ISR: Invertible Symbolic Regression

Anonymous authors
Paper under double-blind review

Abstract

We introduce an Invertible Symbolic Regression (ISR) method. It is a machine learning technique that generates analytical relationships between inputs and outputs of a given dataset via invertible maps (or architectures). The proposed ISR method naturally combines the principles of Invertible Neural Networks (INNs) and Equation Learner (EQL), a neural network-based symbolic architecture for function learning. In particular, we transform the affine coupling blocks of INNs into a symbolic framework, resulting in an end-to-end differentiable symbolic invertible architecture that allows for efficient gradient-based learning. The proposed ISR framework also relies on sparsity promoting regularization, allowing the discovery of concise and interpretable invertible expressions. We show that ISR can serve as a (symbolic) normalizing flow for density estimation tasks. Furthermore, we highlight its practical applicability in solving inverse problems, including a benchmark inverse kinematics problem, and notably, a geoacoustic inversion problem in oceanography aimed at inferring posterior distributions of underlying seabed parameters from acoustic signals.

1 Introduction

In many applications in engineering and science, experts have developed theories about how measurable quantities result from system parameters, known as forward modeling. In contrast, the *Inverse Problem* aims to find unknown parameters of a system that lead to desirable observable quantities. A typical challenge is that numerous configurations of these parameters yield the same observable quantity, especially with underlying complicated nonlinear governing equations and where hidden parameters outnumber the observable variables. To tackle challenging and ill-posed inverse problems, a common method involves estimating a posterior distribution on the unknown parameters, given the observations. Such a probabilistic approach facilitates the uncertainty quantification by analyzing the diversity of potential inverse solutions.

An established computationally expensive approach in finding the posterior distribution is to directly generate samples using acceptance/rejection. In this scope, the Markov Chain Monte Carlo (MCMC) methods (Brooks et al., 2011; Andrieu et al., 2003; Doucet & Wang, 2005; Murphy, 2012; Goodfellow et al., 2016; Korattikara et al., 2014; Atchadé & Rosenthal, 2005; Kungurtsev et al., 2023) offer a strong alternative for achieving near-optimum Bayesian estimation (Constantine et al., 2016; Conrad et al., 2016; Dosso & Dettmer, 2011). However, MCMC methods can be inefficient MacKay (2003) as the number of unknown parameter increases.

When the likelihood is unknown or intractable, the Approximate Bayesian Computation (ABC) Csilléry et al. (2010) is often used to estimate the posterior distribution. However, similar to MCMC, this method also suffers from poor scalability Cranmer et al. (2020a); Papamakarios et al. (2019). A more efficient alternative is to approximate the posterior using a tractable distribution, i.e. the variational method Blei et al. (2017); Salimans et al. (2015); Wu et al. (2018). However, the performance of the variational method deteriorates as the true posterior becomes more complicated.

Since direct sampling of the posterior distribution requires many runs of the forward map, often a trained and efficient surrogate model is used instead of the exact model. Surrogate models are considered as an efficient representation of the forward map trained on the data. Popular approaches include recently introduced Physics-Informed Neural Networks (Raissi et al., 2019). However, due to the black-box nature of neural networks, it is beneficial to express the forward map symbolically instead for several reasons. First, Symbolic

Regression (SR) can provide model interpretability, while understanding the inner workings of deep Neural Networks is challenging (Kim et al., 2020; Gilpin et al., 2018). Second, studying the symbolic outcome can lead to valuable insights and provide nontrivial relations and/or physical laws (Udrescu & Tegmark, 2020; Udrescu et al., 2020; Liu & Tegmark, 2021; Keren et al., 2023). Third, they may achieve better results than Neural Networks in out-of-distribution generalization Cranmer et al. (2020b). Fourth, unlike conventional regression methods, such as least squares (Wild & Seber, 1989), likelihood-based (Edwards, 1984; Pawitan, 2001), and Bayesian regression techniques (Lee, 1997; Leonard & Hsu, 2001; Vanslette et al., 2020; Tohme et al., 2020), SR does not rely on a fixed parametric model structure.

The attractive properties of SR, such as interpretability, often come at high computational cost compared to standard Neural Networks. This is because SR optimizes for model structure and parameters simultaneously. Therefore, SR is thought to be NP-hard (Petersen et al., 2021; Virgolin & Pissis, 2022). However tractable solutions exist, that can approximate the optimal solution suitable for applications. For instance, genetic algorithms (Koza & Koza, 1992; Schmidt & Lipson, 2009; Tohme et al., 2023; Orzechowski et al., 2018; La Cava et al., 2021) and machine learning algorithms, such as neural networks, and transformers (Sahoo et al., 2018; Jin et al., 2019; Udrescu et al., 2020; Cranmer et al., 2020b; Kommenda et al., 2020; Burlacu et al., 2020; Biggio et al., 2021; Mundhenk et al., 2021; Petersen et al., 2021; Valipour et al., 2021; Zhang et al., 2022; Kamienny et al., 2022), are used to solve SR efficiently.

Related Works. In recent years, a branch of Machine Learning methods has emerged and is dedicated to finding data-driven invertible maps. While they are ideal for data generation and inverse problem, they lack interpretablity. On the other hand, several methods have been developed to achieve interpretablity in representing the forward map via Symbolic Regression. Hence, it is natural to incorporate SR in the invertible map for the inverse problem. Next, we review the related works in the scope of this work.

Normalizing Flows: The idea of this class of methods is to train an invertible map such that in the forward problem, the input samples are mapped to a known distribution function, e.g. the normal distribution function. Then, the unknown distribution function is found by inverting the trained map with the normal distribution as the input. This procedure is called the normalizing flow technique (Rezende & Mohamed, 2015; Dinh et al., 2016; Kingma & Dhariwal, 2018; Durkan et al., 2019; Tzen & Raginsky, 2019; Kobyzev et al., 2020; Wang & Marzouk, 2022). This method has been used for re-sampling unknown distributions, e.g. Boltzmann generators (Noé et al., 2019), as well as density recovery such as AI-Feynmann (Udrescu & Tegmark, 2020; Udrescu et al., 2020).

Invertible Neural Networks (INNs): This method can be categorized in the class of normalizing flows. The invertibility of INN is rooted in their architecture. The most popular design is constructed by concatenating affine coupling blocks Kingma & Dhariwal (2018); Dinh et al. (2016; 2014), which limits the architecture of the neural network. INNs have been shown to be effective in estimating the posterior of the probabilistic inverse problems while outperforming MCMC, ABC, and variational methods. Applications include epidemiology Radev et al. (2021), astrophysics Ardizzone et al. (2019a), medicine Ardizzone et al. (2019a), optics Luce et al. (2023), geophysics Zhang & Curtis (2021); Wu et al. (2023), and reservoir engineering Padmanabha & Zabaras (2021). However, similar to standard neural networks, INNs lead to a model that cannot be evaluated with interpretable mathematical formula.

Equation Learner (EQL): Among SR methods, the EQL network is one of the attractive methods since it incorporates gradient descent in the symbolic regression task for better efficiency (Martius & Lampert, 2016; Sahoo et al., 2018; Kim et al., 2020). EQL devises a neural network-based architecture for SR task by replacing commonly used activation functions with a dictionary of operators and use back-propagation for training. However, in order to obtain a symbolic estimate for the inverse problem efficiently, it is necessary to merge such efficient SR method with an invertible architecture, which is the goal of this paper.

Our Contributions. We present Invertible Symbolic Regression (ISR), a machine learning technique that identifies mathematical relationships that best describe the forward and inverse map of a given dataset through the use of invertible maps. ISR is based on an invertible symbolic architecture that bridges the concepts of Invertible Neural Networks (INNs) and Equation Learner (EQL), i.e. a neural network-based symbolic architecture for function learning. In particular, we transform the affine coupling blocks of INNs

into a symbolic framework, resulting in an end-to-end differentiable symbolic inverse architecture. This allows for efficient gradient-based learning. The symbolic invertible architecture is easily invertible with a tractable Jacobian, which enables explicit computation of posterior probabilities. The proposed ISR method, equipped with sparsity promoting regularization, captures complex functional relationships with concise and interpretable invertible expressions. In addition, as a byproduct, we naturally extend ISR into a conditional ISR (cISR) architecture by integrating the EQL network within conditional INN (cINN) architectures present in the literature. We further demonstrate that ISR can also serve as a symbolic normalizing flow (for density estimation) in a number of test distributions. We demonstrate the applicability of ISR in solving inverse problems, and compare it with INN on a benchmark inverse kinematics problem, as well as a geoacoustic inversion problem in oceanography (see (Chapman & Shang, 2021) for further information). Here, we aim to characterize the undersea environment, such as water-sediment depth, sound speed, etc., from acoustic signals. To the best of our knowledge, this work is the first attempt towards finding interpretable solutions to general nonlinear inverse problems by establishing analytical relationships between measurable quantities and unknown variables via symbolic invertible maps.

The remainder of the paper is organized as follows. In Section 2, we go through an overall background about Symbolic Regression (SR) and review the Equation Learner (EQL) network architecture. In Section 3, we introduce and present the proposed Invertible Symbolic Regression (ISR) method. We then show our results in Section 4, where we demonstrate the versatility of ISR as a density estimation method on a variety of examples (distributions), and then show its applicability in inverse problems on an inverse kinematics benchmark problem and through a case study in ocean geoacoustic inversion. Finally in Section 5, we provide our conclusions and outlook.

2 Background

Before diving into the proposed ISR method, we first delve into a comprehensive background of the Symbolic Regression (SR) task, as well as the Equation Learner (EQL) network architecture.

Symbolic Regression. Given a dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$ consisting of N independent and identically distributed (i.i.d.) paired examples, where $\mathbf{x}_i \in \mathbb{R}^{d_\mathbf{x}}$ represents the input variables and $\mathbf{y}_i \in \mathbb{R}^{d_\mathbf{y}}$ the corresponding output for the i-th observation, the objective of SR is to find an analytical (symbolic) expression f that best maps inputs to outputs, i.e. $\mathbf{y}_i \approx f(\mathbf{x}_i)$. SR seeks to identify the functional form of f from the space of functions $\mathcal{S}$ defined by a set of given arithmetic operations (e.g. $+$, $-$, $\times$, $\div$) and mathematical functions (e.g. sin, cos, exp, etc.) that minimizes a predefined loss function $\mathcal{L}(f, \mathcal{D})$, which measures the discrepancy between the true outputs $\mathbf{y}_i$ and the predictions $f(\mathbf{x}_i)$ over all observations in the dataset. Unlike conventional regression methods that fit parameters within a predefined model structure, SR dynamically constructs the model structure itself, offering a powerful means to uncover underlying physical laws and/or nontrivial relationships.

