIDA that is used to infer new parameter trials. A concise summary of the generative model is also presented in Appendix B.1.

Throughout this section, we will make use of factor graphs for visualisation of probabilistic models. We focus on FFGs, as introduced in [25] with notational conventions adopted from [69].

FFGs represent factorised functions by undirected graphs, whose nodes represent individual factors of the global function. Nodes are connected by edges representing mutual arguments of the connected factors. In an FFG, a node can be connected to an arbitrary number of edges, but edges are constrained to have a maximum degree of two.

### 4.3.1 Acoustic Model

Our acoustic model of the observed signal and HA 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 the observed acoustic signal  $x$  consists of a speech signal (or more generally, a target signal that the HA client wishes to focus on) and an additive noise signal (that the HA client is not interested in), as

$$x_t = z_t + n_t \quad (4.2)$$

where  $x_t \in \mathbb{R}$  represents the observed signal at time  $t$ , i.e. the input to the HA. Speech and noise signals are denoted by  $z_t \in \mathbb{R}$  and  $n_t \in \mathbb{R}$ , respectively. Now we need to specify the source dynamics of  $z_t$  and  $n_t$ . We model the speech signal by a time-varying auto-regressive (TVAR) model and the noise signal by a context-dependent AR model. The remainder of this subsection will elaborate on both of these source models and how the HA output is generated. The FFG of the acoustic model is depicted in Fig. 4.3.

Historically, AR models have been widely used to represent speech signals [53, 80]. As the dynamics of the vocal tract exhibit non-stationary behaviour,

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 phonetic structure. Therefore the quasi-stationarity assumption is likely to be violated and time-varying dynamics occur within the segmented frames [111]. To address this issue, we employ a time-varying prior for the coefficients of the AR model, leading to a TVAR [96]

$$p(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}) = \mathcal{N}(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}, \omega \mathbf{I}) \quad (4.3a)$$

$$p(\mathbf{z}_t | \mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t | \mathbf{A}(\boldsymbol{\theta}_t) \mathbf{z}_{t-1}, \mathbf{V}(\gamma)) \quad (4.3b)$$

where  $\boldsymbol{\theta}_t = [\theta_{1t}, \theta_{2t}, \dots, \theta_{Kt}]^T \in \mathbb{R}^K$ ,  $\mathbf{z}_t = [z_t, z_{t-1}, \dots, z_{t-K+1}]^T \in \mathbb{R}^K$  are the coefficients and states of an  $K$ -th order TVAR model for the speech signal  $z_t = \mathbf{e}_1^T \mathbf{z}_t$ . We use  $\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}, \boldsymbol{\Sigma})$  to denote a Gaussian distribution over  $\mathbf{x}$  with mean  $\boldsymbol{\mu}$  and covariance matrix  $\boldsymbol{\Sigma}$ . In this model, the AR coefficients  $\boldsymbol{\theta}_t$  are represented by a Gaussian random walk with process noise covariance  $\omega \mathbf{I}$ , with  $\mathbf{I}_k$  denoting the  $K \times K$  identity matrix, scaled by  $\omega \in \mathbb{R}_{>0}$ .  $\gamma \in \mathbb{R}_{>0}$  represents the process noise precision of the AR process. Here, we have adopted the state-space formulation of TVAR models as in [86], where  $\mathbf{V}(\gamma) = (1/\gamma) \mathbf{e}_1 \mathbf{e}_1^T$  creates a covariance matrix with a single non-zero entry. We use  $\mathbf{e}_i$  to denote an appropriately sized Cartesian standard unit vector, i.e. a column vector of zeros where only the  $i$ 'th entry is 1.  $\mathbf{A}(\boldsymbol{\theta})$  denotes the  $M \times M$  companion matrix, defined as

$$\mathbf{A}(\boldsymbol{\theta}) = \begin{bmatrix} & \boldsymbol{\theta}^T & \\ \mathbf{I} & & \\ & & \mathbf{0} \end{bmatrix}. \quad (4.4)$$

Multiplication of a state vector by the companion matrix,  $\mathbf{A}(\boldsymbol{\theta}_t) \mathbf{z}_{t-1}$ , is equivalent to performing two operations: an inner product  $\boldsymbol{\theta}_t^T \mathbf{z}_{t-1}$  and a shift of  $\mathbf{z}_{t-1}$  by one time step to the past.

The acoustic model also encompasses a model for background noise, such as the sounds of a bar or train station. Many of these background sounds can be represented by coloured noise [88], which in turn can be modeled by a low-order AR model [40, 42]

$$p(\mathbf{n}_t | \phi_k, \mathbf{n}_{t-1}, \tau_k) = \mathcal{N}(\mathbf{n}_t | \mathbf{A}(\phi_k) \mathbf{n}_{t-1}, \mathbf{V}(\tau_k)), \quad \text{for } t = t^-, t^- + 1, \dots, t^+ \quad (4.5)$$



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

$$k = \left\lceil \frac{t}{W} \right\rceil. \quad (4.6)$$

Here  $\lceil \cdot \rceil$  denotes the ceiling function that returns the largest integer smaller or equal to its argument, while  $W$  is the window length. Intuitively, the above equation makes sure that  $k$  is 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  $k$ , we define  $t^- = (k-1)W + 1$  and  $t^+ = kW$  as implicit functions of  $k$ .

However, abrupt changes in the dynamics of background noise may still occasionally occur. For example, if the user moves from a quiet street to the inside of a bar, the parameters of the AR model that are attributed to the street inadequately capture 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:

$$p(\phi_k | \mathbf{o}_k) = \prod_{l=1}^L \mathcal{N}(\phi_k | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)^{c_{lk}} \quad (4.7a)$$

$$p(\tau_k | \mathbf{o}_k, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{l=1}^L \Gamma(\tau_k | \alpha_l, \beta_l)^{c_{lk}} \quad (4.7b)$$

The context at time index  $k$ , denoted by  $\mathbf{o}_k$ , comprises a 1-of- $L$  binary vector with elements  $c_{lk} \in \{0, 1\}$ , which are constrained by  $\sum_l c_{lk} = 1$ .  $\Gamma(\tau | \alpha, \beta)$  represents a Gamma distribution over  $\tau$  with shape and rate parameters  $\alpha$  and  $\beta$ . The hyperparameters  $\boldsymbol{\mu}_l$ ,  $\boldsymbol{\Sigma}_l$ ,  $\alpha_l$  and  $\beta_l$  define characteristics of the different background noise environments.

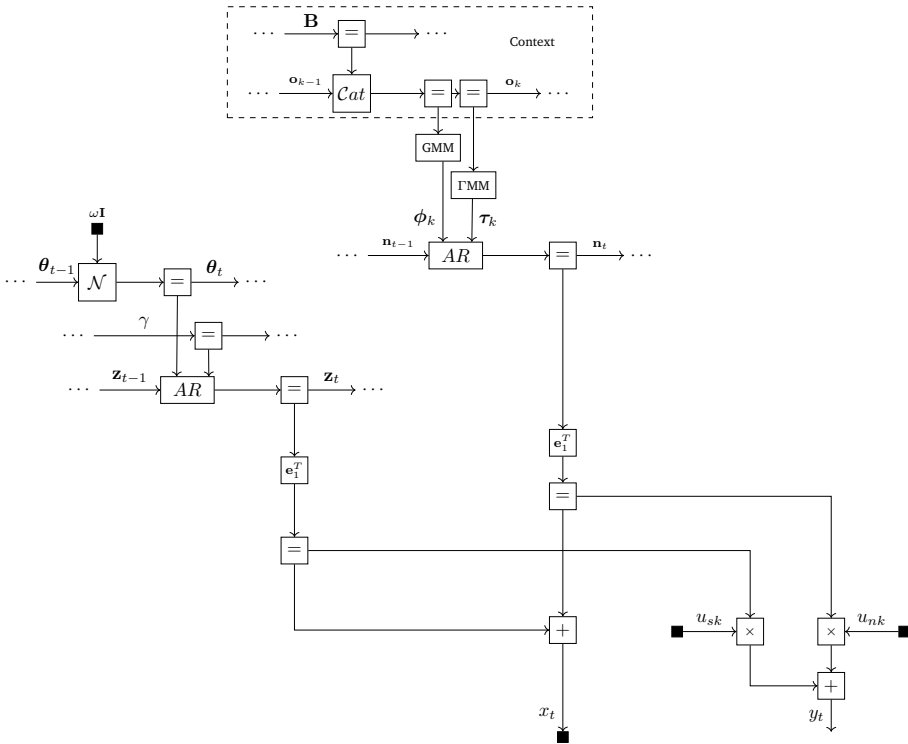


Figure 4.3: An FFG representation of the acoustic source signals model as specified by (4.3)-(4.11) at time index  $t$ . The observation  $x_t$  is given as the sum of a latent speech signal  $z_t$  and a latent noise signal  $n_t$ . The speech signal is modeled by a TVAR process, where the coefficients  $\theta_t$  are modeled by a Gaussian random walk. The noise signal is a context-dependent AR process, modeled by Gaussian mixture models (GMMs) and Gamma mixture models ( $\Gamma$ MMs) for the parameters  $\phi_k$  and  $\tau_k$ , respectively. The selection variable of both mixture models represents the context  $o_k$ . The model for the context dynamics is enclosed by the dashed box. The composite AR factor node represents the auto-regressive transition dynamics specified by (4.3b). The output of the HA  $y_t$  is modeled as the weighted sum of the extracted speech and noise signals.

Having specified the acoustic model of the environment, we now extend it in order to obtain a HA algorithm. The principal goal of a HA algorithm is to improve 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 incoming signal. Assuming that we can infer the constituent source signals  $\mathbf{z}_t$  and  $\mathbf{n}_t$  from received signal  $x_t$ , the desired HA output signal can be modeled by

$$y_t = u_{zk} \mathbf{e}_1^T \mathbf{z}_t + u_{nk} \mathbf{e}_1^T \mathbf{n}_t \quad \text{for } t = t^-, t^- + 1, \dots, t^+ \quad (4.8)$$

where  $\mathbf{u}_k = [u_{zk}, u_{nk}]^T \in [0, 1]^2$  represents a vector of 2 tuning parameters or source-specific gains for the speech and background noise signal, respectively. In Eq. 4.8, the output of the HA is modeled by a weighted sum of the constituent source signals.

The gains  $\mathbf{u}_k$  control the amplification of the extracted speech and noise signals individually, allowing the user to perform source-specific filtering or soundscaping [24]. Because of imperfections during inference of the source signals (see Section 4.4), the gains simultaneously reflect a trade-off between residual noise and speech distortion.

Finding good values for the gains  $\mathbf{u}_k$  can be a difficult task because the ideal parameter settings depend both on the individual listener and 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 driving in a car, doing groceries, watching TV at home, etc.) the preferred parameter settings for HA algorithms vary. The context signal allows the HA to distinguish between different acoustic environments.

The hidden context state variable  $\mathbf{o}_k$  at time index  $k$  is a 1-of- $L$  encoded vector with elements  $o_{lk} \in \{0, 1\}$ , constrained by  $\sum_l o_{lk} = 1$ . The context is responsible for the operations of the noise model in (4.7). Context transitions are supported by a dynamic model

$$p(\mathbf{o}_k \mid \mathbf{B}, \mathbf{o}_{k-1}) = \text{Cat}(\mathbf{o}_k \mid \mathbf{B}\mathbf{o}_{k-1}), \quad (4.9)$$

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

$$p(\mathbf{B}_j | \boldsymbol{\alpha}_j) = \text{Dir}(\mathbf{B}_j | \boldsymbol{\alpha}_j), \quad (4.10)$$

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

$$p(\mathbf{o}_0 | \boldsymbol{\pi}) = \text{Cat}(\mathbf{o}_0 | \boldsymbol{\pi}) = \prod_{l=1}^L \pi_l^{c_{l0}} \quad \text{such that} \quad \sum_{l=1}^L \pi_l = 1, \quad (4.11)$$

where the vector  $\boldsymbol{\pi} = [\pi_1, \pi_2, \dots, \pi_L]^T$  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 Fig. 4.3.

### 4.3.2 AIDA's User Response Model

The goal of AIDA is to continually provide the most “interesting” settings for the HA tuning parameters  $\mathbf{u}_k$ , where interesting is operationalised as minimising EFE. But how does AIDA know what the client wants? In order to learn the client's preferences, she is invited to cast an appraisal  $r_k \in \{\emptyset, 0, 1\}$  of current HA performance at any time. To keep the user interface very light, we assume that appraisals are binary, encoded by  $r_k = 0$  for disapproval and  $r_k = 1$  indicating satisfaction. If a user does not cast an appraisal, we will record a missing value, i.e.,  $r_k = \emptyset$ . The subscript  $k$  for  $r_k$  indicates that we record appraisals at the same rate as the context dynamics.

If a client submits a negative appraisal  $r_k = 0$ , AIDA interprets this as an expression that the client is not happy with the current HA settings  $\mathbf{u}_k$  in the current acoustic context  $\mathbf{o}_k$  (and vice versa for positive appraisals). To *learn* client preferences from these appraisals, AIDA holds a context-dependent generative model for *predicting* user appraisals and updates this model after observing actual appraisals. In this chapter, we opt for a 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 [18, 47, 50]. For an in-depth discussion on GPs, we refer the reader to [90]. The context-dependent user response model is defined as

$$p(v_k(\cdot) \mid \mathbf{o}_k) = \prod_{l=1}^L \text{GP}(m_l(\cdot), \mathbf{K}_l(\cdot, \cdot))^{o_{lk}} \quad (4.12a)$$

$$p(r_k \mid v_k, \mathbf{u}_k) = \text{Ber}(\Phi(v_k(\mathbf{u}_k))), \quad \text{with } r_k \in \{0, 1\} \quad (4.12b)$$

In (4.12a),  $v_k(\cdot)$  is a latent function drawn from a mixture of GPs with mean functions  $m_l(\cdot)$  and kernels  $\mathbf{K}_l(\cdot, \cdot)$ . Evaluating  $v_k(\cdot)$  at the point  $\mathbf{u}_k$  provides an estimate of user preferences. Without loss of generality, we can set  $m_l(\cdot) = 0$ . Since  $\mathbf{o}_k$  is one-hot encoded, raising to the power  $o_{lk}$  serves to select the GP that corresponds to the active context.  $\Phi(\cdot)$  denotes the Gaussian cumulative distribution function, defined as  $\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x \exp(-t^2/2) dt$ . This map in (4.12b) casts  $v_k(\mathbf{u}_k)$  to a Bernoulli-distributed variable  $r_k$ .

## 4.4 Solving Tasks by Probabilistic Inference

This section elaborates on our solution for the three tasks of Section 4.2.2:

1. Context classification.
2. Trial design.
3. HA algorithm execution.

All tasks can be solved through probabilistic inference in the generative model specified by Eq. (4.2)-(4.12b) in Section 4.3. In this section, the inference goals are formally specified based on the previously proposed generative model.

To perform inference we employ VMP in a factor graph representation of the generative model. Message passing (MP)-based inference is highly efficient, modular, and scales well to large inference tasks [19, 70]. With MP, inference tasks in the generative model reduce to automatable procedures involving only local computations on the corresponding FFG.

### 4.4.1 Inference for Context Classification

The acoustic context  $\mathbf{o}_k$  describes the dynamics of the background noise model through Eq. (4.5) and Eq. (4.7). To determine the current acoustic environment

of the user, we need to infer the current context based on preceding observations. Technically, we are interested in determining the marginal distribution  $p(\mathbf{o}_k \mid \mathbf{x}_{1:t+})$ , where the index range over  $t$  of  $\mathbf{x}$  takes into account the relation between  $t$  and  $k$  as defined in Eq. (4.6). In the online setting, we wish to calculate this marginal distribution by iteratively solving

$$\begin{aligned}
 \underbrace{p(\mathbf{o}_k \mid \mathbf{x}_{1:t+})}_{\text{posterior}} \propto & \int \underbrace{p(\mathbf{s}_{t-:t+}, \Psi_k, \mathbf{x}_{t-:t+} \mid \mathbf{s}_{t-1}, \mathbf{o}_k)}_{\text{observation model}} \underbrace{p(\mathbf{o}_k, \mathbf{T} \mid \mathbf{o}_{k-1})}_{\text{context dynamics}} \\
 & \cdot \underbrace{p(\mathbf{o}_{k-1}, \mathbf{s}_{t-1} \mid \mathbf{x}_{1:t-1})}_{\text{prior}} d\mathbf{s}_{t-1:t+} d\Psi_k d\mathbf{o}_{k-1} d\mathbf{T},
 \end{aligned} \tag{4.13}$$

where  $\mathbf{s}_t$  and  $\Psi_k$  denote the sets of dynamic states and static parameters  $\mathbf{s}_t = \{\boldsymbol{\theta}_t, \mathbf{z}_t, \mathbf{n}_t\}$  and  $\Psi_k = \{\gamma, \tau_k, \phi_k\}$ , respectively. The observation model is given by the model specification in Section 4.3, similar for the context dynamics. The prior distribution is a joint result of the iterative execution of both Eq. (4.13) and Eq. (4.18), where the latter refers to the HA algorithm execution from Section 4.4.3. Calculation of this marginal distribution is intractable and therefore exact context inference is not possible. This issue stems from two sources:

- Intractability of the autoregressive model. We further elaborate on this in Section 4.4.3.
- Intractability as a result of performing message passing with mixture models.

In Eq. (4.7), the model contains both a Gaussian and Gamma mixture model for the AR-coefficients and process noise precision, respectively. Exact inference through message passing with these mixture models quickly becomes intractable, especially when multiple background noise models are involved. Therefore, we need to resort to a variational approximation where output messages of the mixture models are constrained to be within the exponential family.

Although variational inference with mixture models is feasible [10, 63, 87], 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 be either Gaussian or Gamma distributions, possibly losing important multi-modal information, and as a result, they can lead to sub-optimal inference of the context variable. Because the context is vital for the

underdetermined source separation stage, we wish to limit the amount of (variational) approximations during context inference. At the cost of increased computational complexity, we can remove the variational approximation around the mixture models and instead expand the mixture components into a series of distinct models. This means each distinct model now contains the mixture component for a given context, resulting in exact messages originating from the priors of  $\phi_k$  and  $\tau_k$ . Therefore we only need to resort to a variational approximation for the AR node. By expanding the mixture models into a series of distinct models to reduce the number of variational approximations, calculation of the posterior distribution of the context  $p(\mathbf{o}_k \mid \mathbf{x}_{1:t+})$  reduces to an approximate Bayesian model comparison problem, similarly to the procedure described in [24].

Appendix B.2.1 gives a more in-depth description of how we use Bayesian model comparison to solve the inference task in Eq. (4.13).

#### 4.4.2 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 minimisation over desired future user responses can be achieved through a procedure called expected free energy (EFE) minimisation [33, 98].

EFE as a trial selection criterion induces a natural trade-off between explorative (information seeking) and exploitative (reward seeking) behaviour. In the context of situated HA personalisation, this is desirable because soliciting user feedback can be burdensome and invasive, as described in Section 4.2.2. 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 [33, 83]. The EFE is defined as [33]

$$G[q; \mathbf{u}] = \mathbb{E}_{q(r,v|\mathbf{u})} \left[ \log \frac{q(v \mid \mathbf{u})}{p(r, v \mid \mathbf{u})} \right], \quad (4.14)$$

Note that Eq. (4.14) is a functional of  $q$  but a function of trial parameters  $\mathbf{u}$ . We indicate this difference by separating them with a semicolon. The EFE can be decomposed into [33]

$$G[q; \mathbf{u}] \approx \underbrace{-\mathbb{E}_{q(r|\mathbf{u})} \left[ \log p(r) \right]}_{\text{Utility drive}} - \underbrace{\mathbb{E}_{q(r,v|\mathbf{u})} \left[ \ln \frac{q(v | \mathbf{u}, r)}{q(v | \mathbf{u})} \right]}_{\text{Information gain}}, \quad (4.15)$$

which contains an information gain term and a utility-driven term. Minimisation of the EFE reduces to maximization of both these terms. Maximising the utility drive pushes the agent towards matching predicted user responses  $q(r | \mathbf{u})$  to a goal prior over *desired* user responses  $p(r)$  that we wish to observe. Setting the goal prior to match positive user responses then drives the agent towards parameter settings that it believes will make the user happy in the future. The information gain term in Eq. (4.15) on the other hand, drives EFE-minimising agents to seek out responses that are maximally informative about latent states  $v$ , where  $v$  represents the users inferred preferences. To select the next set of gains  $\mathbf{u}$  to propose to the user, we need to find

$$\mathbf{u}^* = \arg \min_{\mathbf{u}} \left( \arg \min_{q \in \mathcal{Q}} G[q; \mathbf{u}] \right). \quad (4.16)$$

where  $\mathcal{Q}$  denotes the family of candidate variational distributions. Intuitively, one can think of Eq. (4.16) as a two-step procedure with an inner and an outer loop. The inner loop finds the optimal approximate posterior  $q$  using (approximate) Bayesian inference, conditioned on a particular action parameter  $\mathbf{u}$ . The outer loop evaluates the resulting EFE as a function of  $\mathbf{u}$  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 GPC indexed by context  $\mathbf{o}_k$ . For simplicity, we will restrict inference to the GP corresponding to the MAP estimate of  $\mathbf{o}_k$ . Between trials, the corresponding GP needs to be updated to adapt to new data gathered from the user. Specifically, we are interested in finding the posterior over the latent user preference function

$$p(v^* | \mathbf{u}_{1:k}, \mathbf{r}_{1:k-1}) = \int p(v^* | \mathbf{u}_{1:k-1}, \mathbf{u}_k, v) p(v | \mathbf{u}_{1:k-1}, \mathbf{r}_{1:k-1}) dv. \quad (4.17)$$

where we assume AIDA has access to a dataset consisting of previous parameter settings  $\mathbf{u}_{1:k-1}$  and appraisals  $\mathbf{r}_{1:k-1}$  and we are querying the model at  $\mathbf{u}_k$ .



While this inference task in the GPC is also intractable, there exist several techniques for approximate inference, such as variational Bayesian methods, expectation propagation (EP), and the Laplace approximation [90]. Appendix B.2.2 describes the exact details of the inference realisation of AIDA.

### 4.4.3 Inference for Executing the Hearing Aid Algorithm

The main goal of the proposed HA algorithm is to improve audibility and intelligibility by re-weighting inferred source signals in the HA output signal. In our model of the observed signal in Eq. (4.2)-(4.7) we are interested in iteratively inferring the marginal distribution over the latent speech and noise signals  $p(\mathbf{z}_t, \mathbf{n}_t \mid \mathbf{x}_{1:t})$ . This inference task is also referred to as informed source separation [58] in prior work. Inferring the latent speech and noise signals tries to optimally disentangle each signal from the observed signal, based on the sub-models of the speech and noise source.

This requires us to compute the posterior marginal distributions associated with both speech and noise signals. To do so, we perform probabilistic inference by MP in the acoustic model of Eq. (4.2)-(4.7). The posterior distributions can be calculated in an online manner using sequential Bayesian updating by solving the Chapman-Kolmogorov equation [99]

$$\underbrace{p(\mathbf{s}_t, \Psi_k \mid \mathbf{x}_{1:t})}_{\text{posterior}} \propto \underbrace{p(x_t \mid \mathbf{s}_t)}_{\text{observation}} \int \underbrace{p(\mathbf{s}_t \mid s_{t-1}, \Psi_k)}_{\text{state dynamics}} \underbrace{p(s_{t-1}, \Psi_k \mid \mathbf{x}_{1:t-1})}_{\text{prior}} ds_{t-1}, \quad \text{for } t = t^-, t^- + 1, \dots, t^+.$$

(4.18)

Recall that  $\mathbf{s}_t = \{\boldsymbol{\theta}_t, \mathbf{z}_t, \mathbf{n}_t\}$  and  $\Psi_k = \{\gamma, \tau_k, \phi_k\}$ . Here, the states and parameters correspond to the latent AR and TVAR models of Eq. (4.3) and Eq. (4.5). We further assume that the context does not change during inference, i.e.  $k$  is fixed. When the context does change Eq. (4.18) will need to be extended by integrating over the time-varying parameters. Unfortunately, the solution of Eq. (4.18) is not analytically tractable. This happens due to

- Integration over large state spaces.
- The non-conjugate prior-posterior pairing.

- The absence of a closed-form solution for the evidence factor [85].

To circumvent these issues, we resort to a hybrid message passing algorithm that combines (structured) VMP and loopy belief propagation (BP) for the minimisation of the BFE [116].

For further details of the (S)VMP and BP algorithms, we also refer the reader to Section 1.4 and [23, 116].

Due to the modularity of FFGs, the necessary MP update rules can be tabulated and hence only need to be derived once for each of the included factor nodes. The derivations of the BP update rules for elementary factor nodes can be found in [70] and the derived structured variational rules for the composite AR node can be found in [85]. The variational updates in the mixture models can be found in [63, 87]. 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  $q(z) \propto \vec{\nu}(z) \cdot \bar{\nu}(z)$ .

Based on the inferred posterior distributions of  $z_t$  and  $n_t$ , these signals can be used for inferring the hearing aid output through Eq. (4.8) to produce a personalised output which compromises between residual noise and speech distortion.

## 4.5 Experimental Verification and Validation

In this section, we verify our approach to the three design tasks of Section 4.2.2. Section 4.5.1 evaluates our approach to context inference by reporting the classification accuracy our algorithm on simulated signals. In Section 4.5.2 we evaluate the performance of our intelligent agent AIDA on the task of actively online HA tuning and preference learning from a simulated user. We examine the HA algorithm in Section 4.5.3 by evaluating performance on a source separation task intended to simulate the parsing of incoming audio data into signal and noise components. To conclude, we present a demonstrator for the entire system in Section 4.5.4.

All algorithms have been implemented in the scientific programming language Julia [9]. Probabilistic inference in our model is automated using the open source Julia package `ReactiveMP2` [3]. All experiments presented in this section can be found in the AIDA GitHub repository available at <https://github.com/biaslab/AIDA>.

<sup>2</sup>`ReactiveMP` [3] is available at <https://github.com/biaslab/ReactiveMP.jl>.

### 4.5.1 Context Classification Verification

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

$$p(\mathbf{o}_k | \mathbf{B}, \mathbf{o}_{k-1}) = \text{Cat}(\mathbf{o}_k | \mathbf{B}\mathbf{o}_{k-1}) \quad (4.19)$$

with priors

$$\begin{aligned} p(\mathbf{o}_0 | \boldsymbol{\pi}) &= \text{Cat}(\mathbf{o}_0 | \boldsymbol{\pi}) \\ p(\mathbf{B}_j | \boldsymbol{\alpha}_j) &= \text{Dir}(\mathbf{B}_j | \boldsymbol{\alpha}_j) \end{aligned} \quad (4.20)$$

where  $\mathbf{o}_o$  is chosen to have length  $L = 4$ . The event probabilities  $\boldsymbol{\pi}$  and concentration parameters  $\boldsymbol{\alpha}_j$  are given by  $\boldsymbol{\pi} = [0.25, 0.25, 0.25, 0.25]^T$  and  $\boldsymbol{\alpha}_j = [1.0, 1.0, 1.0, 1.0]^T$ , respectively. We generated a sequence of 1000 frames, each containing 100 samples, for a total of 100,000 data points. Each frame is associated with one of 4 different contexts. Each context corresponds to an AR model with the parameters presented in Table 4.1.

AR order	$\phi$				$\tau^{-1}$
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

Table 4.1: Parameters of the AR processes used for generating time series data with simulated context dynamics.

To validate our context classification procedure, we need to demonstrate that it can adequately identify which model best approximates the generated dataset. For this experiment, we employed 4 models with the same specifications as were used to generate the dataset. We used informative priors for the coefficients and precision of the AR models. Additionally, we extended our set of candidate models with an AR (5) model with weakly informative priors as well as a Gaussian i.i.d. model that can also be viewed as an AR model of zero'th order, AR (0). Individual frames containing 100 samples each were processed separately and the resulting BFE computed for each model. If we approximate the true model evidence using the BFE as described in Appendix B.2.1, we can

perform approximate Bayesian model selection by selecting the model with the lowest BFE. This model then corresponds to the most likely context for the frame at hand. The results of our experiment are shown in Fig. 4.4.

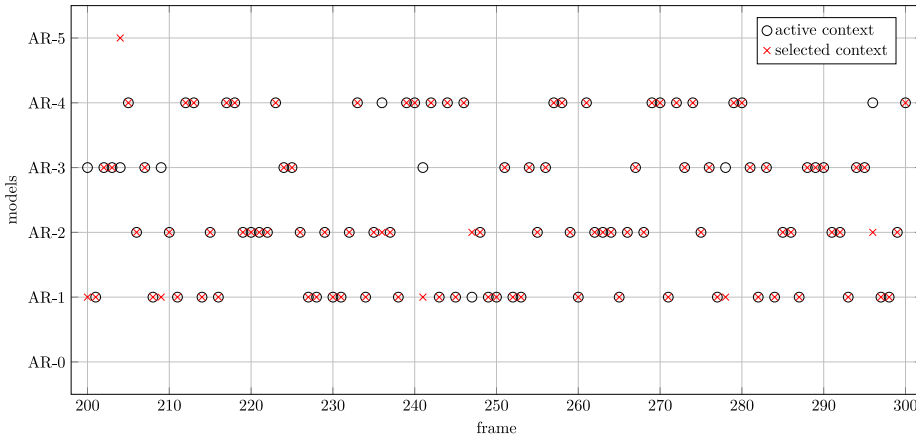


Figure 4.4: True and inferred evolution of contexts from frames 200 to 300. Each frame consists of 100 data points. Circles denote the active contexts that were used to generate the frame. Crosses denote the model that achieves the lowest BFE for a specific frame.

We evaluate the performance of our context classification procedure by computing the categorical accuracy metric defined as

$$acc = \frac{tp + tn}{N} \quad (4.21)$$

where  $tp$ ,  $tn$  are the number of true positive and true negative values, respectively.  $N$  corresponds to the number of total observations  $N = 1000$ . In this context classification experiment, our method achieves a categorical accuracy of  $acc = 0.94$ .

### 4.5.2 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

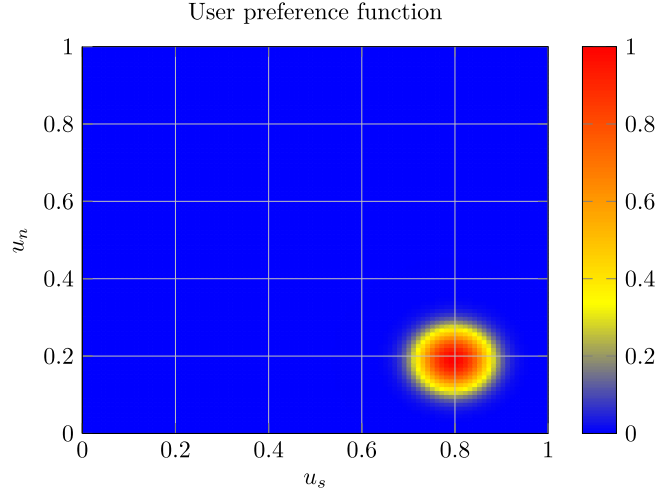


Figure 4.5: Simulated user preference function  $p(r_k = 1 \mid \mathbf{u}^*, \mathbf{u}_k, \Lambda_{\text{user}})$ . The colouring corresponds to the probability of the user giving a positive appraisal for the search space of gains  $\mathbf{u}_k = [u_{sk}, u_{nk}]^T$ .

4

is the optimum in order to obtain more information. As a verification experiment, we can therefore investigate how the agent interacts with a simulated user. Our simulated user samples binary appraisals  $r_k$  based on the HA parameters  $\mathbf{u}_k$  as

$$p(r_k \mid \mathbf{u}^*, \mathbf{u}_k, \Lambda_{\text{user}}) = \text{Ber} \left( r_k \mid \frac{2}{1 + \exp \left( (\mathbf{u}_k - \mathbf{u}^*)^T \Lambda_{\text{user}} (\mathbf{u}_k - \mathbf{u}^*) \right)} \right), \quad (4.22)$$

where  $\mathbf{u}^*$  denotes the optimal parameter setting,  $\mathbf{u}_k$  is the set of parameters proposed by AIDA at time  $k$ ,  $\Lambda_{\text{user}}$  is a diagonal weighting matrix that controls how quickly the probability of positive appraisals decays with the squared distance to  $\mathbf{u}^*$ . The constant 2 ensures that when  $\mathbf{u}_k = \mathbf{u}^*$ , the probability of positive appraisals is 1 instead of 0.5. For our experiments, we set  $\mathbf{u}^* = [0.8, 0.2]^T$  and the diagonal elements of  $\Lambda_{\text{user}}$  to 0.004. This results in the user preference function  $p(r_k = 1 \mid \mathbf{u}^*, \mathbf{u}_k, \Lambda_{\text{user}})$  as shown in Figure 4.5.

The kernel used for AIDA is a squared exponential kernel, given by

$$K(\mathbf{u}, \mathbf{u}') = \sigma^2 \exp \left\{ -\frac{\|\mathbf{u} - \mathbf{u}'\|_2^2}{2l^2} \right\}, \quad (4.23)$$

where  $l$  and  $\sigma$  are hyperparameters. Intuitively,  $\sigma$  is a static noise parameter and  $l$  encodes the smoothness of the kernel function. Both hyperparameters were initialised 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<sup>th</sup> trial using conjugate gradient descent as implemented in `Optim.jl` [52]. We constrain both hyperparameters to the domain  $[0.1, 1]$  to ensure stability of the optimisation procedure. As we will see, for the majority of each experiment, AIDA only receives negative appraisals. The generative model of AIDA is fundamentally a classifier and unconstrained optimisation can therefore lead to degenerate results when the data set only 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 AIDA is assumed to have no a priori knowledge of the user preference function. This random initial proposal leads to distinct behaviour 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 behaviour. 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 separate terms of the EFE decomposition in Eq. (4.15) over time. We expect that when AIDA is primarily exploring, the utility drive is relatively low while the information gain is comparatively high. When AIDA is primarily engaged in exploitation, we expect the opposite pattern. We show these terms separately in Fig. 4.6.

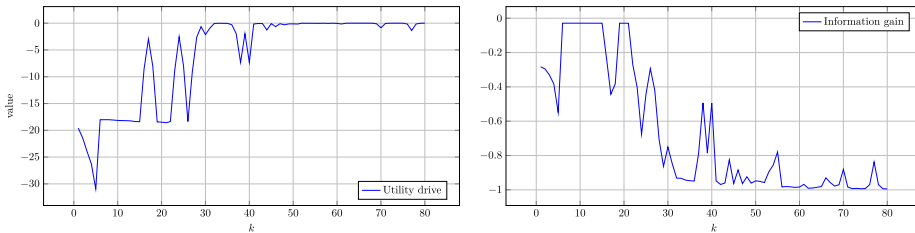


Figure 4.6: Evolution of the utility drive and negative information gain after throughout a single experiment.

Fig. 4.6 shows that there are distinct phases to the experiment. In the beginning ( $k < 5$ ) AIDA sees a sharp decrease in utility drive and information gain terms. This indicates a 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, hyperparameters are optimised and the agent no longer believes it has saturated the search space, which manifests as the jumps in Figure 4.6 from trial 5 to 6. From trials 6 through 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 that yield a positive user appraisal. The agent obtains its first positive appraisal at  $k = 16$ , as evidenced 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 behaviour starting from the 41'st trial. To examine the first transition, we can visualize the EFE landscape at  $k = 5$  and  $k = 6$ , the upper row of Fig. 4.7.

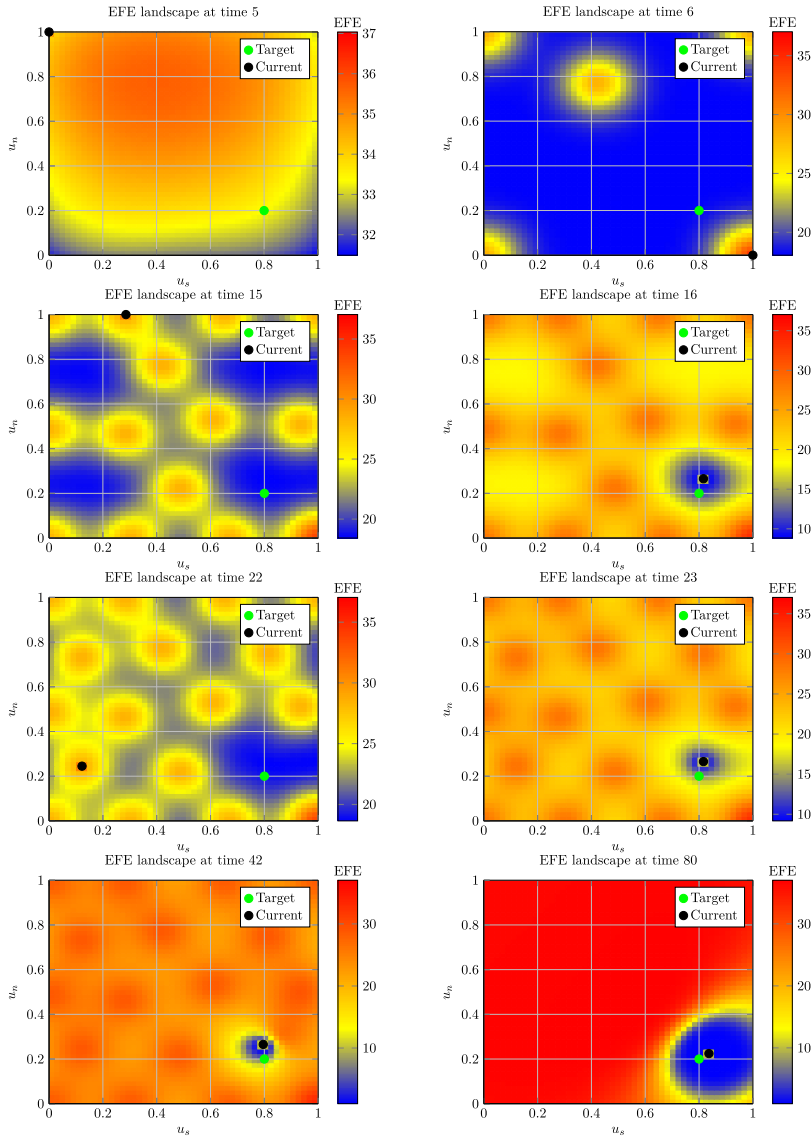


Figure 4.7: Snapshots of EFE landscape at different time points as a function of gains  $u_z$  and  $u_n$ ,  $G(\mathbf{u})$ . The black dot denotes the current parameter settings and the green dot denotes  $\mathbf{u}^*$ .



Since AIDA minimises EFE, it seeks out low values corresponding to blue regions and avoids 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  $k = 6$  indicate regions corresponding to previous proposals of AIDA that resulted in negative appraisals.

We can visualize snapshots of the exploration phase starting from  $k = 6$  in a similar manner. The second row of Fig. 4.7 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 result 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 Fig. 4.6, 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 engages in *local* exploration around where the optimum might be located. However, AIDA was unfortunately located near the boundary of the optimum and next received a negative appraisal. Therefore in trials 18 to 22, AIDA queries points throughout the search space that it deems informative. At trial 23, the position of AIDA in the search space (black dot in the third row of Fig. 4.7) returns to the edge of the user preference function in Fig. 4.5. This causes AIDA to receive a mixture of positive and negative appraisals in the following trials, leading to the oscillations seen in Fig. 4.6. Finally, we can examine the landscape after AIDA has confidently located the optimum and switched to purely exploitative behaviour. This happens at  $k = 42$  where the utility drive goes to 0 and the information gain concentrates around -1.

The last row of Fig. 4.7 shows that once  $\mathbf{u}^*$  is confidently located, AIDA disregards 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, indicating that if a user keeps requesting updated parameters, AIDA will once again perform local exploration around the optimum. This further shows that AIDA accommodates gradual retraining as the user's hearing loss profile changes over time.

Having thoroughly examined an example run and investigated the types of behaviour 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.

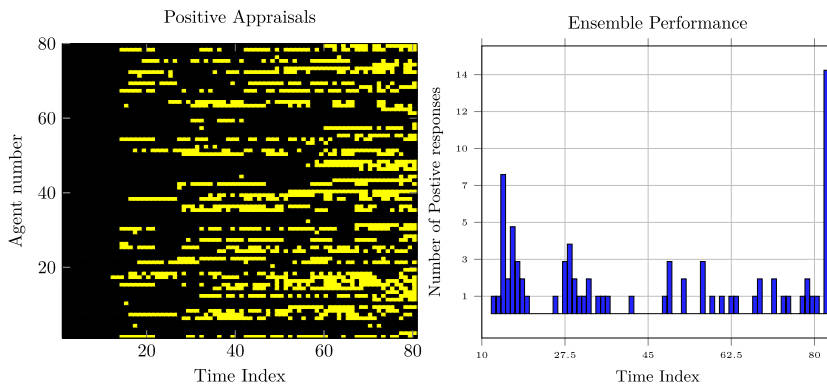


Figure 4.8: (Left) Heatmap showing ensemble performance over 80 agents. Positive and negative responses are indicated with yellow and black squares, respectively. (Right) Histogram showing time indices where agents receive their first positive response. The right-most column indicates agents that failed to obtain a positive appraisal. In total, 66/80 agents solve the task, corresponding to a success rate of 82.5%.

Fig. 4.8 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 sub-optimal with respect to eliciting positive user responses in order to gather more information. Fig. 4.8 also shows a histogram indicating when each agent obtains its first positive appraisal. The right-most column shows agents that failed to locate the optimum within the designated number of trials. In total, 66/80 agents correctly solved 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.

### 4.5.3 HA Algorithm Execution Verification

To verify the proposed inference methodology for the execution of the HA algorithm, we synthesised a dataset by sampling from the following generative model:

$$p(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}, \omega) = \mathcal{N}(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}, \omega \mathbf{I}) \quad (4.24a)$$

$$p(\mathbf{z}_t | \mathbf{z}_{t-1}, \boldsymbol{\theta}, \tau) = \mathcal{N}(\mathbf{z}_t | \mathbf{A}(\boldsymbol{\theta})\mathbf{z}_{t-1}, \mathbf{V}(\tau)) \quad (4.24b)$$

$$p(\mathbf{n}_t | \mathbf{n}_{t-1}, \boldsymbol{\phi}, \tau) = \mathcal{N}(\mathbf{n}_t | \mathbf{A}(\boldsymbol{\phi})\mathbf{n}_{t-1}, \mathbf{V}(\tau)) \quad (4.24c)$$

$$x_t = \mathbf{e}_1^T \mathbf{z}_t + \mathbf{e}_1^T \mathbf{n}_t, \quad (4.24d)$$

with priors

$$p(\boldsymbol{\theta}_0 | \omega) = \mathcal{N}(\boldsymbol{\theta}_0 | \mathbf{0}, \omega \mathbf{I}) \quad (4.25a)$$

$$p(\boldsymbol{\phi}) = \mathcal{N}(\boldsymbol{\phi} | \mathbf{0}, \mathbf{I}) \quad (4.25b)$$

$$p(\gamma) = \Gamma(\gamma | 1.0, 1e - 4) \quad (4.25c)$$

$$p(\tau) = \Gamma(\tau | 1.0, 1.0) \quad (4.25d)$$

$$\omega = 1e - 4 \quad (4.25e)$$

We use an uninformative prior for the output of the HA  $y_t$  as in Fig. 4.3 to prevent unwanted 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  $N$  were sampled from the discrete domains  $[4, 8]$  and  $[1, 4]$ . Priors that resulted in unstable AR or TVAR processes were resampled.

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 Eq.(4.24) however, we used non-informative priors for the states and parameters of the model that corresponds to the TVAR process in Eq.(4.24b). To ensure identifiability of the separated sources, we used weakly informative priors for the parameters of the AR process in Eq.(4.24c). Specifically, the mean of the prior for  $\boldsymbol{\phi}$  was centered around the real AR coefficients that were used in the data generation process. The goals of the experiment are

- To verify that the proposed inference procedure recovers the hidden states  $\boldsymbol{\theta}_t$ ,  $z_t$  and  $n_t$  for each generated dataset.
- To verify convergence of the BFE. Since the graph contains loops, this is not guaranteed [77].

For a typical realisation, inference results for the hidden states  $z_t$  and  $n_t$  are shown in the top row of Fig. 4.9

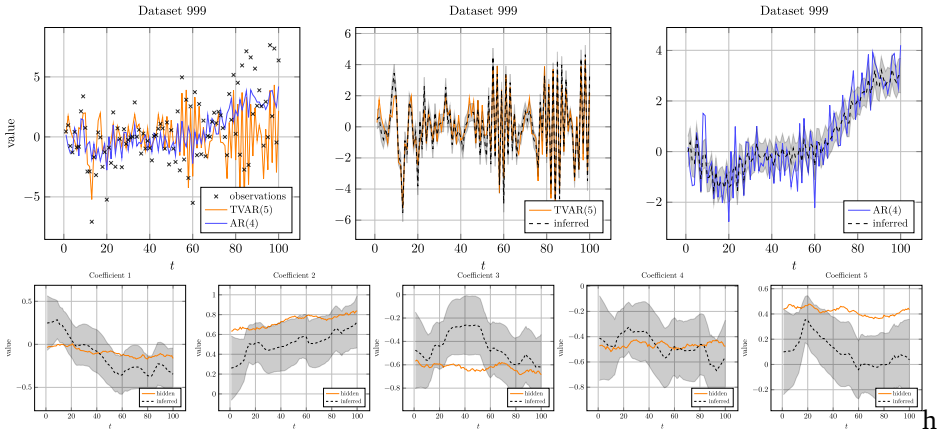


Figure 4.9: (Top) Inference results for the hidden states  $z_t$  and  $n_t$  of coupled (TV)AR process on dataset 999. (left) The generated observed signal  $x_t$  with underlying generated signals  $z_t$  and  $n_t$ . (center) The latent signal  $z_t$  and its corresponding posterior approximation. (right) The latent signal  $n_t$  and its corresponding posterior approximation. The dashed lines correspond to the mean of the posterior estimates. The transparent regions represent the corresponding remaining uncertainty as  $\pm$  one standard deviation from the mean. (Bottom) Inference results for the coefficients  $\theta_t$  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  $\pm$  one standard deviation from the mean of the estimated coefficients.

The bottom row of Fig. 4.9 shows the tracking of the time-varying coefficients  $\theta_t$ . Fig. 4.9 does not show 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.

A different kind of interesting behaviour is shown in Fig. 4.10. Here we see that inference results for the latent states  $z_t$  and  $n_t$  are swapped with respect to the true underlying signals. This type of behaviour is undesirable in standard algorithms when the output of the HA is based on hard-coded gains. However,

due to the gains being decided by the intelligent agent, we can still find optimal gains in this situation.

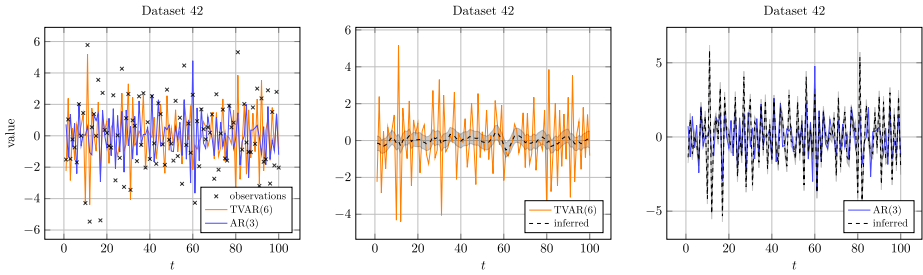


Figure 4.10: Inference results for the hidden states  $z_t$  and  $n_t$  of coupled (TV)AR process on dataset 42. In this particular case, the inferred states are swapped with respect to the true underlying signals. However, the accompanying intelligent agent is able to cope with these kinds of situations, such that the HA clients do not experience any problems as a result.

4

As seen from Fig. 4.11, the BFE averaged over all generated time series monotonically decreases. Note that even though the proposed hybrid MP algorithm results in a stationary solution, it does not provide convergence guarantees.

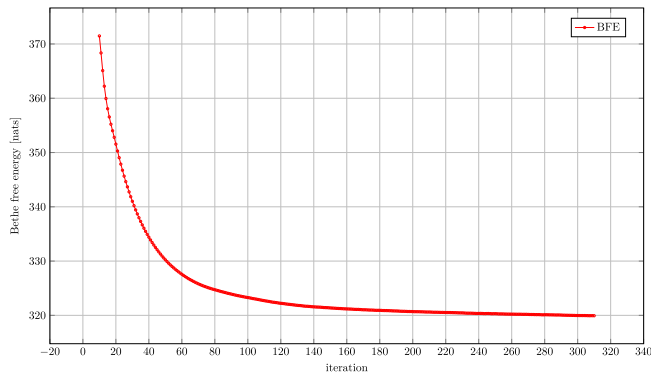


Figure 4.11: Evolution of the BFE for the coupled AR model averaged over all generated time series. The iteration index specifies the number of marginal updates for all edges in the graph.

#### 4.5.4 Validation Experiments

For the validation of the proposed HA algorithm and AIDA, we created an interactive web application<sup>3</sup> to demonstrate the joint system. Fig. 4.12 shows the interface of the demonstrator.

---

<sup>3</sup>A web application of AIDA is available at <https://github.com/biaslab/AIDA-app/>.

4

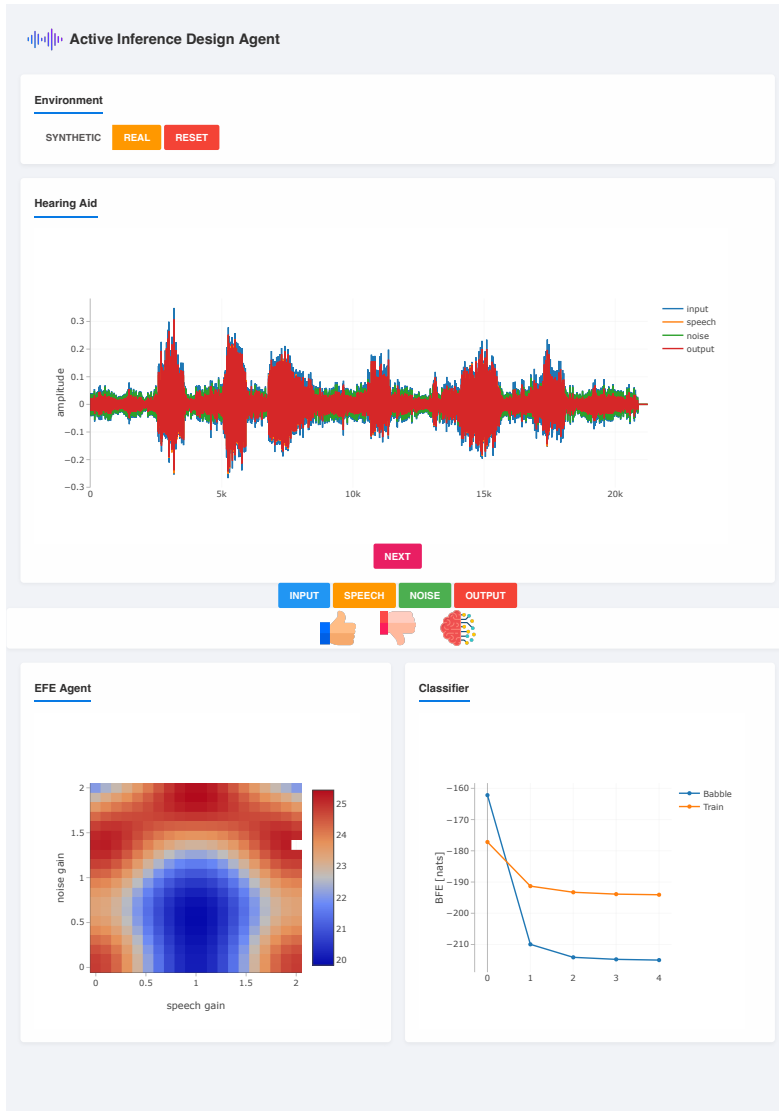


Figure 4.12: Screenshot of the interactive web application of AIDA. The dashboard consists of four distinct cells. The top cell *Environment* allows the user to change between an artificially generated and a real noise signal. It also contains a reset button for resetting the application. The *HA* cell provides an interactive plot of the input, separated speech and 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 provide AIDA with positive and negative appraisals. The brain button optimises hyperparameters of the 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 BFE score for different models corresponding to different contexts.

The user listens to the output of the HA algorithm by pressing the output button. The buttons `speech` and `noise` correspond to AIDA's beliefs about the constituent signals of the HA input. Note that in reality the user would not have access to this information and can only listen to HA output. After listening to the output signal, the user can assess the performance of the current HA setting. The user can send positive and negative appraisals by pressing the `thumbs up` or `thumbs down` buttons respectively. Once an appraisal is sent, AIDA updates its beliefs about the parameter space and provides new settings for the HA algorithm, in an attempt to make the user happy. As AIDA models user appraisals using a GPC, we provide an additional button that initiates optimisation of the hyperparameters of said GPC.

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, either interference from a sinusoidal wave or a power drill. In the synthetic environment, the HA algorithm exploits knowledge about acoustic contexts by using informative priors for the AR noise model. The real environment uses data from the NOIZEUS speech corpus<sup>4</sup>. Concretely the real environment consists of 30 sentences spoken in two different noise environments - either the background noise of a train station or babble noise. In the real environment, the HA algorithm instead uses weakly informative priors for the background noise which naturally influences the performance of the HA algorithm. Both the HA algorithm and AIDA determine acoustic context based on the BFE score, also shown in the demonstrator. The context with the lower BFE score corresponds to the selected acoustic context.

## 4.6 Related Work

The problem of HA personalisation has been explored in various works. In [78], HA parameters are tuned according to pairwise user assessment tests, during which the user's perception is modelled using GPs. 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 [78] 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

---

<sup>4</sup>The NOIZEUS database is available at <https://ecs.utdallas.edu/loizou/speech/noizeus/>.



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 overall less data to locate optimal gains. Secondly, AIDA can be continually trained and retrained. 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 the 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 [2], personalisation of the HA compression algorithm is framed in terms of deep reinforcement learning. Our work instead takes inspiration from the AIF framework where agents act to maximise model evidence of their underlying generative model. Importantly, this does not require us to explicitly specify a loss function that balances explorative and exploitative behaviour. In the recent work of [51], the HA preference learning algorithm is implemented through sequential Bayesian optimisation with pairwise comparisons. Their HA system comprises two subsystems representing a user with their preferences and an agent that guides the learning process. However, [51] focuses only on exploration through maximising 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 [51] to employ the full EFE is an exciting potential direction for future work. Finally neither [78] nor [51] takes context dependence into account.

[38] introduces Active Listening, which performs speech recognition based on the principles of AIF. In [38], they regard listening as an active process that is largely influenced by lexical, speaker, and prosodic information. [38] distinguishes itself from conventional audio processing algorithms because it explicitly includes the process of word boundary selection before word classification and recognition, and treats this as an active process. Word boundaries are selected from a group of candidate word boundaries through Bayesian model selection by choosing the word boundary that optimises the VFE during classification. In the future, we see potential improvements from incorporating the Active Listening approach into AIDA.

The audio processing components of AIDA essentially perform informed source separation [58], where sources are separated based on prior knowledge. Even though blind source separation approaches [68, 113] always use some degree of prior information, we have elected not to focus on this direction and

instead actively try to model the underlying sources based on variations of AR processes. For audio processing applications, source separation has often been performed in the log-power domain [27, 93, 94]. However, the interaction of the signals in this domain is no longer linear, resulting in intractability issues. The intractability that results from performing exact inference in this model is often resolved by simplifying the interaction function [45, 89]. Although this approach has shown to be successful in the past, its performance is limited because it neglects phase information.

## 4.7 Discussion

We have introduced AIDA, an Active Inference Design Agent that is capable of tuning context-dependent parameters of a HA 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 4.3.1 describes the dynamics of a speech signal perturbed by coloured noise. Even though the proposed inference algorithm allows for the decomposition of acoustic input signals into speech and noise components, it has limitations that must be highlighted. First, identifiability of the coupled AR model depends on the choice of priors. Non-informative priors can lead to poor source estimation [48, 57]. To tackle the identifiability issue, we used informative context-dependent priors for our experiments. In other words, for each context, we use a different set of priors that better capture the dynamics of the acoustic signal in that context. Second, throughout our experiments, we used fixed orders of both TVAR and AR models. In real-world applications, we do not have access to apriori information about the actual order of the underlying signals. Therefore, to continuously update our models of the underlying sources we would need to perform active order selection, which can be realized using Bayesian model reduction [31, 32]. Third, our model assumes that the HA device only has access to a monaural input, meaning the observed signal originates from a single microphone. As a result, we do not use any spatial information about the 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 HA algorithms rather than building a real-world HA engine. Fortunately, the proposed framework allows for easy substitution of source models with more versatile ones that might be better suited for speech. For instance, one can use several microphones, as commonly done in beamforming [79], or use a frequency decomposition for im-

proving source separation performance [27, 93, 94]. However, a more complex model is likely to result in a higher computational burden which will pose a new set of challenges when working on embedded devices.

The power of the intelligent 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 non-parametric model with very few assumptions on the underlying function. Placing constraints on the preference function, such as was done in [20, 51], is likely to improve the data efficiency of the agent. Arguably, a core move of [20, 51] 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. In turn, working with a less flexible model predictably leads to higher data efficiency, which can aid the agent's performance. 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 4.3.2, we make a number of assumptions on the control variable  $\mathbf{u}_k$  and user appraisals  $r_k$ . First, we set the domain of the elements of control variable  $\mathbf{u}_k$  to  $[0, 1]$ . This is an arbitrary constraint that we use for illustrative purposes. The domain can be easily rescaled without loss of generality. For example, our demonstrator uses the default domain of  $\mathbf{u}_k \in [0, 2]^2$ . Secondly, we opt for binary user appraisals, i.e.  $r_k \in \{\emptyset, 0, 1\}$ . This design choice follows from the desire to allow users to communicate covertly with AIDA since binary user appraisal can more easily be linked to for example covert wrist movements when wearing a smartwatch. With continuous user appraisals, e.g.  $r_k \in [0, 1]$ , or pairwise comparison tests, the convergence of AIDA can be greatly improved as these appraisals yield more information per trial. However, providing AIDA with these appraisals requires more user attention, which can be undesirable in certain circumstances.

Real-world testing of AIDA has not been included in our work. Performance evaluation with human HA clients is not straightforward. To evaluate the performance of AIDA, we would need to conduct a randomised controlled trial, where HA clients are randomly assigned to either an experimental group or a control group. While the intelligent agent AIDA can interact with users in real-time, the source separation framework is currently limiting actual, real-time deployment of the system. Given the current model assumptions - two AR filters under a

variational approximation - we obtain good source separation performance at the cost of computational complexity. As a consequence, the complete system is not yet suitable for a proper randomised controlled trial setting. Nonetheless, we provide a demo that simulates AIDA and can be tested freely. In future work, we will focus on specifying source models that only require cheap computations, allowing us to run the source separation algorithms in real-time.

## 4.8 Conclusions

This chapter has presented AIDA, an active inference design agent for novel situation-aware personalised HA algorithms. AIDA and the corresponding HA algorithm are based on probabilistic generative models that model both user preferences and the underlying speech and context-dependent background noise signals of an observed acoustic signal. Through probabilistic inference by means of MP, we perform informed source separation and use the separated signals to perform source-specific filtering. AIDA then learns personalised source-specific gains through user interaction, depending on the environment that the user is currently in. Users can at any time provide a binary appraisal after which the agent will make an improved proposal, based on EFE minimisation, balancing both exploitative and explorative behaviour.

Experimental results indicate that hybrid MP procedures are capable of correctly inferring hidden states of the coupled AR model that are associated with speech and noise components of the observed acoustic signal. Moreover, Bayesian model selection has been demonstrated adequate for the context inference problem when each source is modelled by an AR process. The experiments on preference learning showed the potential of applying EFE minimisation for finding optimal settings of the HA algorithm. Although real-world implementations still present challenges, this novel class of audio processing algorithms provides an alternative to leading approaches to HA algorithm design.

## Data Availability Statement

The datasets used in this study can be found at the AIDA-data repository<sup>5</sup>.

---

<sup>5</sup>The datasets used in this study can be found at the AIDA-data repository at <https://github.com/biaslab/AIDA-data>.



# Chapter 5

## Realising Synthetic Active Inference Agents

*”Such is the advantage of a well constructed language that its simplified notation often becomes the source of profound theories. ”*

*– Pierre Simon Laplace*

The Free Energy Principle (FEP) is a theoretical framework for describing how (intelligent) systems self-organise into coherent, stable structures by minimising a free energy functional. Active Inference (AIF) is a corollary of the FEP that specifically details how systems that can plan for the future (agents) function by minimising particular free energy functionals that incorporate information seeking components. In this chapter, we derive a local version of the free energy functionals used for AIF. This enables us to construct a version of AIF that applies to arbitrary graphical models and interfaces with prior work on message passing algorithms. We derive the required message passing updates as well and demonstrate an algorithm for direct policy inference on the classic T-maze task. A key advantage of performing direct policy inference is that it circumvents a long standing scaling issue that has so far hindered the application of AIF in industrial settings.

We also identify a gap in the graphical notation used for factor graphs. While factor graphs are great at expressing a generative model, they have so far been

unable to specify the full optimisation problem including constraints. To solve this problem we develop constrained Forney-style factor graph (CFFG) notation which permits a fully graphical description of variational inference objectives. We proceed to show how CFFGs can be used to reconstruct prior algorithms for AIF as well as derive new ones.

The present chapter is based on the twin original works referenced below. The contents of both papers were established in close collaboration with T. W. van de Laar. The present chapter focuses mainly on Part I and includes content from Part II where necessary to form a complete document.

**Koudahl, M. T., van de Laar, T. W., & de Vries, B. (2023).** *Realising Synthetic Active Inference Agents, Part I: Epistemic Objectives and Graphical Specification Language* arXiv:2306.08014

**van de Laar, T. W., Koudahl, M. T., & de Vries, B. (2023).** *Realising Synthetic Active Inference Agents, Part II: Variational Message Passing Updates* arXiv:2306.02733

## 5.1 Introduction

Active Inference (AIF) is an emerging framework for modelling intelligent agents interacting with an environment. Originating in the field of computational neuroscience, it has since been spread to numerous other fields such as modern machine learning. At its core, AIF relies on variational inference techniques to minimise a free energy functional. A key differentiator of AIF compared to other approaches is the use of custom free energy functionals such as Expected and Generalised free energies (expected free energy (EFE) and generalised free energy (GFE), respectively). These functionals are specifically constructed so as to elicit epistemic, information seeking behaviour when used to infer actions.

Optimisation of these functionals have so far relied on custom algorithms that require evaluating very large search trees which has rendered upscaling of AIF difficult. Many recent works, such as Branching Time AIF [16] and sophisticated inference [35] have investigated algorithmic ways to prune the search tree in order to solve this problem.

In this chapter, we take a different approach and formulate a variation of the GFE optimisation problem using a custom Lagrangian derived from constrained Bethe free energy (CBFE) on factor graphs. Using variational calculus we then derive a custom message passing (MP) algorithm that can directly solve for fixed

points of local GFE terms, removing the need for a search tree. This allows us to construct a purely optimisation-based approach to AIF which we name Lagrangian Active Inference (LAIF).

LAIF applies to arbitrary graph topologies and interfaces with generic MP algorithms which allow for scaling up of AIF using off-the-shelf tools. We accomplish this by constructing a node-local GFE on generic factor graphs.

The present chapter is structured as follows: In Section 5.2, we review relevant background material concerning Forney-style factor graphs (FFGs) and Bethe free energy (BFE). In Section 5.3 we formalise what we mean by "epistemics" and construct an objective that is local to a single node on an FFG and possesses an epistemic, information-seeking drive. This objective turns out to be a local version of the GFE which we review in Section 5.3.2.

Once we start modifying the free energy functional, we recognise a problem with current FFG notation. FFGs visualise generative models but fail to display a significant part of the optimisation problem, namely, the variational distribution and the functional to be optimised. To remedy this, we develop the CFFG graphical notation in Section 5.5 as a method for visualising both the variational distribution and any adaptations to the free energy functional that are needed for AIF. These tools form the basis of the update rules derived in Appendix C.3.

With the CFFG notation in hand, in Section 5.8 we then proceed to demonstrate how to recover prior algorithms for AIF as message passing on a CFFG. AIF is often described as message passing on a probabilistic graphical model, see [22, 33, 59, 103] for examples. However, this relationship has not been properly formalised before, in part because adequate notation has been lacking. Using CFFGs it is straightforward to accurately write down this relation. Further, due to the modular nature of CFFGs it becomes easy to devise extensions to prior AIF algorithms that can be implemented using off-the-shelf MP tools.

Finally, Section 5.9 demonstrates a new algorithm for policy inference using LAIF that scales linearly in the planning horizon, providing a potential solution to a long-standing barrier for scaling AIF to larger models and more complex tasks.

## 5.2 The Lagrangian Approach to Message Passing

In this section, we review the Bethe free energy along with FFGs. These concepts form the foundation from which we will build towards local epistemic, objectives.



As a free energy functional, the BFE is unique because stationary points of the BFE correspond to solutions of the belief propagation algorithm [84, 114, 116] which provides exact inference on tree-structured graphs.

Furthermore, by adding constraints to the BFE using Lagrange multipliers, one can form a custom Lagrangian for an inference problem. Taking the first variation of this Lagrangian, one can then solve for stationary points and obtain MP algorithms that solve the desired problem. Prior work [114, 116] have shown that adding additional constraints to the BFE allows for deriving a host of different message passing algorithms including variational message passing (VMP) [112] and expectation propagation (EP) [75] among others. We refer interested readers to [116] for a comprehensive overview of this technique and how different choices of constraints can lead to different algorithms.

Constraints are specified at the level of nodes and edges, meaning this procedure can produce hybrid MP algorithms, foreshadowing the approach we are going to take for deriving a local, epistemic objective for AIF.

### 5.2.1 Bethe Free Energy and Forney-style Factor Graphs

Throughout the remainder of this chapter, we will use FFGs to visualise probabilistic models. In Section 5.5 we extend this notation to additionally allow for specifying constraints on the variational optimisation objective. Following [116] we define an FFG as a graph  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  with nodes  $\mathcal{V}$  and edges  $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ . For a node  $a \in \mathcal{V}$ , we denote the connected edges by  $\mathcal{E}(a)$ . Similarly for an edge  $i \in \mathcal{E}$ , we denote the connected nodes by  $\mathcal{V}(i)$ .

An FFG can be used to represent a factorised function (model) over variables  $\mathbf{s}$ , as

$$f(\mathbf{s}) = \prod_{a \in \mathcal{V}} f_a(\mathbf{s}_a), \quad (5.1)$$

where  $\mathbf{s}_a$  collects the argument variables of the factor  $f_a$ . Throughout this chapter, we will use *cursivebold* font to denote collections of variables. In the corresponding FFG, the factor  $f_a$  is denoted by a square node, and connected edges  $\mathcal{E}(a)$  represent the argument variables  $\mathbf{s}_a$ .

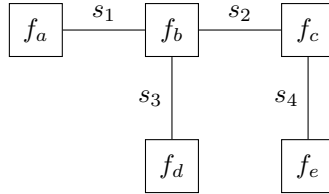


Figure 5.1: Example of an FFG

As an example, we can consider the FFG shown in Fig. 5.1 which corresponds to the model

$$f(s_1, s_2, s_3, s_4) = f_a(s_1)f_b(s_1, s_2, s_3)f_c(s_2, s_4)f_d(s_3)f_e(s_4) \quad (5.2)$$

In Fig. 5.1, the vertex set is  $\mathcal{V} = \{a, \dots, e\}$  and the edge set is  $\mathcal{E} = \{1, \dots, 4\}$ . As an example, the neighbouring edges of the node  $c$  are given by  $\mathcal{E}(c) = \{2, 4\}$ . In this way, FFGs allow for simple visualisation of the factorisation properties of a high-dimensional function.

The problem we will focus on concerns the minimisation of a free energy functional over a generative model. More formally, given a model (5.1) and a "variational" distribution  $q(\mathbf{s})$ , the variational free energy (VFE) is defined as

$$F[q] \triangleq \int q(\mathbf{s}) \log \frac{q(\mathbf{s})}{f(\mathbf{s})} d\mathbf{s}. \quad (5.3)$$

Variational inference concerns minimising this functional, leading to the solution

$$q^* = \arg \min_{q \in \mathcal{Q}} F[q], \quad (5.4)$$

with  $\mathcal{Q}$  denoting the admissible family of functions  $q$ . The optimised VFE upper-bounds the negative log-evidence (surprisal), as

$$F[q^*] = \underbrace{\int q^*(\mathbf{s}) \log \frac{q^*(\mathbf{s})}{p(\mathbf{s})} d\mathbf{s}}_{\text{Posterior divergence}} - \underbrace{\log Z}_{\text{Surprisal}}, \quad (5.5)$$

with  $Z = \int f(\mathbf{s})d\mathbf{s}$  is the model evidence and the exact posterior is given by  $p(\mathbf{s}) = f(\mathbf{s})/Z$ .

The BFE applies the Bethe assumption to the factorisation of  $q$  which yields an objective that decomposes into a sum of local free energy terms, each local to a node on the corresponding FFG. Each node local free energy will include entropy terms from all connected edges. Since an edge is connected to (at most) two nodes, that means the corresponding entropy term would be counted twice. To prevent overcounting of the edge entropies, the BFE includes additional terms entropy terms that cancel out overcounted terms.

Under the Bethe approximation  $q(\mathbf{s})$  is given by

$$q(\mathbf{s}) = \prod_{a \in \mathcal{V}} q_a(\mathbf{s}_a) \prod_{i \in \mathcal{E}} q_i(s_i)^{1-d_i} \quad (5.6)$$

with  $d_i$  the degree of edge  $i$ . As an example, on the FFG shown in Fig. 5.1 this would correspond to a variational distribution of the form

$$q(s_1, \dots, s_4) = \frac{q_a(s_1)q_b(s_1, s_2, s_3)q_c(s_2, s_4)q_d(s_3)q_e(s_4)}{q_1(s_1)q_2(s_2)q_3(s_3)q_4(s_4)} \quad (5.7)$$

where we see terms for the edges in the denominator and for the nodes in the numerator. With this definition, the free energy factorises over the FFG as

$$F[q] = \sum_{a \in \mathcal{V}} \underbrace{\int q_a(\mathbf{s}_a) \log \frac{q_a(\mathbf{s}_a)}{f_a(\mathbf{s}_a)} d\mathbf{s}_a}_{F[q_a]} + \sum_{i \in \mathcal{E}} (1 - d_i) \underbrace{\int q_i(s_i) \log \frac{1}{q_i(s_i)} ds_i}_{H[q_i]} \quad (5.8)$$

Eq. (5.8) defines the BFE. Note that  $F$  defines a free energy functional which can be either local or global depending on its arguments. More specifically,  $F[q]$  defines the free energy for the entire model, while  $F[q_a]$  defines a node local (to the node  $a$ ) BFE contribution of the same functional form.

Under optimisation of the BFE- solving Eq. (5.4) - the admissible set of functions  $\mathcal{Q}$  enforces consistent normalisation and marginalisation of the node and edge local distributions, such that

$$\begin{aligned}
 \int q_i(s_i) ds_i &= 1 \text{ for all } i \in \mathcal{E} \\
 \int q_a(s_a) ds_a &= 1 \text{ for all } a \in \mathcal{V} \\
 \int q_a(s_a) ds_{a \setminus i} &= q_i(s_i) \text{ for all } a \in \mathcal{V}, i \in \mathcal{E}(a).
 \end{aligned} \tag{5.9}$$

Because we always assume the constraints given by Eq. (5.9) to be in effect, we will omit the subscript on individual  $q$ 's moving forwards and instead let the arguments determine which marginal we are referring to, for example writing  $q(s_a)$  instead of  $q_a(s_a)$ .

A core aspect of FFGs is that they allow for easy visualisation of MP algorithms. MP algorithms are a family of distributed inference algorithms with the common trait that they can be viewed as messages flowing on an FFG.

As a general rule, to infer a marginal for a variable, messages are passed on the graph toward the associated edge for that variable. Multiplication of colliding (forwards and backwards) messages on an edge yields the desired posterior marginal. We denote a message on an FFG by arrows pointing in the direction that the message flows.

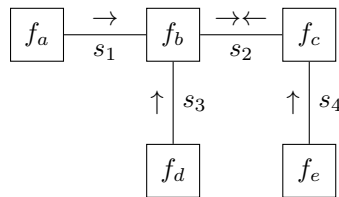


Figure 5.2: Example of messages flowing on an FFG

We show an example in Fig. 5.2 of inferring a posterior marginal for the variable  $s_2$ . In this example, messages are flowing on the FFG of Fig. (5.1) towards the variable  $s_2$  where we see two arrows colliding. The exact form of the individual messages depends on the algorithm being used.

### 5.3 Defining Epistemic Objectives

Now we move on to the central topic of AIF, namely agents that interact with the world they inhabit. Under the heading of AIF, an agent entails a generative model of its environment and is engaged in the process of achieving future goals (specified by a target distribution or goal prior) through actions. This task can be cast as a process of free energy minimisation [28, 35, 37]. A natural question to ask is then, what should this free energy functional look like and why? Here we wish to highlight a core feature of AIF that sets it apart from other approaches: Systematic information gathering by targeted exploration of an environment. Whatever the form of our free energy functional, it should lead to agents that possess an exploratory drive consistent with AIF.

While it is tempting to default to a standard VFE, prior work [102] has shown that directly optimising BFE or VFE when inferring a sequence of actions (which we will refer to as a *policy*) does not lead to agents that systematically explore their environment. Instead, directly inferring a policy by minimising a VFE/BFE leads to Kullback-Leibler divergence (KL)-control [54]<sup>1</sup>. This means that VFE/BFE is not the correct choice when we desire an agent that actively samples its environment with the explicit purpose of gathering information.

Instead a hallmark feature of AIF is the use of alternative functionals in place of either the VFE or the BFE, specifically for inferring policies. The goal of these alternative functionals is often specifically to induce an epistemic, explorative term that drives AIF agents to seek out information.

Epistemics, epistemic behaviour or "foraging for information" are commonly used terms in the AIF literature and related fields. While we have used the term colloquially until this point, we now clarify formally how we use the term in the present paper and how it relates to the objective functionals we consider. A core problem is that epistemics is most often defined in terms of the behaviour of agents ("what does my agent *do*?") rather than from a mathematical point of view. Prior work on this point includes [43, 74].

We take the view that epistemics arise from the optimisation of either an mutual information (MI) term or a bound thereon. The MI (between two variables  $x$  and  $z$ ) is defined as [71]

$$I[x, z] = \iint p(x, z) \log \frac{p(x, z)}{p(x)p(z)} dx dz. \quad (5.10)$$

<sup>1</sup>Other works will sometimes use  $\pi$  as a symbol to denote a policy

To gain an intuition for why maximising MI leads to agents that seek out information, we can rewrite MI as

$$\begin{aligned} I[x, z] &= \iint p(x, z) \log \frac{p(z|x)}{p(z)} dx dz \\ &= H[z] - H[z|x] = H[x] - H[x|z] \end{aligned} \quad (5.11)$$

Since MI is symmetric in its arguments, Eq. (5.11) can equally well be written in terms of  $x$  rather than  $z$ . Eq. (5.11) shows that MI decomposes as the difference between the marginal entropy  $z$  and the expected entropy of  $z$  conditional on  $x$ .

If we let  $z$  denote an internal state of an agent and  $x$  an observation and allow our agent to choose  $x$  - for instance through acting on an environment - we can see why maximising Eq. (5.10) biases the agent towards seeking our observations that reduce entropy in  $z$ . Maximising Eq. (5.10) means the agent will prefer observations that provide useful (in the sense of reducing uncertainty) information about its internal states  $z$ . For this reason MI is also known as Information Gain.

Actually computing Eq. (5.10) is often intractable and in practice, a bound is often optimised instead.

### 5.3.1 Constructing a Local Epistemic Objective

At this point, we have seen that the BFE is defined over arbitrary FFG's, yet does not lead to epistemic behaviour. On the other hand, maximising MI leads to the types of epistemic behaviour we desire, yet is not distributed like the BFE. The question becomes whether there is a way to merge the two and obtain a distributed functional - like the BFE- that includes an epistemic term?

We will now show how to construct such a functional. Our starting point will be the BFE given by Eq. (5.8). We will focus on a single node  $a$  and its associated local energy term and partition the incoming edges into two sets,  $x$  and  $z$ . This gives the node local free energy

$$F[q_a] = \iint q(x, z) \log \frac{q(x, z)}{f(x, z)} dx dz \quad (5.12)$$

Now we need to add on an MI term to induce epistemics. Since we are minimising our free energy functional and want to maximise MI, we augment the free energy with a negative MI term as

$$\mathcal{G}[q_a] = \overbrace{\iint q(\mathbf{x}, z) \log \frac{q(\mathbf{x}, z)}{f(\mathbf{x}, z)} d\mathbf{x}dz}^{\text{Variational free energy}} + \overbrace{\iint q(\mathbf{x}, z) \log \frac{q(\mathbf{x})q(z)}{q(\mathbf{x}, z)} d\mathbf{x}dz}^{\text{Negative mutual information}} \quad (5.13a)$$

$$= \iint q(\mathbf{x}, z) \log \frac{q(\mathbf{x}, z)}{f(\mathbf{x}, z)} d\mathbf{x}dz + \iint q(\mathbf{x}, z) \log \frac{q(\mathbf{x})q(z)}{q(\mathbf{x}, z)} d\mathbf{x}dz \quad (5.13b)$$

$$= \iint q(\mathbf{x}, z) \log \frac{q(\mathbf{x})q(z)}{f(\mathbf{x}, z)} d\mathbf{x}dz \quad (5.13c)$$

which we can recognise as a node-local GFE [81]. In Section 5.3.2 we review the results of [81] to expand upon this statement. Eq. (5.13a) provides a straightforward explanation of the kinds of behaviour that we can expect out of agents optimising a GFE. Namely, minimising BFE with a goal prior corresponds to performing KL-control [54] while the MI term adds an epistemic, information seeking component. Viewed in this way, there is nothing mysterious about the kind of objective optimised by AIF agents: It is simply the sum of two well known and established objectives that are each widely used within the control and reinforcement learning communities.

## 5

### 5.3.2 Generalised Free Energy

The objective derived in Eq. (5.13) is a node local version of the GFE originally introduced by [81]. In this section we review the GFE as constructed by [81] in order to relate our construction to prior work on designing AIF functionals. In Section 5.8 we show how to reconstruct the exact method of [81] using the tools we develop in this paper. Prior to [81], the functional of choice was the expected free energy (EFE) [22, 33]. [81] identified some issues with the EFE and proposed the GFE as a possible solution.

We show how to reconstruct the original EFE-based algorithm of [33] using our local objective in Section 5.8. We also provide a detailed description of the EFE in Appendix. C.1 since it is still a popular choice for designing AIF agents.

A core issue with the EFE is that it is strictly limited to planning over future time steps. This means that AIF agents that utilise the EFE functional need to maintain two separate models: One for inferring policies (using EFE) and one for state inference (using VFE/BFE) that updates as observations become available. The key advantage of EFE is that it induces the epistemic drive that we desire from AIF agents.

The goal of [81] was to extend upon the EFE by introducing a functional that could induce similar epistemic behaviour when used to infer policies while at the same time reducing to a VFE when dealing with past data points. In this way an agent would no longer have to maintain two separate models and could instead utilise only one.

The GFE as introduced by [81] is tied to a specific choice of generative model. The model introduced by [81] is given by

$$p(\mathbf{x}, \mathbf{z} \mid \hat{\mathbf{u}}) \propto p(\mathbf{z}_0) \prod_{k=1}^T p(\mathbf{x}_k \mid \mathbf{z}_k) p(\mathbf{z}_k \mid \hat{\mathbf{u}}_k, \mathbf{z}_{k-1}) \tilde{p}(\mathbf{x}_k) \quad (5.14)$$

where  $\mathbf{x}$  denotes observations,  $\mathbf{z}$  denotes latent states and  $\hat{\mathbf{u}}$  denotes a fixed policy. Throughout this paper we will use  $t$  to refer to the current time step in a given model and denote fixed values by a hat, here exemplified by  $\hat{\mathbf{u}}$ . Further, we also use  $\mathbf{z}$  to denote vectors.

Given a current time step  $t$ , we have that for future time steps  $k > t$ ,  $\tilde{p}(\mathbf{x}_k)$  defines a goal prior over desired future observations. For past time steps  $k \leq t$ , we instead have  $\tilde{p}(\mathbf{x}_k) = 1$  which makes it uninformative<sup>2</sup>. Writing the model in this way allows for a single model for both perception (integrating past data points) and action (inferring policies) since both past and future time steps are included. GFE is defined as [81]

$$\mathcal{G}[q; \hat{\mathbf{u}}] = \sum_{k=1}^T \iint q(\mathbf{x}_k \mid \mathbf{z}_k) q(\mathbf{z}_k \mid \hat{\mathbf{u}}_k) \log \frac{q(\mathbf{x}_k \mid \hat{\mathbf{u}}_k) q(\mathbf{z}_k \mid \hat{\mathbf{u}}_k)}{\tilde{p}(\mathbf{x}_k) p(\mathbf{x}_k, \mathbf{z}_k \mid \hat{\mathbf{u}}_k)} d\mathbf{x}_k d\mathbf{z}_k \quad (5.15)$$

where

$$q(\mathbf{x}_k \mid \mathbf{z}_k) = \begin{cases} \delta(\mathbf{x}_k - \hat{\mathbf{x}}_k) & \text{if } k \leq t \\ p(\mathbf{x}_k \mid \mathbf{z}_k) & \text{if } k > t \end{cases} \quad (5.16)$$

and  $\hat{\mathbf{x}}_k$  denotes the observed data point at time step  $k$ .  $p(\mathbf{x}_k, \mathbf{z}_k \mid \hat{\mathbf{u}}_k)$  can be found recursively by Bayesian smoothing, see [60, 99] for details. To see how the GFE reduces to a VFE when data is available, we refer to Appendix C.2.

The GFE introduced by [81] improves upon prior work utilising EFE but still has some issues. The first lies in tying it to the model definition in Eq. (5.14).

<sup>2</sup>[81] writes this as the prior being flat to achieve a similar effect



Being committed to a model specification apriori severely limits what the GFE can be applied to since not all problems are going to fit the model specification. Additionally, a more subtle issue lies in the commitment to a hard time horizon. If  $t$  denotes the current time step, at some point time will advance to the point  $t > T$ . When this happens Eq. (5.14) loses the capacity to plan and becomes static since all observations are clamped by virtue of Eq. (5.16). A further complication arises from  $\hat{u}$  being a fixed parameter and not a random variable. Being a fixed parameter means it is not possible to perform inference for  $\hat{u}$  which in turn makes scaling difficult. Moving to a fully local version of the GFE instead means we can construct a new approach to AIF that addresses these issues.

## 5.4 LAIF - Lagrangian Active Inference

Armed with the node local GFE derived in Eq. (5.13), we can construct a Lagrangian for AIF. The goal is to adapt the Lagrangian approach to MP sketched in Section 5.2 to derive a MP algorithm that optimises local GFE in order to have a distributed inference procedure that incorporates epistemic terms. Naively applying the method of [116] to Eq. (5.13) does not yield useful results because the numerator differs from the term we take the expectation with respect to. To obtain a useful solution we need that

$$q(\mathbf{x}, \mathbf{z}) = q(\mathbf{x} | \mathbf{z})q(\mathbf{z}) \quad (5.17)$$

and add the original assumption in [81], given by Eq. (5.16)

$$q(\mathbf{x} | \mathbf{z}) \triangleq p(\mathbf{x} | \mathbf{z}). \quad (5.18)$$

With these additional assumptions, we can obtain meaningful solutions and derive a MP algorithm that optimises a local GFE and induces epistemic behaviour which we demonstrate in Section 5.9. The detailed derivation of these results can be found in Section 5.6.

At this point, we will instead show a different way to arrive at the local GFE. This approach is more "mechanical" but has the advantage that it can more easily be written in terms of constraints on the BFE. Our starting point will once again be a free energy term local to the node  $a$

$$F[q_a] = \int q(\mathbf{s}_a) \log \frac{q(\mathbf{s}_a)}{f(\mathbf{s}_a)} d\mathbf{s}_a \quad (5.19)$$

To turn Eq. (5.19) into a local GFE we need to perform two steps. The first is to enforce a mean field factorisation

$$q(\mathbf{s}_a) = \prod_{i \in \mathcal{E}(a)} q(s_i) \quad (5.20)$$

The second is to change the expectation to obtain

$$\int q(\mathbf{s}_a) \log \frac{q(\mathbf{s}_a)}{f(\mathbf{s}_a)} d\mathbf{s}_a \implies \int p(s_i | \mathbf{s}_{a \setminus i}) q(\mathbf{s}_{a \setminus i}) \log \frac{q(\mathbf{s}_a)}{f(\mathbf{s}_a)} d\mathbf{s}_a \quad (5.21)$$

where  $i \in \mathcal{E}(a)$ . This partitions the connected variables into two sets: A set of variables where the expectation is modified and any remainder that is not modified.  $p(s_i | \mathbf{s}_{a \setminus i})$  denotes a conditional probability distribution. We write  $p$  here instead of  $f$  to emphasise that the conditional needs to be normalised in order for us to be able to take the expectation.

We refer to this move as a *P-substitution*. Once the mean field factorisation is enforced, we can recognise Eq. (5.21) as a node local GFE  $G[q_a]$ .

As an example of constructing a node local GFE with this approach, we can consider a node with two connected variables,  $\{x, z\}$ .

The local free energy becomes

$$F[q_a] = \iint q(x, z) \log \frac{q(x, z)}{p(x, z)} dx dz \quad (5.22)$$

Now we apply a mean field factorisation

$$F[q_a] = \iint q(x)q(z) \log \frac{q(x)q(z)}{p(x, z)} dx dz \quad (5.23)$$

And finally, perform P-substitution to obtain a node local GFE

$$\iint q(x)q(z) \log \frac{q(x)q(z)}{p(x, z)} dx dz \implies \iint p(x | z)q(z) \log \frac{q(x)q(z)}{p(x, z)} dx dz \quad (5.24)$$

We denote the set of nodes for which we want to perform P-substitution with  $\mathcal{P} \subseteq \mathcal{V}$ . When performing P-substitution, we are replacing a local *variational* free energy  $F[q_a]$  with a local *generalised* free energy  $\mathcal{G}[q_a]$ . Armed with P-substitution, we can now write the simplest instance of a Lagrangian for Active Inference

$$\begin{aligned}
\mathcal{L}[q] = & \underbrace{\sum_{a \in \mathcal{P}} \mathcal{G}[q_a]}_{\substack{\text{P-substituted subgraph} \\ \text{w / naive mean field}}} + \underbrace{\sum_{b \in \mathcal{V} \setminus \mathcal{P}} F[q_b]}_{\substack{\text{Node local} \\ \text{free energies}}} + \sum_{i \in \mathcal{E}} (1 - d_i) \underbrace{\text{H}[q(s_i)]}_{\text{Edge entropy}} \\
& + \underbrace{\sum_{a \in \mathcal{V}} \sum_{i \in \mathcal{E}} \int \lambda_{ia}(s_i) \left[ q(s_i) - \int q(\mathbf{s}_a) d\mathbf{s}_{a \setminus i} \right] ds_i}_{\text{Marginalisation}} \\
& + \underbrace{\sum_{a \in \mathcal{V}} \lambda_a \left[ \int q(\mathbf{s}_a) d\mathbf{s}_a - 1 \right]}_{\text{Normalisation of node marginals}} + \underbrace{\sum_{i \in \mathcal{E}} \lambda_i \left[ \int q(s_i) ds_i - 1 \right]}_{\text{Normalisation of edge marginals}}.
\end{aligned} \tag{5.25}$$

With the Active Inference Lagrangian in hand, we can now solve for stationary points using variational calculus and obtain MP algorithms for LAIF. The key insight is that messages flowing out of  $\mathcal{P}$  derived from stationary points of Eq. (5.25) will correspond to stationary points of the local GFE rather than BFE, meaning the result will include an epistemic component. This paves the way for a localised version of AIF that applies to arbitrary graph structures, does not suffer from scaling issues as the planning horizon increases, and can be solved efficiently and asynchronously using MP.

## 5.5 Constrained Forney-style Factor Graphs

While FFGs are a useful tool for writing down generative models, we have by now established the importance of knowing the exact functional to be minimised. This requires specifying not just the model  $f$  but also the family  $\mathcal{Q}$  through constraints and any potential P-substitutions. This is important if we want to be able to succinctly specify not just the model but also the exact inference problem we aim to solve.

We will now develop just such a new notation for writing constraints directly as part of the FFG. We refer to FFGs with added constraints specification as CFFGs.

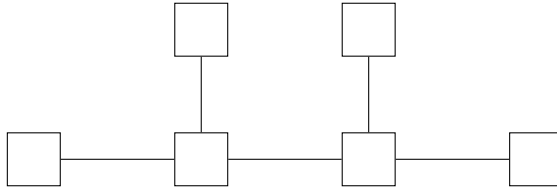


Figure 5.3: An example FFG.

Fig 5.3 shows a model comprised of five edges and six nodes. FFGs traditionally represent the model  $f$  using squares connected by lines as shown in Fig. 5.3. The squares represent factors and the connections between them represent variables. Connecting an edge to a square node indicates that the variable on that edge is an argument of the factor it is connected to.

The notation for CFFGs adheres to similar principles when specifying  $f$ . However, we augment the FFG with circular beads to indicate the constraints that define our family  $\mathcal{Q}$ . Each factor of the variational distribution in  $q$  will correspond to a bead and the position of a bead indicates to which marginal it refers - a bead on an edge denotes an edge marginal  $q(s_i)$  and a bead inside a node denotes a node marginal  $q(s_a)$ . An empty bead will denote the default normalisation constraints while a connection between beads indicates marginalisation constraints following Eq. (5.9). These beads form the basic building blocks of our notation.

To write the objective corresponding to the model in Fig. 5.3 given the default constraints of Eq. (5.9), we add beads for every term and extend edges through the node boundary to connect variables that are under marginalisation constraints as shown in Fig. 5.4.

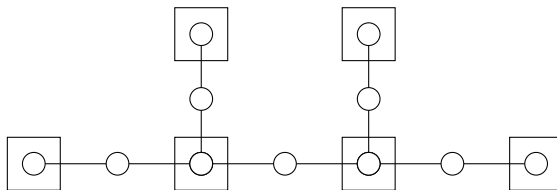


Figure 5.4: Example CFFG with normalisation and marginalisation constraints.

### 5.5.1 Factorisation Constraints

We will now extend our notation with the most common types of constraints used for defining  $\mathcal{Q}$ . A common choice is factorisation of the variational distribution with the most well-known example being the naive mean field approximation. Under a naive mean field factorisation all marginals are considered independent. Formally this means we enforce

$$q(\mathbf{s}_a) = \prod_{i \in \mathcal{E}(a)} q(s_i). \quad (5.26)$$

To write Eq. (5.26) on a CFFG we need to replace the joint node marginal with the product of adjacent edge marginals.

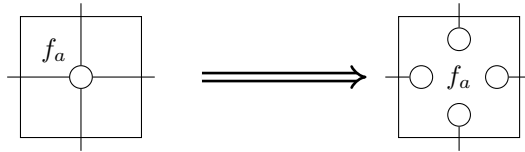


Figure 5.5: Changing a local joint factorisation to a naive mean field assumption on a CFFG.

To do this we can replace the bead indicating the joint marginal with a bead for each edge marginal in Eq. (5.26) as shown in Fig. 5.5.

The naive mean field is the strongest factorisation possible. It is possible to utilise less aggressive factorisations by appealing to a structured mean field approximation instead. The structured mean field constraint takes the form

$$q(\mathbf{s}_a) = \prod_{n \in l(a)} q^n(\mathbf{s}_a^n) \quad (5.27)$$

where  $l(a)$  denotes a set of one or more edges connected to the node  $a$  such that each element in  $\mathcal{E}(a)$  can only appear in  $l(a)$  once [116]. For example if  $\mathcal{E}(a) = \{i, j, k\}$  corresponding to variables  $\{x, y, z\}$ , we can factorise  $q(x, y, z)$  as  $q(x)q(y, z)$  or  $q(z)q(x, y)$  but not as  $q(x, y)q(y, z)$  since  $y$  appears twice. The naive mean field is a special case of the structured mean field where every variable appears only by itself.

To write a structured mean field factorisation on a CFFG we can apply a similar logic and replace the single bead denoting the joint with beads that match the structure of  $l(a)$ . Each set of variables that are factorised together corresponds to a single bead connected to the edges in the set that factor together.

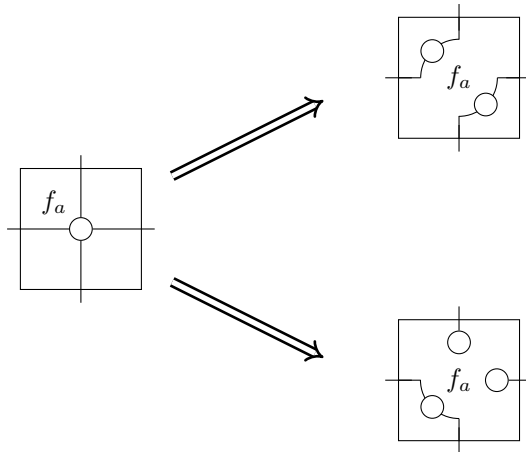


Figure 5.6: Changing a local joint factorisation to structured mean field on a CFFG.

Fig. 5.6 shows two example factorisations. The first option factorises the four incoming edges into two sets of two while the second partitions the incoming edges into two sets of one and a single set of two. Using these principles it is possible to specify complex factorisation constraints as part of the CFFG by augmenting each node on the original FFG.

The final situation we need to consider is the case when a single node has a variable or very high number of incoming edges. An example could be a Gaussian Mixture Model with a variable number of mixture components. On a CFFG we indicate variable or large numbers of identical edges by drawing two of the relevant edges and separating them with dots ( $\dots$ ) as shown in Fig. 5.7. To indicate factorisation constraints we can write either a joint or a naive mean field factorisation between the two edges, letting the dots denote that a similar factorisation applies to the remaining edges.



Figure 5.7: Mean field (left) and joint (right) factorisation constraints for variable numbers of edges on a CFFG.

### 5.5.2 Form Constraints

We will now extend CFFG notation with constraints on the functional form of nodes and edges. Form constraints are used to enforce a particular form for a local marginal on either an edge or a node. For an edge  $s_i$  they enforce

$$\int q(\mathbf{s}_a) d\mathbf{s}_{a \setminus i} = q(s_i) = g(s_i) \quad (5.28)$$

where  $g(s_i)$  denotes the functional form we are constraining the edge marginal  $s_i$  to take. Form constraints on node marginals take the form

$$q(\mathbf{s}_a) = g(\mathbf{s}_a). \quad (5.29)$$

Conventionally FFGs denote the form of a factor by a symbol inside the node. We adopt a similar convention to denote form constraints on  $q$  by adding symbols within the corresponding beads. For instance, we can indicate a Gaussian form constraint on an edge as shown in Fig 5.8

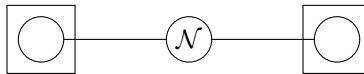


Figure 5.8: Notation for enforcing a Gaussian form constraint on an edge.

Note that this is not dependent on the form of the neighbouring factors. This is a subtle point as it allows us to write approximations into the specification of  $Q$ . As an example, the unconstrained marginal in Fig. 5.8 might be bimodal or highly skewed but by adding a form constraint, we are enforcing a Gaussian approximation. Outside of a few special cases, enforcing form constraints on edges is rarely done in practice since the functional form of  $q$  most often follows from optimisation [116].

A special case of form constraints is the case of dangling edges (edges that are not terminated by a factor node). Technically these would not warrant a bead since they would not appear explicitly in the BFE due to having degree 1. Intuitively this means that the edge marginal is only counted once and we therefore do not need to correct for overcounting. However without a bead, there is nowhere to annotate a form constraint which is problematic.

The solution for CFFG notation is to simply draw the bead anyway, in case a form constraint is needed. This is formally equivalent to terminating the dangling edge by a factor node with the node function  $f_a(s_a) = 1$ . Terminating the edge in this way means the edge in question now has degree 2 and therefore warrants a bead. This is always a valid move since multiplication by 1 does not change the underlying function [116].

We can denote form constraints on node marginals in the same manner as edge marginals. We show an example in Fig. 5.9 where we enforce a Gaussian form constraint on one node marginal and a Wishart on the other. Again it is important to note that these are constraints on  $q$  and not part of the underlying model specification  $f$ .

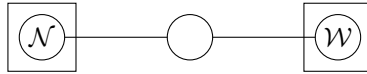


Figure 5.9: Notation for enforcing form constraints on nodes.

Two kinds of form constraints warrant extra attention:  $\delta$ -constraints and moment matching. We will now deal with these in turn.

### 5.5.3 $\delta$ -constraints and Data Points

$\delta$ -constraints are the most commonly used form constraints because they allow us to incorporate data points into a model. A  $\delta$ -constraint on an edge defines the function  $g(s_i)$  in Eq. (5.28) to be

$$g(s_i) = \delta(s_i - \hat{s}_i) \quad (5.30)$$

What makes the  $\delta$ -constraint special is that  $\hat{s}_i$  can either be a known value or a parameter to optimise [116]. In the case where  $\hat{s}_i$  is known, it commonly corresponds to a data point. We will refer to this case as a data constraint and denote it with a filled circle as shown in Fig 5.10





Figure 5.10: Terminating and non-terminating notation for data constraints.

Data constraints are special because they denote observations. They also block any information flow across the edge in question [116]. Because they block information flow, CFFG notation optionally allows data constraints to terminate edges.

Here we wish to raise a subtle point about prior FFG notation. Previous work has used small black *squares* to denote data constraints following [92]. In keeping with our convention, a small black square on a CFFG denotes a  $\delta$ -distributed variable in the model  $f$  rather than the variational distribution  $q$ . Being able to differentiate data-constrained variables in  $q$  and apriori fixed parameter of the model  $f$  allows us to be explicit about what actually constitutes a data point for the inference problem at hand [14].

In the case where  $\hat{s}_j$  is not known, it can be treated as a parameter to be optimised. We refer to this case as a  $\delta$ -constraint or a pointmass constraint and notate it with an unfilled circle as shown in Fig 5.11

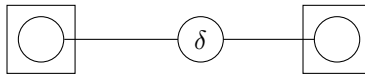


Figure 5.11: Notation for  $\delta$ -constraints.

Unlike data constraints, the  $\delta$ -constraint allows messages to pass and is therefore not allowed to terminate an edge. Optimising the value of  $\hat{s}_i$  under a  $\delta$ -constraint leads to EM as message passing [116].

### 5.5.4 Moment Matching Constraints

Moment matching constraints are special in that they replace the hard marginalisation constraints of Eq. (5.8) with constraints of the form

$$q(s_i) = \int q(\mathbf{s}_a) ds_{a \setminus i} \implies \int q(\mathbf{s}_a) \mathbf{T}_i(s_i) ds_a = \int q(s_i) \mathbf{T}_i(s_i) ds_i \quad (5.31)$$

where  $\mathbf{T}_i(s_i)$  are the sufficient statistics of an exponential family distribution. This move loosens the marginalisation constraint by instead only requiring

that the moments in question align. When taking the first variation and solving, one obtains the expectation propagation (EP) algorithm [75].

For notational purposes, moment matching constraints are unique in that they involve both an edge- and a node-marginal. That means the effects are not localised to a single bead. To indicate which beads are involved, we replace the solid lines between them with dashed lines instead.

We denote moment matching constraints by an  $\mathbb{E}$  inside the corresponding edge-bead as shown in Fig 5.12. Choosing the edge-bead over the node-bead is an arbitrary decision made mainly for convenience.



Figure 5.12: Notation for moment matching with a single-sided (left) and double-sided (right) node/edge pairs.

The left side of Fig. 5.12 shows notation for constraining a single node/edge pair by moment matching. If both nodes connected to an edge are under moment matching constraints, the double-sided notation on the right of Fig. 5.12 applies.

Given the modular nature of CFFG notation it is easy to compose different local constraints to accurately specify a Lagrangian and by extension an inference problem. Adding custom marginal constraints to a CFFG is also straightforward as it simply requires defining the meaning of a symbol inside a bead.

### 5.5.5 P-substitution on CFFGs

The final piece needed to represent the Active Inference Lagrangian on a CFFG is P-substitution. Being able to represent LAIF on a CFFG is the reason for constructing the local GFE using a meanfield factorisation and P-substitution. This construction is much more amenable to the tools we have developed so far as we will now demonstrate.

Recall that P-substitution involves substituting part of the model  $p$  for  $q$  in the expectation only. To write P-substitution on a CFFG, the logical notation is therefore to replace a circle with a square. Fig. 5.13 shows an example of adding a P-substitution to a meanfield factorised node marginal

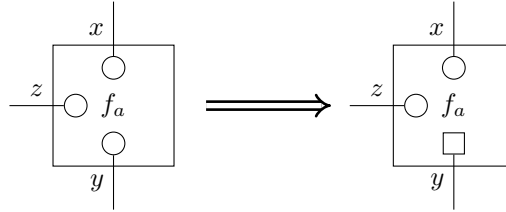


Figure 5.13: P-substitution on a CFFG with naive mean field factorisation.

The square notation for P-substitution on CFFGs implies a conditioning of the P-substituted variable on all other connected variables that are not P-substituted. For example, in Fig. 5.13 the P-substitution changes local VFE to a GFE by

$$\begin{aligned} & \iiint q(y)q(x)q(z) \log \frac{q(y)q(x)q(z)}{p(y, x, z)} dy dx dz \\ \implies & \iiint p(y | x, z) q(x)q(z) \log \frac{q(y)q(x)q(z)}{p(y, x, z)} dy dx dz \end{aligned} \quad (5.32)$$

Here the P-substituted variable is  $y$  and the remainder are  $\{x, z\}$ .

5

### 5.5.6 CFFG Compression

The value of CFFG notation is measured by how much it aids other researchers and practitioners in expressing their ideas accurately and succinctly. We envision two main groups for whom CFFGs might be of particular interest. The first group is comprised of mathematical researchers working on constrained free energy optimisation on FFGs. For this group we expect that the notation developed so far will be both useful and practically applicable since work is often focused on the intricacies of performing local optimisation. Commonly an FFG in this tradition is small but a very high level of accuracy is desired in order to be mathematically rigorous.

However there is a second group composed of applied researchers for whom the challenge is to accurately specify a larger inference problem and its solution in order to solve an auxiliary goal - for instance controlling a drone, transmitting a coded message or simulating some phenomenon using AIF. For this group of researchers in particular, CFFG notation as described so far might be too verbose

and the overhead of using it may not outweigh the benefits gained. To this end, we now complete CFFG notation by a mandatory compression step.

The compression step is designed to remove redundant information by enforcing an emphasis on *deviations from a default BFE*. By default, we mean no constraints other than normalisation and marginalisation and with a joint factorisation around every node. Recall that default normalisation constraints are denoted by empty, round beads and marginalisation by connected lines. A joint factorisation means all incoming edges are connected and the node only has a single, internal bead.

To provide a recipe for compressing a CFFG, we will need the concept of a *bead chain*. A bead chain is simply a series of beads connected by edges. In the following recipe, a bead will only be summarised as part of a chain if it contains no additional information, meaning if it is round and empty. To compress a CFFG, we follow a series of four steps:

1. Summarise every bead chain by their terminating beads.
2. For nodes with no factorisation constraints, remove empty, internal beads.
3. For nodes with no factorisation constraints, remove all internal edges.
4. For all factor nodes, push the remaining internal beads to the border of the corresponding factor node.

After performing these steps, we are left with a compressed version of the original CFFG where each node can be much smaller and more concise. To exemplify, we apply the recipe to the CFFG in Fig. 5.14.

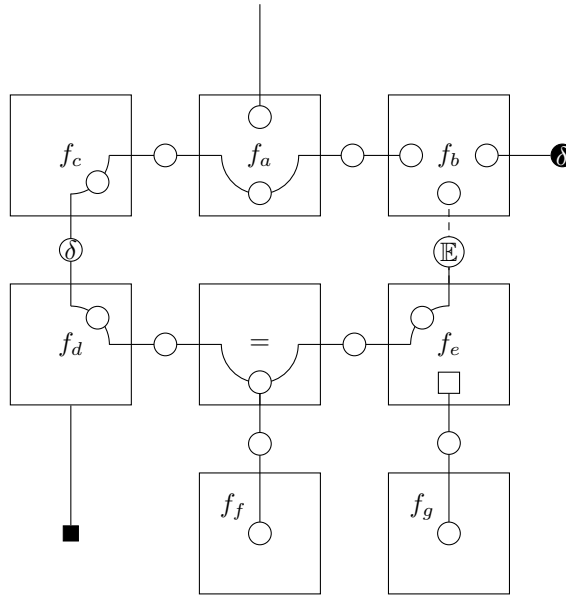


Figure 5.14: Initial CFFG before compression.

5

Fig. 5.14 is a complicated graph with loops, dangling edges, and multiple different factorisations in play. As a result, Fig. 5.14 covers a lot of special cases that one might encounter on a CFFG. We will now apply the steps in sequence, starting by removing empty beads on chains. This removes most of the beads in the inner loop as shown in Fig. 5.15

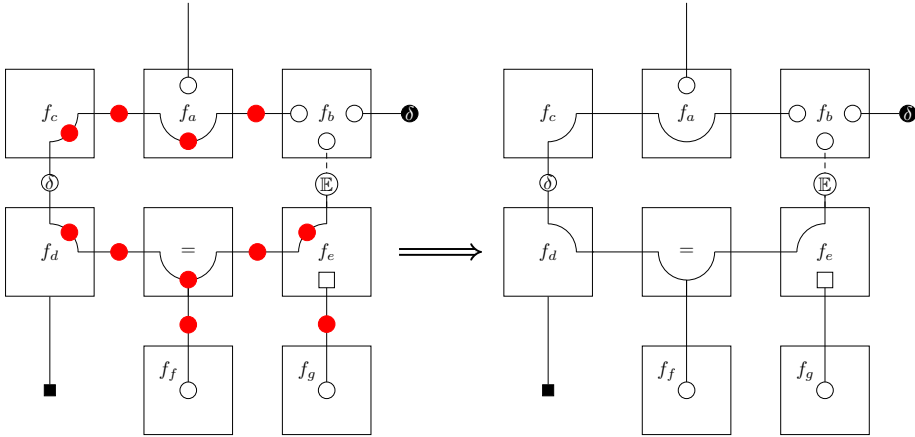


Figure 5.15: Step 1: Removing empty beads from chains. Affected beads are highlighted in red.

Note that the dangling edge extending from  $f_a$  is treated as terminated by a bead and the bead on the edge is removed. Next, we remove any internal beads for nodes with no factorisation constraints. We show this step in Fig. 5.16

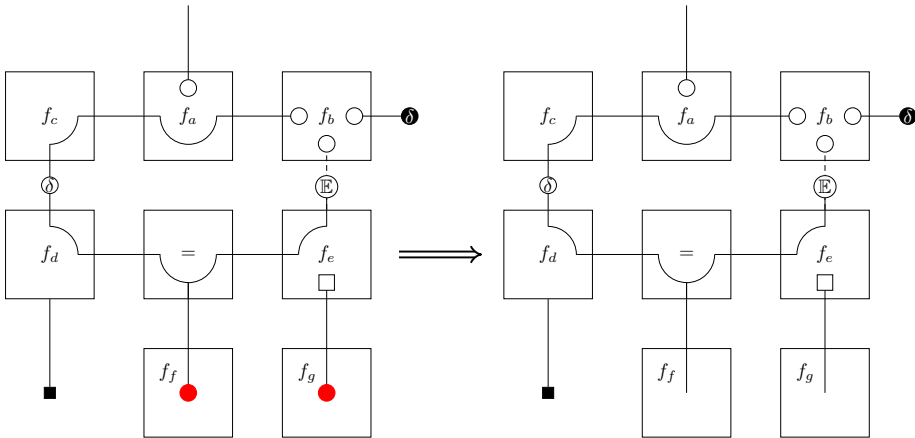


Figure 5.16: Step 2: Removing beads on nodes with default factorisation. Affected nodes are highlighted in red.

The next step is to remove any internal edge extensions for nodes with no factorisation constraints. We demonstrate this step in Fig. 5.17. Note that the internal edge of the node  $f_d$  does not get cancelled since one of the connected edges is a pointmass in the model and therefore not present in  $q$ , meaning  $f_d$  does not have a default factorisation.

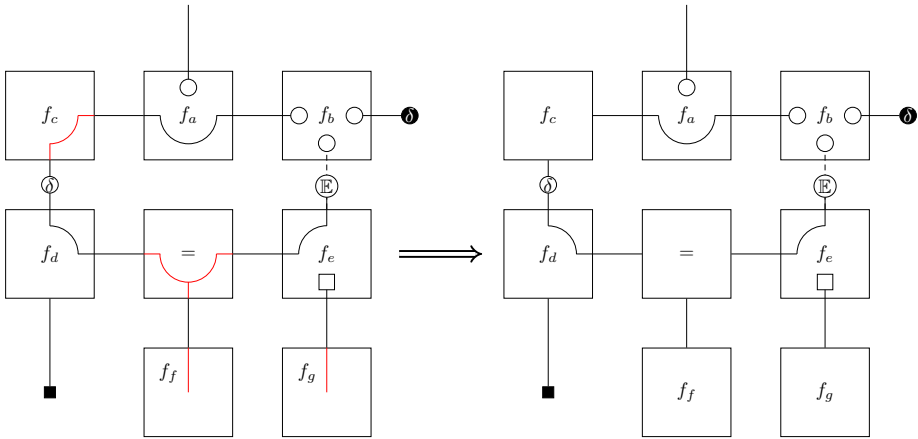


Figure 5.17: Step 3. Removing internal edges. Affected edges are highlighted in red.

Finally, we can push any remaining internal beads to their node border, illustrated in Fig. 5.18. This step allows for writing the CFFG much more compactly, as we can see in Fig. 5.19.

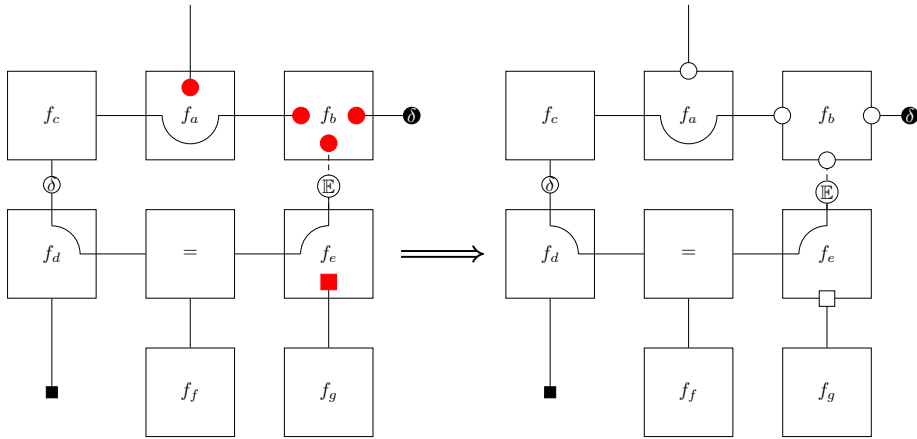


Figure 5.18: Step 4. Pushing beads to node borders. Affected beads are highlighted in red.

At this point, we have removed most of the beads and edges and still retain most of the relevant information around all nodes. What is left is only what deviates from a default BFE specification. The goal here is as stated initially to make it easier to work with CFFG's for larger models which necessitate working with smaller nodes.



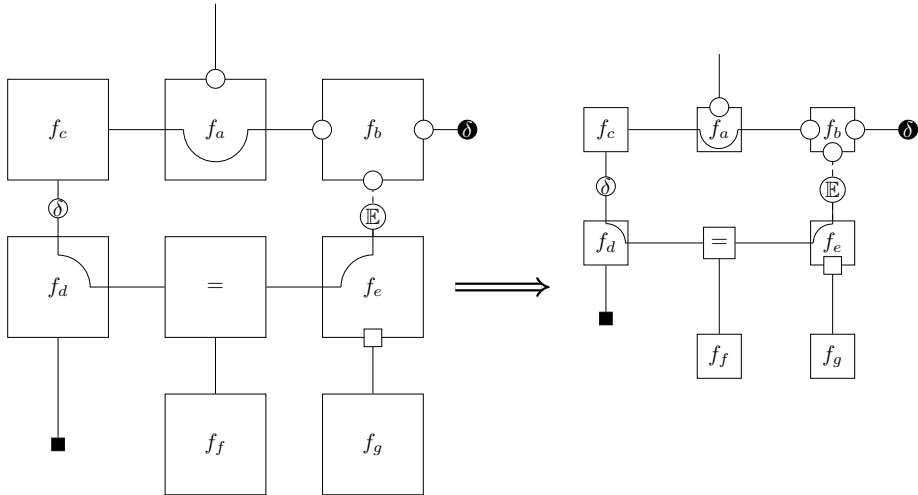


Figure 5.19: Compression allows for much more compact CFFG's which in turn are better suited for larger inference problems.

From Fig. 5.19 it is clear that compression allows for much more compact CFFG's which in turn are more amenable to larger CFFG's. This is especially true in the absence of any factorisation or form constraints and P-substitutions in which case the compressed CFFG and underlying FFG are identical. We see this with nodes  $f_c$ ,  $f_f$ ,  $f_g$  and  $=$  in Fig. 5.19.

## 5.6 General GFE-Based Message Updates

Having defined a node-local GFE as well as the necessary notation for expressing the constrained free energy optimisation problem graphically, we are now ready to derive the message update rules required to solve said optimisation problem. To adequately reproduce the behaviour of the original GFE formulation in Eq. 5.15, we define a generalised goal and observation model, which simultaneously impose dual constraints on the observation variable  $x$ . The observation model  $p(x | z, \theta)$  consists of states  $z$  and parameters  $\theta$ . The goal prior extends to a goal model  $\tilde{p}(x | w, \phi)$ , with states  $w$  and parameters  $\phi$ .

The CFFG of Fig. 5.20 draws the observation and goal model as two facing nodes. Crucially, from the perspective of the CFFG the role of these nodes in the

bigger model is irrelevant, expanding the range of applicability beyond observation and goal models. The facing nodes are contained by a composite structure that acts as a Markov blanket for communication with the remaining graph.

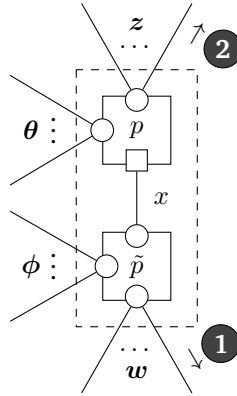


Figure 5.20: CFFG of two facing nodes with indicated p-substitution and messages.

The CFFG of Fig. 5.20 then defines the free energy objective,

$$F[q] = \mathbb{E}_{q(x,z,\theta,w,\phi)} \left[ \log \frac{q(x,z,\theta,w,\phi)}{p(x|z,\theta)\tilde{p}(x|w,\phi)} \right], \quad (5.33)$$

under local normalisation and marginalisation constraints, the imposed factorisation

$$q(x,z,\theta,w,\phi) \triangleq q(x)q(z)q(\theta)q(w)q(\phi), \quad (5.34)$$

and p-substitution of

$$q(x) \rightarrow p(x|z,\theta), \quad (5.35)$$

in the expectation term.

### 5.6.1 Local Lagrangian

After substitution of the factorisation and applying the p-substitution to the local VFE objective, we obtain the local GFE,

$$\mathcal{G}[q] = \mathbb{E}_{p(x|z,\theta)q(z)q(\theta)q(w)q(\phi)} \left[ \log \frac{q(x)q(z)q(\theta)q(w)q(\phi)}{p(x|z,\theta)\tilde{p}(x|w,\phi)} \right]. \quad (5.36)$$

To find local stationary solutions to this GFE, we introduce Lagrange multipliers that enforce the normalisation and marginalisation constraints in  $\mathcal{Q}$ . The node-local Lagrangian then becomes

$$\begin{aligned} \mathcal{L}[q] = \mathcal{G}[q] &+ \sum_{a \in \mathcal{V}} \psi_a \left( \int q(\mathbf{s}_a) d\mathbf{s}_a - 1 \right) + \sum_{i \in \mathcal{E}} \psi_i \left( \int q(s_i) ds_i - 1 \right) + \\ &\sum_{a \in \mathcal{V}} \sum_{i \in \mathcal{E}(a)} \int \lambda_{ia}(s_i) \left( q(s_i) - \int q(\mathbf{s}_a) d\mathbf{s}_{a \setminus i} \right) ds_i, \end{aligned} \quad (5.37)$$

with  $s_i \in \{x, z, \theta, w, \phi\}$  a generic variable. This Lagrangian is then optimised under a free-form variational density

$$q^* = \arg \min_{q \in \mathcal{Q}} \mathcal{L}[q], \quad (5.38)$$

for all individual factors in the variational distribution factorisation.

### 5.6.2 Local Stationary Solutions

We are now prepared to derive the stationary points of the node-local Lagrangian Eq.(5.37). We start by considering the node-local Lagrangian as a functional of the variational factor  $q_x$ .

**Lemma 1.** *Stationary points of  $\mathcal{L}[q]$  as a functional of  $q_x$ ,*

$$\mathcal{L}[q_x] = \mathcal{G}[q_x] + \psi_x \left[ \int q(x) dx - 1 \right] + C_x, \quad (5.39)$$

where  $C_x$  collects all terms independent from  $q_x$ , are given by

$$q^*(x) = \mathbb{E}_{q(\mathbf{z})q(\boldsymbol{\theta})}[p(x | \mathbf{z}, \boldsymbol{\theta})] . \quad (5.40)$$

*Proof.* The proof is given by Appendix C.3.1.  $\square$

Next, we derive the stationary points of Eq.(5.37) as a functional of  $q_{\mathbf{z}}$ . Note that, by symmetry, a similar result applies to  $q_{\boldsymbol{\theta}}$ .

**Lemma 2.** *Stationary points of  $\mathcal{L}[q]$  as a functional of  $q_{\mathbf{z}}$ ,*

$$\begin{aligned} \mathcal{L}[q_{\mathbf{z}}] = & \mathcal{G}[q_{\mathbf{z}}] + \psi_{\mathbf{z}} \left[ \int q(\mathbf{z}) d\mathbf{z} - 1 \right] \\ & + \sum_{i \in \mathcal{E}(\mathbf{z})} \int \lambda_{ip}(z_i) \left[ q(z_i) - \int q(\mathbf{z}) d\mathbf{z}_{\setminus i} \right] dz_i + C_{\mathbf{z}} , \end{aligned} \quad (5.41)$$

where  $C_{\mathbf{z}}$  collects all terms independent from  $q_{\mathbf{z}}$ , are given by

$$q^*(\mathbf{z}) = \frac{\tilde{f}(\mathbf{z}) \prod_{z_i \in \mathbf{z}} \tilde{\mu}(z_i)}{\int \tilde{f}(\mathbf{z}) \prod_{z_i \in \mathbf{z}} \tilde{\mu}(z_i) d\mathbf{z}} , \quad (5.42)$$

with

$$\tilde{f}(\mathbf{z}) = \exp \left( \mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\boldsymbol{\theta})} \left[ \log \frac{p(x | \mathbf{z}, \boldsymbol{\theta}) \tilde{f}(x)}{q(x)} \right] \right) \quad (5.43a)$$

$$\tilde{f}(x) = \exp(\mathbb{E}_{q(\mathbf{w})q(\boldsymbol{\phi})}[\log \tilde{p}(x | \mathbf{w}, \boldsymbol{\phi})]) . \quad (5.43b)$$

*Proof.* The proof is given by Appendix C.3.2.  $\square$

Finally, we derive the stationary points of Eq.(5.37) with respect to  $q_{\mathbf{w}}$ . Again, by symmetry, a similar result follows for  $q_{\boldsymbol{\phi}}$ .

**Lemma 3.** *Stationary points of  $\mathcal{L}[q]$  as a functional of  $q_{\mathbf{w}}$ ,*

$$\begin{aligned} \mathcal{L}[q_{\mathbf{w}}] &= \mathcal{G}[q_{\mathbf{w}}] + \psi_{\mathbf{w}} \left[ \int q(\mathbf{w}) d\mathbf{w} - 1 \right] \\ &+ \sum_{i \in \mathcal{E}(\mathbf{w})} \int \lambda_i \tilde{p}(w_i) \left[ q(w_i) - \int q(\mathbf{w}) d\mathbf{w}_{\setminus i} \right] dw_i + C_{\mathbf{w}}, \end{aligned} \quad (5.44)$$

where  $C_{\mathbf{w}}$  collects all terms independent from  $q_{\mathbf{w}}$ , are given by

$$q^*(\mathbf{w}) = \frac{\tilde{f}(\mathbf{w}) \prod_{w_i \in \mathbf{w}} \tilde{\mu}(w_i)}{\int \tilde{f}(\mathbf{w}) \prod_{w_i \in \mathbf{w}} \tilde{\mu}(w_i) d\mathbf{w}}, \quad (5.45)$$

with

$$\tilde{f}(\mathbf{w}) = \exp(\mathbb{E}_{q(x)q(\phi)}[\log \tilde{p}(x | \mathbf{w}, \phi)]). \quad (5.46)$$

*Proof.* The proof is given by Appendix C.3.3.  $\square$

## 5

### 5.6.3 Message Updates

In this section, we show that the stationary solutions of Section 5.6.2 correspond to the fixed points of a fixed-point iteration scheme. The corresponding messages are shown in Fig. 5.20. We first derive the update rule for message ① on  $w_j \in \mathbf{w}$ . By symmetry, a similar result applies to  $\phi_j \in \phi$ .

**Theorem 1.** *Given the stationary points of the node-local Lagrangian  $\mathcal{L}[q]$ , the stationary message  $\tilde{\mu}(w_j)$  corresponds to a fixed point of the iterations*

$$\tilde{\mu}^{(n+1)}(w_j) = \int \tilde{f}(\mathbf{w}) \prod_{\substack{w_i \in \mathbf{w} \\ w_i \neq w_j}} \tilde{\mu}^{(n)}(w_i) d\mathbf{w}_{\setminus j}, \quad (5.47)$$

with  $n$  an iteration index, and  $\tilde{f}(\mathbf{w})$  given by Eq. (5.46).

*Proof.* The proof is given by Appendix C.4.1.  $\square$

We now derive the update rule for message ② on  $z_j \in \mathbf{z}$  (Fig. 5.20), where we apply the same procedure as before. By symmetry, a similar result applies to  $\theta_j \in \boldsymbol{\theta}$ .

**Theorem 2.** *Given the stationary points of the node-local Lagrangian  $\mathcal{L}[q]$ , the stationary message  $\bar{\mu}(z_j)$  corresponds to a fixed point of the iterations*

$$\bar{\mu}^{(n+1)}(z_j) = \int \tilde{f}(\mathbf{z}) \prod_{\substack{z_i \in \mathbf{z} \\ z_i \neq z_j}} \bar{\mu}^{(n)}(z_i) d\mathbf{z}_{\setminus j}, \quad (5.48)$$

with  $n$  an iteration index, and  $\tilde{f}(\mathbf{z})$  given by Eq.(5.43),

*Proof.* The proof is given by Appendix C.4.2. □

#### 5.6.4 Convergence Considerations

While direct application of Eq.(5.48) works well in some cases, this message update may also yield algorithms for which the GFE actually diverges over iterations. This perhaps counter-intuitive effect then has major implications for the practical implementation of Eq.(5.48).

This divergence issue relates to a subtlety about what is actually proven by our theorems. While our theorems prove that the stationary messages correspond to fixed-points of the node-local Lagrangian, the theorems do not guarantee that iterations of the fixed-point equations actually converge to said fixed-points. In order to guarantee that updates approach the fixed-points, we derive an alternative message update rule for message ② in Fig. 5.21.

**Corollary 1.** *Given the stationary points of the node-local Lagrangian  $\mathcal{L}[q]$ , the stationary message  $\bar{\mu}(z_j)$  corresponds to*

$$\bar{\mu}(z_j) \propto \frac{\int q(\mathbf{z}; \nu^*) d\mathbf{z}_{\setminus j}}{\bar{\mu}(z_j)}, \quad (5.49)$$

with  $\nu^*$  a solution to

$$q(\mathbf{z}; \nu) \stackrel{!}{=} \frac{\tilde{f}(\mathbf{z}; \nu) \prod_{z_i \in \mathbf{z}} \tilde{\mu}(z_i)}{\int \tilde{f}(\mathbf{z}; \nu) \prod_{z_i \in \mathbf{z}} \tilde{\mu}(z_i) d\mathbf{z}}, \quad (5.50)$$

where  $q_{\mathbf{z}}$  is parameterised by statistics  $\nu$ , and where  $\tilde{f}(\mathbf{z}; \nu)$  is given by Eq.(5.43), with

$$q(x; \nu) = \mathbb{E}_{q(\mathbf{z}; \nu) q(\boldsymbol{\theta})} [p(x | \mathbf{z}, \boldsymbol{\theta})], \quad (5.51)$$

which is parameterised by  $\nu$  through a recursive dependence on  $q_{\mathbf{z}}$ .

*Proof.* The proof is given by Appendix C.4.3. □

The result of Corollary 1 avoids the possibly diverging fixed-point iterations of Theorem 2. For example, the optimal parameters  $\nu^*$  can now be found through Newton's method.

## 5

## 5.7 Application to a Discrete-Variable Model

In this section, we apply the general message update rules of Section 5.6.3 to a discrete-variable model that is often used in AIF practice. Using the general results we derive messages for this specific model.

### 5.7.1 Goal-Observation Submodel

We consider a discrete state  $z \in \mathcal{Z}$  and observation variable  $x \in \mathcal{X}$ . To conveniently model these variables with categorical distributions, we convert them to a one-hot representation, with  $\mathbf{x} = \mathbf{e}_i$ , the standard unit vector on  $\mathcal{X}$  with  $x_i = 1$  at the index for  $x$ , and 0 otherwise (and similar for  $\mathbf{z}$ ). The notation  $Cat(\mathbf{x} | \boldsymbol{\rho}) = \prod_i \rho_i^{x_i}$  then represents the categorical distribution on  $\mathbf{x}$  (one-hot) with probability vector  $\boldsymbol{\rho}$ . We relate the state and observation variables by transition probability matrix  $\mathbf{A} \in \mathcal{X} \times \mathcal{Z}$ . The columns of  $\mathbf{A}$  are normalised such that  $\mathbf{A}\mathbf{z}$  represents a probability vector. With notation in place, we define the observation model and goal prior for the constrained submodel,

$$p(\mathbf{x} \mid \mathbf{z}, \mathbf{A}) = \text{Cat}(\mathbf{x} \mid \mathbf{A}\mathbf{z}) \quad (5.52a)$$

$$\tilde{p}(\mathbf{x} \mid \mathbf{c}) = \text{Cat}(\mathbf{x} \mid \mathbf{c}), \quad (5.52b)$$

as shown in Fig. 5.21 (left).

### 5.7.2 GFE-Based Message Updates

The CFFG of Fig. 5.21 (left) defines the local GFE objective, with an incoming message  $\mu_{\textcircled{2}}(\mathbf{z}) = \text{Cat}(\mathbf{z} \mid \mathbf{d})$ . We denote by

$$\mathbf{h}(\mathbf{A}) = -\text{diag}(\mathbf{A}^T \log \mathbf{A}), \quad (5.53)$$

the vector of entropies of the columns of the conditional probability matrix  $\mathbf{A}$ .

As a notational convention in this context, we use an over-bar shorthand to denote an expectation,  $\bar{\mathbf{z}} = \mathbb{E}_{q(\mathbf{z})}[\mathbf{z}]$ . The table in Fig. 5.21 (right) summarises the resulting message updates and average energy, with

$$\xi(\mathbf{A}) = \mathbf{A}^T (\overline{\log \mathbf{c}} - \log(\overline{\mathbf{A}\bar{\mathbf{z}}})) - \mathbf{h}(\mathbf{A}) \quad (5.54a)$$

$$\rho = \overline{\mathbf{A}^T (\log \mathbf{c} - \log(\overline{\mathbf{A}\bar{\mathbf{z}}}))} - \overline{\mathbf{h}(\mathbf{A})}. \quad (5.54b)$$

The full derivations are available in Appendix C.5.

Fig. 5.21 shows the message updates towards all connected variables as well as the average energy term ( $U_x$ ) for the composite node.  $\mathbf{d}$  denotes the parameters of the incoming message  $\mu_{\textcircled{2}}(\mathbf{z})$ . Interestingly, the energy term  $U_x$  corresponds exactly to the expected free energy as used in standard AIF [22, 33].

The message  $\textcircled{3}$  does not follow a standard exponential family distribution as a function of  $\mathbf{A}$ . To circumvent this problem, we can pass on the logpdf directly as a message. When we need to compute the marginal  $q(\mathbf{A})$ , we can then use sampling procedures to estimate the necessary expectations [1].

## 5.8 Classical AIF and the Original GFE Algorithm as Special Cases of LAIF

An integral part of working with MP algorithms is the choice of schedule - the order in which messages are passed. Many MP algorithms are iterative in na-



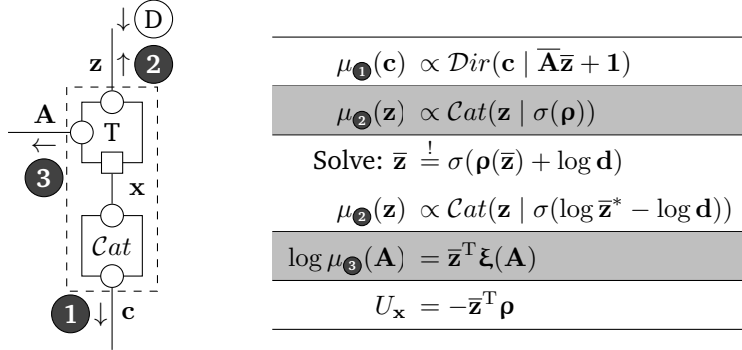


Figure 5.21: CFFG of the discrete-variable submodel (left) and message updates (right), with  $\sigma$  a softmax function. Updates for message two indicate the direct and indirect computation respectively.

ture and can therefore be sensitive to scheduling. LAIF is also an iterative MP algorithm and might consequently be sensitive to the choice of schedule.

A particularly interesting observation is that we can recover both the classical AIF planning algorithm of [33] and the scheme proposed by [81] as special cases of LAIF by carefully choosing the schedule and performing model comparison. This result extends upon prior work by [59] who reinterpreted the classical algorithm through the lens of MP on an FFG.

With our newly derived messages for the goal-constrained composite node in hand, we can now restate prior work unambiguously using CFFGs, starting with the classical algorithm of [33]. To reconstruct the algorithm of [33], we start by defining the generative model. The generative model is a discrete partially observed Markov decision process (POMDP) over future time steps given by

$$\begin{aligned}
 & p(\underbrace{\mathbf{x}_{t+1:T}, \mathbf{z}_{t:T}}_{\text{Future}} \mid \underbrace{\hat{\mathbf{u}}_{t+1:T}}_{\text{Policy}}, \underbrace{\mathbf{x}_{1:t}, \mathbf{u}_{1:t}}_{\text{Past}}) \\
 & \propto \underbrace{p(\mathbf{z}_t \mid \mathbf{x}_{1:t}, \mathbf{u}_{1:t})}_{\text{State Prior}} \prod_{k=t+1}^T \underbrace{p(\mathbf{x}_k \mid \mathbf{z}_k)}_{\text{Likelihood}} \underbrace{p(\mathbf{z}_k \mid \mathbf{z}_{k-1}, \hat{\mathbf{u}}_k)}_{\text{State Transition}} \underbrace{\tilde{p}(\mathbf{x}_k)}_{\text{Goal prior}}
 \end{aligned}
 \tag{5.55}$$

where

5

$$\begin{aligned}
 p(\mathbf{z}_t \mid \mathbf{x}_{1:t}, \mathbf{u}_{1:t}) &= \text{Cat}(\mathbf{z}_t \mid \mathbf{d}) \\
 p(\mathbf{z}_k \mid \mathbf{z}_{k-1}, \hat{\mathbf{u}}_k) &= \text{Cat}(\mathbf{z}_k \mid \mathbf{B}_{\hat{\mathbf{u}}_k} \mathbf{z}_{k-1}) \\
 p(\mathbf{x}_k \mid \mathbf{z}_k) &= \text{Cat}(\mathbf{x}_k \mid \mathbf{A} \mathbf{z}_k) \\
 \tilde{p}(\mathbf{x}_k) &= \text{Cat}(\mathbf{x}_k \mid \mathbf{c}_k).
 \end{aligned}
 \tag{5.56}$$

Here we let  $\mathbf{x}_k$  denote observations,  $\mathbf{z}_k$  latent states, and  $\hat{\mathbf{u}}_k$  a fixed control, with the subscript  $k$  indicating the future time step in question. The initial state  $\mathbf{z}_t$  represents the filtering solution given the agents trajectory so far which we summarise in the parameter vector  $\mathbf{d}$ . Control signals correspond to particular transition matrices  $\mathbf{B}_{\hat{\mathbf{u}}_k}$  where we use the subscript to emphasise that each  $\mathbf{B}$  matches a particular control  $\hat{\mathbf{u}}_k$ . The observation model is given by the known matrix  $\mathbf{A}$ . Note that Eq. (5.55) is not normalised as it includes goal priors over  $\mathbf{x}$ . The CFFG of (5.55) is shown in Fig 5.22.

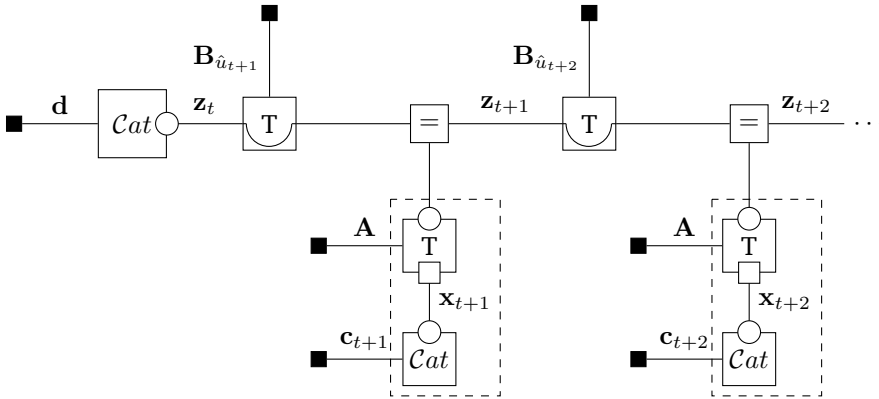


Figure 5.22: CFFG of discrete POMDP as used for planning in standard AIF models.

where  $T$  nodes denote a discrete state transition (multiplication of a categorical variable by a transition matrix). Given a generative model with a fixed set of controls, the next step is to compute the EFE [22, 33, 59, 60]. We provide a brief description here and refer to Appendix C.1 and [22, 59, 60, 97] for more detailed descriptions. The EFE is given by

$$G(\hat{\mathbf{u}}_{t+1:T}) = \sum_{k=t}^T \iint p(\mathbf{x}_k | \mathbf{z}_k) q(\mathbf{z}_k | \hat{\mathbf{u}}_k) \log \frac{q(\mathbf{z}_k | \hat{\mathbf{u}}_k)}{p(\mathbf{x}_k, \mathbf{z}_k | \hat{\mathbf{u}}_k)} d\mathbf{x}_k d\mathbf{z}_k \quad (5.57)$$

With a slight abuse of notation, we can compute EFE by first applying transition matrices to the latent state and generating predicted observations as

$$\begin{aligned} \mathbf{z}_k &= \mathbf{B}_{\hat{\mathbf{u}}_k} \mathbf{z}_{k-1} \\ \mathbf{x}_k &= \mathbf{A} \mathbf{z}_k . \end{aligned} \quad (5.58)$$

In Eq. (5.58) we slightly abuse notation by having  $\mathbf{z}_k$  (resp.  $\mathbf{x}_k$ ) refer to the prediction after applying the transition matrix  $\mathbf{B}_{\hat{\mathbf{u}}_k}$  (resp.  $\mathbf{A}$ ) instead of the random variable as we have done elsewhere.

We can recognise these operations as performing a forwards MP sweep using BP messages. With this choice of generative model, the EFE of a policy  $\hat{\mathbf{u}}_{t+1:T}$  is found by [22, Eq. D.2-3].

$$G(\hat{\mathbf{u}}_{t+1:T}) = \sum_{k=t+1}^T -\text{diag}(\mathbf{A}^T \log \mathbf{A})^T \mathbf{z}_k + \mathbf{x}_k^T (\log \mathbf{x}_k - \log \mathbf{c}_k) \quad (5.59)$$

where  $\mathbf{c}_k$  denotes the parameter vector of a goal prior at the  $k$ 'th time step. To select a policy we simply pick the sequence  $\hat{\mathbf{u}}_{t+1:T}$  that results in the lowest numerical value when solving Eq. (5.59).

To write this method on the CFFG in Fig. 5.22, we can simply add messages to the CFFG and note that the sum of energy terms of the P-substituted composite nodes in Fig. 5.21 matches Eq. (5.59). We show this result in Fig. 5.23.

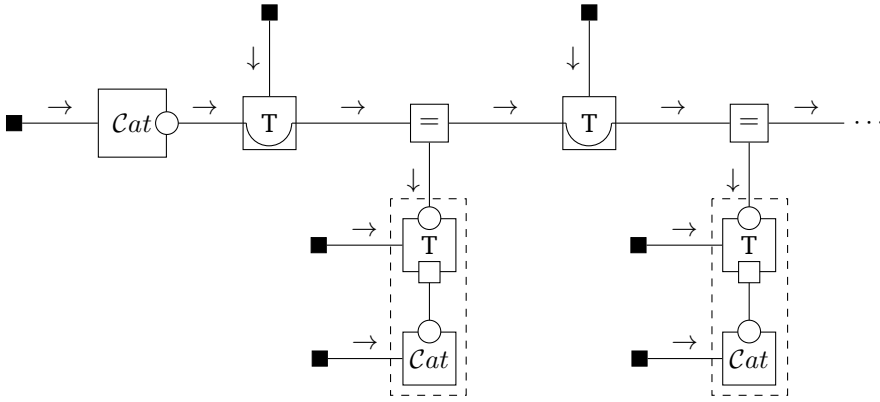


Figure 5.23: Classical EFE computation on the corresponding CFFG.

Comparing different policies (choices of  $\hat{\mathbf{u}}_{t+1:T}$ ) and computing the energy terms of the P-substituted composite nodes is then exactly equal to the EFE-computation detailed in [33].

### 5.8.1 Reconstructing the Original GFE Method

We can further exemplify the capabilities of CFFG notation by recapitulating the update rules given in the original GFE paper [81]. To recover the procedure of [81], we need to extend the generative model to encompass past observations as

$$\begin{aligned}
 & p(\mathbf{x}_{1:T}, \mathbf{z}_{0:T} \mid \hat{\mathbf{u}}_{1:T}) \\
 & \propto p(\mathbf{z}_0) \underbrace{\prod_{l=1}^t p(\mathbf{x}_l \mid \mathbf{z}_l) p(\mathbf{z}_l \mid \mathbf{z}_{l-1}, \hat{\mathbf{u}}_l)}_{\text{Past}} \underbrace{\prod_{k=t+1}^T p(\mathbf{x}_k \mid \mathbf{z}_k) p(\mathbf{z}_k \mid \mathbf{z}_{k-1}, \hat{\mathbf{u}}_k)}_{\text{Future}} \overbrace{\tilde{p}(\mathbf{x}_k)}^{\text{Goal prior}}
 \end{aligned}
 \tag{5.60}$$

where

$$\begin{aligned}
 p(\mathbf{z}_0) &= \text{Cat}(\mathbf{z}_0 \mid \mathbf{d}) \\
 p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \hat{\mathbf{u}}_t) &= \text{Cat}(\mathbf{z}_t \mid \mathbf{B}_{\hat{\mathbf{u}}_t} \mathbf{z}_{t-1}) \\
 p(\mathbf{x}_t \mid \mathbf{z}_t) &= \text{Cat}(\mathbf{x}_t \mid \mathbf{A} \mathbf{z}_t) \\
 \tilde{p}(\mathbf{x}_k) &= \text{Cat}(\mathbf{x}_k \mid \mathbf{c}_k).
 \end{aligned}
 \tag{5.61}$$

For past time steps, we add data constraints and for future time steps we perform P-substitution. We also apply a naive mean field factorisation for every node. The final ingredient we need is a schedule that includes both forwards and backwards passes as hinted at in [59]. For  $t = 1$  the corresponding CFFG is shown in Fig. 5.24 with messages out of the P-substituted nodes highlighted in red. These messages are given by  $\mu_{\otimes}(\mathbf{z})$  in Fig. 5.21.

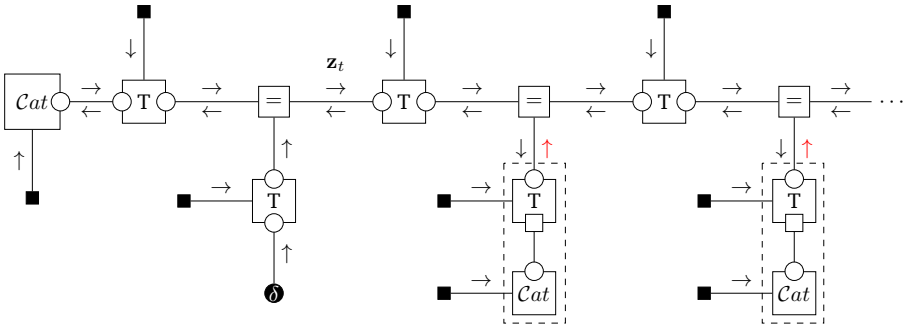


Figure 5.24: GFE computation on the CFFG.

With these choices, the update equations become identical to those of [81]. Indeed, careful inspection of the update rules given in [81] reveals the component parts of the P-substituted message. However, by using CFFGs and P-substitution, we can cast their results as MP on generic graphs which immediately generalises their results to free form graphical models.

Once inference has converged, we note that [81] shows that GFE evaluates identically to the EFE, meaning we can use the same model comparison procedure to select between policies as we used for reconstructing the classical algorithm. This shows how we can obtain the algorithm of [81] as a special case of LAIF.

## 5.9 LAIF for Policy Inference

The tools presented in this chapter are not limited to restating prior work. Indeed, LAIF offers several advantages over prior methods, one of which is the ability to directly infer a policy instead of relying on a post hoc comparison of energy terms. To demonstrate, we solve two variations of the classic T-maze task [33]. This is a well-studied setting within the AIF literature and therefore constitutes a good minimal benchmark. In the T-maze experiment, the agent lives in a maze with four locations as depicted in Fig. 5.25. The agent (☺) starts in position 1 and knows that a reward is present at either position 2 or 3, but not which one. At position 4 is a cue that informs the agent which arm contains the reward. The optimal action to take is therefore to first visit 4 and learn which arm contains the reward before going to the rewarded arm. Because this course of action requires delaying the reward, an agent following a greedy policy behaves sub-optimally. The T-maze is therefore considered a reasonable minimal example of the epistemic, information-seeking behaviour that is a hallmark of AIF agents. We implemented our experiments in the reactive MP toolbox `RxInfer` [3]. The source code for our simulations is available at <https://github.com/biaslab/LAIF>.

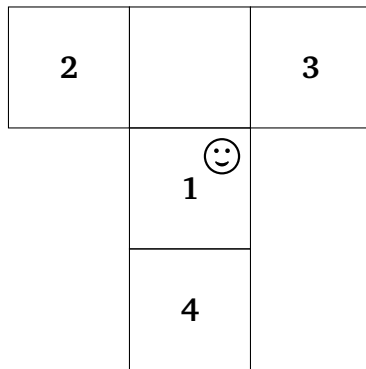


Figure 5.25: The T-maze environment

### 5.9.1 Model Specification

The generative model for the T-maze is an adaptation of the discrete POMDP used by [33, 81]. We assume the agent starts at some current time step  $t$  and

wants to infer a policy up to a known time horizon  $T$ . The generative model is then

$$p(\mathbf{x}, \mathbf{z}, \mathbf{u}) \propto \underbrace{p(\mathbf{z}_t)}_{\text{Initial state}} \prod_{k=t+1}^T \underbrace{p(\mathbf{x}_k | \mathbf{z}_k)}_{\text{Observation model}} \underbrace{p(\mathbf{z}_k | \mathbf{z}_{k-1}, \mathbf{u}_k)}_{\text{Transition model}} \underbrace{p(\mathbf{u}_k)}_{\text{Control prior}} \underbrace{\tilde{p}(\mathbf{x}_k)}_{\text{Goal prior}} \quad (5.62)$$

where

$$\begin{aligned} p(\mathbf{z}_t) &= \text{Cat}(\mathbf{z}_t | \mathbf{d}) \\ p(\mathbf{x}_k | \mathbf{z}_k) &= \text{Cat}(\mathbf{x}_k | \mathbf{A}\mathbf{z}_k) \\ p(\mathbf{z}_k | \mathbf{z}_{k-1}, \mathbf{u}_k) &= \prod_n \text{Cat}(\mathbf{z}_k | \mathbf{B}_n \mathbf{z}_{k-1})^{\mathbf{u}_{nk}} \\ p(\mathbf{u}_k) &= \text{Cat}(\mathbf{u}_k | \mathbf{b}_k) \\ \tilde{p}(\mathbf{x}_k) &= \text{Cat}(\mathbf{x}_k | \mathbf{c}_k). \end{aligned} \quad (5.63)$$

The notation  $\mathbf{u}_{nk}$  picks the  $n$ 'th entry of  $\mathbf{u}_k$ . We show the corresponding CFFG in Fig. 5.26.

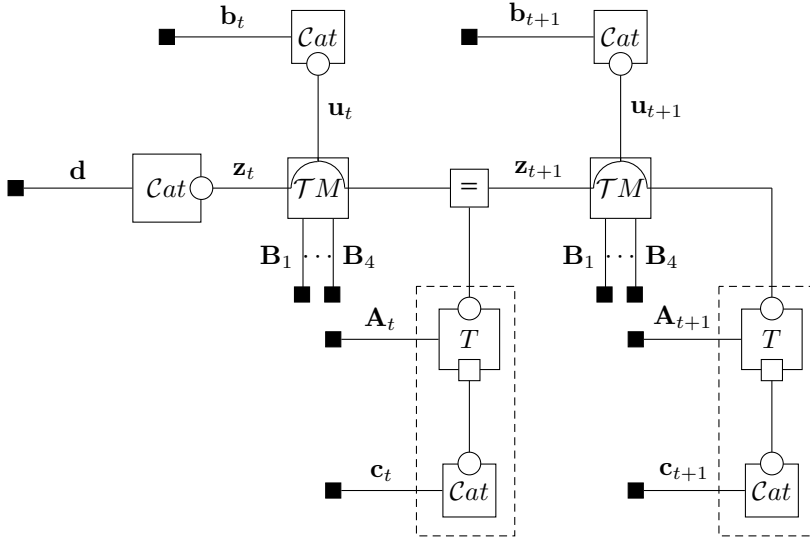


Figure 5.26: CFFG for the T-maze experiment

Eq. (5.63) defines a mixture model over candidate transition matrices indexed by  $\mathbf{u}_k$ . We give the details of this node function and the required messages in Appendix C.6. The MP schedule is shown in Fig. 5.27 with GFE-based messages highlighted in red

Following [33] we define the initial state and control prior as

$$\begin{aligned} \mathbf{d} &= (1, 0, 0, 0)^T \otimes (0.5, 0.5)^T \\ \mathbf{b}_k &= (0.25, 0.25, 0.25, 0.25) \forall k \end{aligned} \quad (5.64)$$

with  $\otimes$  denoting the Kronecker product. The transition mixture node requires a set of candidate transition matrices. The T-maze utilises four possible transitions, given below



$$\begin{aligned}
 \mathbf{B}_1 &= \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \otimes \mathbf{I}_2, \mathbf{B}_2 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \otimes \mathbf{I}_2 \\
 \mathbf{B}_3 &= \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \otimes \mathbf{I}_2, \mathbf{B}_4 = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \otimes \mathbf{I}_2
 \end{aligned} \tag{5.65}$$

where  $\mathbf{I}_2$  denotes a  $2 \times 2$  identity matrix. Note that these differ slightly from the original implementation of [33]. In [33] invalid transitions are represented by an identity mapping where we instead model invalid transitions by sending the agent back to position 1. The likelihood matrix  $\mathbf{A}$  is given by four blocks, corresponding to the observation likelihood in each position.

5

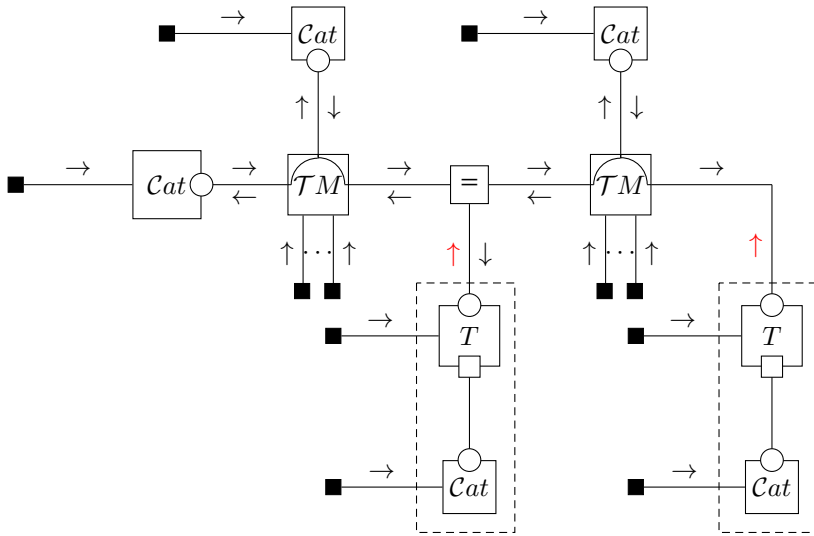


Figure 5.27: MP schedule for the T-maze experiment

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & & & \\ & \mathbf{A}_2 & & \\ & & \mathbf{A}_3 & \\ & & & \mathbf{A}_4 \end{bmatrix}, \quad (5.66)$$

with everything outside the blocks being set to 0. The blocks are

$$\begin{aligned} \mathbf{A}_1 &= \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, & \mathbf{A}_2 &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \alpha & 1 - \alpha \\ 1 - \alpha & \alpha \end{bmatrix} \\ \mathbf{A}_3 &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 - \alpha & \alpha \\ \alpha & 1 - \alpha \end{bmatrix}, & \mathbf{A}_4 &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \end{aligned} \quad (5.67)$$

with  $\alpha$  being the probability of observing a reward. The goal prior is given by

$$\mathbf{c}_k = \sigma((0, 0, c, -c)^T \otimes (1, 1, 1, 1)^T) \forall k \quad (5.68)$$

with  $c$  being the utility ascribed to a reward and  $\sigma(\cdot)$  the softmax function. Inference for the parts of the model not in  $\mathcal{P}$  can be accomplished using BP. We follow the experimental setup of [33] and let  $c = 2, \alpha = 0.9$ . For inference, we perform two iterations of our MP procedure and use 20 Newton steps for obtaining the parameters of the outgoing message from the P-substituted nodes.

We show the results in Fig. 5.28. The number in each cell is the posterior probability mass assigned to the corresponding action, with the most likely actions highlighted in red.

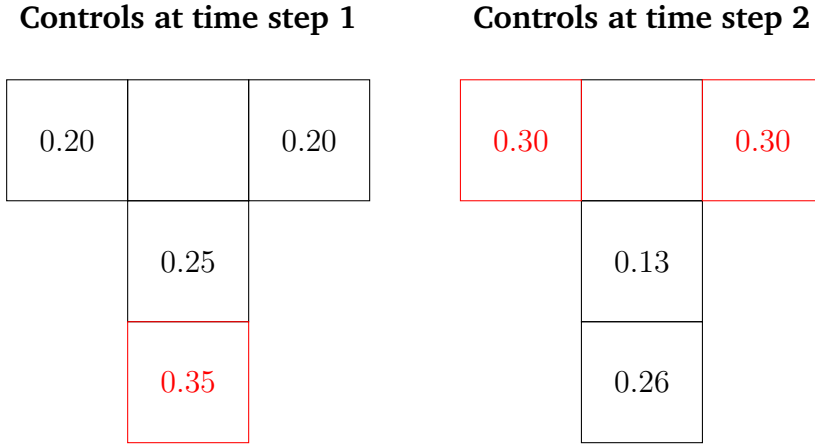


Figure 5.28: Posterior controls for the T-maze experiment

Fig. 5.28 shows an agent that initially prefers the epistemic action (move to state 4) at time  $t + 1$  and subsequently exhibits a preference for either of the potentially rewarding arms (indifferent between states 2 and 3). This shows that LAIF is able to infer the optimal policy and that our approach can reproduce prior results on the T-maze.

Since CFFG's are inherently modular, they allow us to modify the inference task without changing the model. To demonstrate, we now add  $\delta$ -constraints to the control variables. This corresponds to selecting the MAP estimate of the control posterior [116] and results in the CFFG shown in Fig. 5.29. The schedule and all messages remain the same as our previous experiment - however now the agent will instead select the most likely course of action instead of providing us with a posterior distribution.

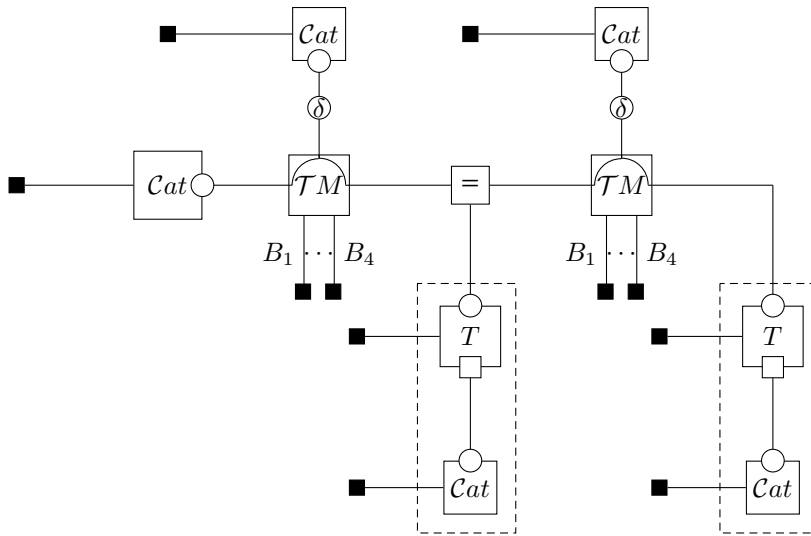


Figure 5.29: CFFG for the T-maze model with additional  $\delta$ -constraints

Performing this experiment yields the policy shown in Fig. 5.30

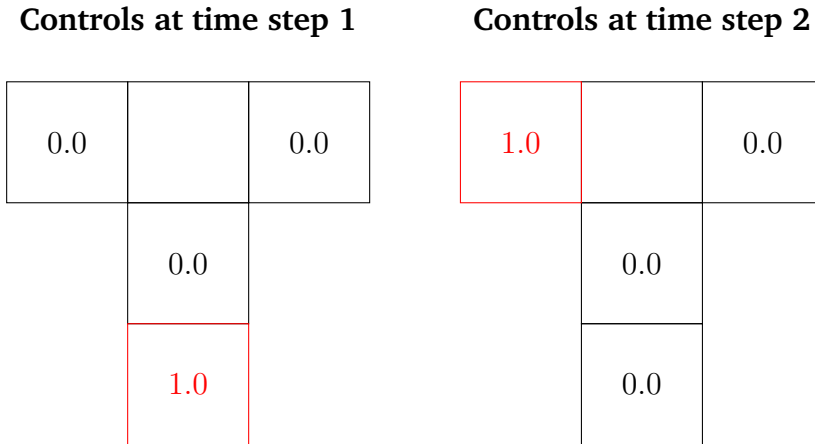


Figure 5.30: Posterior controls for the T-maze experiment with  $\delta$ -constraints

Fig. 5.30 once again shows that the agent is able to infer the optimal policy for solving the task. For repeated runs, the agent will randomly select to move to either position 2 or 3 at the second step due to minute differences in the Monte Carlo estimates used for computing the messages. The  $\delta$ -constraint obscures this since it forces the marginal to put all mass on the MAP estimate.

The point of repeating the experiments with  $\delta$ -constraints is not to show that the behaviour of the agent changes dramatically. Instead, the idea is to demonstrate that CFFG's allow for modular specification of AIF agents by adapting parts of the model without having to touch the rest. In this case, the only parts of the inference procedure that change are those involving the control marginal. This means all messages out of the P-substituted composite nodes are unaffected since the  $\delta$ -constraints are only applied to the control marginals.

## 5.10 Conclusions

In this chapter, we have proposed a novel approach to AIF based on Lagrangian optimisation which we have named Lagrangian Active Inference (LAIF). We demonstrated LAIF on a classic benchmark problem from the AIF literature and found that it inherits the epistemic drive that is a trademark feature of AIF. LAIF presents three main advantages over previous algorithms for AIF.

Firstly, an advantage of LAIF is the computational efficiency afforded by being able to pass backward messages instead of needing to perform forwards rollouts for every policy. While LAIF is still an iterative procedure, the computational complexity of each iteration scales linearly in the size of the planning horizon  $T$  instead of exponentially.

A second advantage is that LAIF allows for directly inferring posteriors over control signals instead of relying on a model comparison step based on EFE/GFE. This means that LAIF unifies inference for perception, learning, and actions into a single procedure without any overhead - everything becomes part of the same inference procedure.

Thirdly, LAIF is inherently modular and consequently works for freely definable CFFGs, while prior work has focused mostly on specific generative models. Extensions to hierarchical or heterarchical models are straightforward and only require writing out the corresponding CFFG.

We have also introduced CFFG notation for writing down constraints and P-substitutions on an FFG. CFFGs are useful not just for AIF but for specifying free energy functionals in general. CFFGs accomplish this through a simple and intuitive graphical syntax. Our hope is that CFFGs can become a standard tool similar to FFGs when it is desirable to write not just a model  $f$  but also a family of approximating distributions  $\mathcal{Q}$ . Specifically in the context of AIF we have also introduced P-substitution as a way to modify the underlying free energy functional. In doing so we have formalised the relation between AIF and MP on a CFFG, paving the way for future developments.

In future work, we plan to extend LAIF to more node constructions to further open up the scope of available problems that can be attacked using AIF.



# Chapter 6

## Conclusions and Future Outlook

*"It's not about the money, it's about sending a message."*

– *The Joker*, 2008

So far, AIF has largely been developed in the field of neuroscience and more recently reinforcement learning. Throughout this dissertation, we have worked towards bringing AIF to engineering more generally as a viable methodology for designing intelligent systems. The overall theme has been the following:

**Theme:** *Can we develop a practical toolset for describing and constructing artificial Active Inference agents?*

Under this heading, we formulated a series of research questions to guide the concrete projects we embarked on. The resulting work makes up the body of this dissertation. At this point, it is prudent to take a step back and critically assess what we have learned and what open questions remain. The first research question we addressed was:

**Q1:** *Can factor graphs be used to accurately describe the model assumptions and inference processes of an Active Inference agent?*



We approached this question in Chapter 2 and to a lesser degree 3. The main challenge we faced was that AIF uses a custom free energy functional — the EFE— the computation of which requires special considerations. We translated EFE computation to a two-step algorithm on an FFG. This work resulted in a method that we employed for Chapters 3 and 4.

The idea behind this method is to express EFE computation on the FFG of the generative model we are working with. From there, we can derive an expression for EFE given a particular likelihood model. Since EFE is a function of a particular policy, we can then evaluate a set of candidate policies in our model and select the one resulting in the lowest EFE value.

However, just like with any other technical piece of work, the devil is in the details. The details in question for Chapter 2 lie with what is not included in the analysis. Specifically, the results of Chapter 2 rely on a particular version of the EFE where goals are expressed in terms of observations rather than states. While this is practically how most software implementations work and is accurate for the models covered in both Chapters 2 and 3, there exist alternate ways of writing the EFE that express goals over states instead [22, 35, 115]. Further, expressing goals as observations leads to an additional term as we explore in Appendix A.3. For the two formulations to be equal, that term has to go to zero which requires being able to perform exact inference. The fully generic version requires expressing the EFE natively on a factor graph, which we only accomplish by Chapter 5.

Nevertheless, we used this method as an entry point for our second research question:

**Q2:** *Is a linear Gaussian dynamical system a useful generative model for an Active Inference agent?*

To answer this question, in 3 we formulated AIF on the FFG of an linear Gaussian dynamical system (LGDS) model. We then derived the required expressions for EFE computation and the resulting planning algorithm. In doing so, we uncovered the unexpected result that epistemics are absent in the linear Gaussian case. Concretely this means that an AIF agent based on an LGDS model does not possess the information-seeking properties that are often touted as a hallmark feature of AIF agents.

In terms of **Q2**, we are left with an ambivalent answer. Extending AIF to the case of LGDS undoubtedly makes it more practical for engineering applications. When designing engineering applications, we often have to deal with systems that are most conveniently described using a linear Gaussian generative model.

However, the absence of epistemics raises the question as to what advantages AIF presents over other, more established approaches like KL-control in this particular case.

On the topic of engineering applications, Chapter 4 addresses our third research question:

**Q3:** *Can an artificial Active Inference agent meaningfully support situated personalisation of hearing aid algorithms?*

Our answer to **Q3** comes in the form of AIDA, the Active Inference Design Agent. AIDA is a complex, interwoven system composed of multiple interacting models that handle audio processing in a hearing aid (HA) device. A core feature of AIDA is the ability to learn *in situ* from user feedback through the use of AIF. The generative model employed was a Gaussian process classifier (GPC) and the method used was similar to Chapters 2 and 3. Using AIF with a GPC gave AIDA the capability to adaptively search the space of user preferences while trading off exploration (learning what the user prefers) and exploitation (satisfying user preferences). The key to a successful application here is to balance this trade-off. HA users are often elderly and providing feedback to the HA system is an annoying and distracting task. This means data efficiency is key and balancing the need for informative data with correctly tuning the HA device becomes the core challenge. AIDA handles this trade-off through the balance of epistemic and goal-directed drives afforded by the EFE.

However for AIDA to be a viable practical solution still requires addressing at least two core challenges. The first lies with processing time. The system as it currently stands does not achieve *real-time* audio processing on a laptop which makes it unsuitable for practical application. In order to provide a seamless interface between the user and her environment, AIDA needs to process incoming audio fast enough that the user does not experience a disconnect with her other senses. The loopy belief propagation algorithm that was used for inference in the audio-processing system is simply too slow for real-time performance at the time of writing. The second challenge that needs to be addressed lies with the performance of the preference learning agent. Binary feedback signals convey very little information in the present setting since the first positive appraisal generally means that the task is complete and AIDA has found a good enough set of parameters to satisfy the user's preferences. Practically this means AIDA is primarily trying to learn a classifier from only negative examples which is an exceedingly difficult task. These issues mean that AIDA still requires 80 trials to reliably tune a two-parameter HA which is still too much - especially given that

actual HAs are much more complex and requires tuning many more parameters. A possible solution lies in letting the user provide a continuous feedback signal. Instead of asking "Are these parameters good enough?" the question then becomes "How good are these parameters on a scale from 1-10?". A continuous feedback signal would provide a gradient that AIDA can then follow in order to search for parameters that are *better* than the current proposal instead of needing to find the *optimal* parameters in one go. Developing AIDA V2 by addressing these challenges constitutes an exciting potential research project.

Finally, we focused on the methodology of AIF itself when addressing **Q4**:

**Q4:** *Can free energy minimisation in artificial Active Inference agents be formulated as an automatable optimisation problem by message passing on a factor graph?*

Free energy minimisation for AIF by design requires optimising both a variational and an Expected or Generalised free energy, the latter of which turned out to be the key to **Q4**.

To address **Q4** we first developed a new graphical notation for writing not just the generative model but also the exact free energy functional to be optimised. We based our notation on FFGs and augmented them with a set of constraints on the implied (Bethe) free energy. This new constrained Forney-style factor graph (CFFG) notation allowed us to identify a new way of writing the GFE as a node-local functional on a factor graph.

Starting from a constrained free energy functional on a CFFG, we constructed a node-local version of the GFE. We then solved for node-local stationary points of the resulting Lagrangian and derived the necessary message updates for integrating local GFE optimisation into generic, hybrid MP algorithms. The result is a form of Lagrangian Active Inference (LAIF) where GFE is optimised directly, rather than evaluated for a set of candidate policies. Classical AIF algorithms are recoverable as special cases of LAIF, meaning our proposed method is strictly more general.

Practically, LAIF still requires some implementational details to be practically useful in the discrete case we use for our experiments. These are documented in Chapter 5 but should be reiterated here. The first is that finding a stable solution to the backwards message requires running a Newton scheme. This inherently slows down inference since every round of MP passing now requires solving an optimisation problem per backwards message. Given the wide-reaching simplicity of the FEP proper, there should be a simpler and more elegant solution to this problem. The lack of elegance is further exacerbated when we examine

the message towards the parameters of our composite node construction. This message is not even a known exponential family distribution and to properly infer marginals we have to rely on sampling-based approximations. Again, this is not a particularly elegant solution and we hope that there exists a more beautiful way out of this conundrum. Finding these elegant solutions is still in the domain of future work.

On the other hand, being a pure MP passing technique, LAIF allowed us to do direct policy inference rather than evaluating EFE/GFE as a function of a specific policy and choosing the candidate policy with the lowest numerical value. In terms of efficiency, this puts AIF on the same level as KL-control when the latter is treated as a variational inference problem. Concretely, each iteration of LAIF scales linearly in the planning horizon, rather than exponentially. LAIF is therefore a potential solution to the scaling problem that has been a core obstacle for industry adoption of AIF so far.

As a welcome side-effect, CFFGs allow for writing generic inference problems on arbitrary graphical models that include GFE terms. Since LAIF is a local operation on a CFFG this extends the scope of AIF to include any generative model that is expressible as a CFFG- including for example the LGDS considered in **Q2**. In fact, the techniques used in Chapters 2, 3 and 4 are all expressible as instances of LAIF on a CFFG.

## 6.1 Future Outlook

Throughout this dissertation, we have concerned ourselves with AIF from the point of view of engineering, specifically using factor graphs. Our goal has been to work towards developing the requisite tooling to make AIF a practical engineering approach - and the true measure of the value of a tool is whether it is useful. Assessing whether we have succeeded in our endeavour is therefore not up to us, but is to be judged by the AIF community at large. Simply put, if the work presented in this dissertation ends up being used for research and industrial applications of AIF, we have succeeded in what we set out to do. To conclude this dissertation, we will therefore muse on the possible opportunities and obstacles for adoption that we see in the future.

Paradoxically, a potential obstacle to wider adoption is our choice to approach the problem from the angle of engineering. Donning the hat of the engineer means that we have not concerned ourselves with neurobiology and any related features of AIF. Since AIF is still primarily a neuroscientific endeavour, initial adoption of our work is likely to largely come from neuroscientists whose

objectives might be different from ours. Where we have focused on designing tools for practical applications, a neuroscientist might be more concerned with biological plausibility which we have not made a primary concern — instead we have operationalised our original question of "what the brain does" as AIF and proceeded to focus on implementations using MP. We do however note that there is a reasonable body of work supporting the idea that the brain performs some version of MP which would be compatible with the work presented in this dissertation.

In terms of developing tooling, our main contribution lies in the introduction of CFFGs and Lagrangian Active Inference in Chapter 5 with prior chapters forming the foundation for this development. CFFGs are a new, concise way of accurately specifying AIF problems - including prior work such as [33] and [81]. A notable omission from the analyses in Chapter 5 is sophisticated inference (SI) [35]. This is important because SI is the state-of-the-art method for AIF at the time of writing. As alluded to in Chapter 2, SI does not easily conform to a pure MP description due to a number of algorithmic moves made in order to both prune and expand the forward search tree. We hope to be able to address this shortcoming as part of future research.

Still, writing AIF agents in terms of their CFFG allows for a syntax that is both flexible, easily interpretable, has a low barrier of entry, and is mathematically precise. All of these are desirable qualities that we wish to bring to the broader AIF community.

As with any body of scientific work, we have found that the insights gained throughout writing this dissertation have raised more questions than they have answered. While future work is still needed to build the necessary tools for the widespread adoption of AIF, we believe that CFFGs provide the requisite foundational work for designing just such a class of tools. Only once such tools become widely adopted and field-tested on industrial applications and cutting-edge research, will we know the answer to our original, overarching question.

For now, thanks for staying with us to the end,

A handwritten signature in black ink, appearing to read "Magnus". The signature is fluid and cursive, with a long, sweeping tail on the final letter.

# Appendix A

## Appendices for Chapter 3

### A.1 Perception as Bayesian filtering

To derive the filtering equations in Section 3.4, we need to infer  $p(z_t | \mathbf{x}_{1:t})$ . We consider a model of the form

$$p(x_t, z_t, u_t | z_{t-1}) = p(x_t | z_t)p(z_t | z_{t-1}, u_t)p(u_t). \quad (\text{A.1})$$

We can write the inference task as

$$p(z_t | \mathbf{x}_{1:t}) = p(z_t | x_t, \mathbf{x}_{1:t-1}) \quad (\text{A.2a})$$

$$= \frac{1}{p(x_t | \mathbf{x}_{1:t-1})} p(x_t, z_t | \mathbf{x}_{1:t-1}) \quad (\text{A.2b})$$

$$= \frac{1}{p(x_t | \mathbf{x}_{1:t-1})} p(x_t | z_t)p(z_t | \mathbf{x}_{1:t-1}) \quad (\text{A.2c})$$

$$= \frac{1}{p(x_t | \mathbf{x}_{1:t-1})} p(x_t | z_t) \iint p(z_t, z_{t-1}, u_t | \mathbf{x}_{1:t-1}) dz_{t-1} du_t \quad (\text{A.2d})$$

$$= \frac{1}{\underbrace{p(x_t | \mathbf{x}_{1:t-1})}_{\text{Evidence}}} \underbrace{p(x_t | z_t)}_{\text{Likelihood}} \times \iint \underbrace{p(z_t | z_{t-1}, u_t)}_{\text{State transition}} \underbrace{p(u_t)}_{\text{Control prior}} \underbrace{p(z_{t-1} | \mathbf{x}_{1:t-1})}_{\text{State prior}} dz_{t-1} du_t. \quad (\text{A.2e})$$

Now assuming that observations  $\mathbf{x}_{1:t} = \hat{\mathbf{x}}_{1:t}$  are available and the state prior is available from the last time step, we can select a control by setting

$$p(u_t) = \delta(u_t - \hat{u}_t). \quad (\text{A.3})$$

The inference problem becomes

$$\begin{aligned} p(z_t | \hat{\mathbf{x}}_{1:t}) &= \frac{1}{p(\hat{x}_t | \hat{\mathbf{x}}_{1:t-1})} p(\hat{x}_t | z_t) \\ &\quad \times \iint p(z_t | z_{t-1}, u_t) \delta(u_t - \hat{u}_t) p(z_{t-1} | \hat{\mathbf{x}}_{1:t-1}) dz_{t-1} du_t \end{aligned} \quad (\text{A.4a})$$

$$= \frac{1}{p(\hat{x}_t | \hat{\mathbf{x}}_{1:t-1})} p(\hat{x}_t | z_t) \int p(z_t | z_{t-1}, \hat{u}_t) p(z_{t-1} | \hat{\mathbf{x}}_{1:t-1}) dz_{t-1}. \quad (\text{A.4b})$$

The likelihood is then available from the generative model and the state prior is available from the previous time step. We can find the evidence by marginalising out  $z_t$  as

$$p(\hat{x}_t | \hat{\mathbf{x}}_{1:t-1}) = \int p(\hat{x}_t | z_t) \int p(z_t | z_{t-1}, \hat{u}_t) p(z_{t-1} | \hat{\mathbf{x}}_{1:t-1}) dz_{t-1} dz_t. \quad (\text{A.5})$$

Solving these equations amount to performing Bayesian filtering, synonymous with perception.

## A.2 Linearly Related Gaussian variables

In this section we show how to obtain a posterior marginal, given linearly related and jointly Gaussian variables. We use this result throughout our derivations in 3.4, for instance when moving from Eq. (3.11c) to Eq. (3.11d). Using  $\mathbf{x}$  and  $\mathbf{z}$  for generic variables, the goal is to obtain the posterior

$$p(\mathbf{x}) = \int p(\mathbf{x} | \mathbf{z}) p(\mathbf{z}) d\mathbf{z} \quad (\text{A.6})$$

where

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z} \mid \boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z) \quad (\text{A.7a})$$

$$p(\mathbf{x} \mid \mathbf{z}) = \mathcal{N}(\mathbf{x} \mid \mathbf{A}\mathbf{z} + \mathbf{b}, \boldsymbol{\Sigma}_x) \quad (\text{A.7b})$$

We can view the problem of obtaining  $p(\mathbf{x})$  as first applying a linear transform ( $\mathbf{A}\mathbf{z} + \mathbf{b}$ ) and then adding Gaussian noise with mean  $\mathbf{0}$  and variance  $\boldsymbol{\Sigma}_x$ . We will deal with each of these steps in turn, starting with the linear transform. The posterior mean  $\boldsymbol{\mu}$  is given by

$$\boldsymbol{\mu} = \mathbb{E}[\mathbf{A}\mathbf{z} + \mathbf{b}] = \mathbf{A}\mathbb{E}[\mathbf{z}] + \mathbf{b} = \mathbf{A}\boldsymbol{\mu}_z + \mathbf{b} \quad (\text{A.8})$$

where  $\mathbb{E}$  denotes the expectation operation. Here we first factor the terms that do not depend on  $\mathbf{z}$  out of the expectation and then identify  $\mathbb{E}[\mathbf{z}] = \boldsymbol{\mu}_z$  as  $\mathbf{z}$  is Gaussian. To find the covariance matrix  $\boldsymbol{\Sigma}$  we can proceed by plugging the terms we know into the definition of covariance

$$\boldsymbol{\Sigma} = \mathbb{E}[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T] \quad (\text{A.9a})$$

$$= \mathbb{E}\left[\underbrace{(\mathbf{A}\mathbf{z} + \mathbf{b})}_{\mathbf{x}} - \underbrace{\mathbf{A}\boldsymbol{\mu}_z + \mathbf{b}}_{\boldsymbol{\mu}} \left( \underbrace{(\mathbf{A}\mathbf{z} + \mathbf{b})}_{\mathbf{x}} - \underbrace{\mathbf{A}\boldsymbol{\mu}_z + \mathbf{b}}_{\boldsymbol{\mu}} \right)^T\right] \quad (\text{A.9b})$$

$$= \mathbb{E}\left[(\mathbf{A}\mathbf{z} - \mathbf{A}\boldsymbol{\mu}_z)(\mathbf{A}\mathbf{z} - \mathbf{A}\boldsymbol{\mu}_z)^T\right] \quad (\text{A.9c})$$

Now we can factor  $\mathbf{A}$  out of the expectation

$$= \mathbb{E}\left[\mathbf{A}(\mathbf{z} - \boldsymbol{\mu}_z)(\mathbf{z} - \boldsymbol{\mu}_z)^T \mathbf{A}^T\right] \quad (\text{A.10a})$$

$$= \mathbf{A} \underbrace{\mathbb{E}\left[(\mathbf{z} - \boldsymbol{\mu}_z)(\mathbf{z} - \boldsymbol{\mu}_z)^T\right]}_{\boldsymbol{\Sigma}_z} \mathbf{A}^T \quad (\text{A.10b})$$

$$= \mathbf{A}\boldsymbol{\Sigma}_z\mathbf{A}^T. \quad (\text{A.10c})$$

In the last line we recognise the definition of the prior covariance matrix  $\boldsymbol{\Sigma}_z$ . To obtain our final result we now need to add Gaussian noise with mean  $\mathbf{0}$  and variance  $\boldsymbol{\Sigma}_x$ . We know that if two Gaussian variables are independent, the variance of their sum is the sum of their variances. We can use this to write the final covariance matrix as



$$\Sigma = \mathbf{A}\Sigma_z\mathbf{A}^T + \Sigma_x \quad (\text{A.11})$$

Which gives the result utilised in the main text as

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x} \mid \mathbf{A}\boldsymbol{\mu}_z + \mathbf{b}, \mathbf{A}\Sigma_z\mathbf{A}^T + \Sigma_x) \quad (\text{A.12})$$

### A.3 Mutual Information Bound

In this section, we examine the result of substituting  $q$  for  $p$  in Eq. (3.32). We see that the substitution turns Eq. (3.32) into a bound on the expected free energy that becomes exact when we can do exact inference. We can show this by writing

$$\begin{aligned} & \iint q(x_k, z_k \mid u_k) \log \frac{q(z_k \mid u_k)}{q(z_k \mid x_k, u_k)} dx_k dz_k \\ &= \iint q(x_k, z_k \mid u_k) \log \frac{q(z_k \mid u_k)}{q(z_k \mid x_k, u_k)} \frac{p(z_k \mid x_k, u_k)}{p(z_k \mid x_k, u_k)} dx_k dz_k \end{aligned} \quad (\text{A.13a})$$

$$= \iint q(x_k, z_k \mid u_k) \log \frac{q(z_k \mid u_k)}{p(z_k \mid x_k, u_k)} \frac{p(z_k \mid x_k, u_k)}{q(z_k \mid x_k, u_k)} dx_k dz_k \quad (\text{A.13b})$$

$$\begin{aligned} &= \iint q(x_k, z_k \mid u_k) \log \frac{q(z_k \mid u_k)}{p(z_k \mid x_k, u_k)} dx_k dz_k \\ &\quad - \underbrace{\iint q(x_k, z_k \mid u_k) \log \frac{q(z_k \mid x_k, u_k)}{p(z_k \mid x_k, u_k)} dx_k dz_k}_{\mathbb{E}_{q(x_k|z_k)} [\text{KL}[q(z_k|x_k, u_k) \parallel p(z_k|x_k, u_k)]]} \end{aligned} \quad (\text{A.13c})$$

$$\leq \iint q(x_k, z_k \mid u_k) \log \frac{q(z_k \mid u_k)}{p(z_k \mid x_k, u_k)} dx_k dz_k, \quad (\text{A.13d})$$

where we recognise the upper bound since the expected Kullback-Leibler divergence (KL) is non-negative. When the expected KL goes to 0, which is the case when we do exact inference, the bound becomes exact. That is the case for all models considered in Chapter 3 since all relations are linear and all distributions Gaussian.

## A.4 Mutual information derivation

In this section, we derive the expression for mutual information in detail. We restate the definition given in Eq. (3.35) here as a starting point

$$I[x, z] = \iint p(x, z) \log \frac{p(x, z)}{p(x)p(z)} dx dz = H[z] - H[z | x] = H[x] - H[x | z]. \quad (\text{A.14})$$

Note that we can write Eq. (3.35) in terms of entropies of either  $x$  or  $z$  since mutual information is symmetric in its arguments.

Recall that Eq. (3.34c) optimises mutual information between expected observations  $\mathbf{x}_k$  and latent states  $\mathbf{z}_k$ . This means that we need the marginal and conditional distributions of  $\mathbf{x}_k$  in order to calculate the requisite entropies. The marginal is given by Eq. (3.17) and the conditional by Eq. (3.18). Since both distributions are Gaussian, their entropies are available in closed form as

$$H[\mathbf{x}_k] = \frac{1}{2} (n \log 2\pi + \log |\boldsymbol{\Sigma}_{22}| + n) \quad (\text{A.15a})$$

$$H[\mathbf{x}_k | \mathbf{z}_k] = \frac{1}{2} (n \log 2\pi + \log |\boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\Sigma}_{12}| + n), \quad (\text{A.15b})$$

where  $n$  denotes the dimensionality. Before we proceed, we need the generalised matrix determinant lemma which can be written as

$$|\mathbf{A} + \mathbf{U}\mathbf{V}^T| = |\mathbf{I} + \mathbf{V}^T \mathbf{A}^{-1} \mathbf{U}| |\mathbf{A}|, \quad (\text{A.16})$$

where  $\mathbf{I}$  denotes the identity matrix and  $\mathbf{A}$  is invertible. By setting  $\mathbf{V} = \mathbf{U} = \mathbf{I}$  we obtain a form which will be convenient moving forwards

$$|\mathbf{A} + \mathbf{U}\mathbf{V}^T| = |\mathbf{A} + \mathbf{I}| = |\mathbf{I} + \mathbf{I}^T \mathbf{A}^{-1} \mathbf{I}| |\mathbf{A}| \quad (\text{A.17a})$$

$$= |\mathbf{I} + \mathbf{A}^{-1}| |\mathbf{A}|, \quad (\text{A.17b})$$

which also implies

$$|\mathbf{I} - \mathbf{A}^{-1}| = \frac{|\mathbf{I} - \mathbf{A}|}{-|\mathbf{A}|}. \quad (\text{A.18})$$

Now we can write the MI as

$$I[\mathbf{x}_k, \mathbf{z}_k] = H[\mathbf{x}_k] - H[\mathbf{x}_k | \mathbf{z}_k] \quad (\text{A.19a})$$

$$= \frac{1}{2} \log |\boldsymbol{\Sigma}_{22}| - \frac{1}{2} \log |\boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\Sigma}_{12}| \quad (\text{A.19b})$$

$$= -\frac{1}{2} \log |\boldsymbol{\Sigma}_{22}|^{-1} |\boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\Sigma}_{12}| \quad (\text{A.19c})$$

$$= -\frac{1}{2} \log |\mathbf{I} - \boldsymbol{\Sigma}_{22}^{-1} \boldsymbol{\Sigma}_{21} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\Sigma}_{12}| \quad (\text{A.19d})$$

$$= -\frac{1}{2} \log |\mathbf{I} - \underbrace{[\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T + \boldsymbol{\Sigma}_x]^{-1}}_{\boldsymbol{\Sigma}_{22}^{-1}} \underbrace{\mathbf{A} \boldsymbol{\Sigma}_{z_k}}_{\boldsymbol{\Sigma}_{21}} \underbrace{\boldsymbol{\Sigma}_{z_k}^{-1}}_{\boldsymbol{\Sigma}_{11}^{-1}} \underbrace{\boldsymbol{\Sigma}_{z_k} \mathbf{A}^T}_{\boldsymbol{\Sigma}_{12}}| \quad (\text{A.19e})$$

$$= -\frac{1}{2} \log |\mathbf{I} - [\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T + \boldsymbol{\Sigma}_x]^{-1} \mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T| \quad (\text{A.19f})$$

$$= -\frac{1}{2} \log |\mathbf{I} - \underbrace{[\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T]^{-1} \mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T}_{\text{Evaluates to } \mathbf{I}} + [\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T]^{-1} \boldsymbol{\Sigma}_x|^{-1}| \quad (\text{A.19g})$$

$$= -\frac{1}{2} \log |\mathbf{I} - [\mathbf{I} + [\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T]^{-1} \boldsymbol{\Sigma}_x]^{-1}|. \quad (\text{A.19h})$$

Using Eq. (A.18) we can rewrite Eq. (A.19h) as

$$-\frac{1}{2} \log |\mathbf{I} - [\mathbf{I} + [\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T]^{-1} \boldsymbol{\Sigma}_x]^{-1}| = -\frac{1}{2} \log \left( \frac{|\mathbf{I} - [\mathbf{I} + (\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T)^{-1} \boldsymbol{\Sigma}_x]|}{-|\mathbf{I} + (\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T)^{-1} \boldsymbol{\Sigma}_x|} \right) \quad (\text{A.20a})$$

$$= -\frac{1}{2} \log \left( \frac{-|(\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T)^{-1} \boldsymbol{\Sigma}_x|}{-|\mathbf{I} + (\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T)^{-1} \boldsymbol{\Sigma}_x|} \right) \quad (\text{A.20b})$$

$$= -\frac{1}{2} \log \left( \frac{|(\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T)^{-1} \boldsymbol{\Sigma}_x|}{|\mathbf{I} + (\mathbf{A} \boldsymbol{\Sigma}_{z_k} \mathbf{A}^T)^{-1} \boldsymbol{\Sigma}_x|} \right). \quad (\text{A.20c})$$

Applying Eq. (A.17a) in the denominator, we now rewrite Eq. (A.20c) as

$$-\frac{1}{2} \log \left( \frac{|(\mathbf{A}\Sigma_{z_k}\mathbf{A}^T)^{-1}\Sigma_x|}{|\mathbf{I} + (\mathbf{A}\Sigma_{z_k}\mathbf{A}^T)^{-1}\Sigma_x|} \right) = -\frac{1}{2} \log \left( \frac{|(\mathbf{A}\Sigma_{z_k}\mathbf{A}^T)^{-1}\Sigma_x|}{|\mathbf{I} + [(\mathbf{A}\Sigma_z\mathbf{A}^T)^{-1}\Sigma_x]^{-1} |(\mathbf{A}\Sigma_{z_k}\mathbf{A}^T)^{-1}\Sigma_x|} \right) \quad (\text{A.21a})$$

$$= -\frac{1}{2} \log \left( \frac{|(\mathbf{A}\Sigma_{z_k}\mathbf{A}^T)^{-1}\Sigma_x|}{|\mathbf{I} + [(\mathbf{A}\Sigma_z\mathbf{A}^T)^{-1}\Sigma_x]^{-1} |(\mathbf{A}\Sigma_{z_k}\mathbf{A}^T)^{-1}\Sigma_x|} \right) \quad (\text{A.21b})$$

$$= -\frac{1}{2} \log \left( \frac{1}{|\mathbf{I} + [(\mathbf{A}\Sigma_z\mathbf{A}^T)^{-1}\Sigma_x]^{-1}|} \right) \quad (\text{A.21c})$$

$$= \frac{1}{2} \log \left( \left| \mathbf{I} + [(\mathbf{A}\Sigma_z\mathbf{A}^T)^{-1}\Sigma_x]^{-1} \right| \right) \quad (\text{A.21d})$$

$$= \frac{1}{2} \log |\mathbf{I} + \Sigma_x^{-1}\mathbf{A}\Sigma_{z_k}\mathbf{A}^T|, \quad (\text{A.21e})$$

which gives the expression found in Eq. (3.36).



# Appendix B

## Appendices for Chapter 4

### B.1 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 4.5.

#### B.1.1 Acoustic Model

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

$$x_t = z_t + n_t \tag{B.1}$$

The speech signal  $z_t = \mathbf{e}_1^T \mathbf{z}_t$  is modeled by a time-varying auto-regressive (TVAR) process as

$$p(\mathbf{z}_t | \mathbf{z}_{t-1}) = \mathcal{N}(\mathbf{z}_t | \mathbf{A}(\boldsymbol{\theta}_t)\mathbf{z}_{t-1}, \mathbf{V}(\gamma)) \tag{B.2}$$

The auto-regressive (AR)-coefficients of the speech signal are time-varying with dynamics given by

$$p(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}) = \mathcal{N}(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}, \omega \mathbf{I}) \tag{B.3}$$

The noise signal  $n_t = \mathbf{e}_1^T \mathbf{n}_t$  is also modeled by an AR process

$$p(\mathbf{n}_t | \phi_k, \mathbf{n}_{t-1}, \tau_k) = \mathcal{N}(\mathbf{n}_t | \mathbf{A}(\phi_k)\mathbf{n}_{t-1}, \mathbf{V}(\tau_k)), \quad \text{for } t = t^-, t^- + 1, \dots, t^+ \quad (\text{B.4})$$

The parameters of the noise model are dependent on context through

$$p(\phi_k | \mathbf{o}_k) = \prod_{l=1}^L \mathcal{N}(\phi_k | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)^{\mathbf{o}_{lk}} p(\tau_k | \mathbf{o}_k, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \prod_{l=1}^L \Gamma(\tau_k | \alpha_l, \beta_l)^{\mathbf{o}_{lk}} \quad (\text{B.5})$$

where the context  $\mathbf{o}_k$  evolves over a different time scale indexed by  $k$  as

$$p(\mathbf{o}_k | \mathbf{B}, \mathbf{o}_{k-1}) = \text{Cat}(\mathbf{o}_k | \mathbf{B}\mathbf{o}_{k-1}), \quad (\text{B.6})$$

Each column of the context transition matrix is modeled as

$$p(\mathbf{B}_j | \boldsymbol{\alpha}_j) = \text{Dir}(\mathbf{B}_j | \boldsymbol{\alpha}_j), \quad (\text{B.7})$$

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

$$y_t = u_{zk} \mathbf{e}_1^T \mathbf{z}_t + u_{nk} \mathbf{e}_1^T \mathbf{n}_t \quad \text{for } t = t^-, t^- + 1, \dots, t^+ \quad (\text{B.8})$$

### B.1.2 AIDA's User Response Model

The user responses are modeled by a Bernoulli distribution parameterised by a Gaussian cumulative probability function that enforces the output  $v_k(\mathbf{u}_k) \in [0, 1]$  as

$$p(r_k | v_k, \mathbf{u}_k) = \text{Ber}(\Phi(v_k(\mathbf{u}_k))), \quad \text{with } r_k \in \{0, 1\} \quad (\text{B.9})$$

$v_k(\mathbf{u}_k)$  encodes our beliefs about the user response function (evaluated at  $\mathbf{u}_k$ ), modeled by a mixture of Gaussian processes as

$$p(v_k(\cdot) | \mathbf{o}_k) = \prod_{l=1}^L \text{GP}(m_l(\cdot), \mathbf{K}_l(\cdot, \cdot))^{o_{lk}} \quad (\text{B.10})$$

whose kernel function is defined as

$$K(\mathbf{u}, \mathbf{u}') = \sigma^2 \exp \left\{ -\frac{\|\mathbf{u} - \mathbf{u}'\|_2^2}{2l^2} \right\} \quad (\text{B.11})$$

where  $\sigma$  denotes noise and  $l$  the length scale of the kernel.

## B.2 Inference Realisation

This appendix describes in detail how the inference tasks of Sections 4.4.1 and 4.4.2 are realized. The inference task of Section 4.4.3 is performed by message passing (MP) using the update rules of [85].

### B.2.1 Realisation of Inference for Context Classification

The inference task for context classification of Eq. (4.13) is intractable as discussed in Section 4.4.1. To circumvent this problem, we will instead approach this task as a Bayesian model comparison problem.

In a Bayesian model comparison problem, we are interested in calculating the posterior probability  $p(m_l | \mathbf{x})$  of some model  $m_l$  after observing data  $\mathbf{x}$ .

The posterior model probability  $p(m_l | \mathbf{x})$  can be calculated using Bayes rule as

$$p(m_l | \mathbf{x}) = \frac{p(\mathbf{x} | m_l)p(m_l)}{\sum_j p(\mathbf{x} | m_j)p(m_j)}, \quad (\text{B.12})$$

where the denominator represents the weighted model evidence  $p(\mathbf{x})$ , i.e. the model evidence obtained for the individual models  $p(\mathbf{x} | m_l)$ , weighted by their priors  $p(m_l)$ .

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  $\mathbf{o}_{k-1}$  and  $\mathbf{s}_{t-1}$  in Eq. (4.13) separately, and therefore we implicitly assume a factorisation of our prior  $p(\mathbf{o}_{k-1}, \mathbf{s}_{t-1} \mid \mathbf{x}_{1:t-1})$  as

$$p(\mathbf{o}_{k-1}, \mathbf{s}_{t-1} \mid \mathbf{x}_{1:t-1}) = p(\mathbf{o}_{k-1} \mid \mathbf{x}_{1:t-1}) p(\mathbf{s}_{t-1} \mid \mathbf{x}_{1:t-1}). \quad (\text{B.13})$$

As a result Eq. (4.13) can be rewritten as

$$p(\mathbf{o}_k \mid \mathbf{x}_{1:t+}) \propto \underbrace{\int p(\mathbf{o}_k, \mathbf{B} \mid \mathbf{o}_{k-1}) p(\mathbf{o}_{k-1} \mid \mathbf{x}_{1:t-1}) d\mathbf{B} d\mathbf{o}_{k-1}}_{\vec{\mu}(\mathbf{o}_k)} \cdot \underbrace{\int p(\mathbf{s}_{t-:t+}, \Psi_k, \mathbf{x}_{t-:t+} \mid \mathbf{s}_{t-1}, \mathbf{o}) p(\mathbf{s}_{t-1} \mid \mathbf{x}_{1:t-1}) d\mathbf{s}_{t-1:t+} d\Psi_k}_{p(\mathbf{x}_{t-:t+} \mid \mathbf{x}_{1:t-1}, \mathbf{o}_k)}. \quad (\text{B.14})$$

The first term  $\vec{\mu}(\mathbf{o}_k)$  can be regarded as the forward message towards the context  $\mathbf{o}_k$  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  $\mathbf{B}$ . The second term  $p(\mathbf{x}_{t-:t+} \mid \mathbf{x}_{1:t-1}, \mathbf{o}_k)$  can be regarded as the incremental model evidence under some given context  $\mathbf{o}_k$ . Comparison of Eq. (B.14) and Eq. (B.12) allows us to formulate our inference problem in Eq. (4.13) into a Bayesian model comparison problem by defining

$$p(m_l) = \vec{\mu}(\mathbf{o}_k = \mathbf{e}_l), \quad (\text{B.15a})$$

$$p(\mathbf{x} \mid m_l) = p(\mathbf{x}_{t-:t+} \mid \mathbf{x}_{1:t-1}, \mathbf{o}_k = \mathbf{e}_l). \quad (\text{B.15b})$$

We can therefore define a model  $m_l$  by clamping the context variable in generative model as  $\mathbf{o}_k = \mathbf{e}_l$ . 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  $p(\mathbf{x}_{t-:t+} \mid \mathbf{x}_{1:t-1}, \mathbf{o}_k = \mathbf{e}_l)$  cannot be computed exactly as the AR source models lead to intractable inference.

Instead, we approximate the model evidence in Eq. (B.15b) using the Bethe free energy (BFE), as defined in Eq. (1.14)

$$p(\mathbf{x} | m_l) \approx \exp\{-F[q, m_l]\}, \quad (\text{B.16})$$

where  $F[q, m_l]$  denotes the BFE observed after convergence of the inference algorithm for model  $m_l$ . Similarly the calculation of Eq. (B.15a) is intractable. Therefore we will approximate the model prior with the variational message towards  $\mathbf{o}_k$  instead as

$$p(m_l) \approx \vec{v}(\mathbf{o}_k = \mathbf{e}_l). \quad (\text{B.17})$$

## B.2.2 Realisation 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 expected free energy (EFE) decomposition in Eq. (4.15) can be obtained for the Gaussian process classifier (GPC) using results from [47].

The first term in the decomposition, the negative utility drive, is a cross entropy loss between our goal prior and posterior marginal. Since user responses are binary, the resulting cross-entropy term has a closed-form expression as [47]

$$\begin{aligned} -\mathbb{E}_{q(r|\mathbf{u})} [\log p(r)] = \\ \Phi \left( \frac{\mu_{\mathbf{u}, \mathbf{D}}}{\sqrt{\sigma_{\mathbf{u}, \mathbf{D}}^2 + 1}} \right) \log \mathbb{E}_{p(r)}[r] + \left( 1 - \Phi \left( \frac{\mu_{\mathbf{u}, \mathbf{D}}}{\sqrt{\sigma_{\mathbf{u}, \mathbf{D}}^2 + 1}} \right) \right) \log (1 - \mathbb{E}_{p(r)}[r]), \end{aligned} \quad (\text{B.18})$$

where  $\mu_{\mathbf{u}, \mathbf{D}}$  and  $\sigma_{\mathbf{u}, \mathbf{D}}^2$  denote the posterior mean and variance returned by the GPC when queried at the point  $\mathbf{u}$  given some data set  $\mathbf{D} = \{\mathbf{u}_{1:k-1}, \mathbf{r}_{1:k-1}\}$ , respectively. More concretely, the GPC returns a Gaussian distribution from which the posterior mean and variance are extracted as  $v(\mathbf{u}) = \mathcal{N}(\mu_{\mathbf{u}, \mathbf{D}}, \sigma_{\mathbf{u}, \mathbf{D}}^2)$ .

$\Phi(\cdot)$  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  $\mathbf{D}$ .

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 ( $I[r, v | \mathbf{D}, \mathbf{u}]$ ) can be expressed in a GPC as [47]

$$I[r, v | \mathbf{D}, \mathbf{u}] \approx h \left( \Phi \left( \frac{\mu_{\mathbf{u}, \mathbf{D}}}{\sqrt{\sigma_{\mathbf{u}, \mathbf{D}}^2 + 1}} \right) \right) - \frac{C}{\sqrt{\sigma_{\mathbf{u}, \mathbf{D}}^2 + C^2}} \exp \left( -\frac{\mu_{\mathbf{u}, \mathbf{D}}^2}{2(\sigma_{\mathbf{u}, \mathbf{D}}^2 + C^2)} \right), \quad (\text{B.19})$$

where the constant  $C$  is defined as  $C = \sqrt{\frac{\pi \log 2}{2}}$  and where  $h(\cdot)$  is defined as  $h(p) = -p \log(p) - (1-p) \log(1-p)$ .

### Inference in the Gaussian Process Classifier

To perform inference in the GPC for our experiments, we use the Laplace approximation as described in [90, Chapter 3.4]. 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  $p(v | \mathbf{D})$  amounts to calculating

$$p(v | \mathbf{D}) = \frac{p(\mathbf{r}_{1:k-1} | v)p(v | \mathbf{u}_{1:k-1})}{p(\mathbf{r}_{1:k-1} | \mathbf{u}_{1:k-1})} \quad (\text{B.20a})$$

$$\propto p(\mathbf{r}_{1:k-1} | v)p(v | \mathbf{u}_{1:k-1}). \quad (\text{B.20b})$$

Taking the logarithm of Eq. (B.20b) and differentiating twice with respect to  $v$  gives

$$\nabla_v \log p(v | \mathbf{D}) = \nabla_v \log p(\mathbf{r}_{1:k-1} | v) - \mathbf{K}^{-1}v \quad (\text{B.21a})$$

$$\nabla_v \nabla_v \log p(v | \mathbf{D}) = \nabla_v \nabla_v \log p(\mathbf{r}_{1:k-1} | v) - \mathbf{K}^{-1} = -\mathbf{W} - \mathbf{K}^{-1} \quad (\text{B.21b})$$

where  $\nabla_v$  denotes the gradient with respect to  $v$ ,  $\mathbf{K} = \mathbf{K}(\mathbf{u}_{1:k-1}, \mathbf{u}_{1:k-1})$  is the kernel matrix over the queries  $\mathbf{u}_{1:k-1}$  and  $\mathbf{W} = -\nabla_v \nabla_v \log p(\mathbf{r}_{1:k-1} | v)$  is a

diagonal matrix since the likelihood factorises over independent observations. At the mode  $v^*$  Eq. (B.21a) equals zero which implies

$$v^* = \mathbf{K} \nabla_v \log p(\mathbf{r}_{1:k-1} | v^*). \quad (\text{B.22})$$

Directly solving Eq. (B.22) is intractable, because of the recursive non-linear relationship. Instead we can estimate  $v^*$  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 [90, Algorithm 3.1]. We determine the step size using a line search as implemented in `Optim.jl` [52]. Having found the mode  $v^*$ , we can construct our posterior approximation as

$$p(v | \mathbf{D}) \approx \mathcal{N}\left(v^*, (\mathbf{K}^{-1} + \mathbf{W})^{-1}\right), \quad (\text{B.23})$$

where  $\mathbf{W}$  is evaluated at  $v = v^*$ . If we now recall that evaluating a Gaussian process (GP) at any finite number of points results in a Gaussian, we see that under the Laplace approximation the solution can now be obtained using standard results for marginalisation of jointly Gaussian variables. We define the shorthand  $\mathbf{K}(\mathbf{u}_k, \mathbf{u}_{1:k-1}) = \mathbf{K}_{1:k}$  and  $\mathbf{K}(\mathbf{u}_k, \mathbf{u}_k) = \mathbf{K}_k$  and find the posterior mean  $\mu_{\mathbf{u}}$  as [90, p. 44]

$$\mu_{\mathbf{u}, \mathbf{D}} = \mathbf{K}_{1:k}^T \mathbf{K}^{-1} v^* = \mathbf{K}_{1:k}^T \nabla \log p(\mathbf{r}_{1:k-1} | v^*). \quad (\text{B.24})$$

The posterior covariance  $\sigma_{\mathbf{u}, \mathbf{D}}^2$  is given by [90, p. 44]

$$\sigma_{\mathbf{u}, \mathbf{D}}^2 = \mathbf{K}_k - \mathbf{K}_{1:k}^T (\mathbf{K} + \mathbf{W}^{-1})^{-1} \mathbf{K}_{1:k}. \quad (\text{B.25})$$



# Appendix C

## Appendices for Chapter 5

### C.1 Expected Free Energy

This section will focus on the the most common alternative functional used for AIF, the Expected Free Energy (EFE).

EFE is the standard choice for AIF models and is what is found in most AIF focused software such as SPM [39] and PyMDP [44]. In this section we will cover the details of how EFE is commonly treated.

EFE is defined specifically over future time steps and on a particular choice of generative model. The model in question is a state space model (SSM) of the form

$$p(\mathbf{x}, \mathbf{z} \mid \hat{\mathbf{u}}) = \underbrace{p(z_t)}_{\text{State prior}} \prod_{k=t+1}^T \underbrace{p(x_k \mid z_k)}_{\text{Observation model}} \underbrace{p(z_k \mid z_{k-1}, \hat{u}_k)}_{\text{Transition model}} \quad (\text{C.1})$$

where  $\hat{u}_k$  denotes a particular control parameter which is known apriori and fixed. Note that Eq. (C.1) does not include a prior over actions  $\mathbf{u}$  and is instead conditional on a fixed policy  $\hat{\mathbf{u}} = \hat{u}_{t+1:T}$ . We use the  $\hat{\cdot}$  notation on  $\mathbf{u}$  to indicate that it is treated as a known value instead of a random variable.

The EFE is evaluated as a function of a particular policy. The EFE is defined as [33]

$$G[q; \hat{\mathbf{u}}] = \sum_{k=t+1}^T \iint p(x_k | z_k) \underbrace{q(z_k | \hat{\mathbf{u}}_k) \log \frac{q(z_k | \hat{\mathbf{u}}_k)}{p(x_k, z_k | \hat{\mathbf{u}}_k)}}_{\text{VFE conditioned on } \hat{\mathbf{u}}_k} dx_k dz_k \quad (\text{C.2a})$$

$$= \sum_{k=t+1}^T \iint p(x_k | z_k) q(z_k | \hat{\mathbf{u}}_k) \log \frac{q(z_k | \hat{\mathbf{u}}_k)}{p(z_k | x_k, \hat{\mathbf{u}}_k)} - \log p(x_k) dx_k dz_k \quad (\text{C.2b})$$

where  $p(x_k)$  denotes a goal prior over preferred observations. Note that  $p(x_k)$  is not part of the generative model in Eq. (C.1). To compute Eq. (C.2) we also need

$$p(x_k, z_k | \hat{\mathbf{u}}_k) = \int p(x_k | z_k) p(z_k | z_{k-1}, \hat{\mathbf{u}}_k) q(z_{k-1}) dz_{k-1} \quad (\text{C.3})$$

meaning we use the **forward prediction** from the previous time step to compute  $p(x_k, z_k | \hat{\mathbf{u}}_k)$ . If we further we assume that

$$q(x_k, z_k | \hat{\mathbf{u}}_k) = p(x_k | z_k) q(z_k | \hat{\mathbf{u}}_k) \quad (\text{C.4})$$

It can be shown [22, 33, 60, 97] that (C.2) can be decomposed into a bound on a mutual information term and a cross-entropy loss between predicted observations and the goal prior.

## C.2 VFE and GFE

In this section, we walk through how the formulation of generalised free energy (GFE) given by [81] reduces to the variational free energy (VFE) when observations are available. To show how this comes about, we note that [81] assumes that

$$q(\mathbf{x}_k | \mathbf{z}_k) = \begin{cases} \delta(\mathbf{x}_k - \hat{\mathbf{x}}_k) & \text{if } k \leq t \\ p(\mathbf{x}_k | \mathbf{z}_k) & \text{if } k > t \end{cases}. \quad (\text{C.5})$$

where  $t$  is the current time step and  $\hat{\mathbf{x}}_k$  denotes the observed data point at time step  $k$ . This is a problematic move since  $q(\mathbf{x}_k | \mathbf{z}_k)$  is no longer a function of  $\mathbf{z}_k$  for  $k \leq t$ . However if we do take Eq. (5.16) as valid and further assume

$$q(\mathbf{x}_k, \mathbf{z}_k | \hat{\mathbf{u}}_k) = q(\mathbf{x}_k | \mathbf{z}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k) \quad (\text{C.6a})$$

$$\tilde{p}(\mathbf{x}_k) = 1 \text{ if } k \leq t \quad (\text{C.6b})$$

and note that

$$q(\mathbf{x}_k | \hat{\mathbf{u}}_k) = \int q(\mathbf{x}_k | \mathbf{z}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k)d\mathbf{z}_k. \quad (\text{C.7})$$

We can plug this result into Eq. (5.15) and obtain for  $k \leq t$

$$\begin{aligned} & \sum_{k=1}^t \iint q(\mathbf{x}_k | \mathbf{z}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k) \log \frac{q(\mathbf{x}_k | \hat{\mathbf{u}}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k)}{p(\mathbf{x}_k, \mathbf{z}_k | \hat{\mathbf{u}}_k)\tilde{p}(\mathbf{x}_k)} d\mathbf{x}_k d\mathbf{z}_k \\ &= \sum_{k=1}^t \iint q(\mathbf{x}_k | \mathbf{z}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k) \log \frac{\int [q(\mathbf{x}_k | \mathbf{z}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k)] d\mathbf{z}_k q(\mathbf{z}_k | \hat{\mathbf{u}}_k)}{p(\mathbf{x}_k, \mathbf{z}_k | \hat{\mathbf{u}}_k)} d\mathbf{x}_k d\mathbf{z}_k \end{aligned} \quad (\text{C.8})$$

where the last line follows from Eq. (C.6b). Then by Eq. (5.16) we have

$$= \sum_{k=1}^t \iint \delta(\mathbf{x}_k - \hat{\mathbf{x}}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k) \log \frac{\int [\delta(\mathbf{x}_k - \hat{\mathbf{x}}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k)] d\mathbf{z}_k q(\mathbf{z}_k | \hat{\mathbf{u}}_k)}{p(\mathbf{x}_k, \mathbf{z}_k | \hat{\mathbf{u}}_k)} d\mathbf{x}_k d\mathbf{z}_k \quad (\text{C.9})$$

Now we pull  $\delta(\mathbf{x}_k - \hat{\mathbf{x}}_k)$  out of the integral and solve to find

$$= \sum_{k=1}^t \iint \delta(\mathbf{x}_k - \hat{\mathbf{x}}_k)q(\mathbf{z}_k | \hat{\mathbf{u}}_k) \log \frac{\delta(\mathbf{x}_k - \hat{\mathbf{x}}_k) \overbrace{\left[ \int q(\mathbf{z}_k | \hat{\mathbf{u}}_k) d\mathbf{z}_k \right]}^{\text{Integrates to 1}} q(\mathbf{z}_k | \hat{\mathbf{u}}_k)}{p(\mathbf{x}_k, \mathbf{z}_k | \hat{\mathbf{u}}_k)} d\mathbf{x}_k d\mathbf{z}_k \quad (\text{C.10a})$$

$$= \sum_{k=1}^t \int q(\mathbf{z}_k | \hat{\mathbf{u}}_k) \log \frac{q(\mathbf{z}_k | \hat{\mathbf{u}}_k)}{p(\hat{\mathbf{x}}_k, \mathbf{z}_k | \hat{\mathbf{u}}_k)} d\mathbf{z}_k \quad (\text{C.10b})$$



which we can recognise as a VFE with data constraints.

### C.3 Proofs of Local Stationary Solutions in Section 5.6.2

This section derives the stationary points of a local GFE constrained objective, as defined by Fig. 5.20.

#### C.3.1 Proof of Lemma 1

*Proof.* Writing out the terms in the Lagrangian and simplifying, we obtain

$$\mathcal{L}[q_x] = \mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\mathbf{z})q(\boldsymbol{\theta})}[\log q(x)] + \psi_x \left[ \int q(x) dx - 1 \right] + C_x. \quad (\text{C.11})$$

The functional derivative then becomes

$$\frac{\delta \mathcal{L}}{\delta q_x} = \frac{\mathbb{E}_{q(\mathbf{z})q(\boldsymbol{\theta})}[p(x|\mathbf{z}, \boldsymbol{\theta})]}{q(x)} + \psi_x \stackrel{!}{=} 0. \quad (\text{C.12})$$

Solving this equation for  $q_x$  results in Eq.(5.40).  $\square$

#### C.3.2 Proof of Lemma 2

*Proof.* Writing out the Lagrangian, we obtain

$$\begin{aligned} \mathcal{L}[q_{\mathbf{z}}] = & \mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\mathbf{z})q(\boldsymbol{\theta})q(\mathbf{w})q(\boldsymbol{\phi})} \left[ \log \frac{q(x)}{p(x|\mathbf{z}, \boldsymbol{\theta})\tilde{p}(x|\mathbf{w}, \boldsymbol{\phi})} \right] + \mathbb{E}_{q(\mathbf{z})}[\log q(\mathbf{z})] \\ & + \psi_{\mathbf{z}} \left[ \int q(\mathbf{z}) d\mathbf{z} - 1 \right] + \sum_{i \in \mathcal{E}(\mathbf{z})} \int \lambda_{ip}(z_i) \left[ q(z_i) - \int q(\mathbf{z}) d\mathbf{z}_{\setminus i} \right] dz_i + C_{\mathbf{z}}. \end{aligned} \quad (\text{C.13})$$

The functional derivative then becomes

$$\frac{\delta \mathcal{L}}{\delta q_{\mathbf{z}}} = \mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\boldsymbol{\theta})q(\mathbf{w})q(\phi)} \left[ \log \frac{q(x)}{p(x|\mathbf{z}, \boldsymbol{\theta})\tilde{p}(x|\mathbf{w}, \phi)} \right] + \log q(\mathbf{z}) + 1 + \psi_{\mathbf{z}} - \sum_{i \in \mathcal{E}(\mathbf{z})} \lambda_{ip}(z_i) \quad (\text{C.14})$$

$$= \mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\boldsymbol{\theta})} \left[ \log \frac{q(x)}{p(x|\mathbf{z}, \boldsymbol{\theta})\tilde{f}(x)} \right] + \log q(\mathbf{z}) - \sum_{i \in \mathcal{E}(\mathbf{z})} \lambda_{ip}(z_i) + Z_{\mathbf{z}} \quad (\text{C.15})$$

$$= -\log \tilde{f}(\mathbf{z}) + \log q(\mathbf{z}) - \sum_{i \in \mathcal{E}(\mathbf{z})} \lambda_{ip}(z_i) + Z_{\mathbf{z}}, \quad (\text{C.16})$$

where  $Z_{\mathbf{z}}$  absorbs all terms independent of  $\mathbf{z}$ , and with  $\tilde{f}(\mathbf{z})$  and  $\tilde{f}(x)$  given by Eq.(5.43a) and Eq.(5.43b) respectively. Setting to zero and solving for  $q_{\mathbf{z}}$ , we obtain

$$\log q^*(\mathbf{z}) = \log \tilde{f}(\mathbf{z}) + \sum_{i \in \mathcal{E}(\mathbf{z})} \lambda_{ip}(z_i) - Z_{\mathbf{z}}. \quad (\text{C.17})$$

Exponentiating on both sides, identifying  $\vec{\mu}(z_i) = \exp \lambda_{ip}(z_i)$ , and normalizing then results in Eq.(5.42).  $\square$

### C.3.3 Proof of Lemma 3

*Proof.* Writing out the Lagrangian, we obtain

$$\begin{aligned} \mathcal{L}[q_{\mathbf{w}}] &= \mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\mathbf{z})q(\boldsymbol{\theta})q(\mathbf{w})q(\phi)} \left[ \log \frac{q(x)}{p(x|\mathbf{z}, \boldsymbol{\theta})\tilde{p}(x|\mathbf{w}, \phi)} \right] \\ &\quad + \mathbb{E}_{q(\mathbf{z})} [\log q(\mathbf{z})] + \psi_{\mathbf{w}} \left[ \int q(\mathbf{w}) d\mathbf{w} - 1 \right] \\ &\quad + \sum_{i \in \mathcal{E}(\mathbf{w})} \int \lambda_{i\tilde{p}}(w_i) \left[ q(w_i) - \int q(\mathbf{w}) d\mathbf{w}_{\setminus i} \right] dw_i + C_{\mathbf{w}}. \end{aligned} \quad (\text{C.18})$$

The functional derivative then becomes

$$\frac{\delta \mathcal{L}}{\delta q_{\mathbf{w}}} = \mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\mathbf{z})q(\boldsymbol{\theta})q(\boldsymbol{\phi})} \left[ \log \frac{q(x)}{p(x|\mathbf{z}, \boldsymbol{\theta})\tilde{p}(x|\mathbf{w}, \boldsymbol{\phi})} \right] + \log q(\mathbf{w}) + 1 + \psi_{\mathbf{w}} - \sum_{i \in \mathcal{E}(\mathbf{w})} \lambda_{i\tilde{p}}(w_i) \quad (\text{C.19})$$

$$= -\mathbb{E}_{p(x|\mathbf{z}, \boldsymbol{\theta})q(\mathbf{z})q(\boldsymbol{\theta})q(\boldsymbol{\phi})} [\log \tilde{p}(x|\mathbf{w}, \boldsymbol{\phi})] + \log q(\mathbf{w}) - \sum_{i \in \mathcal{E}(\mathbf{w})} \lambda_{i\tilde{p}}(w_i) + Z_{\mathbf{w}} \quad (\text{C.20})$$

$$= -\mathbb{E}_{q(x)q(\boldsymbol{\phi})} [\log \tilde{p}(x|\mathbf{w}, \boldsymbol{\phi})] + \log q(\mathbf{w}) - \sum_{i \in \mathcal{E}(\mathbf{w})} \lambda_{i\tilde{p}}(w_i) + Z_{\mathbf{w}} \quad (\text{C.21})$$

$$= -\log \tilde{f}(\mathbf{w}) + \log q(\mathbf{w}) - \sum_{i \in \mathcal{E}(\mathbf{w})} \lambda_{i\tilde{p}}(w_i) + Z_{\mathbf{w}} \quad (\text{C.22})$$

where  $Z_{\mathbf{w}}$  absorbs all terms independent of  $\mathbf{w}$ , the second-to-last step uses the result of Eq.(5.40), and where  $\tilde{f}(\mathbf{w})$  is given by Eq.(5.46).

Setting to zero and solving for  $q_{\mathbf{w}}$ , we obtain

$$\log q^*(\mathbf{w}) = \log \tilde{f}(\mathbf{w}) + \sum_{i \in \mathcal{E}(\mathbf{w})} \lambda_{i\tilde{p}}(w_i) - Z_{\mathbf{w}}. \quad (\text{C.23})$$

Exponentiating on both sides, identifying  $\tilde{\mu}(w_i) = \exp \lambda_{i\tilde{p}}(w_i)$ , and normalizing results in Eq.(5.45). □

## C.4 Proofs of Message Update Expressions in Section 5.6.3

### C.4.1 Proof of Theorem 1

*Proof.* Firstly, Lemma 3 provides us with the stationary solutions to  $\mathcal{L}[q]$  as a functional of  $q_{\mathbf{w}}$ . Secondly, the stationary solution of  $\mathcal{L}[q]$  as a functional of the edge-local variational distribution  $q_j(w_j)$ , defined as

$$\begin{aligned} \mathcal{L}[q_j] &= \mathbb{H}[q_j] + \psi_j \left[ \int q(w_j) dz_j - 1 \right] \\ &+ \sum_{a \in \mathcal{V}(j)} \int \lambda_{ja}(w_j) \left[ q(w_j) - \int q(\mathbf{w}) d\mathbf{w}_{\setminus j} \right] dw_j + C_j, \end{aligned} \quad (\text{C.24})$$

where  $C_j$  absorbs all terms independent of  $q_j$ , directly follows from [116, Lemma 2], as

$$q^*(w_j) = \frac{\bar{\mu}(w_j)\bar{\mu}(w_j)}{\int \bar{\mu}(w_j)\bar{\mu}(w_j) dw_j}. \quad (\text{C.25})$$

We then apply the marginalisation constraint on the edge- and node-local variational distributions

$$q^*(w_j) = \int q^*(\mathbf{w}) d\mathbf{w}_{\setminus j}. \quad (\text{C.26})$$

Substituting the stationary solutions we can directly apply [116, Theorem 2]. It then follows that fixed points of Eq.(5.47) correspond to stationary solutions of  $\mathcal{L}[q]$ .  $\square$

The notation  $\mu_{\bullet}(w_j) = \bar{\mu}^{(n+1)}(w_j)$  then conveniently represents the recursive message update schedule.

### C.4.2 Proof of Theorem 2

*Proof.* We follow the same procedure as before. Firstly, Lemma 2 provides us with the stationary solutions of  $\mathcal{L}[q]$  as a functional of  $z_j$ . Secondly, the Lagrangian as a functional of  $q_j(z_j)$  is then constructed as

$$\begin{aligned} \mathcal{L}[q_j] &= \mathbb{H}[q_j] + \psi_j \left[ \int q(z_j) dz_j - 1 \right] \\ &+ \sum_{a \in \mathcal{V}(j)} \int \lambda_{ja}(z_j) \left[ q(z_j) - \int q(\mathbf{z}) dz_{\setminus j} \right] dz_j + C_j, \end{aligned} \quad (\text{C.27})$$

where  $C_j$  absorbs all terms independent of  $q_j$ . The stationary solution again follows from [116, Lemma 2],

$$q^*(z_j) = \frac{\bar{\mu}(z_j)\bar{\mu}(z_j)}{\int \bar{\mu}(z_j)\bar{\mu}(z_j) dz_j}. \quad (\text{C.28})$$

From the marginalisation constraint

$$q^*(z_j) = \int q^*(\mathbf{z}) dz_{\setminus j}, \quad (\text{C.29})$$

we can again directly apply [116, Theorem 2], from which it follows that fixed points of Eq.(5.48) correspond to stationary solutions of  $\mathcal{L}[q]$ .  $\square$

In the schedule, the fixed-point iteration is then represented by  $\mu_{\otimes}(z_j) = \bar{\mu}^{(n+1)}(z_j)$ .

### C.4.3 Proof of Corollary 1

*Proof.* From the marginalisation constraint we obtain Eq.(5.49). We then parameterise  $q_{\mathbf{z}}$  with statistics  $\nu$  and substitute Eq.(5.42), Eq.(5.43) and Eq.(5.40) to obtain a recursive dependence on  $\nu$ .  $\square$

## C.5 Derivations of Message Updates in Figure 5.21

Here we derive the message updates for the discrete-variable submodel of Fig. 5.21. To streamline the derivations, we first derive some intermediate results.

### C.5.1 Intermediate Results

First we express the log-observation model,

$$\log p(\mathbf{x} | \mathbf{z}, \mathbf{A}) = \log \text{Cat}(\mathbf{x} | \mathbf{A}\mathbf{z}) \quad (\text{C.30a})$$

$$= \sum_j \sum_k x_j \log(A_{jk}) z_k \quad (\text{C.30b})$$

$$= \mathbf{x}^T \log(\mathbf{A})\mathbf{z}, \quad (\text{C.30c})$$

where the final logarithm is taken element-wise. Then, from Eq.(5.40)

$$\log q(\mathbf{x}) = \log (\mathbb{E}_{q(\mathbf{z})q(\mathbf{A})} [p(\mathbf{x} | \mathbf{z}, \boldsymbol{\theta})]) \quad (\text{C.31a})$$

$$= \log (\mathbb{E}_{q(\mathbf{z})q(\mathbf{A})} [\text{Cat}(\mathbf{x} | \mathbf{A}\mathbf{z})]) \quad (\text{C.31b})$$

$$\approx \log \text{Cat}(\mathbf{x} | \overline{\mathbf{A}\mathbf{z}}) \quad (\text{C.31c})$$

$$= \mathbf{x}^T \log(\overline{\mathbf{A}\mathbf{z}}), \quad (\text{C.31d})$$

Where we used a tentative decision approximation to compute the expectations with respect to  $q(\mathbf{A}) \stackrel{!}{=} \delta(\mathbf{A} - \overline{\mathbf{A}})$ . Next, from Eq.(5.43),

$$\log \tilde{f}(\mathbf{x}) = \mathbb{E}_{q(\mathbf{c})} [\log \tilde{p}(\mathbf{x} | \mathbf{c})] \quad (\text{C.32a})$$

$$= \mathbb{E}_{q(\mathbf{c})} [\log \text{Cat}(\mathbf{x} | \mathbf{c})] \quad (\text{C.32b})$$

$$= \mathbb{E}_{q(\mathbf{c})} [\mathbf{x}^T \log \mathbf{c}] \quad (\text{C.32c})$$

$$= \mathbf{x}^T \overline{\log \mathbf{c}}. \quad (\text{C.32d})$$

Combining these results, from Eq.(5.43),

$$\log \tilde{f}(\mathbf{z}) = \mathbb{E}_{p(\mathbf{x}|\mathbf{z},\mathbf{A})q(\mathbf{A})} \left[ \log \frac{p(\mathbf{x} | \mathbf{z}, \mathbf{A}) \tilde{f}(\mathbf{x})}{q(\mathbf{x})} \right] \quad (\text{C.33a})$$

$$= \mathbb{E}_{p(\mathbf{x}|\mathbf{z},\mathbf{A})q(\mathbf{A})} [\mathbf{x}^T \log(\mathbf{A}\mathbf{z}) + \mathbf{x}^T \overline{\log \mathbf{c}} - \mathbf{x}^T \log(\overline{\mathbf{A}\mathbf{z}})] \quad (\text{C.33b})$$

$$= \mathbb{E}_{q(\mathbf{A})} [(\mathbf{A}\mathbf{z})^T \log(\mathbf{A}\mathbf{z}) + (\mathbf{A}\mathbf{z})^T \overline{\log \mathbf{c}} - (\mathbf{A}\mathbf{z})^T \log(\overline{\mathbf{A}\mathbf{z}})] \quad (\text{C.33c})$$

$$= \mathbb{E}_{q(\mathbf{A})} \left[ \mathbf{z}^T \text{diag}(\mathbf{A}^T \log \mathbf{A}) + (\mathbf{A}\mathbf{z})^T \overline{\log \mathbf{c}} - (\mathbf{A}\mathbf{z})^T \log(\overline{\mathbf{A}\mathbf{z}}) \right] \quad (\text{C.33d})$$

$$= \mathbb{E}_{q(\mathbf{A})} [-\mathbf{z}^T \mathbf{h}(\mathbf{A}) + (\mathbf{A}\mathbf{z})^T \overline{\log \mathbf{c}} - (\mathbf{A}\mathbf{z})^T \log(\overline{\mathbf{A}\mathbf{z}})] \quad (\text{C.33e})$$

$$= -\mathbf{z}^T \overline{\mathbf{h}(\mathbf{A})} + (\overline{\mathbf{A}\mathbf{z}})^T \overline{\log \mathbf{c}} - (\overline{\mathbf{A}\mathbf{z}})^T \log(\overline{\mathbf{A}\mathbf{z}}) \quad (\text{C.33f})$$

$$= \mathbf{z}^T \boldsymbol{\rho}, \quad (\text{C.33g})$$

with

$$\boldsymbol{\rho} = \overline{\mathbf{A}}^T (\overline{\log \mathbf{c}} - \log(\overline{\mathbf{A}\mathbf{z}})) - \overline{\mathbf{h}(\mathbf{A})}, \quad (\text{C.34})$$

and

$$\mathbf{h}(\mathbf{A}) = -\text{diag}\left(\mathbf{A}^T \log \mathbf{A}\right), \quad (\text{C.35})$$

the entropies of the columns of matrix  $\mathbf{A}$ . With these results we can derive the local GFE and messages.

### C.5.2 Average Energy

$$U_{\mathbf{x}}[q] = \mathbb{E}_{p(\mathbf{x}|\mathbf{z},\mathbf{A})q(\mathbf{z})q(\mathbf{A})q(\mathbf{c})} \left[ \log \frac{q(\mathbf{x})}{p(\mathbf{x} | \mathbf{z}, \mathbf{A})\tilde{p}(\mathbf{x} | \mathbf{c})} \right] \quad (\text{C.36a})$$

$$= -\mathbb{E}_{p(\mathbf{x}|\mathbf{z},\mathbf{A})q(\mathbf{z})q(\mathbf{A})} \left[ \log \frac{p(\mathbf{x} | \mathbf{z}, \mathbf{A})\tilde{f}(\mathbf{x})}{q(\mathbf{x})} \right] \quad (\text{C.36b})$$

$$= -\mathbb{E}_{q(\mathbf{z})} \left[ \log \tilde{f}(\mathbf{z}) \right] \quad (\text{C.36c})$$

$$= -\bar{\mathbf{z}}^T \boldsymbol{\rho}, \quad (\text{C.36d})$$

with  $\boldsymbol{\rho}$  given by Eq.(C.34).

### C.5.3 Message ①

We apply the result of Theorem 1 and express the downward message,

$$\log \mu_{\bullet}(\mathbf{c}) = \log \tilde{f}(\mathbf{c}) \quad (\text{C.37a})$$

$$= \mathbb{E}_{q(\mathbf{x})} [\log \tilde{p}(\mathbf{x} | \mathbf{c})] \quad (\text{C.37b})$$

$$= \mathbb{E}_{q(\mathbf{x})} [\mathbf{x}^T \log \mathbf{c}] \quad (\text{C.37c})$$

$$= (\bar{\mathbf{A}}\bar{\mathbf{z}})^T \log \mathbf{c}. \quad (\text{C.37d})$$

Exponentiation on both sides then yields

$$\mu_{\bullet}(\mathbf{c}) \propto \text{Dir}(\mathbf{c} | \bar{\mathbf{A}}\bar{\mathbf{z}} + \mathbf{1}). \quad (\text{C.38})$$

### C.5.4 Direct Result for Message ②

Here we apply the result of Theorem 2 to directly compute the backward message for the state. As explained before, this update may lead to diverging FE for some algorithms.

$$\log \mu_{\ominus}(\mathbf{z}) = \log \tilde{f}(\mathbf{z}) \quad (\text{C.39a})$$

$$= \mathbf{z}^T \boldsymbol{\rho}, \quad (\text{C.39b})$$

with  $\boldsymbol{\rho}$  given by Eq.(C.34). Exponentiation on both sides then yields

$$\mu_{\ominus}(\mathbf{z}) \propto \text{Cat}(\mathbf{z} \mid \sigma(\boldsymbol{\rho})), \quad (\text{C.40})$$

with  $\sigma$  the softmax function.

### C.5.5 Indirect Result for Message ②

Here we apply the result of Corollary 1. We set the statistic  $\nu = \bar{\mathbf{z}}$ , assume message ① (proportionally) Categorical, and use Eq.(5.50) to express

$$\log q(\mathbf{z}; \bar{\mathbf{z}}) = \log \tilde{f}(\mathbf{z}; \bar{\mathbf{z}}) + \log \mu_{\oplus}(\mathbf{z}) + C_{\mathbf{z}} \quad (\text{C.41a})$$

$$= \mathbf{z}^T \boldsymbol{\rho}(\bar{\mathbf{z}}) + \mathbf{z}^T \log \mathbf{d} + C_{\mathbf{z}}, \quad (\text{C.41b})$$

with  $\boldsymbol{\rho}(\bar{\mathbf{z}})$  given by Eq.(C.34), where the circular dependence on  $\bar{\mathbf{z}}$  has been made explicit.

Exponentiating on both sides and normalizing, we obtain

$$q(\mathbf{z}; \bar{\mathbf{z}}) = \text{Cat}(\mathbf{z} \mid \bar{\mathbf{z}}), \text{ with} \quad (\text{C.42})$$

$$\bar{\mathbf{z}} = \sigma(\boldsymbol{\rho}(\bar{\mathbf{z}}) + \log \mathbf{d}),$$

and  $\sigma$  the softmax function. We then approach this equation as a root-finding problem, and use Newton's method to find an  $\bar{\mathbf{z}}^*$  that solves for Eq.(C.42). We can then compute the backward message through Eq.(5.49), as

$$\mu_{\ominus}(\mathbf{z}) \propto q(\mathbf{z}; \bar{\mathbf{z}}^*) / \mu_{\oplus}(\mathbf{z}) \quad (\text{C.43a})$$

$$= \text{Cat}(\mathbf{z} \mid \bar{\mathbf{z}}^*) / \text{Cat}(\mathbf{z} \mid \mathbf{d}) \quad (\text{C.43b})$$

$$\propto \text{Cat}(\mathbf{z} \mid \sigma(\log \bar{\mathbf{z}}^* - \log \mathbf{d})). \quad (\text{C.43c})$$



### C.5.6 Direct Result for Message ③

Here we apply the result of Theorem 2 and use the symmetry between  $\mathbf{z}$  and  $\mathbf{A}$  to directly compute the backward message for the state, as

$$\log \mu_{\textcircled{3}}(\mathbf{A}) = \mathbb{E}_{p(\mathbf{x}|\mathbf{z},\mathbf{A})q(\mathbf{z})} \left[ \log \frac{p(\mathbf{x} | \mathbf{z}, \mathbf{A}) \tilde{f}(\mathbf{x})}{q(\mathbf{x})} \right] \quad (\text{C.44a})$$

$$= \mathbb{E}_{p(\mathbf{x}|\mathbf{z},\mathbf{A})q(\mathbf{z})} \left[ \mathbf{x}^T \log(\mathbf{A})\mathbf{z} + \mathbf{x}^T \overline{\log \mathbf{c}} - \mathbf{x}^T \log(\overline{\mathbf{A}\mathbf{z}}) \right] \quad (\text{C.44b})$$

$$= \mathbb{E}_{q(\mathbf{z})} \left[ (\mathbf{A}\mathbf{z})^T \log(\mathbf{A})\mathbf{z} + (\mathbf{A}\mathbf{z})^T \overline{\log \mathbf{c}} - (\mathbf{A}\mathbf{z})^T \log(\overline{\mathbf{A}\mathbf{z}}) \right] \quad (\text{C.44c})$$

$$= \mathbb{E}_{q(\mathbf{z})} \left[ \mathbf{z}^T \text{diag}(\mathbf{A}^T \log \mathbf{A}) + (\mathbf{A}\mathbf{z})^T \overline{\log \mathbf{c}} - (\mathbf{A}\mathbf{z})^T \log(\overline{\mathbf{A}\mathbf{z}}) \right] \quad (\text{C.44d})$$

$$= \mathbb{E}_{q(\mathbf{z})} \left[ -\mathbf{z}^T \mathbf{h}(\mathbf{A}) + (\mathbf{A}\mathbf{z})^T \overline{\log \mathbf{c}} - (\mathbf{A}\mathbf{z})^T \log(\overline{\mathbf{A}\mathbf{z}}) \right] \quad (\text{C.44e})$$

$$= \overline{\mathbf{z}}^T \xi(\mathbf{A}), \quad (\text{C.44f})$$

with

$$\xi(\mathbf{A}) = \mathbf{A}^T (\overline{\log \mathbf{c}} - \log(\overline{\mathbf{A}\mathbf{z}})) - \mathbf{h}(\mathbf{A}). \quad (\text{C.45})$$

## C.6 The Transition Mixture Node

In this section we derive the message passing rules required for the Transition Mixture node used in our experiments on policy inference. Before doing so, we first establish some preliminary results for the categorical distribution and the standard transition node. In this section we will also on occasion use an overbar to indicate an expectation,  $\mathbb{E}_{q(x)}[g(x)] = \overline{g(x)}$ . The categorical distribution is given by

$$\text{Cat}(\mathbf{x} | \mathbf{z}) = \prod_{i=1}^I \mathbf{z}_i^{\mathbf{x}_i} \quad (\text{C.46a})$$

$$= \sum_{i=1}^I \mathbf{x}_i \mathbf{z}_i \quad (\text{C.46b})$$

where  $\mathbf{x}$  and  $\mathbf{z}$  are both one-hot encoded vectors. The move Eq. (C.46a) to Eq. (C.46b) is possible only because  $\mathbf{x}$  is one-hot encoded. Similarly, we have for the standard Transition node that

$$\mathcal{C}at(\mathbf{x} \mid \mathbf{A}\mathbf{z}) = \prod_{i=1}^I \prod_{j=1}^I [\mathbf{A}_{ij}]^{\mathbf{x}_i \mathbf{z}_j} \quad (\text{C.47a})$$

$$= \sum_{i=1}^I \sum_{j=1}^I \mathbf{x}_i \mathbf{z}_j \mathbf{A}_{ij} \quad (\text{C.47b})$$

and again, since  $\mathbf{x}$  and  $\mathbf{z}$  are one-hot encoded

$$\log \mathcal{C}at(\mathbf{x} \mid \mathbf{A}\mathbf{z}) = \log \left[ \prod_{i=1}^I \prod_{j=1}^I [\mathbf{A}_{ij}]^{\mathbf{x}_i \mathbf{z}_j} \right] \quad (\text{C.48a})$$

$$= \sum_{i=1}^I \sum_{j=1}^I \log \left[ \mathbf{A}_{ij}^{\mathbf{x}_i \mathbf{z}_j} \right] \quad (\text{C.48b})$$

$$= \sum_{i=1}^I \sum_{j=1}^I \mathbf{x}_i \mathbf{z}_j \log \mathbf{A}_{ij} \quad (\text{C.48c})$$

Now we are ready to start derivations for the transition mixture node. The transition mixture node has the node function

$$f(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{A}) = \prod_k \mathcal{C}at(\mathbf{x} \mid \mathbf{A}_k \mathbf{z})^{\mathbf{y}_k} \quad (\text{C.49a})$$

$$= \prod_k \left[ \prod_i \prod_j [\mathbf{A}_k]_{ij}^{\mathbf{z}_i \mathbf{x}_j} \right]^{\mathbf{y}_k} \quad (\text{C.49b})$$

$$= \prod_{i,j,k} \mathbf{A}_{ijk}^{\mathbf{z}_i \mathbf{x}_j \mathbf{y}_k} \quad (\text{C.49c})$$

Where we use  $i$  to index over columns,  $j$  to index over rows and  $k$  to index over factors.  $\mathbf{A}$  is a 3-tensor that is normalized over columns. In other words, each slice of  $\mathbf{A}$  corresponds to a valid transition matrix and slices are indexed by  $k$ . We assume a structured mean field factorisation such that

$$q(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{A}) = q(\mathbf{x}, \mathbf{y}, \mathbf{z}) \prod_k q(\mathbf{A}_k) \quad (\text{C.50})$$

The constrained Forney-style factor graph (CFFG) of the transition mixture node is shown in Fig. C.1

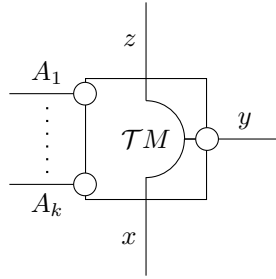


Figure C.1: The transition mixture node

We now define

$$\log f_{\mathbf{A}}(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \mathbb{E}_{q(\mathbf{A})} \left[ \log f(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{A}) \right] \quad (\text{C.51a})$$

$$= \mathbb{E}_{q(\mathbf{A})} \left[ \sum_{ijk} \mathbf{z}_i \mathbf{x}_j \mathbf{y}_k \log \mathbf{A}_{ijk} \right] \quad (\text{C.51b})$$

$$= \sum_k \mathbf{y}_k \mathbb{E}_{q(\mathbf{A}_k)} \left[ \sum_{ij} \mathbf{z}_i \mathbf{x}_j \log \mathbf{A}_{ijk} \right] \quad (\text{C.51c})$$

$$= \sum_k \mathbf{y}_k \sum_{ij} \mathbf{z}_i \mathbf{x}_j \left[ \overline{\log \mathbf{A}_k} \right]_{ij} \quad (\text{C.51d})$$

which implies that

$$f_A(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \exp \left( \sum_k \mathbf{y}_k \sum_{ijk} \mathbf{z}_i \mathbf{x}_j \left[ \overline{\log \mathbf{A}_k} \right]_{ij} \right) \quad (\text{C.52a})$$

$$= \prod_{ijk} \exp \left( \overline{\log \mathbf{A}_k} \right)^{\mathbf{z}_i \mathbf{x}_j \mathbf{y}_k} \quad (\text{C.52b})$$

$$= \sum_{ijk} \mathbf{z}_i \mathbf{x}_j \mathbf{y}_k \underbrace{\exp \left( \overline{\log \mathbf{A}_k} \right)}_{\tilde{\mathbf{A}}_{ijk}} \quad (\text{C.52c})$$

Now we are ready to derive the desired messages. The first message  $\nu(\cdot)$  will be towards  $\mathbf{x}$ . If we use  $\mu(\mathbf{y})$  to denote the incoming message on the edge  $\mathbf{y}$  (similar for  $\mu(\mathbf{z})$ ), then the message towards  $\mathbf{x}$  is given by

$$\nu(\mathbf{x}) = \sum_{\mathbf{z}} \sum_{\mathbf{y}} \mu(\mathbf{z}) \mu(\mathbf{y}) \exp \left( \mathbb{E}_{q(\mathbf{A})} \log f(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{A}) \right) \quad (\text{C.53a})$$

$$= \sum_{\mathbf{z}} \sum_{\mathbf{y}} \mu(\mathbf{z}) \mu(\mathbf{y}) f_A(\mathbf{x}, \mathbf{y}, \mathbf{z}) \quad (\text{C.53b})$$

$$= \sum_{\mathbf{z}} \sum_{\mathbf{y}} \mu(\mathbf{z}) \mu(\mathbf{y}) \sum_{ijk} \mathbf{z}_i \mathbf{x}_j \mathbf{y}_k \tilde{\mathbf{A}}_{ijk} \quad (\text{C.53c})$$

Assuming the incoming messages are Categorically distributed, we can continue as

$$= \mathbb{E}_{\text{Cat}(\mathbf{z}|\pi_z)} \mathbb{E}_{\text{Cat}(\mathbf{y}|\pi_y)} \sum_{ijk} \mathbf{z}_i \mathbf{x}_j \mathbf{y}_k \tilde{\mathbf{A}}_{ijk} \quad (\text{C.54a})$$

$$= \sum_{ijk} \pi_{zi} \pi_{yk} \mathbf{x}_j \tilde{\mathbf{A}}_{ijk} \quad (\text{C.54b})$$

$$= \prod_j \left[ \sum_{ik} \pi_{zi} \pi_{yk} \tilde{\mathbf{A}}_{ijk} \right]^{\mathbf{x}_j} \quad (\text{C.54c})$$

where the last line follows from  $\mathbf{x}$  being one-hot encoded. At this point, we can recognise the message

$$\nu(\mathbf{x}) \propto \text{Cat}(\mathbf{x} | \boldsymbol{\rho}) \text{ where } \boldsymbol{\rho}_j = \frac{\sum_{ik} \pi_{zi} \pi_{yk} \tilde{\mathbf{A}}_{ijk}}{\sum_{ijk} \pi_{zi} \pi_{yk} \tilde{\mathbf{A}}_{ijk}} \quad (\text{C.55})$$

By symmetry, similar results hold for messages  $\nu(\mathbf{z})$  and  $\nu(\mathbf{y})$ .  $\nu(\mathbf{z})$  is given by

$$\nu(\mathbf{z}) = \sum_x \sum_y \mu(x)\mu(y)f_A(\mathbf{x}, \mathbf{y}, \mathbf{z}) \quad (\text{C.56a})$$

$$\propto \text{Cat}(\mathbf{z} \mid \boldsymbol{\rho}) \text{ where } \rho_i = \frac{\sum_{jk} \pi_{xj} \pi_{yk} \tilde{\mathbf{A}}_{ijk}}{\sum_{ijk} \pi_{xj} \pi_{yk} \tilde{\mathbf{A}}_{ijk}}, \quad (\text{C.56b})$$

and  $\nu(\mathbf{y})$  evaluates to

$$\nu(\mathbf{y}) = \sum_x \sum_z \mu(\mathbf{x})\mu(\mathbf{z})f_A(\mathbf{x}, \mathbf{y}, \mathbf{z}) \quad (\text{C.57a})$$

$$\propto \text{Cat}(\mathbf{y} \mid \boldsymbol{\rho}) \text{ where } \rho_k = \frac{\sum_{ij} \pi_{x,j} \pi_{z,i} \tilde{\mathbf{A}}_{ijk}}{\sum_{ijk} \pi_{x,j} \pi_{z,i} \tilde{\mathbf{A}}_{ijk}}. \quad (\text{C.57b})$$

To compute the message towards the  $n$ 'th candidate transition matrix  $\mathbf{A}_n$ , we will need to take an expectation with respect to  $q(\mathbf{x}, \mathbf{y}, \mathbf{z})$ . It is given by

$$q(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \mu(\mathbf{y})\mu(\mathbf{x})\mu(\mathbf{z})f_A(\mathbf{x}, \mathbf{y}, \mathbf{z}) \quad (\text{C.58a})$$

$$= \prod_{ijk} \pi_{xj}^{\mathbf{x}_j} \pi_{zi}^{\mathbf{z}_i} \pi_{yk}^{\mathbf{y}_k} \tilde{\mathbf{A}}_{ijk}^{\mathbf{x}_j \mathbf{z}_i \mathbf{y}_k} \quad (\text{C.58b})$$

$$= \prod_{ijk} \left[ \pi_{xj} \pi_{zi} \pi_{yk} \tilde{\mathbf{A}}_{ijk} \right]^{\mathbf{x}_j \mathbf{z}_i \mathbf{y}_k} \quad (\text{C.58c})$$

$$\propto \text{Cat}(\mathbf{x}, \mathbf{y}, \mathbf{z} \mid \mathbf{B}) \quad (\text{C.58d})$$

where  $\mathbf{B}$  is a three-dimensional contingency tensor with entries

$$\mathbf{B}_{ijk} = \frac{\pi_{xj} \pi_{zi} \pi_{yk} \tilde{\mathbf{A}}_{ijk}}{\sum_{ijk} \pi_{xj} \pi_{zi} \pi_{yk} \tilde{\mathbf{A}}_{ijk}} \quad (\text{C.59})$$

Now we can compute the message towards a transition matrix  $\mathbf{A}_n$  as

$$\log \nu(\mathbf{A}_n) = \mathbb{E}_{q(\mathbf{A}_{k \setminus n})q(\mathbf{x}, \mathbf{y}, \mathbf{z})} \left[ \log f(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{A}) \right] \quad (\text{C.60a})$$

$$= \mathbb{E}_{q(\mathbf{A}_{k \setminus n})q(\mathbf{x}, \mathbf{y}, \mathbf{z})} \left[ \sum_{ijk} \mathbf{x}_j \mathbf{z}_i \mathbf{y}_k \log \mathbf{A}_{ijk} \right] \quad (\text{C.60b})$$

$$= \mathbb{E}_{q(\mathbf{A}_{k \setminus n})} \left[ \sum_{ijk} \mathbf{B}_{ijk} \log \mathbf{A}_{ijk} \right] \quad (\text{C.60c})$$

$$= \underbrace{\sum_{k \setminus n} \left[ \sum_{ijk} \mathbf{B}_{ijk} \log \mathbf{A}_{ijk} \right]}_{\text{Constant w.r.t } \mathbf{A}_n} + \sum_{ij} \mathbf{B}_{ijn} \log \mathbf{A}_{ijn} \quad (\text{C.60d})$$

$$\propto \text{tr} \left( \mathbf{B}_{ijn} \log \mathbf{A}_{ijn} \right) \quad (\text{C.60e})$$

which implies that

$$\nu(\mathbf{A}_n) \propto \text{Dir}(\mathbf{A}_n \mid \mathbf{B}_n + 1). \quad (\text{C.61})$$

We see that messages towards each component in  $\mathbf{A}$  are distributed according to a Dirichlet distribution with parameters  $\mathbf{B}_n + 1$ . The final expression we need to derive is the average energy term for the transition mixture node. It is given by

$$U_x[q] = -\mathbb{E}_{q(\mathbf{A})q(\mathbf{x}, \mathbf{y}, \mathbf{z})} \left( \log f(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{A}) \right) \quad (\text{C.62a})$$

$$= -\mathbb{E}_{q(\mathbf{x}, \mathbf{y}, \mathbf{z})} \left( f_A(\mathbf{x}, \mathbf{y}, \mathbf{z}) \right) \quad (\text{C.62b})$$

$$= -\mathbb{E}_{q(\mathbf{x}, \mathbf{y}, \mathbf{z})} \left[ \sum_{ijk} \mathbf{x}_j \mathbf{z}_i \mathbf{y}_k \overline{[\log \mathbf{A}_k]}_{ij} \right] \quad (\text{C.62c})$$

$$= -\sum_{ijk} \mathbf{B}_{ijk} \overline{[\log \mathbf{A}_k]}_{ij} \quad (\text{C.62d})$$

$$= -\sum_k \text{tr} \left( \mathbf{B}_k^T \overline{[\log \mathbf{A}_k]} \right) \quad (\text{C.62e})$$



# Bibliography

- [1] Semih Akbayrak, Ivan Bocharov, and Bert de Vries. “Extended variational message passing for automated approximate Bayesian inference”. In: *Entropy* 23.7 (2021), p. 815 (cit. on p. 147).
- [2] Nasim Alamdari, Edward Lobarinas, and Nasser Kehtarnavaz. “Personalization of Hearing Aid Compression by Human-In-Loop Deep Reinforcement Learning”. In: *arXiv:2007.00192 [cs, eess]* (June 2020). arXiv: 2007.00192 (cit. on pp. 74, 108).
- [3] Dmitry Bagaev and Bert de Vries. “Reactive Message Passing for Scalable Bayesian Inference”. en. In: (2022). Submitted to the Journal of Machine Learning Research. (cit. on pp. 39, 93, 153).
- [4] S. Bai et al. “Information-theoretic exploration with Bayesian optimization”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Issn: 2153-0866. Oct. 2016, pp. 1816–1822. DOI: 10.1109/iroso.2016.7759289 (cit. on p. 59).
- [5] Manuel Baltieri and Christopher Buckley. “PID Control as a Process of Active Inference with Linear Generative Models”. In: *Entropy* 21 (Mar. 2019), p. 257. DOI: 10.3390/e21030257 (cit. on p. 43).
- [6] Manuel Baltieri and Christopher L. Buckley. “An active inference implementation of phototaxis”. In: *arXiv:1707.01806 [q-bio]* (July 2017). arXiv: 1707.01806 (cit. on p. 43).



- [7] John G. Beerends et al. “Perceptual Objective Listening Quality Assessment (POLQA), The Third Generation ITU-T Standard for End-to-End Speech Quality Measurement Part I—Temporal Alignment”. English. In: *Journal of the Audio Engineering Society* 61.6 (July 2013). Publisher: Audio Engineering Society, pp. 366–384 (cit. on p. 77).
- [8] Glen Berseth et al. “SMiRL: Surprise Minimizing RL in Dynamic Environments”. en. In: *arXiv:1912.05510 [cs, stat]* (Dec. 2019). arXiv: 1912.05510 (cit. on p. 59).
- [9] J. Bezanson et al. “Julia: A Fresh Approach to Numerical Computing”. In: *SIAM Review* 59.1 (Jan. 2017), pp. 65–98. ISSN: 0036-1445. DOI: 10.1137/141000671 (cit. on p. 93).
- [10] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., 2006. ISBN: 0-387-31073-8 (cit. on pp. 33, 89).
- [11] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. “Variational Inference: A Review for Statisticians”. en. In: *Journal of the American Statistical Association* 112.518 (Apr. 2017), pp. 859–877. ISSN: 0162-1459, 1537-274x. DOI: 10.1080/01621459.2017.1285773 (cit. on p. 56).
- [12] Christopher L. Buckley et al. “The free energy principle for action and perception: A mathematical review”. en. In: *Journal of Mathematical Psychology* 81 (Dec. 2017), pp. 55–79. ISSN: 00222496. DOI: 10.1016/j.jmp.2017.09.004 (cit. on p. 43).
- [13] Mona Buisson-Fenet, Friedrich Solowjow, and Sebastian Trimpe. “Actively Learning Gaussian Process Dynamics”. en. In: (), p. 11 (cit. on p. 59).
- [14] Ariel Caticha. *Entropic Inference and the Foundations of Physics*. ru. EBEB-2012, the 11th Brazilian Meeting on Bayesian Statistics, 2012 (cit. on pp. 20, 132).
- [15] Théophile Champion, Marek Grześ, and Howard Bowman. “Realising Active Inference in Variational Message Passing: the Outcome-blind Certainty Seeker”. In: *arXiv:2104.11798 [cs]* (Apr. 2021). arXiv: 2104.11798 (cit. on pp. 23, 40).
- [16] Théophile Champion et al. “Branching Time Active Inference: The theory and its generality”. In: *Neural Networks* 151 (2022), pp. 295–316. ISSN: 0893-6080. DOI: <https://doi.org/10.1016/j.neunet.2022.03.036> (cit. on pp. 40, 114).

- [17] Michael Chinen et al. “ViSQOL v3: An Open Source Production Ready Objective Speech and Audio Metric”. In: *arXiv:2004.09584 [cs, eess]* (Apr. 2020) (cit. on p. 77).
- [18] Wei Chu and Zoubin Ghahramani. “Preference learning with Gaussian processes”. en. In: *Proceedings of the 22nd international conference on Machine learning - ICML '05*. Bonn, Germany: ACM Press, 2005, pp. 137–144. ISBN: 978-1-59593-180-1. DOI: 10.1145/1102351.1102369 (cit. on p. 87).
- [19] Marco Cox, Thijs van de Laar, and Bert de Vries. “A factor graph approach to automated design of Bayesian signal processing algorithms”. In: *International Journal of Approximate Reasoning* 104 (Jan. 2019), pp. 185–204. ISSN: 0888-613x. DOI: 10.1016/j.ijar.2018.11.002 (cit. on p. 88).
- [20] Marco Cox and Bert de Vries. “A parametric approach to Bayesian optimization with pairwise comparisons”. en. In: *NIPS Workshop on Bayesian Optimization (BayesOpt 2017)*. Long Beach, USA, Dec. 2017, pp. 1–5 (cit. on p. 110).
- [21] Maell Cullen et al. “Active Inference in OpenAI Gym: A Paradigm for Computational Investigations Into Psychiatric Illness”. en. In: *Biological Psychiatry: Cognitive Neuroscience and Neuroimaging* 3.9 (Sept. 2018), pp. 809–818. ISSN: 24519022. DOI: 10.1016/j.bpsc.2018.06.010 (cit. on pp. 46, 58).
- [22] Lancelot Da Costa et al. “Active inference on discrete state-spaces: a synthesis”. en. In: *arXiv:2001.07203 [q-bio]* (Jan. 2020). arXiv: 2001.07203 (cit. on pp. 20, 35, 40, 55, 58, 60, 75, 115, 122, 147, 149, 150, 164, 186).
- [23] Justin Dauwels. “On Variational Message Passing on Factor Graphs”. In: *IEEE International Symposium on Information Theory*. Nice, France, June 2007, pp. 2546–2550. DOI: 10.1109/isit.2007.4557602 (cit. on pp. 18, 93).
- [24] Bart van Erp et al. “A Bayesian Modeling Approach to Situated Design of Personalized Soundscaping Algorithms”. en. In: *Applied Sciences* 11.20 (Oct. 2021). Number: 20 Publisher: Multidisciplinary Digital Publishing Institute, p. 9535. DOI: 10.3390/app11209535 (cit. on pp. 86, 90).
- [25] Jr. Forney G.D. “Codes on graphs: normal realizations”. In: *IEEE Transactions on Information Theory* 47.2 (Feb. 2001), pp. 520–548. ISSN: 0018-9448. DOI: 10.1109/18.910573 (cit. on pp. 14, 46, 82).

- [26] Zafeirios Fountas et al. “Deep active inference agents using Monte-Carlo methods”. en. In: *arXiv:2006.04176 [cs, q-bio, stat]* (June 2020). arXiv: 2006.04176 (cit. on p. 46).
- [27] Brendan J Frey et al. “ALGONQUIN: Iterating Laplace’s Method to Remove Multiple Types of Acoustic Distortion for Robust Speech Recognition”. en. In: *Proceedings of the Eurospeech Conference*. Aalborg, Denmark, Sept. 2001, pp. 901–904 (cit. on pp. 109, 110).
- [28] Karl Friston. “A free energy principle for a particular physics”. In: *arXiv:1906.10184 [q-bio]* (June 2019). arXiv: 1906.10184 (cit. on pp. 20, 60, 120).
- [29] Karl Friston and Ping Ao. “Free Energy, Value, and Attractors”. en. In: *Computational and Mathematical Methods in Medicine 2012* (2012), pp. 1–27. ISSN: 1748-670x, 1748-6718. DOI: 10.1155/2012/937860 (cit. on p. 43).
- [30] Karl Friston, James Kilner, and Lee Harrison. “A free energy principle for the brain”. In: *Journal of Physiology, Paris* 100.1-3 (Sept. 2006), pp. 70–87. ISSN: 0928-4257. DOI: 10.1016/j.jphysparis.2006.10.001 (cit. on p. 75).
- [31] Karl Friston, Thomas Parr, and Peter Zeidman. “Bayesian model reduction”. In: *arXiv:1805.07092 [stat]* (May 2018). arXiv: 1805.07092 (cit. on p. 109).
- [32] Karl Friston and Will Penny. “Post hoc Bayesian model selection”. In: *Neuroimage* 56.4-2 (June 2011), pp. 2089–2099. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2011.03.062 (cit. on p. 109).
- [33] Karl Friston et al. “Active inference and epistemic value”. In: *Cognitive Neuroscience* 6.4 (Oct. 2015), pp. 187–214. ISSN: 1758-8928. DOI: 10.1080/17588928.2015.1020053 (cit. on pp. 21, 23, 32, 35, 38, 40, 45, 46, 54, 55, 58, 59, 70, 75, 90, 115, 122, 147–149, 151, 153, 155–157, 168, 185, 186).
- [34] Karl Friston et al. “Active inference and learning”. en. In: *Neuroscience & Biobehavioral Reviews* 68 (Sept. 2016), pp. 862–879. ISSN: 01497634. DOI: 10.1016/j.neubiorev.2016.06.022 (cit. on pp. 20, 43, 70).
- [35] Karl Friston et al. “Sophisticated Inference”. In: *Neural Computation* 33.3 (Mar. 2021), pp. 713–763. ISSN: 0899-7667. DOI: 10.1162/neco\_a\_01351 (cit. on pp. 31, 32, 34, 35, 38–40, 46, 55, 58, 72, 75, 114, 120, 164, 168).

- [36] Karl J. Friston, Thomas Parr, and Bert de Vries. “The graphical brain: Belief propagation and active inference”. en. In: *Network Neuroscience* 1.4 (Dec. 2017), pp. 381–414. ISSN: 2472-1751. DOI: 10.1162/NETN\_a\_00018 (cit. on pp. 23, 40).
- [37] Karl J. Friston et al. “Action and behavior: a free-energy formulation”. en. In: *Biological Cybernetics* 102.3 (Mar. 2010), pp. 227–260. ISSN: 0340-1200, 1432-0770. DOI: 10.1007/s00422-010-0364-z (cit. on p. 120).
- [38] Karl J. Friston et al. “Active listening”. en. In: *Hearing Research* 399. Stimulus-specific adaptation, MMN and predicting coding (Jan. 2021), p. 107998. ISSN: 03785955. DOI: 10.1016/j.heares.2020.107998 (cit. on p. 108).
- [39] K.J. Friston, N. Trujillo-Barreto, and J. Daunizeau. “DEM: A variational treatment of dynamic systems”. en. In: *NeuroImage* 41.3 (July 2008), pp. 849–885. ISSN: 10538119. DOI: 10.1016/j.neuroimage.2008.02.054 (cit. on p. 185).
- [40] S. Gannot, D. Burshtein, and E. Weinstein. “Iterative and sequential Kalman filter-based speech enhancement algorithms”. In: *IEEE Transactions on Speech and Audio Processing* 6.4 (July 1998), pp. 373–385. ISSN: 1558-2353. DOI: 10.1109/89.701367 (cit. on p. 83).
- [41] Mohammad Ghavamzadeh et al. “Bayesian Reinforcement Learning: A Survey”. en. In: *Foundations and Trends® in Machine Learning* 8.5-6 (2015). arXiv: 1609.04436, pp. 359–483. ISSN: 1935-8237, 1935-8245. DOI: 10.1561/22000000049 (cit. on p. 45).
- [42] J.D. Gibson, B. Koo, and S.D. Gray. “Filtering of colored noise for speech enhancement and coding”. In: *IEEE Transactions on Signal Processing* 39.8 (Aug. 1991), pp. 1732–1742. ISSN: 1941-0476. DOI: 10.1109/78.91144 (cit. on p. 83).
- [43] Danijar Hafner et al. “Action and Perception as Divergence Minimization”. In: *arXiv:2009.01791 [cs, math, stat]* (Oct. 2020). arXiv: 2009.01791 (cit. on pp. 58, 59, 70, 120).
- [44] Conor Heins et al. “pymdp: A Python library for active inference in discrete state spaces”. In: *arXiv preprint arXiv:2201.03904* (2022) (cit. on pp. 32, 185).
- [45] John R Hershey, Peder Olsen, and Steven J Rennie. “Signal Interaction and the Devil Function”. en. In: *Proceedings of the Interspeech 2010*. Makuhari, Chiba, Japan, 2010, pp. 334–337 (cit. on p. 109).

- [46] Andrew Hines et al. “ViSQOL: an objective speech quality model”. en. In: *EURASIP Journal on Audio, Speech, and Music Processing* 2015.1 (Dec. 2015), p. 13. ISSN: 1687-4722. DOI: 10.1186/s13636-015-0054-9 (cit. on p. 77).
- [47] Neil Houlsby et al. “Bayesian Active Learning for Classification and Preference Learning”. In: *arXiv:1112.5745 [cs, stat]* (Dec. 2011). arXiv: 1112.5745 (cit. on pp. 87, 181, 182).
- [48] Tesheng Hsiao. “Identification of Time-Varying Autoregressive Systems Using Maximum a Posteriori Estimation”. In: *IEEE Transactions on Signal Processing* 56.8 (Aug. 2008), pp. 3497–3509. ISSN: 1941-0476. DOI: 10.1109/tsp.2008.919393 (cit. on p. 109).
- [49] Justus Huebötter et al. “Learning policies for continuous control via transition models”. In: *International Workshop on Active Inference*. Springer, 2022, pp. 162–178 (cit. on p. 45).
- [50] Ferenc Huszar. “A GP classification approach to preference learning”. en. In: *NIPS Workshop on Choice Models and Preference Learning*. Sierra Nevada, Spain, 2011, p. 4 (cit. on p. 87).
- [51] Tanya Ignatenko et al. “On Sequential Bayesian Optimization with Pairwise Comparison”. In: *arXiv:2103.13192 [cs, math, stat]* (Mar. 2021). arXiv: 2103.13192 (cit. on pp. 108, 110).
- [52] Patrick K Mogensen and Asbjørn N Riseth. “Optim: A mathematical optimization package for Julia”. en. In: *Journal of Open Source Software* 3.24 (Apr. 2018), p. 615. ISSN: 2475-9066. DOI: 10.21105/joss.00615 (cit. on pp. 97, 183).
- [53] 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*. Vol. 7. May 1982, pp. 1295–1298. DOI: 10.1109/icassp.1982.1171643 (cit. on p. 82).
- [54] Hilbert J. Kappen, Vicenç Gómez, and Manfred Opper. “Optimal control as a graphical model inference problem”. en. In: *Machine Learning* 87.2 (May 2012), pp. 159–182. ISSN: 0885-6125, 1573-0565. DOI: 10.1007/s10994-012-5278-7 (cit. on pp. 65, 120, 122).
- [55] James Kates and Kathryn Arehart. “Multichannel Dynamic-Range Compression Using Digital Frequency Warping”. In: *EURASIP Journal on Applied Signal Processing* 18 (2005), pp. 3003–3014. DOI: 10.1155/asp.2005.3003 (cit. on p. 74).

- [56] James M. Kates and Kathryn H. Arehart. “The hearing-aid speech quality index (HASQI)”. In: *Journal of the Audio Engineering Society* 58.5 (2010), pp. 363–381 (cit. on p. 77).
- [57] Frank Kleibergen and Henk Hoek. “Bayesian Analysis of ARMA models using Noninformative Priors”. en. In: *CentER Discussion Paper* 1995-116 (1995), p. 24 (cit. on p. 109).
- [58] Kevin H. Knuth. “Informed Source Separation: A Bayesian Tutorial”. In: *arXiv:1311.3001 [cs, stat]* (Nov. 2013). arXiv: 1311.3001 (cit. on pp. 92, 108).
- [59] Magnus Koudahl, Christopher L Buckley, and Bert de Vries. “A Message Passing Perspective on Planning Under Active Inference”. In: *Active Inference: Third International Workshop, IWAI 2022, Grenoble, France, September 19, 2022, Revised Selected Papers*. Springer. 2023, pp. 319–327 (cit. on pp. 115, 148, 149, 152).
- [60] Magnus T. Koudahl, Wouter M. Kouw, and Bert de Vries. “On Epistemics in Expected Free Energy for Linear Gaussian State Space Models”. In: *Entropy* 23.12 (Nov. 2021), p. 1565. ISSN: 1099-4300. DOI: 10.3390/e23121565 (cit. on pp. 35, 38–40, 123, 149, 186).
- [61] Magnus T. Koudahl and Bert de Vries. “A Worked Example of Fokker-Planck-Based Active Inference”. In: *Active Inference*. Ed. by Tim Verbelen et al. Cham: Springer International Publishing, 2020, pp. 28–34. ISBN: 978-3-030-64919-7 (cit. on p. 20).
- [62] Frank R. Kschischang, Brendan J. Frey, and H.-A. Loeliger. “Factor graphs and the sum-product algorithm”. In: *IEEE Transactions on information theory* 47.2 (2001), pp. 498–519. DOI: 10.1109/18.910572 (cit. on p. 16).
- [63] Thijs van de Laar. “Automated Design of Bayesian Signal Processing Algorithms”. PhD thesis. Eindhoven, The Netherlands: Eindhoven University of Technology, 2019 (cit. on pp. 89, 93).
- [64] Thijs van de Laar, Ayça Özçelikkale, and Henk Wymeersch. “Application of the Free Energy Principle to Estimation and Control”. In: *arXiv preprint arXiv:1910.09823* (2019) (cit. on p. 75).
- [65] Thijs van de Laar and Bert de Vries. “A Probabilistic Modeling Approach to Hearing Loss Compensation”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24.11 (Nov. 2016), pp. 2200–2213. ISSN: 2329-9290. DOI: 10.1109/taslp.2016.2599275 (cit. on p. 74).

- [66] Thijs van de Laar et al. “Active Inference and Epistemic Value in Graphical Models”. In: *Frontiers in Robotics and AI* 9 (2022) (cit. on p. 39).
- [67] Thijs W. van de Laar and Bert de Vries. “Simulating Active Inference Processes by Message Passing”. In: *Frontiers in Robotics and AI* 6 (2019). ISSN: 2296-9144. DOI: 10.3389/frobt.2019.00020 (cit. on pp. 40, 43, 65, 75).
- [68] Y. Laufer and S. Gannot. “A Bayesian Hierarchical Model for Blind Audio Source Separation”. In: *2020 28th European Signal Processing Conference (EUSIPCO)*. Issn: 2076-1465. Jan. 2021, pp. 276–280. DOI: 10.23919/Eusipco47968.2020.9287348 (cit. on p. 108).
- [69] H.-A. Loeliger. “An introduction to factor graphs”. In: *Signal Processing Magazine, IEEE* 21.1 (2004), pp. 28–41 (cit. on pp. 14, 15, 47, 82).
- [70] Hans-Andrea Loeliger et al. “The Factor Graph Approach to Model-Based Signal Processing”. In: *Proceedings of the IEEE* 95.6 (June 2007), pp. 1295–1322. ISSN: 0018-9219. DOI: 10.1109/jproc.2007.896497 (cit. on pp. 15, 34, 47, 88, 93).
- [71] David MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003 (cit. on p. 120).
- [72] Beren Millidge. “Deep active inference as variational policy gradients”. In: *Journal of Mathematical Psychology* 96 (2020), p. 102348 (cit. on pp. 40, 46, 59, 75).
- [73] Beren Millidge, Alexander Tschantz, and Christopher L. Buckley. “Whence the Expected Free Energy?” In: *arXiv:2004.08128 [cs]* (Apr. 2020). arXiv: 2004.08128 (cit. on pp. 58, 70).
- [74] Beren Millidge et al. “Understanding the Origin of Information-Seeking Exploration in Probabilistic Objectives for Control”. In: *arXiv:2103.06859 [cs]* (June 2021). arXiv: 2103.06859 (cit. on pp. 70, 120).
- [75] Thomas P. Minka. “Expectation Propagation for Approximate Bayesian Inference”. In: *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*. Uai’01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 362–369. ISBN: 978-1-55860-800-9 (cit. on pp. 19, 116, 133).
- [76] M. Berk Mirza et al. “Scene Construction, Visual Foraging, and Active Inference”. en. In: *Frontiers in Computational Neuroscience* 10 (June 2016). ISSN: 1662-5188. DOI: 10.3389/fncom.2016.00056 (cit. on p. 23).

- [77] 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*. Morgan Kaufmann Publishers Inc., 1999, pp. 467–475. ISBN: 1-55860-614-9 (cit. on p. 102).
- [78] J.B.B. Nielsen, J. Nielsen, and J. Larsen. “Perception-Based Personalization of Hearing Aids Using Gaussian Processes and Active Learning”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23.1 (Jan. 2015), pp. 162–173. ISSN: 2329-9290. DOI: 10.1109/taslp.2014.2377581 (cit. on pp. 74, 107, 108).
- [79] A. Ozerov and C. Fevotte. “Multichannel Nonnegative Matrix Factorization in Convolutional Mixtures for Audio Source Separation”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 18.3 (Mar. 2010), pp. 550–563. ISSN: 1558-7924. DOI: 10.1109/tasl.2009.2031510 (cit. on p. 109).
- [80] 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*. Vol. 12. Dallas, TX, USA, Apr. 1987, pp. 177–180. DOI: 10.1109/icassp.1987.1169756 (cit. on p. 82).
- [81] Thomas Parr and Karl J. Friston. “Generalised free energy and active inference”. en. In: *Biological Cybernetics* 113.5-6 (Dec. 2019), pp. 495–513. ISSN: 0340-1200, 1432-0770. DOI: 10.1007/s00422-019-00805-w (cit. on pp. 32, 40, 46, 59, 122–124, 148, 151–153, 168, 186).
- [82] Thomas Parr and Karl J. Friston. “The Discrete and Continuous Brain: From Decisions to Movement—And Back Again”. en. In: *Neural Computation* 30.9 (Sept. 2018), pp. 2319–2347. ISSN: 0899-7667, 1530-888x. DOI: 10.1162/neco\_a\_01102 (cit. on p. 23).
- [83] Thomas Parr and Karl J. Friston. “Uncertainty, epistemics and active inference”. en. In: *Journal of The Royal Society Interface* 14.136 (Nov. 2017), p. 20170376. ISSN: 1742-5689, 1742-5662. DOI: 10.1098/rsif.2017.0376 (cit. on p. 90).
- [84] Judea Pearl. “Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach”. In: *Proceedings of the Second AAAI Conference on Artificial Intelligence*. Aaai’82. Pittsburgh, Pennsylvania: AAAI Press, 1982, pp. 133–136 (cit. on pp. 16, 116).



- [85] Albert Podusenko, Wouter M. Kouw, and Bert de Vries. “Message Passing-Based Inference for Time-Varying Autoregressive Models”. In: *Entropy* 23.6 (May 2021), p. 683. ISSN: 1099-4300. DOI: 10.3390/e23060683 (cit. on pp. 93, 179).
- [86] 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)*. Issn: 2157-8117. Los Angeles, CA, USA, June 2020, pp. 1337–1342. DOI: 10.1109/isit44484.2020.9174134 (cit. on p. 83).
- [87] Albert Podusenko et al. “Message Passing-Based Inference in the Gamma Mixture Model”. In: *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*. Gold Coast, Australia: Ieee, Oct. 2021, pp. 1–6. ISBN: 978-1-72816-338-3. DOI: 10.1109/mlsp52302.2021.9596329 (cit. on pp. 89, 93).
- [88] 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*. Vol. 2. Issn: 1520-6149. Seattle, WA, USA, May 1998, pp. 997–1000. DOI: 10.1109/icassp.1998.675435 (cit. on p. 83).
- [89] M.H. Radfar et al. “Nonlinear minimum mean square error estimator for mixture-maximisation approximation”. In: *Electronics Letters* 42.12 (June 2006), pp. 724–725. ISSN: 0013-5194. DOI: 10.1049/e1:20060510 (cit. on p. 109).
- [90] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian processes for machine learning*. en. Adaptive computation and machine learning. OCLC: ocm61285753. Cambridge, Mass: MIT Press, 2006. ISBN: 978-0-262-18253-9 (cit. on pp. 87, 92, 182, 183).
- [91] C. Karadagur Ananda Reddy et al. “An Individualized Super-Gaussian Single Microphone Speech Enhancement for Hearing Aid Users With Smartphone as an Assistive Device”. In: *IEEE Signal Processing Letters* 24.11 (Nov. 2017), pp. 1601–1605. ISSN: 1558-2361. DOI: 10.1109/lsp.2017.2750979 (cit. on p. 74).
- [92] Christoph Reller. *State-space methods in statistical signal processing: New ideas and applications*. Vol. 23. ETH Zurich, 2013 (cit. on pp. 14, 132).

- [93] 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*. Taipei, Taiwan, Apr. 2009, pp. 3845–3848. DOI: 10.1109/icassp.2009.4960466 (cit. on pp. 109, 110).
- [94] Steven Rennie et al. “Dynamic noise adaptation”. In: *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*. Vol. 1. Toulouse, France: Ieee, 2006, pp. 1–4. DOI: 10.1109/icassp.2006.1660241 (cit. on pp. 109, 110).
- [95] Antony W. Rix et al. “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*. Vol. 2. Ieee, 2001, pp. 749–752 (cit. on p. 77).
- [96] Daniel Rudoy, Thomas F. Quatieri, and Patrick J. Wolfe. “Time-Varying Autoregressions in Speech: Detection Theory and Applications”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.4 (May 2011), pp. 977–989. ISSN: 1558-7924. DOI: 10.1109/tasl.2010.2073704 (cit. on p. 83).
- [97] Noor Sajid, Philip J. Ball, and Karl J. Friston. “Active inference: demystified and compared”. en. In: *arXiv:1909.10863 [cs, q-bio]* (Jan. 2020). arXiv: 1909.10863 (cit. on pp. 45, 58, 59, 70, 149, 186).
- [98] Noor Sajid et al. “Active inference, Bayesian optimal design, and expected utility”. In: *arXiv:2110.04074 [cs, math, stat]* (Sept. 2021). arXiv: 2110.04074 (cit. on pp. 43, 70, 90).
- [99] Simo Sarkka. *Bayesian Filtering and Smoothing*. en. Cambridge: Cambridge University Press, 2013. ISBN: 978-1-139-34420-3. DOI: 10.1017/cbo9781139344203 (cit. on pp. 51–53, 92, 123).
- [100] Philipp Schwartenbeck et al. *Computational mechanisms of curiosity and goal-directed exploration*. en. preprint. Neuroscience, Sept. 2018. DOI: 10.1101/411272 (cit. on pp. 40, 58).
- [101] Philipp Schwartenbeck et al. “Exploration, novelty, surprise, and free energy minimization”. In: *Frontiers in Psychology* 4 (2013). ISSN: 1664-1078. DOI: 10.3389/fpsyg.2013.00710 (cit. on p. 55).

- [102] Sarah Schwöbel, Stefan Kiebel, and Dimitrije Marković. “Active Inference, Belief Propagation, and the Bethe Approximation”. en. In: *Neural Computation* 30.9 (Sept. 2018), pp. 2530–2567. ISSN: 0899-7667, 1530-888x. DOI: 10.1162/neco\_a\_01108 (cit. on pp. 65, 120).
- [103] Ryan Smith, Karl Friston, and Christopher Whyte. *A Step-by-Step Tutorial on Active Inference and its Application to Empirical Data*. Jan. 2021. DOI: 10.31234/osf.io/b4jm6 (cit. on p. 115).
- [104] Alex J. Smola, S. V. N. Vishwanathan, and Eleazar Eskin. “Laplace propagation”. In: *Nips*. 2004, pp. 441–448 (cit. on p. 19).
- [105] Oleg Solopchuk. “Information theoretic approach to decision making in continuous domains”. PhD thesis. UCL-Université Catholique de Louvain, 2021 (cit. on pp. 62, 72).
- [106] Cees H. Taal et al. “An Algorithm for Intelligibility Prediction of Time–Frequency Weighted Noisy Speech”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 19.7 (Sept. 2011), pp. 2125–2136. ISSN: 1558-7924. DOI: 10.1109/tas1.2011.2114881 (cit. on p. 77).
- [107] Alexander Tschantz, Anil K. Seth, and Christopher L. Buckley. *Learning action-oriented models through active inference*. en. preprint. *Animal Behavior and Cognition*, Sept. 2019. DOI: 10.1101/764969 (cit. on p. 46).
- [108] Alexander Tschantz et al. “Reinforcement Learning through Active Inference”. In: *arXiv:2002.12636 [cs, eess, math, stat]* (Feb. 2020). arXiv: 2002.12636 (cit. on p. 46).
- [109] Alexander Tschantz et al. “Scaling active inference”. In: *2020 international joint conference on neural networks (ijcnn)*. IEEE. 2020, pp. 1–8 (cit. on pp. 46, 75).
- [110] Kai Ueltzhöffer. “Deep Active Inference”. en. In: *Biological Cybernetics* (Oct. 2018). arXiv: 1709.02341. ISSN: 0340-1200, 1432-0770. DOI: 10.1007/s00422-018-0785-7 (cit. on pp. 40, 46).
- [111] J. Vermaak et al. “Particle methods for Bayesian modeling and enhancement of speech signals”. In: *IEEE Transactions on Speech and Audio Processing* 10.3 (Mar. 2002), pp. 173–185. ISSN: 1558-2353. DOI: 10.1109/tsa.2002.1001982 (cit. on p. 83).
- [112] John Winn and Christopher M. Bishop. “Variational message passing”. In: *Journal of Machine Learning Research* 6.4 (2005), pp. 661–694 (cit. on pp. 18, 116).

- [113] S. Xie et al. “Time-Frequency Approach to Underdetermined Blind Source Separation”. In: *IEEE Transactions on Neural Networks and Learning Systems* 23.2 (Feb. 2012), pp. 306–316. ISSN: 2162-2388. DOI: 10.1109/tnnls.2011.2177475 (cit. on p. 108).
- [114] Dan Zhang et al. “Unifying message passing algorithms under the framework of constrained Bethe free energy minimization”. In: *IEEE Transactions on Wireless Communications* 20.7 (2021), pp. 4144–4158 (cit. on pp. 15, 116).
- [115] Ozan Çatal et al. “Learning Generative State Space Models for Active Inference”. In: *Frontiers in Computational Neuroscience* 14 (2020). ISSN: 1662-5188. DOI: 10.3389/fncom.2020.574372 (cit. on pp. 40, 164).
- [116] İsmail Şenöz et al. “Variational Message Passing and Local Constraint Manipulation in Factor Graphs”. en. In: *Entropy* 23.7 (July 2021). Number: 7 Publisher: Multidisciplinary Digital Publishing Institute, p. 807. DOI: 10.3390/e23070807 (cit. on pp. 15, 18, 19, 38, 56, 93, 116, 124, 128, 130–132, 158, 191, 192).



# Acknowledgments

When I got off the plane in Eindhoven in 2018 to start a new chapter, I had many lofty plans and grand ambitions. Over the past 5 years, I learned that Life didn't really care for my plans and had something entirely different in mind which turned into the work you're currently reading. Dealing with all the curveballs, the ups and the downs would not have been possible without a group of people who are very dear to me for various reasons.

**Bert**, we have had quite a rollercoaster of a run. We've lifted (many!) weights, shot one another (with paintballs), and had heated arguments and very deep discussions. In the end, you always had my back when I really needed it. I am thankful for the freedom you afforded me even when we didn't agree. It led me down many rabbit holes, some of which even turned out to be useful. For entirely unrelated reasons, you have also cultivated in me an almost manic obsession with checking subscripts.

**Thijs**, you have taught me so much. You have probably endured more of my questions than most and regardless of how naive or deep they were, you always had a thoughtful response. Over the years, you taught me many things about message passing, mathematics, and teamwork. I admire your dedication to the approach I have come to label "slow-cooker science". I'll forever be proud to roll out the matlab and get cooking.

**Wouter K.**, you turned out to be not just a patient and thorough colleague but also a really good friend. We've shared many laughs and many a time venting over everything and nothing. You were one of the few people that I felt saw the same things I did and believed that they were worth pursuing. You are inspiring to be around in a very down-to-earth way which I admire very much. Thank you for being generous with your time, your ideas and your energy.

**Semih**, attending your and **Melikes** wedding is one of the absolute highlights of the past 5 years. You are one of the brightest and kindest people that

I know. You were always there with a clear mind and the right words when I needed it, no matter if it was tackling the ups and downs of life or a tricky inference problem. I am proud to have you as a friend.

**Dmitry**, I'm pretty sure your native language is code with Russian as a distant second. I'm in awe of the things you can do and the ease with which you do them - especially since you also find time to bang on drums. You showed me the fun of unplanned, late-night jam sessions which is something I'll forever take with me. Also, along with **Albert**, you convinced me that jumping out of a plane is a good idea.

**Albert** you have always been a thoughtful and excellent researcher, which both contrasts with and complements your ability to throw caution to the wind. You were along not just for many of the craziest nights I had in Eindhoven but also for many of the deepest technical discussions. And we did AIDA which is a mountain I could never have climbed alone. Thank you for being a man of opposites.

**Ismail**, you are the closest thing to a wizard that I know. However more than your exceptional mathmagics, I admire your ability to live the good life. Tuning into a meeting and doing heavyweight mathematics while sitting in an open Hawaii-shirt on a sunny beach is the kind of life we should all strive for.

To the rest of my friends in BIASlab: **Martin**, it's :ZZ on this chapter. I hope we have many more to come. **Tim, Hoang, Burak, Bart, Sepideh, Ivan, Raaja, Wouter N.** and **Mykola** thank you for all the coffee-runs and for making my time in the lab into what it was.

Outside of BIASLab, I owe a special thanks to several other friends and collaborators. **Maxwell**, thank you for pulling me out of the reading group and into the inner circle. It's a very special experience to go from being on the outside of a field looking in, to being on the inside looking out.

To **Chris, Conor, Beren, Dimitrije, Tim, Casper, Karl, Toby, Lance, Dalton, Alec, Alex, Riddhi, Mahault, Tommaso**, and everyone else, thank you for being generous with your brilliance, your insights and your time. I have learned so much from each and every one of you and I look forward to continuing to do so.

Thank you to all the friends who kept in touch while I was living abroad: **Nicolas, Nicolai, Julie, Stina, Gina, Mikkel** and **Mossa**, it means more than you think. Thank you to **Anthony, Wout, Michael, Josh** and **Thijs** for all the adventures. To **Lars** and **Mads** for bringing the power and fire of metal back into my life.

And finally, to my beloved kurst **Mai-Britt**. Du bringer lys, grin og lykke ind

i min verden som ingen anden. Tak for at du har holdt ved mig de sidste 5 år selvom vi har været i forskellige lande - uden dig var det her aldrig gået. Tak for dig.

And that's all folks. Horns up,

Magnus T. Koudahl  
Eindhoven, September 2023





# Biography

Magnus was born in 1990 in Copenhagen, Denmark. His academic career began in 2011. The question on his mind then, as now, was "*Why do people do the things they do?*". In the pursuit of an answer, he enrolled in a Psychology degree and obtained his B.Sc from the University of Copenhagen in 2015. During this time he started showing an interest in research and participated in several projects. He worked in a rodent lab, collected EEG data for perceptual experiments and taught introductory statistics, experimental design and cognitive psychology to fellow Bachelors students.

In 2015 he enrolled in an interdisciplinary Masters programme in It & Cognition, also at the University of Copenhagen to increase his knowledge of mathematical modelling. At the same time, he began volunteering at the Danish Research Center for Magnetic Resonance where he quickly got involved in fMRI research on human decision-making. He worked at DRCMR from 2015-2018 while completing his M.Sc degree. In a twist of fate that turned out to be quite important, his immediate supervisor obtained his doctoral degree with Karl Friston. He introduced Magnus to the ideas of Active Inference which came to form the foundations of his later doctoral work.

In 2017 he obtained his M.Sc degree. His thesis investigated the role of the posterior parietal cortex in risky decision-making for consequential losses. The work was based on a large-scale imaging study using a novel experimental design where subjects actually lost out of pocket money.

Having spent a couple of years in research at this point, Magnus decided that this was to be his path moving forwards. However his desire for precision had increasingly started to draw him away from empirical neuroscience and towards machine learning and mathematical modelling instead.

While looking for Ph.D. programmes that fit his interests, he was forwarded an email that simply read "☺". The email also included a forwarded email where

a certain Bert de Vries complained about the difficulties of finding prospective Ph.D. students who knew how to program, knew about neuroscience and were interested in machine learning. Finding himself at the centre of this particular Venn diagram, Magnus decided to send his resume to Bert at BIASLab in Eindhoven. The two got to talking and quickly agreed that the position matched well with Magnus skillset and interests. In the fall of 2018 he enrolled as a Ph.D. student under Bert. Over the next 5 years, Magnus worked hard on producing the body of research that became the dissertation you are currently reading. For the last half of his Ph.D, Magnus has also worked part-time with various startup companies on applying his research to problems in industry.

# List of Publications

Magnus T. Koudahl has the following publications:

## Journals

- **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
- **Koudahl, M. T., Kouw, W. M., & de Vries, B.** (2021). *On Epistemics in Expected Free Energy for Linear Gaussian State Space Models*. *Entropy*, 23(12), 1565. MDPI AG. <http://dx.doi.org/10.3390/e23121565>

## Proceedings and Conference Contributions

- **Koudahl, M., Buckley, C.L., & de Vries, B.** (2023). *A Message Passing Perspective on Planning Under Active Inference*. In: , et al. *Active Inference. IWAI 2022. Communications in Computer and Information Science*, vol 1721. Springer, Cham.

## Preprints

- **Koudahl, M. T., van de Laar, T. W., & de Vries, B.** (2023). *Realising Synthetic Active Inference Agents, Part I: Epistemic Objectives and Graphical Specification Language* arXiv:2306.08014

- **van de Laar, T. W., Koudahl, M. T., & de Vries, B.** (2023). *Realising Synthetic Active Inference Agents, Part II: Variational Message Passing Updates* arXiv:2306.02733

### Not included in this Dissertation

- **Koudahl, M.** (2022). *Particularly average. A comment on “How particular is the physics of the Free Energy Principle” by Aguilera et al.*, *Physics of Life Reviews* vol. 42. p.40-42 <https://ui.adsabs.harvard.edu/abs/2022PhLRv..42...40K>
- **M. T. Koudahl & B. de Vries.** (2022). *Batman: Bayesian Target Modelling For Active Inference* ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 2020, pp. 3852-3856, doi: 10.1109/ICASSP40776.2020.9053624.
- **Koudahl, M.T. & de Vries, B.** (2020). *A Worked Example of Fokker-Planck-Based Active Inference* In: Verbelen, T., Lanillos, P., Buckley, C.L., De Boom, C. (eds) *Active Inference. IWAI 2020. Communications in Computer and Information Science*, vol 1326. Springer, Cham. [https://doi.org/10.1007/978-3-030-64919-7\\_4](https://doi.org/10.1007/978-3-030-64919-7_4)
- Ramstead Maxwell J. D., Sakthivadivel Dalton A. R., Heins Conor, **Koudahl Magnus**, Millidge Beren, Da Costa Lancelot, Klein Brennan & Friston Karl J. (2023) *Bayesian mechanics: a physics of and by beliefs* *Interface Focus* vol. 13. p.20220029 <http://doi.org/10.1098/rsfs.2022.0029>
- Friston, K. J., Ramstead, M. J., Kiefer, A. B., Tschantz, A., Buckley, C. L., Albarracin, M., Pitliya, R. J., Heins, C., Klein, B., Millidge, B., Sakthivadivel, D. A., Smithe, T. S., **Koudahl, M.**, Tremblay, S. E., Petersen, C., Fung, K., Fox, J. G., Swanson, S., Mapes, D., René, G. (2022). *Designing Ecosystems of Intelligence from First Principles.* arXiv:2212.01354
- H. M. H. Nguyen, S. Akbayrak, **M. T. Koudahl & B. de Vries**, (2022) *Gaussian Process-based Amortization of Variational Message Passing Update Rules* 30th European Signal Processing Conference (EUSIPCO), Belgrade, Serbia, pp. 1517-1521, doi: 10.23919/EUSIPCO55093.2022.9909688.
- W. Kouw, A. Podusenko, **M. Koudahl & M. Schoukens**, (2022) *Variational message passing for online polynomial NARMAX identification* *American Control Conference (ACC)*, Atlanta, GA, USA, pp. 2755-2760, doi: 10.23919/ACC53348.2022.9867898.

- 
- van de Laar, T. W., **Koudahl, M.**, van Erp, B., & de Vries, B. (2022). *Active Inference and Epistemic Value in Graphical Models*, *Frontiers in Robotics and AI*, 9, 794464. <https://doi.org/10.3389/frobt.2022.794464>
  - Ergul, B., van de Laar, T., **Koudahl, M.**, Roa-Villescas, M. & de Vries, B. (2020). *Learning Where to Park* In: Verbelen, T., Lanillos, P., Buckley, C.L., De Boom, C. (eds) *Active Inference*. IWAI 2020. *Communications in Computer and Information Science*, vol 1326. Springer, Cham. [https://doi.org/10.1007/978-3-030-64919-7\\_14](https://doi.org/10.1007/978-3-030-64919-7_14)
  - Meder D, Rabe F, Morville T, Madsen KH, **Koudahl MT**, Dolan RJ, et al. (2021) *Ergodicity-breaking reveals time optimal decision making in humans*. *PLoS Comput Biol* 17(9): e1009217. <https://doi.org/10.1371/journal.pcbi.1009217>

