Formulating Detection and Breaking of Parameter Symmetries

Kenny Chiu

December 9, 2019

1 Background

Probabilistic programming is a modern programming paradigm in which inference is automatically performed on probabilistic models that are specified by the user. The probabilistic models are often in the form of a generative Bayesian model and can be conveniently expressed by a graphical model [Rai17]. An important aspect of probabilistic programming is that the choice of inference algorithm used for the given probabilistic model is left up to the underlying inference engine. By abstracting away the inference step, users have the flexibility to work with multiple models without needing to modify their inference procedure between models.

However, existing inference algorithms do not work well in all models with one challenge being the possible existence of *symmetries* in the parameterization of a model [NMT13]. A symmetry can be informally described as a transformation to the parameters that does not change the overall model. If a symmetry is present in a model, it poses a problem of parameter nonidentifiability where the results of inference may be of poor quality and difficult to interpret. Another consequence of having parameter symmetries is that different samples of the model parameters may be strongly correlated as a result. In particular, this may reduce the performance of sampling-based inference algorithms.

The problems of parameter symmetries provide motivation for *detecting* and *breaking* them in probabilistic programs. Symmetry detection refers to determining if a symmetry exists in the parameterization of a model, and symmetry breaking refers to dealing with symmetries in some way such that the problems they cause are no longer of concern. The paper by Nishihara et al. [NMT13] introduces algorithms for automatically detecting various types of parameter symmetries in a probabilistic programming context. We directly build on their work by formulating two mathematical perspectives of symmetry detection and breaking that have potential to be automatic. We first provide an overview of the required concepts and notations in the remainder of this section.

1.1 Definitions and notations

We borrow the notation used by Nishihara et al. We represent probabilistic models using factor graphs (θ, F) with variables $\theta = (\theta_1, ..., \theta_N)$ and factors $F = (F_1, ..., F_K)$. Let Θ denote the space of variable values. $\theta \in \Theta$ contains all parameters, latent variables, and observations (which are fixed). F represents all functions, operations, constraints, priors and likelihoods that the posterior distribution may factorize into. Figure 1 shows an example factor graph from Nishihara et al.

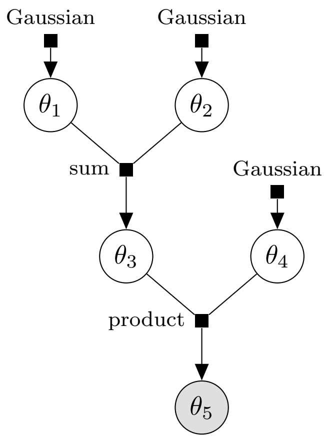


Figure 1: An example factor graph. A variable is represented by a circle and a factor is represented by a square. The shaded circle represents an observed variable.

Given data, the unnormalized posterior distribution can then be expressed as

$$\prod_k F_k(\theta)$$

where factor F_k may not necessarily depend on all of θ. In the context of symmetries, priors are not of interest as it is assumed that the parameters of the prior are chosen by the user. The remainder of this report will refer to non-prior factors when considering F_k. We now formally introduce symmetries as defined in Nishihara et al.

Definition 1. A symmetry $\sigma : \Theta \to \Theta$ is a measurable function with a measurable inverse that satisfies

$$\prod_k F_k(\theta) \propto \prod_k F_k(\sigma(\theta))$$

and where if θ_n is some observed variable, then the symmetry keeps θ_n fixed. In other words, a symmetry is a transformation on the variables that preserves the overall likelihood up to a scaling constant. One may also consider a specific subset of symmetries, referred to as *local symmetries.*

Definition 2. A local symmetry is a symmetry σ that satisfies

$$F_k(\theta) \propto F_k\left(\sigma(\theta)\right)$$

for all non-prior factors F_k. In contrast to a symmetry, a local symmetry is a transformation on the variables that preserves the likelihood at each factor up to a scaling constant. Nishihara et al. mainly focuses on detecting local symmetries as they tend to be easier to identify than non-local symmetries.

The paper by Nishihara et al. discusses several types of symmetries and how they can be detected for a given factor graph. Their work is presented in the context of a probabilistic programming language where it is assumed that all built-in factors have been annotated with constraints and/or labels that correspond to specific types of symmetries. In this report, we will assume that these annotations are intrinsic properties of the factors F_k and hence are known for a given (θ, F). The next two subsections provide an overview of two types of symmetries and the approaches that Nishihara et al. use to detect them.

1.2 Scaling symmetries

A local symmetry that scales each variable by some constant without affecting the model is known as a *scaling symmetry.*

Definition 3. A scaling symmetry is a local symmetry σ such that

$$\sigma : (\theta_1, ..., \theta_N) \longmapsto (r_1\theta_1, ..., r_N\theta_N) = (e^{d_1}\theta_1, ..., e^{d_N}\theta_N)$$

where $r_n \in \mathbb{R}_+$ and $d_n = \log r_n$. The scaling symmety only considers positive scaling by definition. A symmetry that flips the signs of variables is a *sign-flip symmetry* which we only briefly allude to in this report.

Scaling symmetries are detected by solving for the set of constants that the variables can scale by based on the aggregated factor constraints. Examples of factors and the constraints that they impose on scaling are provided in Table 1. For instance, a *sum factor* that sums inputs $a + b = c$ preserves its integrity under scaling only if both a, b, and c are scaled by the same amount. This is enforced by the constraint $d_a = d_b = d_c$ that it imposes.

Let $d = (d_1, ..., d_N)$ be the vector of scaling exponents. Consider the *constraint matrix* $\mathcal{C}$ constructed by stacking all the constraints in a given factor graph. For example, the mentioned sum factor adds the rows $d_a - d_c = 0$ and $d_b - d_c = 0$ to $\mathcal{C}$. The null space $\mathcal{N}(\mathcal{C})$ of the constraint matrix then describes the space of scaling exponents that satisfy all the constraints. The scaling symmetries in the factor graph can then be expressed as the set $\{\sigma_d : d \in \mathcal{N}(\mathcal{C})\}$.

factor	constraints
$c = a + b$	$d_a = d_b = d_c$
$c = a \times b$	$d_c = d_a + d_b$
$x \geq 0$	none

Table 1: Example factors and the constraints they impose on potential scaling symmetries.

1.3 Permutation symmetries

A *permutation symmetry* permutes the variables without changing the overall likelihood. Permutation symmetries are often present in models that have mixture components or latent features and are commonly characterized by the label-switching problem.

Definition 4. A permutation symmetry is a non-local symmetry σ that permutes the components of θ while satisfying

$$\prod_{\text{label}(k)=c} F_k(\sigma(\theta)) = \prod_{\text{label}(k)=c} F_k(\theta)$$

for all *factor labels* c. A factor label is an identifier of a factor that is shared only between factors of the same type. For example, all binary sum factors have the same label but a binary sum factor and a ternary sum factor do not.

To detect permutation symmetries, all factors must have factor labels. Their arguments (incoming edges of the factor) must also be labeled such that two arguments share the same label if and only if the factor is symmetric with respect to those two. For example, the arguments a and b in the sum factor $c = a + b$ would have the same label if the input variables to a and b are symmetric. Given the labels, Nishihara et al. then reduces the detection problem to a graph *automorphism* problem where the permutation symmetries of the factor graph are the automorphisms that preserve the factor and argument labels. In the context of graphs, an automorphism is a mapping of a graph onto itself that retains how the vertices and edges are connected. Detection of graph automorphisms can be done in quasipolynomial time [Bab16] though in practice most graphs can be tested in linear time [BES80].

Note that the permutation symmetries found through this approach is dependent on the structure of the factor graph. For example, the approach will identify the symmetry across arguments a, b and c in the ternary sum factor $a+b+c$ but will only identify the symmetry for a, b and for $a+b$, c in the layered binary sum factors $(a + b) + c$.

1.4 Other symmetries

We briefly describe two other (local) symmetries without giving formal definitions. As mentioned previously, a sign-flip symmetry flips the signs of variables. The symmetry is similar to the scaling symmetry in that each variable is multiplied by some r_n but with the restriction that $r_n = (-1)^{s_n}$ where $s_n \in \{0, 1\}$.

A *translation symmetry* shifts the variables by some amount. The translation symmetry is different from the scaling and sign-flip symmetries in that the translation can depend on other variables, i.e.,

$$\sigma(\theta) = \theta + t(\theta)$$

where $t(\theta)$ can depend on the variables not being translated. Detection of translation symmetries still involves aggregating factor constraints but with the difference that the system of equations is now nonlinear.

1.5 Symmetry breaking

We reference two examples in the literature where parameter symmetries were explicitly dealt with. Most mentions of symmetry breaking are in the context of specific models that have known symmetries. For example, the Rasch model used in political science and given by

$$\mathbb{P}(y_i = 1) = \text{logit}^{-1}\left(\gamma_{k,i}(\alpha_{j,i} - \beta_{k,i})\right)$$

has a scaling symmetry where parameter γ_k can be scaled by a constant and parameters α_j and β_k divided by the same constant. Bafumi et al. [Baf+05] proposed two solutions to resolve this aliasing: constrain α_j to have a fixed distribution (e.g. standard normal) or introduce hyperpriors for the parameters and recover the original parameters after inference through normalization. Stephens et al. [Ste00] looked at permutation symmetries in a mixture model. They proposed a relabelling algorithm based on loss functions as an alternative to imposing ordering constraints on the parameters.

These solutions, along with others in the literature, tend to be specific to the model and context. We note the techniques used in these solutions and consider ways to incorporate them in our formulations under a general model context.

1.6 Equivalence class

One formulation of symmetry detection and breaking that we will present in this report is based on the idea of *equivalence classes*. We provide a brief review of the definitions and properties of equivalence classes that will be relevant for our formulation.

Definition 5. Let S be a set. For all $a, b, c \in S$, an *equivalence relation* Ξ on S is a binary relation that satisfies the following three properties:

1. **reflexivity**: $a \; \Xi \; a$.
2. **symmetry**: if $a \; \Xi \; b$ then $b \; \Xi \; a$.
3. **transitivity**: if $a \; \Xi \; b$ and $b \; \Xi \; c$ then $a \; \Xi \; c$.

An equivalence relation Ξ partitions S into subsets called equivalence classes. If $a \; \Xi \; b$ then a and b belong to the same equivalence class. We denote the equivalence class of a as $[a]$. The set of all equivalence classes in S with respect to Ξ is called the *quotient set* of S by Ξ, denoted S/Ξ.

Definition 6. A *section* is a function $f : S/\Xi \to S$ that maps an equivalence class to one of its members. The member $f([a])$ is called the *representative* of $[a]$ with respect to f.

2 Equivalence class formulation

The first formulation of symmetry detection and breaking that we introduce is based on the concept of equivalence classes. For a given factor graph, symmetry detection can be framed as a problem of finding the set of variables that are symmetric (an equivalence class). Symmetry breaking can then be framed as choosing a single member (a representative) from the set of symmetric variables based on some desired criteria. We formalize these ideas mathematically.

Definition 7. Consider a factor graph (θ, F). Define Ξ to be the equivalence relation such that for $\theta, \theta^* \in \Theta$, we have $\theta \; \Xi \; \theta^*$ if and only if there exists a local symmetry σ such that $\sigma(\theta) = \theta^*$.

We show that Ξ is a proper equivalence relation.

Proof. Consider $\theta_1, \theta_2, \theta_3 \in \Theta$.

1. **reflexivity**: let σ be the identity map. Then $F_k(\theta_1) = F_k(\sigma(\theta_1))$ for all k.

2. **symmetry**: suppose $\theta_1 \; \Xi \; \theta_2$. Then there exists some local symmetry σ such that $\sigma(\theta_1) = \theta_2$ and $F_k(\theta_1) \propto F_k(\theta_2)$ for all non-prior F_k. By definition of symmetry, σ^{-1} exists and $\sigma^{-1}(\theta_2) = \theta_1$. Then

$$F_k(\theta_2) \propto F_k(\theta_1) = F_k(\sigma^{-1}(\theta_2))$$

   for all non-prior F_k and hence $\theta_2 \; \Xi \; \theta_1$.

3. **transitivity**: suppose $\theta_1 \; \Xi \; \theta_2$ and $\theta_2 \; \Xi \; \theta_3$. Then there exists local symmetries σ_1, σ_2 such that

$$\begin{aligned} \sigma_1(\theta_1) &= \theta_2 & \qquad F_k(\theta_1) &\propto F_k(\theta_2) \\ \sigma_2(\theta_2) &= \theta_3 & \qquad F_k(\theta_2) &\propto F_k(\theta_3) \end{aligned}$$

   for all non-prior F_k. Let $\sigma_3 = \sigma_2 \circ \sigma_1$ and so σ_3 is also measurable with a measurable inverse. Then

$$\sigma_3(\theta_1) = \sigma_2(\sigma_1(\theta_1)) = \theta_3$$

   and

$$F_k(\theta_1) \propto F_k(\theta_2) \propto F_k(\theta_3) = F_k(\sigma_3(\theta_1))$$

   for all non-prior F_k. Hence $\theta_1 \; \Xi \; \theta_3$.

□

Notice that Ξ is only well-defined in the context of a factor graph (θ, F). It is therefore mathematically more convenient to consider factor graphs equipped with a Ξ that corresponds to a specific type of symmetry. We denote this as $\mathcal{F} = (\theta, F, \Xi)$. The constraint matrix $\mathcal{C}$ of $\mathcal{F}$ described in Section 1.2 is constructed from the constraints specified by Ξ on each factor F_k. We now formally define symmetry detection and symmetry breaking in terms of functions on equivalence classes.

Definition 8. A *symmetry detector* $\Delta_{\mathcal{F}} : \Theta \to \Theta/\Xi$ is a measurable function that maps θ to $[\theta]$ with respect to Ξ.

Definition 9. A *symmetry breaker* $\phi_{\mathcal{F}} : \Theta/\Xi \to \Theta$ is a measurable section that maps $[\theta]$ to a representative $\theta^* \in [\theta]$.

Definition 10. An *automatic symmetry breaker* $\Phi_{\mathcal{F}} = \phi_{\mathcal{F}} \circ \Delta_{\mathcal{F}} : \Theta \to \Theta$ is the composition of a symmetry breaker and its corresponding symmetry detector. It maps θ to a representative θ^* of its equivalence class.

The dependence of Δ, ϕ, and Φ on the factor graph $\mathcal{F}$ is made clear by the subscript. When considering two functions of the same type, e.g. $\Phi_{\mathcal{F}_1}$ and $\Phi_{\mathcal{F}_2}$, we will view them as being defined on the same factor graph (θ, F) but with Ξ_1 and Ξ_2 corresponding to different types of symmetries.

This formulation is most useful when the main concern of having symmetries is parameter nonidentifiability. If inference for a posterior distribution was performed but it is unclear if the estimated value is "correct", it may instead be more interpretable to work with the symmetry-broken posterior distribution

$$\prod_k F_k(\Phi_{\mathcal{F}_M} \circ ... \circ \Phi_{\mathcal{F}_1}(\theta))$$

where $\Phi_{\mathcal{F}_1}, ..., \Phi_{\mathcal{F}_M}$ correspond to M different types of symmetries that are to be broken. The corresponding $\phi_{\mathcal{F}_1}, ..., \phi_{\mathcal{F}_M}$ are chosen to select representatives from the respective equivalence classes based on some desired criteria.

We now show how detection and breaking of scaling and permutation symmetries are described under the equivalence class formulation.

2.1 Scaling symmetries

We define scaling symmetry detection and breaking under the equivalence class formulation.

Definition 11. Let Σ_d be the diagonal matrix with $(e^{d_1}, ..., e^{d_N})$ on the diagonal. A scaling symmetry σ_d can be written as

$$\sigma_d(\theta) = \Sigma_d \theta$$

where the right-hand side is taken as a matrix-vector product. The matrix Σ_d is positive definite and so the inverse exists as required by the definition of symmetry. We then define the *scaling symmetry detector* to be

$$\Delta_{\mathcal{F}}(\theta) = \{\Sigma_d \theta : d \in \mathcal{N}(\mathcal{C})\} = [\theta]$$

where $\mathcal{C}$ is the constraint matrix of $\mathcal{F}$.

We show that this detector correctly maps the variables θ to their equivalence classes $[\theta]$ with respect to Ξ. That is, the detector maps two variables θ, $\theta^* \in \Theta$ to the same equivalence class if and only if there exists a scaling symmetry σ_d such that $\sigma_d(\theta) = \theta^*$.

Proof. Sufficiency: Suppose that $\theta^* = \Sigma_{d^*}\theta$ for some $d^* \in \mathcal{N}(\mathcal{C})$. Then

$$\begin{aligned}\Delta_{\mathcal{F}}(\theta^*) &= \{\Sigma_d \theta^* : d \in \mathcal{N}(\mathcal{C})\} \\ &= \{\Sigma_d \Sigma_{d^*} \theta : d \in \mathcal{N}(\mathcal{C})\} \\ &= \{\Sigma_{d+d^*} \theta : d \in \mathcal{N}(\mathcal{C})\} \\ &= \{\Sigma_d \theta : d \in \mathcal{N}(\mathcal{C})\} \\ &= \Delta_{\mathcal{F}}(\theta)\end{aligned}$$

where the above follows because Σ_d and Σ_{d^*} are diagonal and $d + d^* \in \mathcal{N}(\mathcal{C})$.

Necessity: We show this by the contrapositive. Suppose that $\theta^* \neq \Sigma_d \theta$ for all $d \in \mathcal{N}(\mathcal{C})$. Then $\theta^* \notin \Delta_{\mathcal{F}}(\theta)$ by definition of the symmetry detector. Also, by definition of equivalence class, $\theta^* \in [\theta^*] = \Delta_{\mathcal{F}}(\theta^*)$. Hence $\Delta_{\mathcal{F}}(\theta) \neq \Delta_{\mathcal{F}}(\theta^*)$ and so the detector maps θ, θ^* to different equivalence classes. □

It is difficult to provide a concrete example of a *scaling symmetry breaker* $\phi_{\mathcal{F}}$ for two reasons. The first reason is that the equivalence class is notationally described by using the input θ as a reference point. If $[\theta_1]$ and $[\theta_2]$ describe the same equivalence class, the breaker must map both to the same representative. The second reason is that the equivalence class is nonlinear in the space of matrices Σ_d (it is described by the linear space $\mathcal{N}(\mathcal{C})$ rather). This means that for a given θ, operations such as scaling may not return a member of the same equivalence class. These two reasons make it challenging to mathematically define a scaling symmetry breaker that correctly extracts the representative for a given equivalence class. An example of a possible breaker is the one that returns the minimum norm, i.e.,

$$\phi_{\mathcal{F}}([\theta]) = \underset{\theta \in [\theta]}{\arg\min} \, \|\theta\|$$

However, this mathematically convenient notation hides the question of how to computationally solve for the minimum norm (if it exists) when $[\theta]$ may be an infinite nonlinear space. Consideration will also need to be given in how to deal with multiple members having the same norm.

2.2 Permutation symmetries

We define permutation symmetry detection and breaking under the equivalence class formulation. As permutation symmetries are not local, the definition of Ξ is modified to consider the existence of a non-local symmetry between two members of an equivalence class in this context.

Definition 12. Let Π be a permutation matrix (a matrix obtained from permuting rows of the identity matrix). A permutation symmetry σ can be written as

$$\sigma(\theta) = \Pi\theta$$

where again the right-hand side is taken to be a matrix-vector product. Permutation matrices are orthogonal and so the inverse exists and is given by Π^T. We say that $\Pi \in \mathcal{F}$ if the rows that correspond to non-permutable variables have 1 on the diagonal. The *permutation symmetry detector* is then defined to be

$$\Delta_{\mathcal{F}}(\theta) = \{\Pi\theta : \Pi \in \mathcal{F}\} = [\theta]$$

Under this definition, it is assumed that there are no restrictions on the permutations of the permutable variables. If variables are permutable with only certain others, multiple permutation symmetry detectors are needed where each detects the symmetries among a subset of permutable variables.

Notice that if $\Pi, \Pi^* \in \mathcal{F}$, then the matrix product $\Pi\Pi^*$ is itself a permutation matrix by properties of permutation matrices. Furthermore, $\Pi\Pi^* \in \mathcal{F}$ as both matrices have 1 on the diagonal in non-permutable rows and thus so does the product. We use this fact to again show that $\Delta_{\mathcal{F}}(\theta) = \Delta_{\mathcal{F}}(\theta^*)$ if and only if there exists a permutation symmetry σ such that $\sigma(\theta) = \theta^*$.

Proof. Sufficiency: Suppose that $\theta^* = \Pi^*\theta$ for some $\Pi^* \in \mathcal{F}$. Then

$$\begin{aligned} \Delta_{\mathcal{F}}(\theta^*) &= \{\Pi\theta^* : \Pi \in \mathcal{F}\} \\ &= \{\Pi\Pi^*\theta : \Pi \in \mathcal{F}\} \\ &= \{\Pi\theta : \Pi \in \mathcal{F}\} \\ &= \Delta_{\mathcal{F}}(\theta) \end{aligned}$$

where the above follows from the aforementioned fact.

Necessity: The proof for the converse follows the same argument as for the scaling symmetry. Hence this permutation symmetry detector correctly maps the variables to their equivalent classes. □

An example of a *permutation symmetry breaker* is easy to describe. For simplicity, suppose that the permutable variables in θ have distinct values. Then an example permutation breaker $\phi_{\mathcal{F}}$ is one such that $\phi_{\mathcal{F}}([\theta]) = \theta^*$ where if θ_i and θ_j are permutable, then $\theta_i^* < \theta_j^*$. In the context of probabilistic programming, this reduces to a sorting problem involving certain entries of θ.

One distinction between permutation equivalence classes and scaling equivalence classes is that the permutation equivalence classes are always finite. A class can have at most $N!$ members corresponding to the possible permutations of the variables. This makes the permutation symmetry breaker computationally easier to implement compared to the scaling symmetry breaker in that in the worst case, the breaker can just iterate over the equivalence class to pick a representative.

2.3 Other symmetries

We end our discussion of the equivalence class formulation with a brief comment about the sign-flip and translation symmetries. The formulation for the sign-flip symmetry would likely closely resemble that of the scaling symmetry where the symmetries are given by the null space of the constraint matrix. An added benefit is that the space of symmetries is also finite due to having only two possible values for each exponent. On the other hand, the translation symmetry appears to be difficult to describe under the equivalence class formulation. The dependence on θ in the symmetry itself makes it challenging to define detection as a proper function.

3 Reparameterization formulation

The equivalence class formulation is useful as a post-processing step after inference when parameter non-identifiability is the primary concern. However, parameter symmetries may also impact the performance of the inference algorithm in which case symmetry breaking should be done before inference. The equivalence class formulation is less relevant in this situation for two reasons. The first is that without having done inference, the equivalence classes are symbolic and generally difficult to work with. The second reason is that the symmetry breaker returns a representative of the equivalence class but does not actually remove the symmetry from the model. These reasons make the equivalence class formulation unhelpful for describing pre-inference symmetry breaking.

We introduce a second formulation of symmetry breaking based on reparameterizing the model which can be done before performing inference. The formulation directly builds on the symmetry detection approaches presented by Nishihara et al. by changing the factor graph in such a way that the approaches detect no symmetries. We preface this section by noting that the ideas presented here are still relatively preliminary. In particular, development of a formal framework and further consideration of how this approach could be automated in a probabilistic program would be necessary.

3.1 Scaling symmetries

The approach for detecting scaling symmetries involves constructing the constraint matrix of a factor graph and finding its null space. In other words, the model has a scaling symmetry if the constraint matrix is underdetermined. Our approach for symmetry breaking under the reparameterization formulation is to modify the factor graph so that the constraint matrix becomes a determined system. We discuss this approach from two perspectives.

The first perspective looks at breaking the symmetry by reducing the number of variables in the factor graph. This corresponds to deleting columns of the constraint matrix and can be done by either removing or merging factors depending on the structure of the factor graph. For example, the ternary sum of common distributions from a single distribution family can usually be specified by another well-known distribution. If a scaling symmetry is introduced through a sum factor that takes in multiple Gaussian inputs, the sum and the Gaussian inputs may be replaced by a single Gaussian factor representing the sum. This breaks the scaling symmetry originating from the sum factor in the original factor graph. Figure 2 demonstrates what this process looks like graphically.

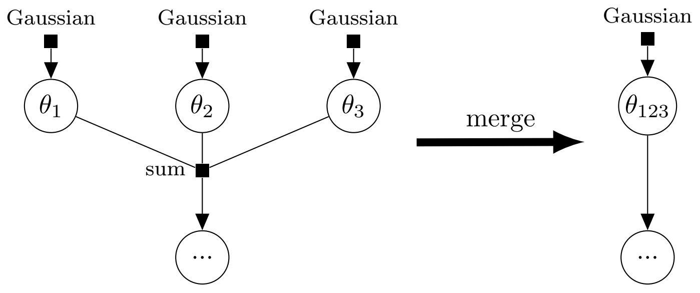


Figure 2: An example of merging factors to break a scaling symmetry.

The second perspective looks at breaking the symmetry by imposing additional constraints, which corresponds to adding new rows to the constraint matrix. The most obvious constraint that can be added is to make a latent variable observed. This fixes the variable so that it cannot be scaled, and is done in the constraint matrix by adding the row $d_n = 0$. The graphical representation of this is shown in Figure 3. Other constraints may be introduced by either adding new factors or imposing additional constraints on existing ones.

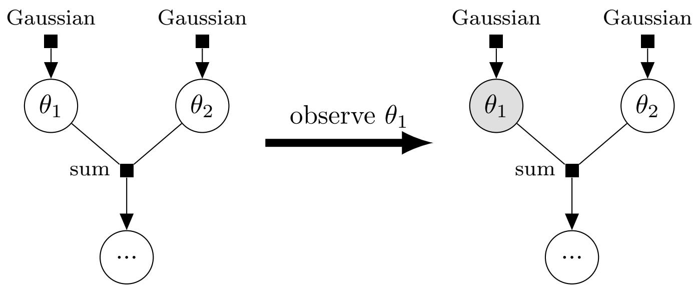


Figure 3: An example of imposing additional constraints to break a scaling symmetry.

3.2 Permutation symmetries

Nishihara et al. reduces detection of permutation symmetries to detection of automorphisms of a labeled graph. An automorphism may exist due to the symmetric arguments of a factor being annotated with the same label. The permutation symmetry can be broken by imposing additional constraints such that the arguments no longer receive the same label. One constraint that achieves this is the ordering constraint. For example, the sum factor $c = a + b$ is no longer symmetric with respect to a and b under the constraint $a < b$. This is equivalent to reparameterizing the factor graph by changing the inputs of the factor to the order statistics. Under this new parameterization, the arguments receive different labels and so the factor is no longer a source of possible automorphism. Figure 4 shows what this reparameterization may look like graphically.

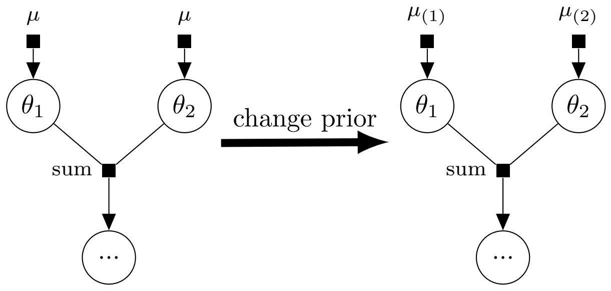


Figure 4: An example of changing the prior distribution μ to the distributions $\mu_{(1)}$ and $\mu_{(2)}$ of the order statistics to break a permutation symmetry.

3.3 Remarks regarding reparameterization

The reparameterization formulation of symmetry breaking covers a broad class of techniques and closely resembles the symmetry breaking done in the literature. However, the flexibility that the formulation allows makes it difficult to automate for a general model in a probabilistic program. A formal framework that reasons about the types of reparameterizations allowed would be needed to make progress towards automation. In addition, changing the factor graph is also likely to change the posterior distribution or at least its assumptions. For example, there may be a practical reason to include multiple Gaussian inputs instead of a single combined Gaussian input. Fixing a variable for mathematical convenience is also something to question. An ideal solution to this problem would be to reparameterize the model in a way such that the original model could be recovered after inference.

4 Open questions and research directions

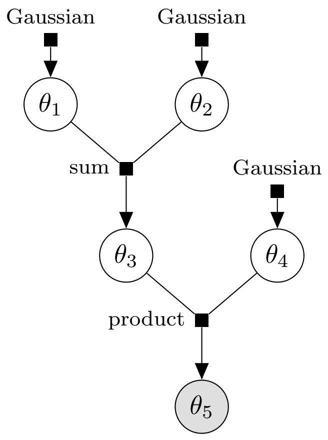
Parameter symmetries in models lead to problems of parameter nonidentifiability and reduced inference performance. Although these symmetries can often be dealt with manually as seen in the literature [Baf+05; Ste00], it is not as easy to handle them automatically in a probabilistic programming context. The paper by Nishihara et al. [NMT13] presents multiple algorithms for automatically detecting certain types of symmetries. Based on their work, we introduced two formulations of symmetry detection and breaking that have potential for automation. We conclude this report with a summary of both formulations and their possible research directions.

The equivalence class formulation frames the problem of detection as finding the set of symmetric variables, and breaking as choosing a member from the set based on some specified criteria. The formulation is most useful for describing a post-processing step after inference as an intuitive solution to parameter nonidentifiability. Describing both detection and breaking as mathematical functions also naturally leads to direct translations of functions in a probabilistic programming language. The main challenges in the equivalence class formulation include specifying valid symmetry breakers and implementing them to handle general classes that are nonlinear spaces. There has been some work in formalizing functions defined on equivalence classes [Pau06] that may be relevant to our objective. Another challenge is the question of how the translation symmetries described by Nishihara et al. could be formulated as functions given their dependence on the input variables. Drawing on results from other areas of mathematics such as group theory may be necessary to solve these problems.

The reparameterization formulation directly works off the algorithms by Nishihara et al. and aims to break symmetries by changing the factor graph. The formulation allows for symmetries to be broken pre-inference and is useful when the performance of the inference algorithm is expected to be affected by the presence of symmetries. However, it is highly dependent on the structure and context of the factor graph in its current state which makes it difficult to implement for a general model. Further development of the formulation is needed to determine how feasible it is to automate it. A research direction worth exploring is to understand the class of factor graph transformations that do not change or at least allow for the original model to be recovered after inference. Such transformations would mitigate the drawback of implicitly changing the model when the factor graph is changed. The paper by Gelman [Gel04] discusses some reparameterization techniques for improving computability of Bayesian models. We expect that some of these techniques can be repurposed to break parameter symmetries in a model.

A Exercises

1. This exercise is a worked extended example from Nishihara et al. [NMT13]. The factor graph from Figure 1 is shown again here.



Some factors and the scaling constraints that they impose is provided in the following table.

factor	constraints
$c = a + b$	$d_a = d_b = d_c$
$c = a \times b$	$d_c = d_a + d_b$
c observed	$d_c = 0$

(a) Find the scaling constraint matrix $\mathcal{C}$ for this factor graph.

Solution. The sum factor adds the rows $d_1 - d_3 = 0$ and $d_2 - d_3 = 0$.
The product factor adds the row $d_3 + d_4 - d_5 = 0$.
The observed factor adds the row $d_5 = 0$.
Aggregating the rows produces the following constraint matrix.

$$\mathcal{C} = \begin{bmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(b) Find the null space of $\mathcal{C}$.

Solution. A basis for the null space of $\mathcal{C}$ is given by the solution to the following system of equations.

$$\begin{bmatrix} 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Solving for the solution leads to the null space

$$\mathcal{N}(\mathcal{C}) = \left\{ c \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ 0 \end{bmatrix} : c \in \mathbb{R} \right\}$$

(c) Under the equivalence class formulation, give the definition of the scaling symmetry detector for this factor graph $\mathcal{F}$.

Solution. The scaling symmetry detector is defined as

$$\Delta_{\mathcal{F}}(\theta) = \{\Sigma_d \theta : d \in \mathcal{N}(\mathcal{C})\}$$

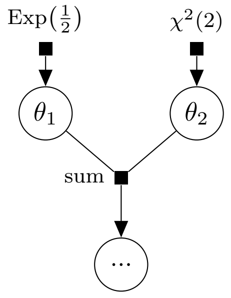
where Σ_d is the diagonal matrix with $(e^{d_1}, ..., e^{d_5})$ on the diagonal and $\mathcal{N}(\mathcal{C})$ is as found in the previous question.

(d) Given $\theta^* = (\theta_1^*, ..., \theta_5^*)^T$, find $[\theta^*]$.

Solution. The equivalence class of θ^* can be found using the symmetry detector defined in the previous question.

$$[\theta^*] = \Delta_{\mathcal{F}}(\theta^*) = \{\Sigma_d \theta^* : d \in \mathcal{N}(\mathcal{C})\} = \left\{ \begin{bmatrix} e^c \theta_1^* \\ e^c \theta_2^* \\ e^c \theta_3^* \\ e^{-c} \theta_4^* \\ \theta_5^* \end{bmatrix} : c \in \mathbb{R} \right\}$$

2. Consider the following factor graph that has an exponential prior with rate $\beta = \frac{1}{2}$ and a chi-squared prior with degrees of freedom $d = 2$:



This factor graph has a permutation symmetry. Explain why and suggest one way to break the symmetry under the reparameterization formulation.

Let λ denote the Lebesgue measure. You may find the following useful:

$$\mathbb{P}(\theta_1 \in dx) = \lambda(dx)\beta e^{-\beta x} \qquad x \in \mathbb{R}_+$$

$$\mathbb{P}(\theta_2 \in dx) = \lambda(dx)\frac{1}{2^{\frac{d}{2}}\Gamma\left(\frac{d}{2}\right)}x^{\frac{d}{2}-1}e^{-\frac{x}{2}} \qquad x \in \mathbb{R}_+$$

Solution. Notice that

$$\mathbb{P}(\theta_1 \in dx) = \lambda(dx)\frac{1}{2}e^{-\frac{1}{2}x} \qquad x \in \mathbb{R}_+$$

and

$$\mathbb{P}(\theta_2 \in dx) = \lambda(dx)\frac{1}{2^{\frac{2}{2}}\Gamma\left(\frac{2}{2}\right)}x^{\frac{2}{2}-1}e^{-\frac{x}{2}} = \lambda(dx)\frac{1}{2}e^{-\frac{x}{2}} \qquad x \in \mathbb{R}_+$$

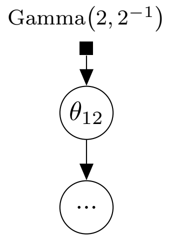
and so θ_1 and θ_2 have the same distribution. Hence the permutation symmetry exists because the inputs to the sum factor are symmetric. In this case, the symmetry can be broken by merging the sum factor with its inputs as the sum is specified by a gamma distribution with shape $\alpha = 2$ and rate $\beta = \frac{1}{2}$. We show this using the Laplace transform. For $r \in \mathbb{R}_+$, we have

$$\mathbb{E}[e^{-r\theta_1}] = \mathbb{E}[e^{-r\theta_2}] = \frac{2^{-1}}{2^{-1} + r}$$

Then by independence of θ_1 and θ_2,

$$\mathbb{E}[e^{-r(\theta_1+\theta_2)}] = \mathbb{E}[e^{-r\theta_1}]\mathbb{E}[e^{-r\theta_2}] = \left(\frac{2^{-1}}{2^{-1} + r}\right)^2$$

which corresponds to the Laplace transform of the mentioned gamma distribution. The new factor graph obtained from merging is



References

[Bab16] L. Babai. "Graph Isomorphism in Quasipolynomial Time". In: *CoRR* abs/1512.03547 (2016). arXiv: 1512.03547. URL: http://arxiv.org/abs/1512.03547.

[BES80] L. Babai, P. Erdős, and S. M. Selkow. "Random Graph Isomorphism". In: *SIAM Journal on Computing* 9.3 (1980), pp. 628–635. DOI: 10.1137/0209047. eprint: https://doi.org/10.1137/0209047. URL: https://doi.org/10.1137/0209047.

[Baf+05] J. Bafumi et al. "Practical Issues in Implementing and Understanding Bayesian Ideal Point Estimation". In: *Political Analysis* 13.2 (2005), pp. 171–187. DOI: 10.1093/pan/mpi010.

[Gel04] A. Gelman. "Parameterization and Bayesian Modeling". In: *Journal of the American Statistical Association* 99.466 (2004), pp. 537–545. DOI: 10.1198/016214504000000458. eprint: https://doi.org/10.1198/016214504000000458. URL: https://doi.org/10.1198/016214504000000458.

[NMT13] R. Nishihara, T. Minka, and D. Tarlow. *Detecting Parameter Symmetries in Probabilistic Models.* 2013. arXiv: 1312.5386 [stat.ML].

[Pau06] L. C. Paulson. "Defining Functions on Equivalence Classes". In: *ACM Trans. Comput. Logic* 7.4 (2006), pp. 658–675. ISSN: 1529-3785. DOI: 10.1145/1183278.1183280. URL: http://doi.acm.org/10.1145/1183278.1183280.

[Rai17] T. Rainforth. "Automating Inference, Learning, and Design using Probabilistic Programming". PhD thesis. University of Oxford, 2017. URL: http://www.robots.ox.ac.uk/~twgr/assets/pdf/rainforth2017thesis.pdf.

[Ste00] M. Stephens. "Dealing With Label-Switching in Mixture Models". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 62 (July 2000). DOI: 10.1111/1467-9868.00265.