Equation Learner Network. The Equation Learner (EQL) network is a multi-layer feed-forward neural network that is capable of performing symbolic regression by substituting traditional nonlinear activation functions with elementary functions. The EQL network was initially introduced by Martius & Lampert (2016) and Sahoo et al. (2018), and further explored by Kim et al. (2020). As shown in Figure 1, the EQL network architecture is based on a fully connected neural network where the ouput $\mathbf{h}^{(i)}$ of the i-th layer is given by

$$\mathbf{g}^{(i)} = \mathbf{W}^{(i)}\mathbf{h}^{(i-1)} \tag{1}$$

$$\mathbf{h}^{(i)} = f\big(\mathbf{g}^{(i)}\big) \tag{2}$$

where $\mathbf{W}^{(i)}$ is the weight matrix of the i-th layer, f denotes the nonlinear (symbolic) activation functions, and $\mathbf{h}^{(0)} = \mathbf{x}$ represents the input data. In regression tasks, the final layer does not typically have an activation function, so the output for a network with L hidden layers is given by

$$\mathbf{y} = \mathbf{h}^{(L+1)} = \mathbf{g}^{(L+1)} = \mathbf{W}^{(L+1)}\mathbf{h}^{(L)}. \tag{3}$$

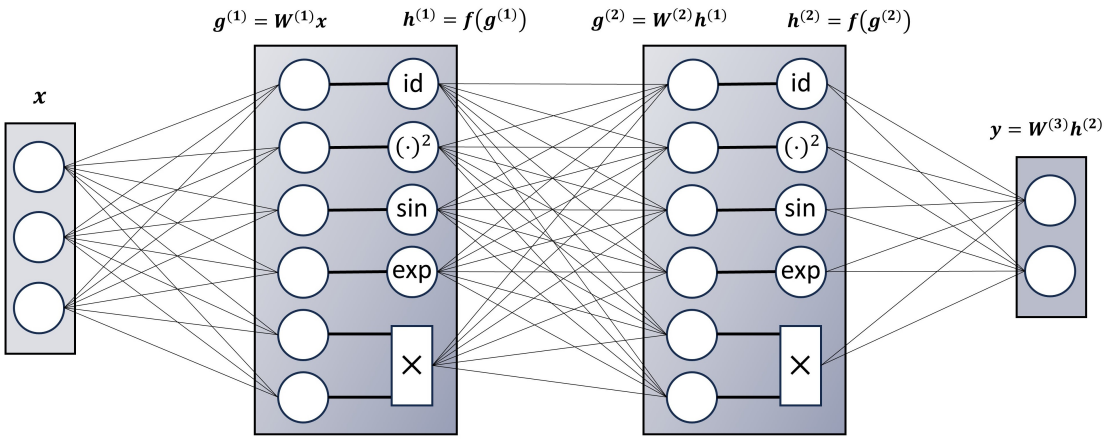

Figure 1: EQL network architecture for symbolic regression. For visual simplicity, we only show 2 hidden layers and 5 activation functions per layer (identity or "id", square, sine, exponential, and multiplication).

In traditional neural networks, activation functions such as ReLU, tanh, or sigmoid are typically employed. However, for the EQL network, the activation function $f(\mathbf{g})$ may consist of a separate primitive function for each component of $\mathbf{g}$ (e.g. the square function, sine, exponential, etc.), and may include functions that take multiple arguments (e.g. the multiplication function). In addition, the primitive functions may be duplicated within each layer (to reduce the training's sensitivity to random initializations).

It is worth mentioning that, for visual simplicity, the schematic shown in Figure 1, shows an EQL network with only two hidden layers, where each layer has only five primitive functions, i.e., the activation function

$$f(\mathbf{g}) \in \big\{\text{identity, square, sine, exponential, multiplication}\big\}.$$

However, the EQL network can in fact include other functions or more hidden layers to fit a broader range (or class) of functions. Indeed, the number of hidden layers can dictate the complexity of the resulting symbolic expression and plays a similar role to the maximum depth of expression trees in genetic programming techniques. Although the EQL network may not offer the same level of generality as traditional symbolic regression methods, it is adequately capable of representing the majority of functions commonly encountered in scientific and engineering contexts. Crucially, the parametrized nature of the EQL network enables efficient optimization via gradient descent (and backpropagation).

After training the EQL network, the identified equation can be directly derived from the network weights. To avoid reaching overly complex symbolic expressions and to maintain interpretability, it is essential to guide the network towards learning the simplest expression that accurately represents the data. In methods based on genetic programming, this simplification is commonly achieved by restricting the number of terms in the expression. For the EQL network, this is attained by applying sparsity regularization to the network weights, which sets as many of these weights to zero as possible (e.g. L_1 regularization (Tibshirani, 1996), $L_{0.5}$ regularization (Xu et al., 2010)). In this work, we use a smoothed $L_{0.5}$ regularization (Wu et al., 2014; Fan et al., 2014), which was also adopted by Kim et al. (2020).

3 Invertible Symbolic Regression

In this section, we delineate the problem setup for inverse problems, and then describe the proposed ISR approach.

3.1 Problem Specification

In various engineering and natural systems, the theories developed by experts describe how measurable (or observable) quantities $\mathbf{y} \in \mathbb{R}^{d_\mathbf{y}}$ result from the unknown (or hidden) properties $\mathbf{x} \in \mathbb{R}^{d_\mathbf{x}}$, known as the

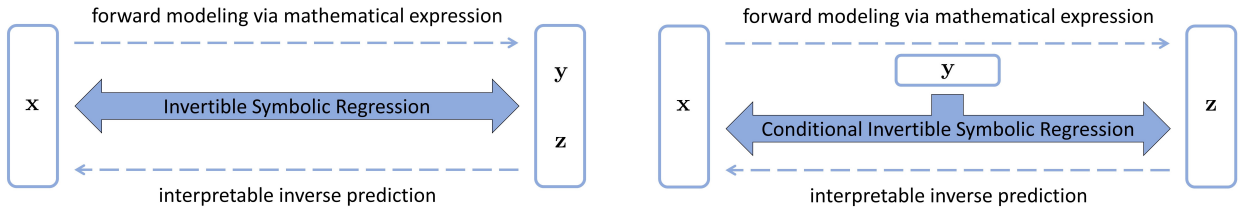

Figure 2: (Left) The proposed ISR framework learns a bijective symbolic transformation that maps the (unknown) variables $\mathbf{x}$ to the (observed) quantities $\mathbf{y}$ while transforming the lost information into latent variables $\mathbf{z}$. (Right) The conditional ISR (cISR) framework learns a bijective symbolic map that transforms $\mathbf{x}$ directly to a latent representation $\mathbf{z}$ given the observation $\mathbf{y}$. As we will show, both the forward and inverse mappings are efficiently computable and possess a tractable Jacobian, allowing explicit computation of posterior probabilities.

forward process $\mathbf{x} \to \mathbf{y}$. The goal of inverse prediction is to predict the unknown variables $\mathbf{x}$ from the observable variables $\mathbf{y}$, through the *inverse process* $\mathbf{y} \to \mathbf{x}$. As critical information is lost during the forward process (i.e. $d_{\mathbf{x}} \geq d_{\mathbf{y}}$), the inversion is usually intractable. Given that $f^{-1}(\mathbf{y})$ does not yield a uniquely defined solution, an effective inverse model should instead estimate the *posterior* probability distribution $p(\mathbf{x} \,|\, \mathbf{y})$ of the hidden variables $\mathbf{x}$, conditioned on the observed variable $\mathbf{y}$.

Invertible Symbolic Regression (ISR). Assume we are given a training dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, collected using forward model $\mathbf{y} = s(\mathbf{x})$ and prior $p(\mathbf{x})$. To counteract the loss of information during the forward process, we introduce latent random variables $\mathbf{z} \in \mathbb{R}^{d_{\mathbf{z}}}$ drawn from a multivariate standard normal distribution, i.e. $\mathbf{z} \sim p_{\mathbf{Z}}(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{d_{\mathbf{z}}})$, where $d_{\mathbf{z}} = d_{\mathbf{x}} - d_{\mathbf{y}}$. These latent variables are designed to capture the information related to $\mathbf{x}$ that is not contained in $\mathbf{y}$ (Ardizzone et al., 2019a). In ISR, we aim to learn a bijective symbolic function $f : \mathbb{R}^{d_{\mathbf{x}}} \to \mathbb{R}^{d_{\mathbf{y}}} \times \mathbb{R}^{d_{\mathbf{z}}}$ from the space of functions defined by a set of mathematical functions (e.g. sin, cos, exp, log) and arithmetic operations (e.g. $+$, $-$, $\times$, $\div$), and such that

$$[\mathbf{y}, \mathbf{z}] = f(\mathbf{x}) = \big[f_{\mathbf{y}}(\mathbf{x}), f_{\mathbf{z}}(\mathbf{x})\big], \qquad \mathbf{x} = f^{-1}(\mathbf{y}, \mathbf{z}) \tag{4}$$

where $f_{\mathbf{y}}(\mathbf{x}) \approx s(\mathbf{x})$ is an approximation of the forward process $s(x)$. As discussed later, we will learn f (and hence f^{-1}) through an invertible symbolic architecture with bi-directional training. The solution of the inverse problem (i.e. the posterior $p(\mathbf{x} \,|\, \mathbf{y}^*)$) can then be found by calling f^{-1} for a fixed observation $\mathbf{y}^*$ while randomly (and repeatedly) sampling the latent variable $\mathbf{z}$ from the same standard Gaussian distribution.

Conditional Invertible Symbolic Regression (cISR). Inspired by works on conditional invertible neural networks (cINNs) (Ardizzone et al., 2019b; 2021; Kruse et al., 2021; Luce et al., 2023), instead of training ISR to predict $\mathbf{y}$ from $\mathbf{x}$ while transforming the lost information into latent variables $\mathbf{z}$, we train them to transform $\mathbf{x}$ directly to latent variables $\mathbf{z}$ given the observed variables $\mathbf{y}$. This is achieved by incorporating $\mathbf{y}$ as an additional input within the bijective symbolic architecture during both the forward and inverse passes (see Figure 2). cISR works with larger latent spaces than ISR since $d_{\mathbf{z}} = d_{\mathbf{x}}$ regardless of the dimension $d_{\mathbf{y}}$ of the observed quantities $\mathbf{y}$. Further details are provided in the following section.

In addition to approximating the forward model via mathematical relations, ISR also identifies an interpretable inverse map via analytical expressions (see Figure 2). Such interpretable mappings are of particular interest in physical sciences, where an ambitious objective involves creating intelligent machines capable of generating novel scientific findings(Udrescu & Tegmark, 2020; Udrescu et al., 2020; Liu & Tegmark, 2021; Keren et al., 2023; Liu et al., 2024). As described next, the ISR architecture is both easily invertible and has a tractable Jacobian, allowing for explicit computation of posterior probabilities.

3.2 Invertible Symbolic Architecture

Inspired by the architectures proposed by Dinh et al. (2016); Kingma & Dhariwal (2018); Ardizzone et al. (2019a), we adopt a fully invertible architecture mainly defined by a sequence of n reversible blocks where

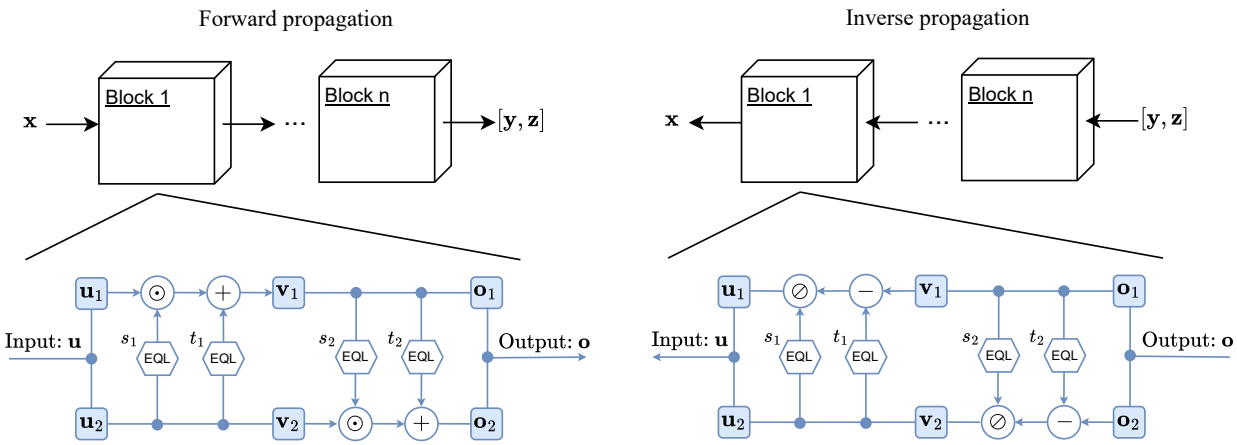

Figure 3: The proposed ISR method integrates EQL within the affine coupling blocks of the INN invertible architecture.[1] This results in a bijective symbolic transformation that is both easily invertible and has a tractable Jacobian. Indeed, the forward and inverse directions both possess identical computational cost. Here, $\odot$ and $\oslash$ denote element-wise multiplication and divison, respectively.

each block consists of two complementary affine coupling layers. In particular, we first split the block's input $\mathbf{u} \in \mathbb{R}^{d_\mathbf{u}}$ into $\mathbf{u}_1 \in \mathbb{R}^{d_{\mathbf{u}_1}}$ and $\mathbf{u}_2 \in \mathbb{R}^{d_{\mathbf{u}_2}}$ (where $d_{\mathbf{u}_1} + d_{\mathbf{u}_2} = d_\mathbf{u}$), which are fed into the coupling layers as follows:

$$\begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{u}_1 \odot \exp\left(s_1(\mathbf{u}_2)\right) + t_1(\mathbf{u}_2) \\ \mathbf{u}_2 \end{bmatrix}, \qquad \begin{bmatrix} \mathbf{o}_1 \\ \mathbf{o}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \odot \exp\left(s_2(\mathbf{v}_1)\right) + t_2(\mathbf{v}_1) \end{bmatrix}, \tag{5}$$

where $\odot$ denotes the Hadamard product or element-wise multiplication. The outputs $[\mathbf{o}_1, \mathbf{o}_2]$ are then concatenated again and passed to the next coupling block. The internal mappings s_1 and t_1 are functions from $\mathbb{R}^{d_{\mathbf{u}_2}} \to \mathbb{R}^{d_{\mathbf{u}_1}}$, and s_2 and t_2 are functions from $\mathbb{R}^{d_{\mathbf{u}_1}} \to \mathbb{R}^{d_{\mathbf{u}_2}}$. In general, s_i and t_i can be arbitrarily complicated functions (e.g. neural networks as in Ardizzone et al. (2019a)). In our proposed ISR approach, they are represented by EQL networks (see Figure 3), resulting in a fully symbolic invertible architecture. Moving forward, we shall refer to them as EQL *subnetworks* of the block.

The transformations above result in upper and lower triangular Jacobians:

$$J_{\mathbf{u} \mapsto \mathbf{v}} = \begin{bmatrix} \mathrm{diag}\big(\exp\left(s_1(\mathbf{u}_2)\right)\big) & \frac{\partial \mathbf{v}_1}{\partial \mathbf{u}_2} \\ 0 & I \end{bmatrix}, \qquad J_{\mathbf{v} \mapsto \mathbf{o}} = \begin{bmatrix} I & 0 \\ \frac{\partial \mathbf{o}_2}{\partial \mathbf{v}_1} & \mathrm{diag}\big(\exp\left(s_2(\mathbf{v}_1)\right)\big) \end{bmatrix}. \tag{6}$$

Hence, their determinants can be trivially computed:

$$\begin{aligned} \det\big(J_{\mathbf{u} \mapsto \mathbf{v}}\big) &= \textstyle\prod_{i=1}^{d_{\mathbf{u}_1}} \exp\big(\,[s_1(\mathbf{u}_2)]_i\,\big) = \exp\Big(\textstyle\sum_{i=1}^{d_{\mathbf{u}_1}} [s_1(\mathbf{u}_2)]_i\Big), \\ \det\big(J_{\mathbf{v} \mapsto \mathbf{o}}\big) &= \textstyle\prod_{i=1}^{d_{\mathbf{u}_2}} \exp\big(\,[s_2(\mathbf{v}_1)]_i\,\big) = \exp\Big(\textstyle\sum_{i=1}^{d_{\mathbf{u}_2}} [s_2(\mathbf{v}_1)]_i\Big)\,. \end{aligned} \tag{7}$$

Then, the resulting Jacobian determinant of the coupling block is given by

$$\begin{aligned} \det\big(J_{\mathbf{u} \mapsto \mathbf{o}}\big) &= \det\big(J_{\mathbf{u} \mapsto \mathbf{v}}\big) \cdot \det\big(J_{\mathbf{v} \mapsto \mathbf{o}}\big) \\ &= \exp\Big(\textstyle\sum_{i=1}^{d_{\mathbf{u}_1}} [s_1(\mathbf{u}_2)]_i\Big) \cdot \exp\Big(\textstyle\sum_{i=1}^{d_{\mathbf{u}_2}} [s_2(\mathbf{v}_1)]_i\Big) \\ &= \exp\Big(\textstyle\sum_{i=1}^{d_{\mathbf{u}_1}} [s_1(\mathbf{u}_2)]_i + \textstyle\sum_{i=1}^{d_{\mathbf{u}_2}} [s_2(\mathbf{v}_1)]_i\Big) \end{aligned}$$

[1] As direct division can lead to numerical issues, we apply the exponential function to s_i (after clipping its extreme values) in the formulation described in Eq. (5). This also guarantees non-zero diagonal entries in the Jacobian matrices.

$$= \exp\Big(\sum_{i=1}^{d_{\mathbf{u}_1}} [s_1(\mathbf{u}_2)]_i + \sum_{i=1}^{d_{\mathbf{u}_2}} \big[s_2\big(\mathbf{u}_1 \odot \exp\left(s_1(\mathbf{u}_2)\right) + t_1(\mathbf{u}_2)\big)\big]_i \Big) \tag{8}$$

which can be efficiently calculated. Indeed, the Jacobian determinant of the whole map $\mathbf{x} \to [\mathbf{y}, \mathbf{z}]$ is the product of the Jacobian determinants of the n underlying coupling blocks (see Figure 3).

Given the output $\mathbf{o} = [\mathbf{o}_1, \mathbf{o}_2]$, the expressions in Eqs. (5) are clearly invertible:

$$\mathbf{u}_2 = \big(\mathbf{o}_2 - t_2(\mathbf{o}_1)\big) \oslash \exp\left(s_2(\mathbf{o}_1)\right), \qquad \mathbf{u}_1 = \big(\mathbf{o}_1 - t_1(\mathbf{u}_2)\big) \oslash \exp\left(s_1(\mathbf{u}_2)\right) \tag{9}$$

where $\oslash$ denotes element-wise division. Crucially, even when the coupling block is inverted, the EQL subnetworks s_i and t_i need *not* themselves be invertible; they are only ever evaluated in the forward direction. We denote the whole ISR map $\mathbf{x} \to [\mathbf{y}, \mathbf{z}]$ as $f(\mathbf{x}; \theta) = \big[f_{\mathbf{y}}(\mathbf{x}; \theta), f_{\mathbf{z}}(\mathbf{x}; \theta)\big]$ parameterized by the EQL subnetworks parameters θ, and the inverse as $f^{-1}(\mathbf{y}, \mathbf{z}; \theta)$.

Remark 3.1. The proposed ISR architecture consists of a sequence of these symbolic reversible blocks. To enhance the model's predictive and expressive capability, we can: i) increase the number of reversible coupling blocks, ii) increase the number of hidden layers in each underlying EQL network, iii) increase the number of hidden neurons per layer in each underlying EQL network, or iv) increase the complexity of the symbolic activation functions used in the EQL network. However, it is worth noting that these enhancements come with a trade-off, as they inevitably lead to a decrease in the model's interpretability.

Remark 3.2. To further improve the model capacity, as in Ardizzone et al. (2019a), we incorporate (random, but fixed) permutation layers between the coupling blocks, which shuffles the input elements for subsequent coupling blocks. This effectively randomizes the configuration of splits $\mathbf{u} = [\mathbf{u}_1, \mathbf{u}_2]$ across different blocks, thereby enhancing interplay between variables.

Remark 3.3. Inspired by Ardizzone et al. (2019a), we split the coupling block's input vector $\mathbf{u} \in \mathbb{R}^{d_{\mathbf{u}}}$ into two halves, i.e. $\mathbf{u}_1 \in \mathbb{R}^{d_{\mathbf{u}_1}}$ and $\mathbf{u}_2 \in \mathbb{R}^{d_{\mathbf{u}_2}}$ where $d_{\mathbf{u}_1} = \lfloor \frac{d_{\mathbf{u}}}{2} \rfloor$ and $d_{\mathbf{u}_2} = d_{\mathbf{u}} - d_{\mathbf{u}_1}$. In the case where $\mathbf{u}$ is one-dimensional (or scalar), i.e. $d_{\mathbf{u}} = 1$ and $\mathbf{u} \in \mathbb{R}$, we pad it with an extra zero (so that $d_{\mathbf{u}} = 2$) along with a loss term that prevents the encoding of information in the extra dimension (e.g. we use the L_2 loss to maintain those values near zero).

Remark 3.4. The proposed ISR architecture is also compatible with the conditional ISR (cISR) framework proposed in the previous section. In essence, cISR identifies a bijective symbolic transformation directly between $\mathbf{x}$ and $\mathbf{z}$ given the observation $\mathbf{y}$. This is attained by feeding $\mathbf{y}$ as an extra input to each coupling block, during both the forward and inverse passes. In particular, and as suggested by Ardizzone et al. (2019b; 2021); Kruse et al. (2021), we adapt the same coupling layers given by Eqs. (5) and (9) to produce a conditional coupling block. Since the subnetworks s_i and t_i are never inverted, we enforce the condition on the observation by concatenating $\mathbf{y}$ to their inputs without losing the invertibility, i.e. we replace $s_1(\mathbf{u}_2)$ with $s_1(\mathbf{u}_2, \mathbf{y})$, etc. In complex settings, the condition $\mathbf{y}$ is first fed into a separate feed-forward conditioning network, resulting in higher-level conditioning features that are then injected into the conditional coupling blocks. Although cISR can have better generative properties (Kruse et al., 2021), it leads to more complex symbolic expressions and less interpretability as it explicitly conditions the map on the observation within the symbolic formulation. We denote the entire cISR forward map $\mathbf{x} \to \mathbf{z}$ as $f(\mathbf{x}; \mathbf{y}, \theta)$ parameterized by θ, and the inverse as $f^{-1}(\mathbf{z}; \mathbf{y}, \theta)$.

3.3 Maximum Likelihood Training of ISR

We train the proposed ISR model to learn a bijective symbolic transformation $f : \mathbb{R}^{d_{\mathbf{x}}} \to \mathbb{R}^{d_{\mathbf{y}}} \times \mathbb{R}^{d_{\mathbf{z}}}$. There are various choices to define the loss functions with different advantage and disadvantages (Grover et al., 2018; Ren et al., 2020; Ardizzone et al., 2019a; 2021; Kruse et al., 2021). As reported in Kruse et al. (2021), there are two main training approaches:

i) A standard supervised L_2 loss for fitting the model's $\mathbf{y}$ predictions to the training data, combined with a Maximum Mean Discrepancy (MMD) (Gretton et al., 2012; Ardizzone et al., 2019a) for fitting the latent distribution $p_{\boldsymbol{Z}}(\mathbf{z})$ to $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_{d_{\mathbf{z}}})$, given samples.

ii) A Maximum Likelihood Estimate (MLE) loss that enforces $\mathbf{z}$ to be standard Gaussian, i.e. $\mathbf{z} \sim p_{\boldsymbol{Z}}(\mathbf{z}) =$

$\mathcal{N}(\mathbf{0}, \boldsymbol{I}_{d_\mathbf{z}})$ and by approximating the distribution on $\mathbf{y}$ with a Gaussian distribution around the ground truth values $\mathbf{y}_{\text{gt}}$ with very low variance σ^2 (Dinh et al., 2016; Ren et al., 2020; Kruse et al., 2021).

Given that MLE is shown to perform well as reported in the literature (Ardizzone et al., 2019a), we apply it here. Next, we demonstrate how this approach is equivalent to minimizing the forward Kullback-Leibler (KL) divergence as the cost (cf. (Papamakarios et al., 2021)). We note that given the map $f(\mathbf{x};\theta) \mapsto [\mathbf{z}, \mathbf{y}]$, parameterized by θ, and assuming $\mathbf{y}$ and $\mathbf{z}$ are independent, the density $p_\mathbf{X}$ relates to $p_\mathbf{Y}$ and $p_\mathbf{Z}$ through the change-of-variables formula

$$p_{\boldsymbol{X}}(\mathbf{x};\theta) = p_{\boldsymbol{Y}}\big(\mathbf{y} = f_\mathbf{y}(\mathbf{x};\theta)\big)\; p_{\boldsymbol{Z}}\big(\mathbf{z} = f_\mathbf{z}(\mathbf{x};\theta)\big) \cdot \left|\det\big(J_{\mathbf{x}\mapsto[\mathbf{z},\mathbf{y}]}(\mathbf{x};\theta)\big)\right|. \tag{10}$$

where $J_{\mathbf{x}\mapsto[\mathbf{z},\mathbf{y}]}(\mathbf{x};\theta)$ denotes the Jacobian of the map f parameterized by θ. This expression is then used to define the loss function, which we derive by following the work in (Papamakarios et al., 2021). In particular, we aim to minimize the forward KL divergence between a target distribution $p^*_{\boldsymbol{X}}(\mathbf{x})$ and our *flow-based* model $p_{\boldsymbol{X}}(\mathbf{x};\theta)$, given by

$$\begin{aligned}
\mathcal{L}(\theta) &= D_{\text{KL}}\big[p^*_{\boldsymbol{X}}(\mathbf{x})\,\big|\big|\,p_{\boldsymbol{X}}(\mathbf{x};\theta)\big] \\
&= -\mathbb{E}_{p^*_{\boldsymbol{X}}(\mathbf{x})}\big[\log p_{\boldsymbol{X}}(\mathbf{x};\theta)\big] + \text{const.} \\
&= -\mathbb{E}_{p^*_{\boldsymbol{X}}(\mathbf{x})}\big[\log p_{\boldsymbol{Y}}\big(f_\mathbf{y}(\mathbf{x};\theta)\big) + \log p_{\boldsymbol{Z}}\big(f_\mathbf{z}(\mathbf{x};\theta)\big) + \log\left|\det\big(J_{\mathbf{x}\mapsto[\mathbf{z},\mathbf{y}]}(\mathbf{x};\theta)\big)\right|\big] + \text{const.}
\end{aligned} \tag{11}$$

The forward KL divergence is particularly suitable for cases where we have access to samples from the target distribution, but we cannot necessarily evaluate the target density $p^*_{\boldsymbol{X}}(\mathbf{x})$. Assuming we have a set of samples $\{\mathbf{x}_i\}_{i=1}^N$ from $p^*_{\boldsymbol{X}}(\mathbf{x})$, we can approximate the expectation in Eq. (11) using Monte Carlo integration as

$$\mathcal{L}(\theta) \approx -\frac{1}{N}\sum_{i=1}^{N}\Big(\log p_{\boldsymbol{Y}}\big(f_\mathbf{y}(\mathbf{x}_i;\theta)\big) + \log p_{\boldsymbol{Z}}\big(f_\mathbf{z}(\mathbf{x}_i;\theta)\big) + \log\left|\det\big(J_{\mathbf{x}\mapsto[\mathbf{z},\mathbf{y}]}(\mathbf{x}_i;\theta)\big)\right| + \text{const.}\Big)\,. \tag{12}$$

As we can see, minimizing the above Monte Carlo approximation of the KL divergence is equivalent to maximizing likelihood (or minimizing negative log-likelihood). Assuming $p_\mathbf{Z}$ is standard Gaussian and $p_\mathbf{Y}$ is a multivariate normal distribution around $\mathbf{y}_{\text{gt}}$, the negative log-likelihood (NLL) loss in Eq. (12) becomes

$$\mathcal{L}_{\text{NLL}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{2}\cdot\frac{\big(f_\mathbf{y}(\mathbf{x}_i;\theta) - \mathbf{y}_{\text{gt}}\big)^2}{\sigma^2} + \frac{1}{2}\cdot f_\mathbf{z}(\mathbf{x}_i;\theta)^2 - \log\left|\det\big(J_{\mathbf{x}\mapsto[\mathbf{z},\mathbf{y}]}(\mathbf{x}_i;\theta)\big)\right|\right)\,. \tag{13}$$

In other words, we find the optimal ISR parameters θ by minimizing the NLL loss in Eq. (13), and the resulting bijective symbolic expression can be directly extracted from the these optimal parameters.

Remark 3.5. We note that cISR is also suited for maximum likelihood training. Given the conditioning observation $\mathbf{y}$, the density $p_{\mathbf{X}\,|\,\mathbf{Y}}$ relates to $p_\mathbf{Z}$ through the change-of-variables formula

$$p_{\mathbf{X}\,|\,\mathbf{Y}}(\mathbf{x}\,|\,\mathbf{y},\theta) = p_{\mathbf{Z}}\big(\mathbf{z} = f(\mathbf{x};\mathbf{y},\theta)\big)\cdot\left|\det\big(J_{\mathbf{x}\mapsto\mathbf{z}}(\mathbf{x};\mathbf{y},\theta)\big)\right|, \tag{14}$$

where $J_{\mathbf{x}\mapsto\mathbf{z}}(\mathbf{x};\mathbf{y},\theta)$ indicates the Jacobian of the map f conditioned on $\mathbf{y}$ and parameterized by θ. Following the same procedure as above, the cISR model can be trained by minimizing the following NLL loss function

$$\mathcal{L}_{\text{NLL}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{2}\cdot f(\mathbf{x}_i;\mathbf{y}_i,\theta)^2 - \log\left|\det\big(J_{\mathbf{x}\mapsto\mathbf{z}}(\mathbf{x}_i;\mathbf{y}_i,\theta)\big)\right|\right)\,. \tag{15}$$

As we will show in the next section, if we ignore the condition on the observation $\mathbf{y}$, the loss in Eq. 15 can also be used for training ISR as a normalizing flow for the unsupervised learning task of approximating a target probability density function from samples (cf. Eq. 18).

4 Results

We evaluate our proposed ISR method on a variety of problems. We first show how ISR can serve as a normalizing flow for density estimation tasks on several test distributions. We then demonstrate the capabilities of ISR in solving inverse problems by considering two synthetic problems and then a more challenging application in ocean acoustics (Jensen et al., 2011; Ali et al., 2023; Huang et al., 2006; Dosso & Dettmer, 2011; Bianco et al., 2019; Holland et al., 2005; Benson et al., 2000). We mainly compare our ISR approach against INN (Ardizzone et al., 2019a) throughout our experiments. Further experimental details can be found in Appendix B.

4.1 Leveraging ISR for Density Estimation via Normalizing Flow

Given N independently and identically distributed (i.i.d.) samples, i.e. $\{\mathbf{X}_i\}_{i=1}^N \sim p_{\mathbf{x}}^{\text{target}}$, we would like to estimate the target density $p_{\mathbf{x}}^{\text{target}}$ and generate new samples from it. This problem categorizes as the density estimation where non-parametric, e.g. Kernel Density Estimation (Sheather, 2004), and parametric estimators, e.g. Maximum Entropy Distribution as the least biased estimator (Tohme et al., 2024), are classically used. In recent years, this problem has been approached using normalizing flow equipped with invertible map, which has gained a great deal of interest in the generative AI task. In an attempt to introduce interpretability in the trained model, we extend invertible normalizing flow to symbolic framework using the proposed ISR architecture.

In order to use the proposed ISR method as the normalizing flow for the unsupervised task of resampling from an intractable target distribution, we drop out $\mathbf{y}$ and enforce $d_{\mathbf{x}} = d_{\mathbf{z}}$.[2] In this case, we aim to learn a an invertible and symbolic map $f : \mathbb{R}^{d_{\mathbf{x}}} \to \mathbb{R}^{d_{\mathbf{z}}}$, parameterized by θ such that

$$\mathbf{z} = f(\mathbf{x};\theta), \qquad \mathbf{x} = f^{-1}(\mathbf{z};\theta), \tag{16}$$

where $\mathbf{z} \sim p_{\mathbf{z}}$ is the standard normal distribution function, which is easy to sample from. Using the change-of-variables formula, the density $p_{\mathbf{x}}$ relates to the density $p_{\mathbf{z}}$ via

$$p_{\mathbf{x}}(\mathbf{x};\theta) = p_{\mathbf{z}}\big(\mathbf{z} = f(\mathbf{x};\theta)\big) \cdot \left|\det\big(J_{\mathbf{x}\mapsto\mathbf{z}}(\mathbf{x};\theta)\big)\right|, \tag{17}$$

where $J_{\mathbf{x}\mapsto\mathbf{z}}$ indicates the Jacobian of the map f parameterized by θ. Following the same procedure outlined in Section 3.3, the model can be trained by minimizing the following NLL loss function

$$\mathcal{L}_{\text{NLL}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{2}\cdot f(\mathbf{x}_i;\theta)^2 - \log\left|\det\big(J_{\mathbf{x}\mapsto\mathbf{z}}(\mathbf{x}_i;\theta)\big)\right|\right) . \tag{18}$$

We compare the proposed ISR approach with INN in recovering several two-dimensional target distributions (i.e. $d_{\mathbf{x}} = d_{\mathbf{z}} = 2$). First, we consider a fairly simple multivariate normal distribution $\mathcal{N}\big(\boldsymbol{\mu}, \boldsymbol{\Sigma}\big)$ with mean $\boldsymbol{\mu} = [0, 3]$ and covariance matrix $\boldsymbol{\Sigma} = \frac{1}{10}\cdot\boldsymbol{I}_2$ as the target density. Then, we consider more challenging distributions: the "Banana," "Mixture of Gaussians (MoG)," and "Ring" distributions that are also considered in Jaini et al. (2019); Wenliang et al. (2019). For each of these target distributions, we draw $N_s = 10^4$ i.i.d. samples and train an invertible map that transport the samples to a standard normal distribution. This is called normalizing flow, where we intend to compare the standard INN with the proposed ISR architecture. Here, we use a single coupling block for the Gaussian and banana cases, and two coupling blocks for the ring and MoG test cases.

As shown in Figure 4, the proposed ISR method finds and generates samples of the considered target densities with slightly better accuracy than INN. We report the bijective symbolic expressions in Table 1 of Appendix A. For instance, the first target distribution considered in Figure 4 is the two-dimensional multivariate Gaussian distribution $\mathcal{N}\big(\boldsymbol{\mu}, \boldsymbol{\Sigma}\big)$ with mean $\boldsymbol{\mu} = [0, 3]$ and covariance matrix $\boldsymbol{\Sigma} = \frac{1}{10}\cdot\boldsymbol{I}_2$. This

[2] In the absence of $\mathbf{y}$, cISR and ISR are equivalent, so we simply refer to them as ISR. Similarly, INN and cINN become equivalent, and we simply refer to them as INN.

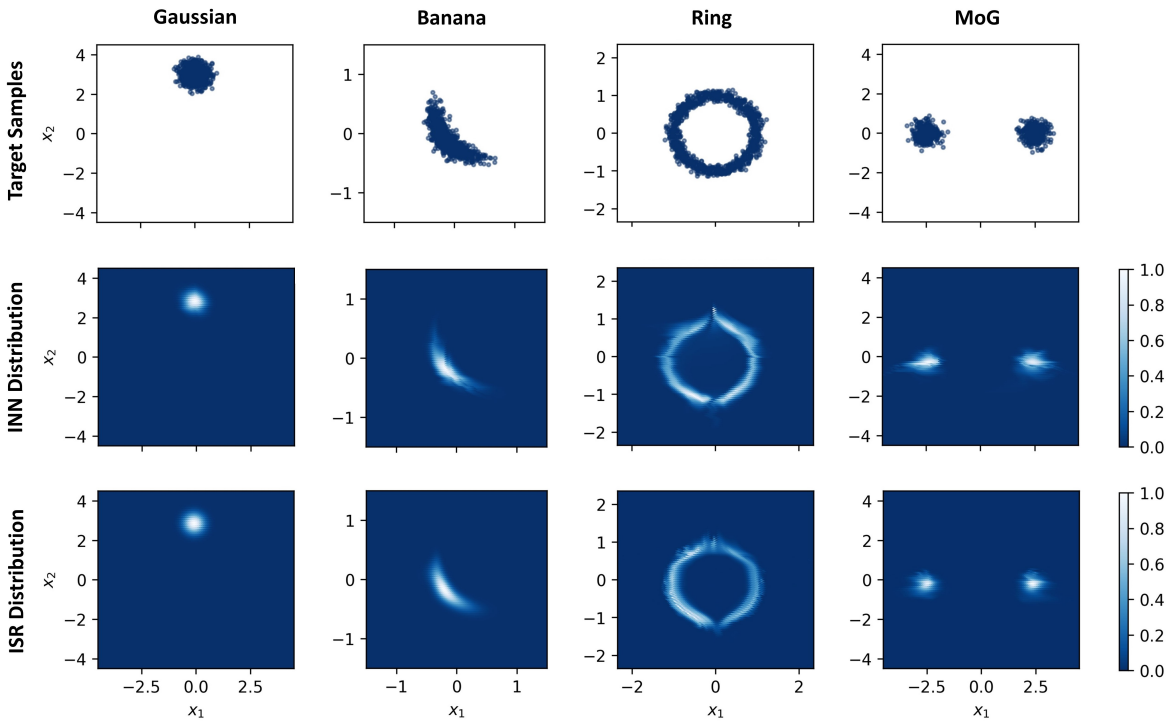

Figure 4: Samples from four different target densities (first row), and their estimated distributions using INN (second row) and the proposed ISR method (third row).

is indeed a shifted and scaled standard Gaussian distribution where we know the analytical solution to the true map:

$$\begin{aligned}\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} &\sim \mathcal{N}\left(\boldsymbol{\mu}, \frac{1}{10} \cdot \boldsymbol{I}_2\right) \\ &\sim \boldsymbol{\mu} + \sqrt{\frac{1}{10}} \cdot \mathcal{N}\left(\mathbf{0}, \boldsymbol{I}_2\right) \\ &= \boldsymbol{\mu} + \frac{1}{\sqrt{10}} \cdot \mathbf{Z} = \begin{bmatrix} 0 \\ 3 \end{bmatrix} + 0.316 \cdot \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} 0.316\, Z_1 \\ 3 + 0.316\, Z_2 \end{bmatrix} . \end{aligned} \tag{19}$$

As shown in Table 1 of Appendix A, for this Gaussian distribution example, the proposed ISR method finds the following bijective expression:

$$\begin{aligned} z_1 &= x_1 \cdot e^{1.16} = 3.19\, x_1 &&\Longleftrightarrow &&x_1 = \frac{1}{3.19}\, z_1 = 0.313\, z_1 \\ z_2 &= x_2 \cdot e^{1.14} - 9.39 = 3.13\, x_2 - 9.39 &&\Longleftrightarrow &&x_2 = \frac{1}{3.13}\left(z_2 + 9.39\right) = 0.319\, z_2 + 3.00 \end{aligned}$$

In other words, the proposed ISR method identifies the true underlying transformation given by Eq. (19) with a high accuracy.

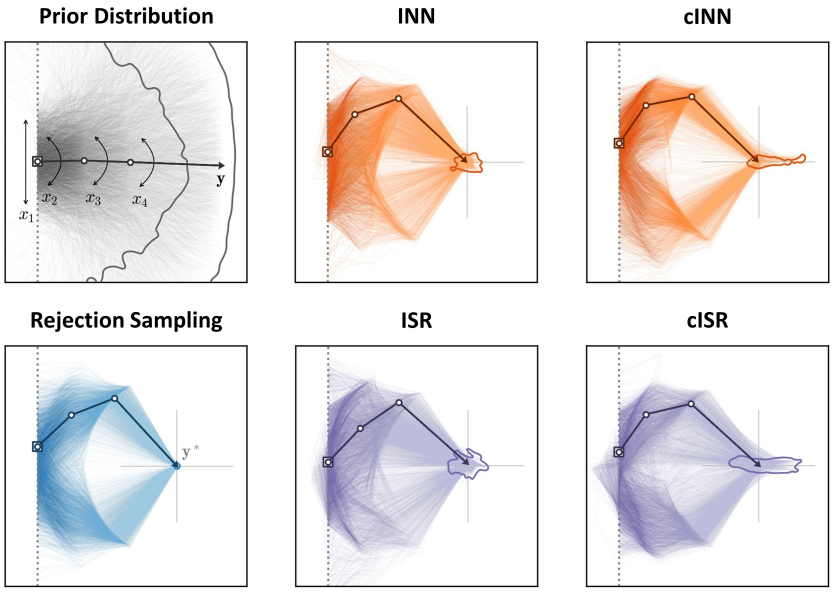

Figure 5: Results for the inverse kinematics benchmark problem. The faint colored lines indicate sampled arm configurations $\mathbf{x}$ taken from each model's predicted posterior $\hat{p}(\mathbf{x} \,|\, \mathbf{y}^*)$, conditioned on the target end point $\mathbf{y}^*$, which is indicated by a gray cross. The contour lines around the target end point enclose the regions containing 97% of the sampled arms' end points. We emphasize the arm with the highest estimated likelihood as a bold line.

4.2 Inverse Kinematics

We now consider a geometrical benchmark example used by Ardizzone et al. (2019a); Kruse et al. (2021), which simulates an inverse kinematics problem in a two-dimensional space: A multi-jointed 2D arm moves vertically along a rail and rotates at three joints. In this problem, we are interested in the configurations (i.e. the four degrees of freedom) of the arm that place the arm's end point at a given position. The forward process computes the coordinates of the end point $\mathbf{y} \in \mathbf{R}^2$, given a configuration $\mathbf{x} \in \mathbf{R}^4$ (i.e. $d_{\mathbf{x}} = 4$, $d_{\mathbf{y}} = 2$, and hence $d_{\mathbf{z}} = 2$). In particular, the forward process takes $\mathbf{x} = [x_1, x_2, x_3, x_4]$ as argument, where x_1 denotes the arm's starting height, and x_2, x_3, x_4 are its three joint angles, and returns the coordinates of its end point $\mathbf{y} = [y_1, y_2]$ given by

$$
\begin{aligned}
y_1 &= \ell_1 \sin(x_2) + \ell_2 \sin(x_2 + x_3) + \ell_3 \sin(x_2 + x_3 + x_4) + x_1 \\
y_2 &= \ell_1 \cos(x_2) + \ell_2 \cos(x_2 + x_3) + \ell_3 \cos(x_2 + x_3 + x_4)
\end{aligned} \tag{20}
$$

where the segment lengths $\ell_1 = 0.5$, $\ell_2 = 0.5$, and $\ell_3 = 1$. The parameters $\mathbf{x}$ follow a Gaussian prior $\mathbf{x} \sim \mathcal{N}\big(\mathbf{0}, \boldsymbol{\sigma}^2 \cdot \boldsymbol{I}_4\big)$ with $\boldsymbol{\sigma}^2 = [0.25^2, 0.25, 0.25, 0.25]$, which favors a configuration with a centered origin and 180° joint angles (see Figure 5). We consider a training dataset of size 10^6, constructed using this Gaussian prior and the forward process in Eq. (20). The inverse problem here asks to find the posterior distribution $p(\mathbf{x} \,|\, \mathbf{y}^*)$ of all possible configurations (or parameters) $\mathbf{x}$ that result in the arm's end point being positioned at a given $\mathbf{y}^*$ location. This inverse kinematics problem, being low-dimensional, offers computationally inexpensive forward (and backward) process, which enables fast training, intuitive visualizations, and an approximation of the true posterior estimates via rejection sampling.[3]

[3]**Rejection sampling.** Assume we require N_s samples of $\mathbf{x}$ from the posterior $p(\mathbf{x} \,|\, \mathbf{y}^*)$ given some observation $\mathbf{y}^*$. After setting some acceptance threshold ϵ, we iteratively generate $\mathbf{x}$-samples from the prior. For each sample, we simulate the corresponding $\mathbf{y}$-values and only keep those with $\text{dist}(\mathbf{y}, \mathbf{y}^*) < \epsilon$. The process is repeated until N_s samples are collected (or accepted). Indeed, the smaller the threshold ϵ, the more $\mathbf{x}$-samples candidates (and hence the more simulations) have to be generated. Hence, we adopt this approach in this low-dimensional inverse kinematics problem, where we can afford to run the forward process (or simulation) a huge number of times.

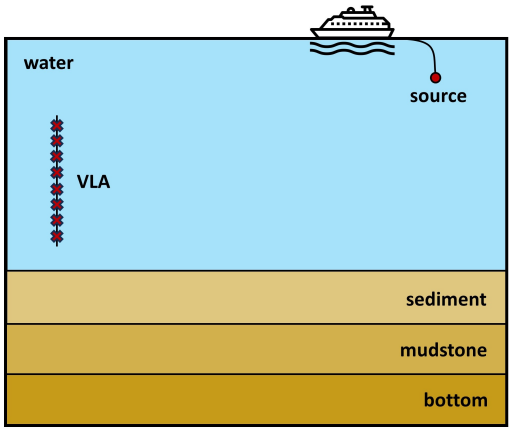

Figure 6: The SWellEx-96 experiment environment. The acoustic source is towed by a research vessel and transmits signals at various frequencies. The acoustic sensor consists of a vertical line array (VLA). Based on the measurements collected at the VLA, the objective is to estimate posterior distributions over parameters of interest (e.g. water depth, sound speed at the water-sediment interface, source range and depth, etc.).

An example of a challenging end point $\mathbf{y}^*$ is shown in Figure 5, where we compare the proposed ISR method against the approximate true posterior (obtained via rejection sampling), as well as INN. The chosen $\mathbf{y}^*$ is particularly challenging, since this end point is unlikely under the prior $p(\mathbf{x})$, and results in a strongly bi-modal posterior $p(\mathbf{x} \,|\, \mathbf{y}^*)$ (Ardizzone et al., 2019a; Kruse et al., 2021). As we can observe in Figure 5, compared to rejection sampling, all the considered architectures (i.e. INN, cINN, ISR, and cISR) are able to capture the two symmetric modes well. However, we can clearly see that they all generate $\mathbf{x}$-samples such that their resulting end points miss the target $\mathbf{y}^*$ by a wider margin. Quantitative results are also provided in Appendix C.

4.3 Application: Geoacoustic Inversion

Predicting acoustic propagation at sea is vital for various applications, including sonar performance forecasting and mitigating noise pollution at sea. The ability to predict sound propagation in a shallow water environment depends on understanding the seabed's geoacoustic characteristics. Inferring those characteristics from ocean acoustic measurements (or signals) is known as geoacoustic inversion (GI). GI involves several components: (i) representation of the ocean environment, (ii) selection of the inversion method, including the forward propagation model implemented, and (iii) quantification of the uncertainty related to the parameters estimates.

We start by describing the ocean environment. We consider the setup of SWellEx-96 Yardim et al. (2010); Meyer & Gemba (2021), which was an experiment done off the coast of San Diego, CA, near Point Loma. This experimental setting is one of the most used, documented, and understood studies in the undersea acoustics community.[4] As depicted in Figure 6, the data is collected via a vertical line array (VLA). The specification of the 21 hydrophones of the VLA and sound speed profile (SSP) in the water column is provided in the SWellEx-96 documentation. The SSP and sediment parameters are considered to be range-independent. Water depth refers to the depth of the water at the array. The source is towed by a research vessel which consists of a comb signal comprising frequencies of $49, 79, 112, 148, 201, 283,$ and 388 Hz. While in the SWellEx-96 experiment the position of the source changes with time, for this task we consider the instant when the source depth is 60 m and the distance (or range) between the source and the VLA is 3 km.

The sediment layer is modeled with the following properties. The seabed consists initially of a sediment layer that is 23.5 meters thick, with a density of 1.76 g/cm^3, and an attenuation of 0.2 dB/kmHz. The sound speed at the bottom of this layer is assumed to be 1593 m/s. The second layer is mudstone that is 800 meters

[4] see `http://swellex96.ucsd.edu/`

thick, possessing a density of 2.06 g/cm^3, and an attenuation of 0.06 dB/kmHz. The top and bottom sound speeds of this layer are 1881 m/s and 3245 m/s respectively. The description of the geoacoustic model of the SWellEx-96 experiment is complemented by a half-space featuring a density of 2.66 g/cm^3, an attenuation of 0.020 dB/kmHz, and a sound speed of 5200 m/s.

Here, we consider two geoacoustic inversion tasks:

Task 1. Based on the measurements at the VLA, the objective of this task is to infer the posterior distribution over the water depth as well as the sound speed at the water-sediment interface. For this task, we assume all the quantities above to be known. The unknown parameters m_1 (the water depth) and m_2 (the sound speed at the water-sediment interface) follow a uniform prior in $[200.5, 236.5]$ m and $[1532, 1592]$ m/s, i.e. $m_1 \sim \mathcal{U}([200.5, 236.5])$ and $m_2 \sim \mathcal{U}([1532, 1592])$, where $\mathcal{U}(\Omega)$ denotes a uniform distribution in the domain Ω.

Task 2. In addition to the two parameters considered in *Task 1* (i.e. the water depth m_1 and the sound speed at the water-sediment interface m_2), we also estimate the posterior distribution over the VLA tilt m_3, as well as the thickness of the first (sediment) layer m_4. All other quantities provided above are assumed to be known. As in *Task 1*, the unknown parameters follow a uniform prior, i.e. $m_1 \sim \mathcal{U}([200.5, 236.5])$, $m_2 \sim \mathcal{U}([1532, 1592])$, $m_3 \sim \mathcal{U}([-2, 2])$, and $m_4 \sim \mathcal{U}([18.5, 28.5])$.

The received pressure $\mathbf{y}$ on each hydrophone and for each frequency is a function of unknown parameters $\mathbf{m}$ (e.g. water depth, sound speed at the water-sediment interface, etc.) and additive noise $\boldsymbol{\epsilon}$ as follows

$$\mathbf{y} = s(\mathbf{m}, \boldsymbol{\epsilon}) = F(\mathbf{m}) + \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \tag{21}$$

where $\boldsymbol{\Sigma}$ is the covariance matrix of data noise. Here, $s(\mathbf{m}, \boldsymbol{\epsilon})$ is a known forward model that, assuming an additive noise model, can be rewritten as $F(\mathbf{m}) + \boldsymbol{\epsilon}$, where $F(\mathbf{m})$ represents the undersea acoustic model Jensen et al. (2011). The SWellEx-96 experiment setup involves a complicated environment and no closed from analytical solution is available for $F(\mathbf{m})$. In this case, $F(\mathbf{m})$ can only be evaluated numerically, and we use the normal-modes program KRAKEN Porter (1992) for this purpose.

Recently, machine learning algorithms have gained attention in the ocean acoustics community (Bianco et al., 2019) for their notable performance and efficiency, especially when compared to traditional methods such as MCMC. In this work, for the first time, we use the concept of invertible networks to estimate posterior distributions in GI. Particularly appealing is that invertible architectures can replace both the forward propagation model as well as the inversion method.

We now discuss the training of the invertible architectures. We use the uniform prior on parameters $\mathbf{m}$ (described in *Task 1* and *Task 2* above) and the forward model in Eq. (21) to construct a synthetic data set for the SWellEx-96 experimental setup using the normal-modes program KRAKEN. For each parameter $\mathbf{m}$, we have $7 \times 21 = 147$ values for the pressure $\mathbf{y}$ received at hydrophones corresponding to the source's 7 different frequencies and the 21 active hydrophones. For inference, we use a test acoustic signal $\mathbf{y}^*$ that corresponds to the actual parameters values from the SWellEx-96 experiment, where the source is 60 m deep and its distance from the VLA is 3 km (i.e. $m_1^* = 216.5$, $m_2^* = 1572.368$, $m_3^* = 0$, $m_4^* = 23.5$). Also, the signal-to-noise ratio (SNR) is 15 dB.

Inspired by Zhang & Curtis (2021), for the invertible architectures, we include data noise $\boldsymbol{\epsilon}$ as additional model parameters to be learned. In this context, as depicted in the Figure 7, the input of the network is obtained by augmenting the unknown parameters $\mathbf{m}$ with additive noise $\boldsymbol{\epsilon}$. There are several ways to use the measurements collected across the 21 hydrophones for training. For instance, one can stack all hydrophones' data and treat them as a single quantity at the network's output. Alternatively, one can treat each hydrophone measurement independently as an individual training example. For the former, the additive noise will be learned separately for each hydrophone pressure $\mathbf{y}$, while for the latter, we essentially learn the effective additive noise over all hydrophones simultaneously. In this experiment, we adopt the latter training approach, which disregards the inter-hydrophone variations, thereby reducing computational overhead.

The pressures received on the hydrophones are considered in the frequency domain, and hence they can be complex numbers. While the invertible architectures can be constructed to address complex numbers, in this case study, we stack the real and imaginary parts of the pressure field at the network's output. That is,

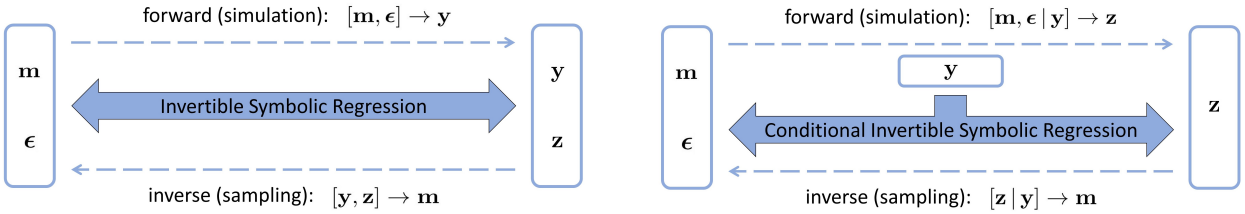

Figure 7: A conceptual figure of ISR (left) and cISR (right) for the geoacoustic inversion task. The posterior distribution of the parameters of interest $\mathbf{m}$ can be obtained by sampling $\mathbf{z}$ (e.g. from a standard Gaussian distribution) for a fixed observation $\mathbf{y}^*$ and running the trained bijective model backwards. To appropriately account for noise in the data, we include random data noise $\boldsymbol{\epsilon}$ as additional model parameters.

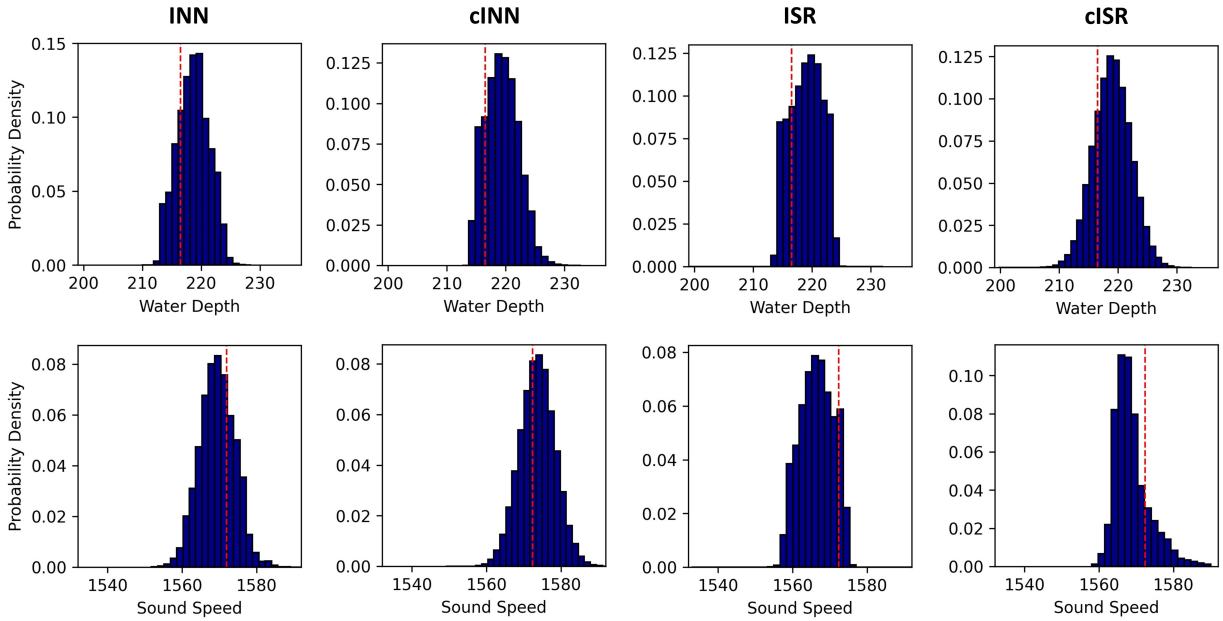

Figure 8: *Task 1.* For a fixed observation $\mathbf{y}^*$, we compare the estimated posteriors $\hat{p}(\mathbf{x} \,|\, \mathbf{y}^*)$ of INN, cINN, and the proposed ISR and cISR methods. Vertical dashed red lines show the ground truth values $\mathbf{x}^*$.

the pressure $y = \mathrm{Re}\{y\} + i\,\mathrm{Im}\{y\}$ will be represented as $\big[\mathrm{Re}\{y\}, \mathrm{Im}\{y\}\big]$ at the network's output. In short, the 7 pressures (corresponding to the 7 source's frequencies) received at each hydrophone are replaced by 14 real numbers at the output of the network. Also, the 14 corresponding additive noises are concatenated with the parameters $\mathbf{m}$ at the network's input. Since the dimension of the network's input is $14 + d_{\mathbf{m}}$, the latent variables $\mathbf{z}$ at the network output will be $d_{\mathbf{m}}$-dimensional for ISR and INN, and $(14 + d_{\mathbf{m}})$-dimensional for cINN and cISR. We compare the performance of the proposed ISR and cISR algorithms against INN and cINN in solving GI.

The inferred posterior distributions via INN, cINN, ISR, and cISR, for the GI *Task 1* and *Task 2* are depicted in Fig. 8 and Fig. 9,respectively. The performance of the ISR and cISR architectures is similar to that of the INN and cINN architectures. All methods produce point estimates – Maximum a Posteriori (MAP) estimates – close to the ground truth values, showcasing the efficacy of invertible architectures in addressing cumbersome inversion tasks.

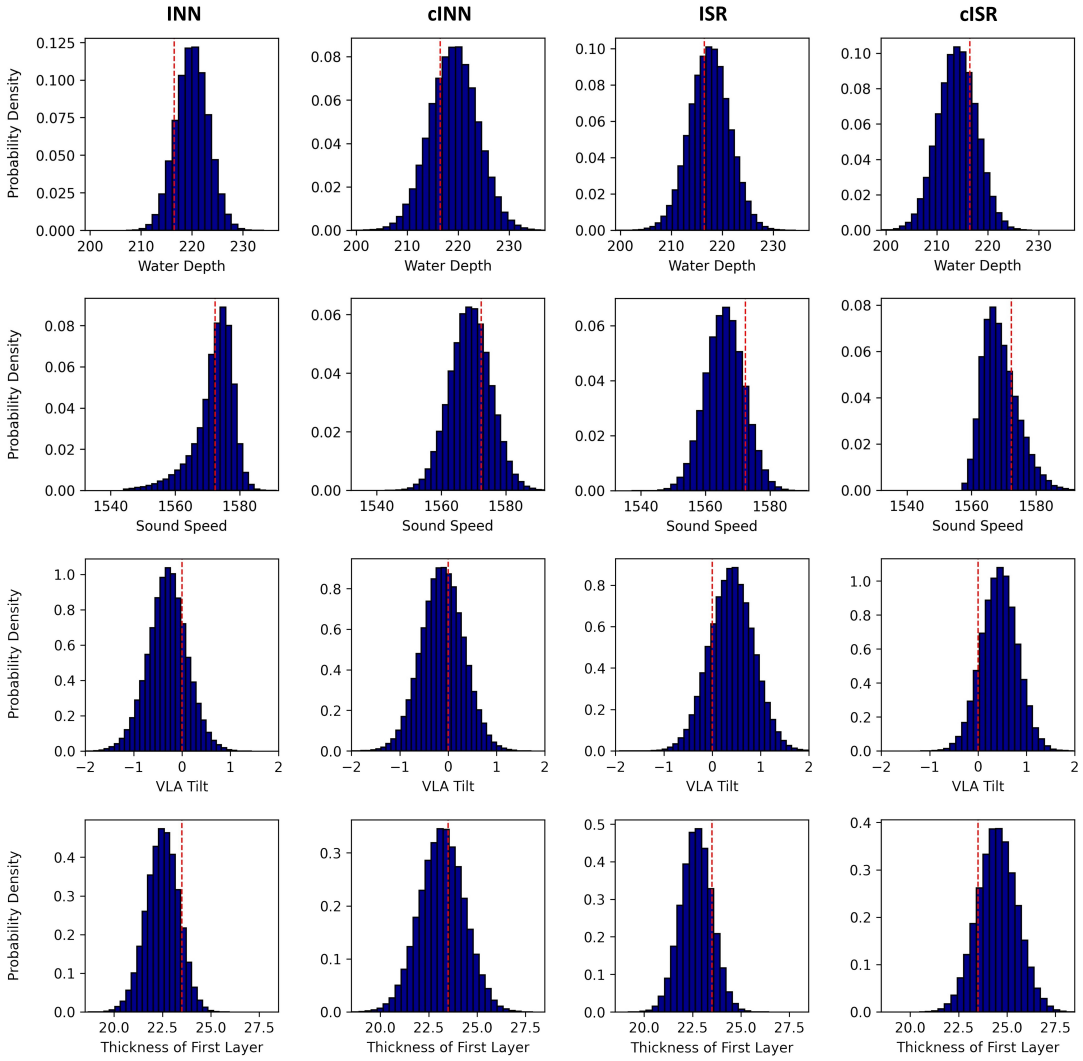

Figure 9: *Task 2.* For a fixed observation $\mathbf{y}^*$, we compare the estimated posteriors $\hat{p}(\mathbf{x} \,|\, \mathbf{y}^*)$ of INN, cINN, and the proposed ISR and cISR methods. Vertical dashed red lines show the ground truth values $\mathbf{x}^*$.

5 Conclusion

In this work, we introduce Invertible Symbolic Regression (ISR), a novel technique that identifies the relationships between the inputs and outputs of a given dataset using invertible architectures. This is achieved by bridging and integrating concepts of Invertible Neural Networks (INNs) and Equation Learner (EQL). This integration transforms the affine coupling blocks of INNs into a symbolic framework, resulting in an end-to-end differentiable symbolic inverse architecture that allows for efficient gradient-based learning. The proposed ISR method, equipped with sparsity promoting regularization, has the ability to not only capture complex functional relationships but also yield concise and interpretable invertible expressions. We demonstrate the versatility of ISR as a normalizing flow for density estimation and its applicability in solving inverse problems, particularly in the context of ocean acoustics, where it shows promising results in inferring posterior distributions of underlying parameters. This work is a first attempt toward creating interpretable symbolic invertible maps. While we mainly focused on introducing the ISR architecture and showing its applicability in density estimation tasks and inverse problems, an interesting research direction would be to explore the practicality of ISR in challenging generative modeling tasks (e.g. image or text generation, etc.).

References

Wael H Ali, Aaron Charous, Chris Mirabito, Patrick J Haley, and Pierre FJ Lermusiaux. MSEAS-ParEq for coupled ocean-acoustic modeling around the globe. In *OCEANS 2023-MTS/IEEE US Gulf Coast*, pp. 1–10. IEEE, 2023.

Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to MCMC for machine learning. *Machine learning*, 50:5–43, 2003.

Lynton Ardizzone, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. In *International Conference on Learning Representations*, 2019a.

Lynton Ardizzone, Carsten Lüth, Jakob Kruse, Carsten Rother, and Ullrich Köthe. Guided image generation with conditional invertible neural networks. *arXiv preprint arXiv:1907.02392*, 2019b.

Lynton Ardizzone, Jakob Kruse, Carsten Lüth, Niels Bracher, Carsten Rother, and Ullrich Köthe. Conditional invertible neural networks for diverse image-to-image translation. In Zeynep Akata, Andreas Geiger, and Torsten Sattler (eds.), *Pattern Recognition*, pp. 373–387, Cham, 2021. Springer International Publishing. ISBN 978-3-030-71278-5.

Yves F Atchadé and Jeffrey S Rosenthal. On adaptive Markov chain Monte Carlo algorithms. *Bernoulli*, 11 (5):815–828, 2005.

Jeremy Benson, N Ross Chapman, and Andreas Antoniou. Geoacoustic model inversion using artificial neural networks. *Inverse Problems*, 16(6):1627, 2000.

Michael J Bianco, Peter Gerstoft, James Traer, Emma Ozanich, Marie A Roch, Sharon Gannot, and Charles-Alban Deledalle. Machine learning in acoustics: Theory and applications. *The Journal of the Acoustical Society of America*, 146(5):3590–3628, 2019.

Luca Biggio, Tommaso Bendinelli, Alexander Neitz, Aurelien Lucchi, and Giambattista Parascandolo. Neural symbolic regression that scales. In *International Conference on Machine Learning*, 2021.

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov chain Monte Carlo*. CRC press, 2011.

Bogdan Burlacu, Gabriel Kronberger, and Michael Kommenda. Operon C++: An efficient genetic programming framework for symbolic regression. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, GECCO '20, pp. 1562–1570, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450371278. doi: 10.1145/3377929.3398099.

N Ross Chapman and Er Chang Shang. Review of geoacoustic inversion in underwater acoustics. *Journal of Theoretical and Computational Acoustics*, 29(03):2130004, 2021.

Patrick R Conrad, Youssef M Marzouk, Natesh S Pillai, and Aaron Smith. Accelerating asymptotically exact MCMC for computationally intensive models via local approximations. *Journal of the American Statistical Association*, 111(516):1591–1607, 2016.

Paul G Constantine, Carson Kent, and Tan Bui-Thanh. Accelerating Markov chain Monte Carlo with active subspaces. *SIAM Journal on Scientific Computing*, 38(5):A2779–A2805, 2016.

Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48):30055–30062, 2020a.

Miles Cranmer, Alvaro Sanchez Gonzalez, Peter Battaglia, Rui Xu, Kyle Cranmer, David Spergel, and Shirley Ho. Discovering symbolic models from deep learning with inductive biases. *Advances in Neural Information Processing Systems*, 33:17429–17442, 2020b.

Katalin Csilléry, Michael GB Blum, Oscar E Gaggiotti, and Olivier François. Approximate Bayesian computation (abc) in practice. *Trends in ecology & evolution*, 25(7):410–418, 2010.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. *arXiv preprint arXiv:1605.08803*, 2016.

Stan E Dosso and Jan Dettmer. Bayesian matched-field geoacoustic inversion. *Inverse Problems*, 27(5): 055009, 2011.

Arnaud Doucet and Xiaodong Wang. Monte Carlo methods for signal processing: a review in the statistical signal processing context. *IEEE Signal Processing Magazine*, 22(6):152–170, 2005.

Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. *Advances in neural information processing systems*, 32, 2019.

Anthony William Fairbank Edwards. *Likelihood*. CUP Archive, 1984.

Qinwei Fan, Jacek M Zurada, and Wei Wu. Convergence of online gradient method for feedforward neural networks with smoothing l1/2 regularization penalty. *Neurocomputing*, 131:208–216, 2014.

Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics (DSAA)*, pp. 80–89. IEEE, 2018.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.

Aditya Grover, Manik Dhar, and Stefano Ermon. Flow-GAN: Combining maximum likelihood and adversarial learning in generative models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Charles W Holland, Jan Dettmer, and Stan E Dosso. Remote sensing of sediment density and velocity gradients in the transition layer. *The Journal of the Acoustical Society of America*, 118(1):163–177, 2005.

Chen-Fen Huang, Peter Gerstoft, and William S Hodgkiss. Uncertainty analysis in matched-field geoacoustic inversions. *The Journal of the Acoustical Society of America*, 119(1):197–207, 2006.

Priyank Jaini, Kira A Selby, and Yaoliang Yu. Sum-of-squares polynomial flow. In *International Conference on Machine Learning*, pp. 3009–3018. PMLR, 2019.

Finn B Jensen, William A Kuperman, Michael B Porter, Henrik Schmidt, and Alexandra Tolstoy. *Computational ocean acoustics*, volume 2011. Springer, 2011.

Ying Jin, Weilin Fu, Jian Kang, Jiadong Guo, and Jian Guo. Bayesian symbolic regression. *arXiv preprint arXiv:1910.08892*, 2019.

Pierre-Alexandre Kamienny, Stéphane d'Ascoli, Guillaume Lample, and François Charton. End-to-end symbolic regression with transformers. *arXiv preprint arXiv:2204.10532*, 2022.

Liron Simon Keren, Alex Liberzon, and Teddy Lazebnik. A computational framework for physics-informed symbolic regression with straightforward integration of domain knowledge. *Scientific Reports*, 13(1):1249, 2023.

Samuel Kim, Peter Y Lu, Srijon Mukherjee, Michael Gilbert, Li Jing, Vladimir Čeperić, and Marin Soljačić. Integration of neural network-based symbolic regression in deep learning for scientific discovery. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020.

Michael Kommenda, Bogdan Burlacu, Gabriel Kronberger, and Michael Affenzeller. Parameter identification for symbolic regression using nonlinear least squares. *Genetic Programming and Evolvable Machines*, 21 (3):471–501, 2020.

Anoop Korattikara, Yutian Chen, and Max Welling. Austerity in MCMC land: Cutting the metropolis-hastings budget. In *International conference on machine learning*, pp. 181–189. PMLR, 2014.

John R Koza and John R Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.

Jakob Kruse, Lynton Ardizzone, Carsten Rother, and Ullrich Köthe. Benchmarking invertible architectures on inverse problems. *arXiv preprint arXiv:2101.10763*, 2021.

Vyacheslav Kungurtsev, Adam Cobb, Tara Javidi, and Brian Jalaian. Decentralized Bayesian learning with Metropolis-adjusted Hamiltonian Monte Carlo. *Machine Learning*, pp. 1–29, 2023.

William La Cava, Patryk Orzechowski, Bogdan Burlacu, Fabricio Olivetti de Franca, Marco Virgolin, Ying Jin, Michael Kommenda, and Jason H Moore. Contemporary symbolic regression methods and their relative performance. 2021.

Peter M Lee. *Bayesian statistics*. Arnold Publication, 1997.

Thomas Leonard and John SJ Hsu. *Bayesian methods: an analysis for statisticians and interdisciplinary researchers*, volume 5. Cambridge University Press, 2001.

Ziming Liu and Max Tegmark. Machine learning conservation laws from trajectories. *Physical Review Letters*, 126(18):180604, 2021.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024.

Alexander Luce, Ali Mahdavi, Heribert Wankerl, and Florian Marquardt. Investigation of inverse design of multilayer thin-films with conditional invertible neural networks. *Machine Learning: Science and Technology*, 4(1):015014, feb 2023. doi: 10.1088/2632-2153/acb48d. URL `https://dx.doi.org/10.1088/2632-2153/acb48d`.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Georg Martius and Christoph H Lampert. Extrapolation and learning equations. *arXiv preprint arXiv:1610.02995*, 2016.

Florian Meyer and Kay L. Gemba. Probabilistic focalization for shallow water localization. *J. Acoust. Soc. Am.*, 150(2):1057–1066, 08 2021.

T Nathan Mundhenk, Mikel Landajuela, Ruben Glatt, Claudio P Santiago, Daniel M Faissol, and Brenden K Petersen. Symbolic regression via neural-guided genetic programming population seeding. *arXiv preprint arXiv:2111.00053*, 2021.

Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Frank Noé, Simon Olsson, Jonas Köhler, and Hao Wu. Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning. *Science*, 365(6457):eaaw1147, 2019.

Patryk Orzechowski, William La Cava, and Jason H Moore. Where are we now? A large benchmark study of recent symbolic regression methods. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1183–1190, 2018.

Govinda Anantha Padmanabha and Nicholas Zabaras. Solving inverse problems using conditional invertible neural networks. *Journal of Computational Physics*, 433:110194, 2021.

George Papamakarios, David Sterratt, and Iain Murray. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 837–848. PMLR, 2019.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

Yudi Pawitan. *In all likelihood: statistical modelling and inference using likelihood*. Oxford University Press, 2001.

Brenden K Petersen, Mikel Landajuela Larma, Terrell N. Mundhenk, Claudio Prata Santiago, Soo Kyung Kim, and Joanne Taery Kim. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. In *International Conference on Learning Representations*, 2021.

Michael B Porter. The kraken normal mode program. *Naval Research Laboratory, Washington DC*, 1992.

Stefan T Radev, Frederik Graw, Simiao Chen, Nico T Mutters, Vanessa M Eichel, Till Bärnighausen, and Ullrich Köthe. Outbreakflow: Model-based Bayesian inference of disease outbreak dynamics with invertible neural networks and its application to the covid-19 pandemics in germany. *PLoS computational biology*, 17(10):e1009472, 2021.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Simiao Ren, