

---

# The Local Inconsistency Resolution Algorithm

---

Oliver Richardson<sup>1</sup>

## Abstract

We present a generic algorithm for learning and approximate inference across a broad class of statistical models, that unifies many approaches in the literature. Our algorithm, called local inconsistency resolution (LIR), has an intuitive epistemic interpretation. It is based on the theory of probabilistic dependency graphs (PDGs), an expressive class of graphical models rooted in information theory, which can capture inconsistent beliefs.

## 1. Introduction

What causes a person to change their mind? According to some, it is a response to inconsistency: the result of discovering new information that contradicts our beliefs, or becoming aware of discrepancies between beliefs we already hold. Inconsistency can be difficult to detect, however, and indeed can only be resolved once we are aware of it. A further complication is that some things may be beyond our control; for example, we might receive conflicting information from two trusted sources and be unable to do resolve their disagreement. So in practice, we resolve inconsistencies *locally*—by looking at only a small part of the picture, and changing another part of it. This can have externalities; fixing one inconsistency can easily create others out of view.

Despite its imperfections, this process of locally resolving inconsistency can be quite useful. As we shall soon see, it is a powerful recipe for learning and approximate inference. We formalize the process in the language of probability and convex optimization, and show how that many popular techniques in the literature arise naturally as instances of it.

Our approach leans heavily on the theory of Probabilistic Dependency Graphs (PDGs), which are very flexible graphical models that allow for arbitrary—even inconsistent—probabilistic information, weighted by confidence (Richard-

son & Halpern, 2021). There is a natural way to measure how inconsistent a PDG is, and many standard loss functions can be viewed as measuring the inconsistency of a PDG that describes the appropriate situation (Richardson, 2022). We introduce an algorithm to operationalize the process of adjusting parameters to resolve this inconsistency.

Computing the degree of inconsistency can be intractable. Variational inference can be understood as adopting extra beliefs to get an overapproximation that is easier to calculate (Richardson, 2022). Our approach also enables the opposite: focusing on small parts of the graph at a time to address tractable underapproximations of the global inconsistency. Thus, it enjoys benefits such as higher potential for parallelization. It is also very expressive.

## 2. Background and Preliminaries

We write  $\mathcal{V}X$  for the set of values that a variable  $X$  can take on, and  $\Delta\mathcal{V}X$  for the set of distributions over  $\mathcal{V}X$ . A conditional probability distribution (cpd) is a map  $p(Y|X) : \mathcal{V}X \rightarrow \Delta\mathcal{V}Y$ . A *directed hypergraph*  $(N, \mathcal{A})$  is a set of nodes  $N$  and a set of arcs  $\mathcal{A}$ , each  $a \in \mathcal{A}$  of which is associated with a set  $S_a \subseteq N$  of source nodes, and  $T_a \subseteq N$  target nodes. We also write  $S \xrightarrow{a} T \in \mathcal{A}$  to specify an arc  $a$  together with its sources  $S = S_a$  and targets  $T = T_a$ .

**Geometry.** We will need various parameter spaces  $\Theta$ . To simplify the presentation, assume that each  $\Theta$  is a convex subset of  $\mathbb{R}^n$  (not necessarily of the same dimension). A *vector field* over  $\Theta$  is a differentiable map  $X$  assigning to each  $\theta \in \Theta$  a vector  $X_\theta \in \mathbb{R}^n$ . The *gradient* of a twice differentiable map  $f : \Theta \rightarrow \mathbb{R}$ , which we write  $\nabla_\Theta f(\Theta)$ , is a vector field. Given a vector field  $X$  and an initial point  $\theta_0 \in \Theta$ , there is a unique trajectory  $y(t)$  that solves the ODE  $\{\frac{d}{dt}y(t) = X_{y(t)}, y(0) = \theta_0\}$ , and we adopt the notation  $\exp_{\theta_0}(X) := y(1)$  for a compact description of it. At first glance,  $\exp$  only gives us access to  $y(1)$ , but it is easily verified that  $\exp_{\theta_0}(tX) = y(t)$ . So altogether, the map  $t \mapsto \exp_\theta(t\nabla_\Theta f(\Theta))$  is the smooth path beginning at  $\theta$  that follows the gradient of  $f$ . It is known as *gradient flow*.

**Probabilistic Dependency Graphs.** A PDG is a directed graph whose arcs carry probabilistic and causal information, weighted by confidence (Richardson & Halpern, 2021). We give an equivalent variant, whose explicit parametric nature

---

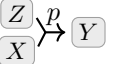

<sup>1</sup>Department of Computer Science, Cornell University, Ithaca NY, USA. Correspondence to: Oliver Richardson <oli@cs.cornell.edu>.

will prove useful for our purposes.

**Definition 2.1.** A *Parametric Probabilistic Dependency Graph* (PPDG)  $\mathcal{M}(\Theta) = (\mathcal{X}, \mathcal{A}, \Theta, \mathbb{P}, \alpha, \beta)$  is a directed hypergraph  $(\mathcal{X}, \mathcal{A})$  whose nodes correspond to variables, each arc  $a \in \mathcal{A}$  of which is associated with:

- a parameter space  $\Theta_a \subseteq \mathbb{R}^n$ , with a default value  $\theta_a^{\text{init}}$ .
- a map  $\mathbb{P}_a : \Theta_a \times \mathcal{V}S_a \rightarrow \Delta \mathcal{V}T_a$  that gives a cpd  $\mathbb{P}_a^\theta(T_a | S_a)$  over  $a$ 's targets given its sources, for every  $\theta \in \Theta_a$ ,
- confidences  $\alpha_a \in \mathbb{R}$  in the functional dependence of  $T_a$  on  $S_a$  expressed by  $a$ , and  $\beta_a \in [0, \infty]$  in the cpd  $\mathbb{P}_a$ .

A PDG is the object obtained by fixing the parameters; thus, a choice of  $\theta \in \Theta := \prod_{a \in \mathcal{A}} \Theta_a$  yields a PDG  $\mathcal{M} = \mathcal{M}(\theta)$ .

Clearly, a PDG is the special case of a PPDG in which every  $\Theta_a = \{\theta_a^{\text{init}}\}$  is a singleton. Conversely, a PPDG may be viewed as a PDG by adding each  $\Theta_a$  as a variable. Either variant can be specified with graphical notation, drawing a cpd  $p(Y|X, Z)$  as  and  $q(A, B)$  as . By default, take  $\beta, \alpha = 1$ . Given PDGs  $\mathcal{M}_1$  and  $\mathcal{M}_2$ , let denote  $\mathcal{M}_1 + \mathcal{M}_2$  the PDG with their combined information.  $\odot$  denotes pointwise multiplication.

**PDG Semantics and Inconsistency.** The power of PDGs comes from their semantics, which sew their (possibly inconsistent) cpds and confidences together into joint probabilistic information. A PDG contains two kinds of information: structural information about causal mechanisms, (the graph  $\mathcal{A}$  and weights  $\alpha$ ), and observational data (the cpds  $\mathbb{P}$  and confidences  $\beta$ ). With respect to a PDG  $\mathcal{M}$ , the *observational incompatibility* of a joint probability measure  $\mu \in \Delta \mathcal{V}\mathcal{X}$  is given by a weighted sum of relative entropies

$$OInc_{\mathcal{M}}(\mu) := \sum_{S \xrightarrow{a} T \in \mathcal{A}} \beta_a D\left(\mu(T, S) \parallel \mathbb{P}_a(T|S)\mu(S)\right), \quad (1)$$

and can be thought of as the excess cost of using codes optimized for each cpd, weighted by the confidence we have in them, if  $\mathcal{X} \sim \mu$ . If  $\beta > 0$ , then  $OInc_{\mathcal{M}}(\mu) = 0$  iff  $\mu$  satisfies the constraints imposed by every cpd of  $\mathbb{P}$ .

We can also score  $\mu$  by its incompatibility with the structural information  $(\mathcal{A}, \alpha)$ . This *structural deficiency* is given by:<sup>1</sup>

$$SDef_{\mathcal{M}}(\mu) := \mathbb{E}_{\mu} \left[ \log \frac{\mu(\mathcal{X})}{\lambda(\mathcal{X})} \prod_{S \xrightarrow{a} T} \left( \frac{\lambda(T|S)}{\mu(T|S)} \right)^{\alpha_a} \right], \quad (2)$$

and, roughly, measures  $\mu$ 's failure to arise as a result of independent causal mechanisms along each edge. If  $\mathcal{A}$  is a qualitative Bayesian Network, for instance, then  $SDef_{\mathcal{A}}(\mu) \geq 0$  with equality iff  $\mu$  has the independencies of  $\mathcal{A}$ . We encourage the reader to consult previous work for further details.

<sup>1</sup>In (2),  $\lambda$  is base measure, a property of  $\mathcal{X}$ . The precise choice doesn't matter, but think: uniform or the appropriate analogue.

With confidence  $\gamma \geq 0$  in the structural information overall, the  $\gamma$ -inconsistency of  $\mathcal{M}$  is the smallest possible overall incompatibility of any distribution with  $\mathcal{M}$ , and denoted

$$\llbracket \mathcal{M} \rrbracket_{\gamma} := \inf_{\mu} \left( OInc_{\mathcal{M}}(\mu) + \gamma SDef_{\mathcal{M}}(\mu) \right). \quad (3)$$

Richardson (2022) argues that this inconsistency measure (3) is a “universal” loss function, largely showing how it specializes to standard loss functions in a wide variety of situations. It follows that, at an abstract level, much of machine learning can be viewed as inconsistency resolution. We take this idea a few steps further, by operationalizing the resolution process, and allowing it to be done locally.

### 3. Local Inconsistency Resolution (LIR)

There are two distinct senses in which inconsistency resolution can be *local*: we can restrict what we can see, or what we can do about it. Correspondingly, there are two “focus” knobs for our algorithm: one restricts our attention to the inconsistency of a subset of arcs  $A \subseteq \mathcal{A}$ , and the other restricts our control to (only) the parameters of a subset  $C \subseteq A$  as we resolve that inconsistency. The former makes for an underestimate of the inconsistency that is easier to calculate, while the latter makes for an easier optimization problem. These restrictions are not just cheap approximations, though: they are also appropriate modeling assumptions for actors that cannot see and control everything at once.

Attention and control need not be black and white. A more general approach is to choose an *attention mask*  $\varphi \in \mathbb{R}^{\mathcal{A}}$  and a *control mask*  $\chi \in [0, \infty]^{\mathcal{A}}$ . Large  $\varphi(a)$  makes  $a$  salient while  $\varphi(a) = 0$  keeps it out of mind; similarly, large  $\chi(a)$  gives significant freedom to change  $a$ 's parameters, small  $\chi(a)$  affords only minor adjustments, and  $\chi(a) = 0$  prevents change altogether.

The full procedure modifies a PPDG  $\mathcal{M}(\Theta)$  so as to be consistent with its context. First, receive context PDG  $Ctx$ , and initialize mutable memory  $\mathcal{M}(\Theta)$ . In each iteration, choose  $\gamma$ , a control mask  $\chi$  over  $\mathcal{M}(\Theta)$ , and an attention mask  $\varphi$  over  $\mathcal{M}(\Theta) + Ctx$ . Calculate  $\llbracket \varphi \odot (\mathcal{M}(\Theta) + Ctx) \rrbracket_{\gamma}$ , the inconsistency of the combined context and memory, weighted by attention. (This can be done with the methods of Richardson et al. (2023).) Then mitigate this local inconsistency by updating mutable memory via (an approximation to) gradient flow, changing  $a$ 's parameters in proportion to control  $\chi(a)$ . This is formalized in Algorithm 1.

Before we can run Algorithm 1, we must supply two more procedures: REFRESH, to get new context in online settings (the identity by default), and REFOCUS, to select focus and control masks. We focus on the case where REFOCUS chooses non-deterministically from a fixed finite set of attn/ctrl mask pairs  $\mathbf{V}$ , which we call *views*.

The ODE described in the final line may be approximated

**Algorithm 1** Local Inconsistency Resolution (LIR)

**Input:** context  $Ctx$ , mutable memory  $\mathcal{M}(\Theta)$ .  
 Initialize  $\theta^{(0)} \leftarrow \theta^{\text{init}}$ ;  
**for**  $t = 0, 1, 2, \dots$  **do**  
      $Ctx \leftarrow \text{REFRESH}(Ctx)$ ; //optional  
      $\varphi, \chi, \gamma \leftarrow \text{REFOCUS}()$ ;  
      $\theta^{(t+1)} \leftarrow \exp_{\theta^{(t)}} \left\{ -\chi \odot \nabla_{\Theta} \left\langle \varphi \odot (Ctx + \mathcal{M}(\Theta)) \right\rangle_{\gamma} \right\}$ ;

with an inner loop running an iterative gradient-based optimization algorithm. Alternatively, if REFOCUS produces small  $\chi$ , then it is well-approximated by a single gradient descent step of size  $\chi$ . At the other extreme, if  $\chi$  is infinite in every component, then so long as the parameterizations  $\mathbb{P}$  are log-concave, the final line reduces to

$$\theta^{(t+1)} \leftarrow \arg \min_{\theta} \left\langle \varphi \odot (Ctx + \mathcal{M}(\theta)) \right\rangle_{\gamma},$$

at least in a many cases of interest, because of

**Proposition 3.1.** *If  $\mathcal{M}(\Theta)$  is a PPDG whose parameterizations  $\mathbb{P}$  are either constant or unconditional and log-concave, then for small enough  $\gamma$ , the map  $\theta \mapsto \left\langle \varphi \odot (Ctx + \mathcal{M}(\theta)) \right\rangle_{\gamma}$  is convex.<sup>2</sup>*

In the remaining sections, we give a sample of some historically important algorithms that are instances of LIR.

#### 4. LIR in the Classification Setting

Consider a parametric classifier  $p_{\theta}(Y|X)$ , perhaps arising from a neural network whose final layer is a softmax. Suppose  $\mathcal{V}Y$  is a finite set of classes. If  $\mathcal{V}X$  is itself a manifold (such as the space of images), we can regard a value  $x \in \mathcal{V}X$  as parameterizing a deterministic cpd, written  $x \rightsquigarrow X$ . Together with a labeled sample  $(x, y)$ , we get a PPDG  $\mathcal{M}(\theta) := x \rightsquigarrow X \xrightarrow{p_{\theta}} Y \leftarrow y$  whose observational inconsistency is  $\langle \mathcal{M} \rangle_0 = -\log p_{\theta}(y|x)$ , the standard training objective for such a classifier (Richardson, 2022). Each cpd plays major role in this inconsistency. What happens when we resolve this it with control over different arcs?

- Adjusting  $\theta$  amounts to training the network in the standard way. In this case, the value  $\chi$  of the control mask corresponds roughly to the product of the learning rate and the number of optimization iterations.
- Adjusting  $y$  is like a forward pass, in that it adjusts  $y$  to match distribution  $p_{\theta}(Y|x)$ .
- Adjusting  $x$  creates an adversarial example. That is, it makes small changes to the input until the (fixed) network gives it the desired label.

**SGD.** Take the mutable state to be the classifier  $p$  as be-

<sup>2</sup>All proofs can be found in the appendix.

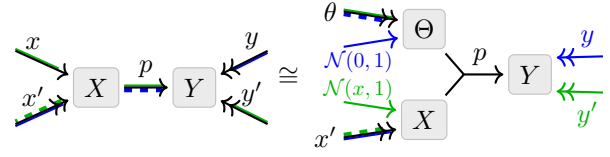


Figure 1. Two equivalent illustrations of adversarial training.

fore. Define REFRESH so that it draws a batch of samples  $\{(x_i, y_i)\}_{i=1}^m$ , and returns a PDG with a single arc describing their empirical distribution  $d(X, Y)$ , and let REFOCUS be such that  $\varphi(d) = \infty$  (reflecting high confidence in the data). If  $\eta := \chi(p)\varphi(p)$  is small, then LIR is stochastic gradient descent with batch size  $m$  and learning rate  $\eta$ .

**Adversarial training.** Suppose we want to slightly alter  $x$  to obtain  $x'$  that is classified as  $y'$  instead of  $y$ . Adding  $x'$  and  $y'$  to  $\mathcal{M}$  and relaxing  $\mathbb{P}_x$  to be a Gaussian centered  $x$  rather than a point mass, we get the PPDG on the left of Figure 1. Taking a view of only the edges marked in green (with control over the dashed green edge) a LIR iteration is an adversarial attack with Euclidean distance (Biggio et al., 2013). The blue view, by contrast, “patches” the adversarial example by adjusting the model parameters to again classify it correctly. Thus, LIR that alternates between the two and refreshes  $(x, y, y')$  is adversarial training, a standard defense to adversarial attacks (Goodfellow et al., 2014).

The ML community’s focus on adversarial examples may appear to be a cultural phenomenon, but at a mathematical level, it is no accident. At this level of abstraction, there is no difference between the network parameters and inputs. Making the parameterization of  $p$  explicit and adding L2 regularization, the symmetry becomes striking (see Figure 1, right). Thus, it is just as sensible to train the inputs, as the network (Kishore et al., 2021).

#### 5. The EM Algorithm as LIR

Suppose we have a generative model  $p(Z, X|\Theta)$  describing the probability over an observable variable  $X$  and a latent one  $Z$ . Given an observation  $X=x$ , the standard approach for trying to learn the parameters despite the missing data is called the EM algorithm. It iteratively computes

$$\theta_{\text{EM}}^{(t+1)} = \arg \max_{\theta} \mathbb{E}_{z \sim p(Z|x, \theta_{\text{EM}}^{(t)})} [\log p(x, z|\theta)].$$

**Proposition 5.1.** *LIR  $\left( x \rightsquigarrow X, X \xrightarrow{p} Z \xleftarrow{q} \right)$  in which REFOCUS fixes  $\varphi, \gamma = 1$  and alternates between full control of  $p$  and  $q$  implements EM, in that  $\theta_{\text{EM}}^{(t)} = \theta_{\text{LIR}}^{(2t)}$ .*

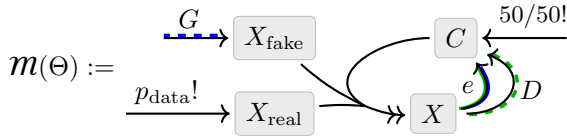
This result is closely related to one due to Neal & Hinton (1998), who view it as an intuitive explanation of why the EM algorithm works. Indeed, it is obvious in this form that

every adjustment reduces the overall inconsistency. The result can also be readily adapted to an entire dataset by replacing  $x$  with a high confidence empirical distribution, or batched with the same technique in Section 4. It also captures fractional EM when  $\chi < \infty$ .

This form of the EM algorithm is closely related to variational inference. Indeed, analogous choices applied to the analysis of Richardson (2022) yields the usual training algorithm for variational autoencoders (VAEs).

## 6. Generative Adversarial Training as LIR

LIR also subsumes more complex training procedures such as the one used to train GANs (Goodfellow et al., 2020). The goal is to train a network  $G$  to generate images that cannot be distinguished from real ones. More precisely, define  $X$  to be either an image  $X_{\text{fake}} \sim G$  or from a dataset  $X_{\text{real}} \sim p_{\text{data}}$ , based on a fair coin  $C$ . A discriminator  $D$  then predicts  $C$  from  $X$ . The generator also has a belief that, even given  $X$ , the coin is equally likely heads as tails (call this  $e$ ). This state of affairs is summarized below.



The GAN objective is typically written as a 2-player mini-max game:  $\min_G \min_D \mathcal{L}^{\text{GAN}}(G, D)$ , where

$$\mathcal{L}^{\text{GAN}}(G, D) = \mathbb{E}_{x \sim p_{\text{data}}} \log D(x) + \mathbb{E}_{x' \sim G} \log(1 - D(x')).$$

**The Discriminator’s View.** The discriminator has full control over  $D$ , and attends to everything but  $e$ . That inconsistency of this PDG is what might be called the discriminator’s objective: the expected KL divergence from  $D$  to the optimal discriminator. If  $D$  also disbelieves that any image is equally likely to be fake as real (by choosing  $\varphi(e) = -1$ ), then the inconsistency becomes  $-\mathcal{L}^{\text{GAN}}$ .

**The Generator’s View.** The generator has control over  $G$ . If it ignores  $D$  attends only to  $e$ , the inconsistency is the Jensen-Shanon Divergence between  $G$  and  $p_{\text{data}}$ . If the generator also disbelieves the discriminator  $D$  (i.e.,  $\varphi(D) = -1$ ), then the inconsistency becomes  $+\mathcal{L}^{\text{GAN}}$ . Standard practice is to use small  $\chi(G)$  and large  $\chi(D)$ .

## 7. Message Passing Algorithms as LIR

A *factor graph* over a set of variables  $\mathcal{X}$  is a set of factors  $\Phi = \{\phi_a : \mathbf{X}_a \rightarrow \mathbb{R}_{\geq 0}\}_{a \in \mathcal{A}}$ , where each  $\mathbf{X}_a \subseteq \mathcal{X}$  is called the *scope* of  $a$ . Conversely, for  $X \in \mathcal{X}$ , let  $\partial X$  be the set of factors with  $X$  in scope.  $\Phi$  specifies a distribution  $\Pr_{\Phi}(\mathcal{X}) \propto \prod_a \phi_a(\mathbf{X}_a)$ , and corresponds to a PDG

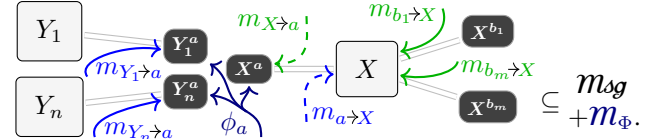
$$m_{\Phi} = \left\{ \frac{\phi_a}{(\alpha, \beta=1)} \rightarrow \boxed{\mathbf{X}_a} \right\}_{a \in \mathcal{A}} \quad \text{that specifies the same distribution, when } \gamma = 1.$$

Sum-product belief propagation (Kschischang et al., 2001) aims to approximate marginals of  $\Pr_{\Phi}$  with only local computations: messages sent between factors and the variables they have in scope. Its state consists of pairs of “messages”  $\{m_{X \rightarrow a}, m_{a \rightarrow X}\}$ , both (unnormalized) distributions over  $X$ , for each pair  $(a, X)$  with  $a \in \partial X$ , which together form a PDG  $\mathcal{M}_{\Phi}$  in the same way as the original factor graph. After initialization, belief propagation repeatedly recomputes:

$$m_{X \rightarrow a}(x) := \prod_{b \in \partial X \setminus a} m_{b \rightarrow X}(x) \quad (4)$$

$$m_{a \rightarrow X}(x) := \sum_{\mathbf{y} \in \mathcal{V}(\mathbf{X}_a \setminus X)} \phi_a(x, \mathbf{y}) \prod_{Y \in \mathbf{X}_a \setminus X} m_{Y \rightarrow a}(Y(\mathbf{y})), \quad (5)$$

where  $Y(\mathbf{y})$  is the value of  $Y$  in the joint setting  $\mathbf{y}$ . Finally, variable marginals  $\{b_X\}_{X \in \mathcal{X}}$ , which we regard as another PDG,  $\mathcal{B}$ , are computed from the messages according to  $b_X(x) \propto \prod_{a \in \partial X} m_{a \rightarrow X}(x)$ . Observe that every calculation is a (marginal of) a product of factors, and thus amounts to inference in some “local” factor graph. The traditional depiction of messages moving between variables and factors (see Appendix A) is not so different from the PDG



Indeed, it can be shown that (4,5) minimize inconsistency of the dotted components in their appropriate contexts (shown in red and blue above, and formalized in Appendix A).

**Proposition 7.1.** *If REFOCUS selects a view non-deterministically from  $\{a \rightarrow X, X \rightarrow a, X\}_{X \in \mathcal{X}, a \in \partial X}$  (details in Appendix A), then the possible runs of  $\text{LIR}(\mathcal{M}_{\Phi}, \mathcal{M}_{\Phi} + \mathcal{B})$  are precisely those of BP for different message schedules.*

[\[link to proof\]](#)

The same construction works for general cluster graphs, and we suspect the same is true of the many other message passing algorithms that are generated from the  $\alpha$ -divergences (Minka, 2005), because of the close relationship those divergences have with simple PDGs (Richardson, 2022).

## 8. Decision Making with LIR

In this section, we show how a few standard decision rules can be viewed as inconsistency minimization. Richardson (2024, §4, §6) established that PDGs can represent expected cost, albeit by articulating some questionable probabilistic beliefs. For the moment, let us entertain the possibility that these strange probabilistic beliefs are in fact an appropriate way to encode preferences. A natural question then emerges: what does it mean to make decisions to as to minimize inconsistency in this context? In this section, we show that the standard picture of rational decision making—that is,

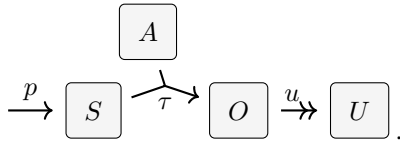


expected utility maximization—can be viewed as the pursuit of consistency in this probabilistic model.

Suppose we are trying to choose an action (i.e., control a variable  $A$ ). To do so reasonably, we need to model some context. Let  $S$  be a variable representing the current state of the world, and  $O$  represent the outcome after taking our action. To complete the standard picture, let's further assume that we have

1. some understanding of how our action  $A$  and the state  $S$  determine the outcome  $O$ , say in the form of a cpd  $\tau(O | S, A)$ ,
2. a belief  $p(S)$  about the current state of the world, and
3. a utility function  $u : \mathcal{VO} \rightarrow \mathbb{R}$  on possible outcomes.

It is easy to encode this information in a PDG:



What's missing is a way to encode the idea that higher utilities are “better” than lower ones. For this, we can add a “soft constraint” (see Richardson (2024, §4.2.2)) that is violated less at higher utilities than lower ones. At a technical level, recall that this means (1) including the variable  $T$  that can technically take on values  $\mathcal{VT} = \{f, t\}$ , yet happens to always on the value  $t$ , and (2) adding a cpd  $b(T | U)$  encoding a constraint violation that is more serious the lower the value of  $U$ . Let  $\mathcal{M}_{(p,\tau,u,b)}$ , or simply  $\mathcal{M}$ , represent the PDG above augmented with such a soft constraint  $b$ .

Intuitively, imagine that there is a part of you that you cannot control, which we will call “Faith”. Faith engages in wishful thinking: she disbelieves outcomes that are undesirable, creating epistemic conflict if she sees outcomes of low utility. (Technically, Faith refers to the sub-PDG consisting of  $u$ ,  $b$ , and  $T=t$ .) If you have no control over Faith, but still have confidence and pay attention to her, then it turns out that selecting actions to minimize your combined inconsistency is a decision rule that interpolates between

- (a) maximizing expected utility (when  $\beta_p \gg \beta_b$ ), and
- (b) maximizing maximum utility (when  $\beta_p \ll \beta_b$ ),

where  $\beta_p$  is your confidence in your prior probabilistic belief  $p(S)$ , and  $\beta_b$  is the confidence of the soft constraint  $b$  (i.e., your “degree of faith”).

**Proposition 8.1.** *Suppose  $b(T=t | U=u) = k \cdot \exp(u)$  for some constant  $k$ .*

1. *If  $\beta_p = \infty$  and  $\beta_b < \infty$ , then the action(s)  $a \in \mathcal{VA}$  that minimize the inconsistency are those that maximize*

*expected utility. Formally, for all  $\gamma \geq 0$ ,*

$$\arg \min_{a \in \mathcal{VA}} \langle \mathcal{M} + A=a \rangle_\gamma = \arg \max_{a \in \mathcal{VA}} \max_{s \sim p, o \sim \tau|s,a} \mathbb{E} [u(o)].$$

2. *If  $\beta_p < \infty$  and  $\beta_b = \infty$ , then the action(s)  $a \in \mathcal{VA}$  that minimize overall inconsistency are those that can lead to the best possible outcome, i.e.,*

$$\arg \min_{a \in \mathcal{VA}} \langle \mathcal{M} + A=a \rangle_\gamma = \arg \max_{a \in \mathcal{VA}} \max_{s \in \mathcal{VS}} \max_{o \sim \tau|s,a} \mathbb{E} [u(o)].$$

We suspect that it may also be possible to implement other decision rules such as minimax or maximin. Like the GAN objective however, these decision rules look like two-player games, and so will likely require two different focuses of LIR, rather than just one. We leave a careful exploration of this avenue to future work.

We conclude our discussion of decision theory with a high-level observation. One's preferences, beliefs, and actions can be jointly modeled with a PDG. When that PDG is inconsistent (i.e., there is a conflict between your preferences, beliefs, and actions), there are, in principle, three possible resolutions. You can resolve the inconsistency by changing your action, which amounts to maximizing expected utility (Proposition 8.1.1); this is thought of as the rational approach. Alternatively, you can change your preferences so as to become content with your current situation, which is perhaps a more zen perspective. Observe that these two resolutions to the internal conflict are two halves of the famous Serenity Prayer (Niebuhr, 1933):

*Give us the courage to change what must be altered,  
serenity to accept what can not be helped,  
and insight to know the one from the other.*

Finally, the third way to resolve the inconsistency involves changing your beliefs about what state you are in, i.e., wishful thinking. Unfortunately, this kind of self delusion typically only leads to more inconsistency in the future, and is seldom a good idea. It also illustrates how, in the process of resolving inconsistency, restricting control to a small set of parameters can be appropriate (and not just a computational shortcut).

## 9. Discussion and Future Work

These examples are only the beginning. Our initial investigations suggest that opinion dynamics models, the training process for diffusion models, and much more, are all naturally captured by LIR. The surprising generality of LIR begs some theoretical questions. What assumptions are needed to prove that it reduces overall inconsistency, as is often the case? How expressive is this mode of computation?

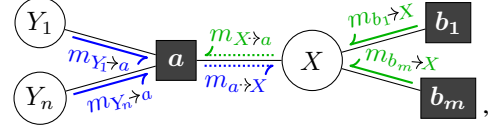
It also suggests a novel approach to structured generative modeling: haphazardly assemble a PDG with many variables, existing models, priors, constraints, and data of all shapes and sizes. Then, train new models to predict variables from one another, using LIR (with random refocusing, say). Is this effective? We are excited to find out!

## References

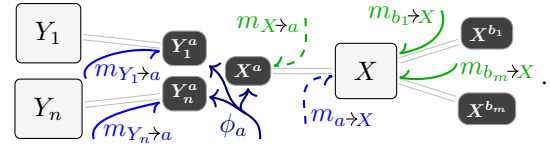
- Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Advanced Information Systems Engineering*, pp. 387–402. Springer Berlin Heidelberg, 2013. doi: 10.1007/978-3-642-40994-3\_25. URL [https://doi.org/10.1007%2F978-3-642-40994-3\\_25](https://doi.org/10.1007%2F978-3-642-40994-3_25).
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.
- Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples, 2014.
- Kishore, V., Chen, X., Wang, Y., Li, B., and Weinberger, K. Q. Fixed neural network steganography: Train the images, not the network. In *International Conference on Learning Representations*, 2021.
- Kschischang, F. R., Frey, B. J., and Loeliger, H.-A. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.
- Minka, T. Divergence measures and message passing. Technical Report MSR-TR-2005–173, Microsoft Research, Cambridge, U.K., 2005.
- Neal, R. M. and Hinton, G. E. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pp. 355–368. Springer, 1998.
- Niebuhr, R. pp. 1, 1933.
- Richardson, O. E. Loss as the inconsistency of a probabilistic dependency graph: Choose your model, not your loss function. *AISTATS '22*, 151, 2022.
- Richardson, O. E. *A Unified Theory of Probabilistic Modeling, Dependence, and Inconsistency*. PhD thesis, December 2024.
- Richardson, O. E. and Halpern, J. Y. Probabilistic dependency graphs. *AAAI '21*, 2021.
- Richardson, O. E., Halpern, J. Y., and Sa, C. D. Inference in probabilistic dependency graphs. *UAI '23*, 2023.

## A. Details on Belief Propagation

The usual schematic illustration of belief propagation is:



This standard diagram is only a schematic, but the PDG  $\mathcal{M}_{\text{log}}$  can be made to look similar. Adding a variable  $X^a$  for every pair  $(X, a)$  with  $X \in \mathbf{X}_a$  along with edges asserting that  $X^a = X$ , we obtain the equivalent PDG in the main body of the paper:



We now define the views. Modulo a small subtlety, the following is essentially true: (4) selects  $C_{X \rightarrow a} := \{m_{X \rightarrow a}\}$  so as to minimize 1-inconsistency in context  $A_{X \rightarrow a} := \{m_{b \rightarrow X}\}_{b \in \partial X \setminus a} \cup \{m_{X \rightarrow a}\}$ , while (5) selects  $C_{a \rightarrow X} := \{m_{a \rightarrow X}\}$  so as to minimize the 1-inconsistency in context  $A_{a \rightarrow X} := \{\phi_a, m_{a \rightarrow X}\} \cup \{m_{Y \rightarrow a}\}_{Y \in \mathbf{X}_a \setminus X}$ .

The only wrinkle is that we do not attend to the *structural* aspect of the edges  $e$  that we’re updating; i.e., we must select  $\varphi$  to effectively set  $\alpha_e = 0$ . Intuitively: although all of the input messages summarize causal information, we’re trying to capture that information with a distribution. Thus, it’s not appropriate to attend to the causal structure of the edges that we’re Thus, for each view  $v \in \mathbf{V} := \bigcup_{a \in \mathbf{A}, X \in \mathbf{X}_a} \{a \rightarrow X, X \rightarrow a, X\}$ , define

$$\varphi(a) := \begin{cases} \begin{pmatrix} 1 \\ 1 \end{pmatrix} & \text{if } a \in A_v \setminus C_v \\ \begin{pmatrix} 1 \\ 0 \end{pmatrix} & \text{if } a \in C_v \\ \begin{pmatrix} 0 \\ 0 \end{pmatrix} & \text{otherwise} \end{cases},$$

where  $\begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix}$  scales  $\beta_a$  by  $\phi_1$  and  $\alpha_a$  by  $\phi_2$ . For completeness, for each view  $v \in \mathbf{V}$ , define

$$\chi(a) := \begin{cases} \infty & \text{if } a \in C_v \\ 0 & \text{otherwise.} \end{cases}$$

With these definitions, Proposition 7.1 follows easily.

## B. Proofs

First, some extra details for Proposition 3.1. By parameteriations  $\mathbb{P}$  log-concave, we mean that, for every  $a \in \mathcal{A}$ , and  $(s, t) \in \mathcal{V}(S_a, T_a)$ , the function

$$\theta \mapsto -\log \mathbb{P}_a^\theta(T_a = t \mid S_a = a) \quad : \Theta_a \rightarrow [0, \infty]$$

is convex. This is true for many families of distributions of interest. For example, if  $S_a, T_a$  is discrete, and the cpd is parameterized by stochastic matrices  $\mathbf{P} = [p_{s,t}] \in [0, 1]^{\mathcal{V}(S_a, T_a)}$ , then

$$-\log \mathbb{P}_a^{\mathbf{P}}(T_a = t \mid S_a = s) = -\log(p_{s,t})$$

which is clearly convex in  $\mathbf{P}$ .

To take another example: if  $\mathbb{P}_a$  is linear Gaussian, i.e.,  $\mathbb{P}_a(T|S) = \mathcal{N}(T|\mathbf{A}s + b, \sigma^2)$ , parameterized by  $(\mathbf{A}, b, 1/\sigma^2)$ , then

$$-\log \mathbb{P}_a^{\mathbf{A}, b, \sigma^2}(t|s) = -\frac{1}{2} \log \frac{2\pi}{\sigma^2} + \frac{1}{2} \left( \frac{t - \mathbf{A}s + b}{\sigma} \right)^2$$

which is convex in  $(\mathbf{A}, b, \frac{1}{\sigma^2})$ . Now, for the proof.

**Proposition 3.1.** *If  $\mathcal{M}(\Theta)$  is a PPDG whose parameterizations  $\mathbb{P}$  are either constant or unconditional and log-concave, then for small enough  $\gamma$ , the map  $\theta \mapsto \langle\langle \varphi \odot (\text{Ctx} + \mathcal{M}(\theta)) \rangle\rangle_\gamma$  is convex.*

*Proof.* By definition,

$$\langle\langle \varphi \odot (\text{Ctx} + \mathcal{M}(\theta)) \rangle\rangle_\gamma = \inf_{\mu} \{ OInc_{\text{Ctx}}(\mu) + OInc_{\mathcal{M}(\theta)}(\mu) + SDef_{\mathcal{M}(\theta)}(\mu) + OInc_{\mathcal{M}(\theta)}(\mu) \}.$$

Only the final term actually depends on  $\theta$ , though. Let  $F(\mu)$  capture the first three terms. For all of our examples, and indeed, if  $\gamma$  is chosen small enough, it will be convex in  $\mu$  (Richardson & Halpern, 2021). Then we have

$$\begin{aligned} \langle\langle \varphi \odot (\text{Ctx} + \mathcal{M}(\theta)) \rangle\rangle_\gamma &= \inf_{\mu} \left( F(\mu) + \mathbb{E}_{\mu} \left[ \sum_{S \stackrel{a}{\rightarrow} T} \beta_a \log \frac{\mu(T|S)}{\mathbb{P}_a(T|S)} \right] \right) \\ &= \inf_{\mu} \left( F(\mu) + \mathbb{E}_{\mu} \left[ \sum_{S \stackrel{a}{\rightarrow} T} \beta_a \log \frac{\mu(T|S)}{\lambda(T|S)} \right] + \mathbb{E}_{\mu} \left[ \sum_{S \stackrel{a}{\rightarrow} T} \beta_a \log \frac{\lambda(T|S)}{\mathbb{P}_a(T|S)} \right] \right) \end{aligned}$$

The second term is then entropy (relative to the base distribution), which is convex in  $\mu$ . The first term,  $F(\mu)$ , is convex in  $\mu$  as well, and neither depend on  $\theta$ . The final term is linear in  $\mu$ . Since  $\mathbb{P}$  is log-convex in  $\theta$ , the  $\log \frac{\lambda(t|s)}{\mathbb{P}_a(t|s)}$  convex in  $\theta$ , and so that third term is a conic combination of expectations that are all convex, and hence itself convex in  $\theta$ . Thus, the sum of all three terms in the infimum is jointly convex in  $\theta$  and in  $\mu$ . Taking an infimum over  $\mu$  pointwise, the result is still convex in  $\theta$ .  $\square$

**Proposition 7.1.** *If REFOCUS selects a view non-deterministically from  $\{a \rightarrow X, X \rightarrow a, X\}_{X \in \mathcal{X}, a \in \partial X}$  (details in Appendix A), then the possible runs of  $\text{LIR}(\mathcal{M}_{\Phi}, \mathcal{M}_{\text{sg}} + \mathcal{B})$  are precisely those of BP for different message schedules.*

*Proof.* When  $\gamma = 1$ , and  $\alpha, \beta = 1$  for all of the input factors, then the optimal distribution  $\mu^*$  that realizes the infimum is just the product of factors. It follows that any distribution that has those marginals will minimize the observational inconsistency.

The different orders that the (4), and (5) can be ordered for different adjacent pairs  $(a, X)$  correspond to both the message passing schedules, and to the possible view selections of LIR.  $\square$

**Proposition 8.1.** *Suppose  $b(T=t \mid U=u) = k \cdot \exp(u)$  for some constant  $k$ .*



1. If  $\beta_p = \infty$  and  $\beta_b < \infty$ , then the action(s)  $a \in \mathcal{VA}$  that minimize the inconsistency are those that maximize expected utility. Formally, for all  $\gamma \geq 0$ ,

$$\arg \min_{a \in \mathcal{VA}} \langle \mathbf{m} + A=a \rangle_\gamma = \arg \max_{a \in \mathcal{VA}} \mathbb{E}_{\substack{s \sim p \\ o \sim \tau | s, a}} [u(o)].$$

2. If  $\beta_p < \infty$  and  $\beta_b = \infty$ , then the action(s)  $a \in \mathcal{VA}$  that minimize overall inconsistency are those that can lead to the best possible outcome, i.e.,

$$\arg \min_{a \in \mathcal{VA}} \langle \mathbf{m} + A=a \rangle_\gamma = \arg \max_{a \in \mathcal{VA}} \max_{s \in \mathcal{VS}} \mathbb{E}_{o \sim \tau | s, a} [u(o)].$$

*Proof.* Since the choice  $a$  is deterministic, the value of  $A$  is determined; likewise the value of  $T$  is also fixed. Similarly, as  $u$  is a deterministic function, the value of  $U$  is determined according to  $u$ . Thus we need only consider distributions  $\mu(S, O)$  in our minimization; the other variables can be found as a function of these. However, we also know that  $\mu(O | S) = \tau(O | S, A=a)$  since it is given with high confidence. Therefore it suffices to restrict our search to distributions over the variable  $S$ .

To simplify notation, let  $EU(s, a) := \mathbb{E}_{o \sim \tau(O | s, a)} [u(o)] + \log k$  be the expected utility of an action, shifted by the constant  $\log k$ . Note that

$$\log \frac{1}{b(T=t | U=u(o))} = -\log(k \cdot \exp(u(o))) = -u(o) - \log k,$$

which in expectation over  $\tau(O | s, a)$ , is  $-EU(s, a)$ . With this in mind, we calculate:

$$\begin{aligned} \langle \mathbf{m}_{p, \tau, u, b} + A=a \rangle_\gamma &= \inf_{\mu(S)} \beta_p \mathbb{E}_{s \sim \mu} \left[ \log \frac{\mu(s)}{p(s)} + \frac{\beta_b}{\beta_p} \mathbb{E}_{o \sim \tau | a, s} \left[ \log \frac{1}{b(T=t | U=u(o))} \right] \right] \\ &= \inf_{\mu(S)} \beta_p \mathbb{E}_{s \sim \mu} \left[ \log \frac{\mu(s)}{p(s)} + \log \circ \exp \left( -\frac{\beta_b}{\beta_p} EU(s, a) \right) \right] \\ &= \inf_{\mu(S)} \beta_p \mathbb{E}_{s \sim \mu} \left[ \log \frac{\mu(s)}{1} \cdot \frac{\exp(-\frac{\beta_b}{\beta_p} EU(s, a))}{p(s)} \cdot \frac{Z}{Z} \right] \\ &= \inf_{\mu(S)} \beta_p \mathbb{E}_{s \sim \mu} \left[ \log \frac{\mu(s)}{1} \cdot \frac{Z}{p(s) \cdot \exp(\frac{\beta_b}{\beta_p} EU(s, a))} \cdot \frac{1}{Z} \right] \end{aligned} \quad (6)$$

At this point, we can use the same trick used repeatedly in the proving results of Richardson (2022): take  $Z$  to be the normalization constant needed to regard the middle fraction as the inverse of a probability distribution  $\nu$ . Once we do so, we are left with an infimum over a KL divergence  $D(\mu \| \nu)$  plus the expectation of a constant:

$$\langle \mathbf{m}_{p, \tau, u, b} + A=a \rangle_\gamma = \beta_p \log \frac{1}{Z} = -\log \sum_s p(s) \exp \left( + \frac{\beta_b}{\beta_p} EU(s, a) \right).$$

We now look at the two extreme cases.

When  $\beta_p \rightarrow \infty$ , then the ratio  $\frac{\beta_b}{\beta_p}$  becomes small, and we can use the fact that  $\exp(\epsilon) \approx 1 + \epsilon$  for small  $\epsilon$  to find that the inconsistency of interest is approximately

$$\approx -\beta_p \log \sum_s p(s) \left[ 1 + \frac{\beta_b}{\beta_p} EU(s, a) \right] = -\beta_p \log(1 + \frac{\beta_b}{\beta_p} EU(s, a)) \approx -\beta_b EU(s, a)$$

Alternatively, more directly, when  $\beta_p = \infty$ , the optimal distribution must be  $\mu = p$ , and so the inconsistency is immediately  $-\mathbb{E}_{s \sim p} [\beta_b EU(s, a)]$ . Either way, minimizing this quantity amounts to maximizing expected utility, as the two differ by a negative affine transformation.

At the other extreme, when  $\beta_b \rightarrow \infty$ , we can write our expression in terms of  $\text{LSE}\{x_1, \dots, x_n\} := \log \sum_{i=1}^n \exp(x_i)$  (LogSumExp) which is a smooth approximation to a max. (In a moment, we will negate its arguments and the final

output, using it as an approximation to a min, instead.) Picking back up from (6) and letting  $t := \frac{\beta_b}{\beta_p}$ , we find

$$\langle\langle m_{p,\tau,u,b} + A=a \rangle\rangle_\gamma = -\beta_b \cdot \frac{-1}{t} \text{LSE}_{s \in \mathcal{VS}} \left[ -t \cdot \left( \frac{1}{t} \log \frac{1}{p(s)} - EU(s, a) \right) \right].$$

Using the standard fact<sup>a</sup> that

$$\min_{i \in [n]} x_i - \frac{1}{t} \log n \leq \frac{-1}{t} \text{LSE}_{i \in [n]} (-tx_i) < \min_{i \in [n]} x_i,$$

we find that, in our case,

$$M - \frac{1}{t} \log |\mathcal{VS}| \leq \frac{1}{\beta_b} \langle\langle m_{p,\tau,u,b} + A=a \rangle\rangle_\gamma \leq M$$

where  $M := \min_{s \in \mathcal{VS}} (-EU(s, a) + \frac{1}{t} \log \frac{1}{p(s)})$ . In particular, when  $\beta_b \rightarrow \infty$ , meaning  $t \rightarrow \infty$ , the gap between the upper and lower bounds shrinks to zero, and the resulting inconsistency becomes  $-\min_{s \in \mathcal{VS}} (-EU(s, a)) = \max_{s \in \mathcal{VS}} EU(s, a)$ , proving the result.  $\square$

<sup>a</sup>Letting  $m := \min_i x_i$ , observe that, for all  $t > 0$ , we have  $\exp(-tm) \leq \sum_i \exp(-tx_i) \leq n \exp(-tm)$ . Apply a logarithm and multiply by  $-\frac{1}{t}$  to get the promised result.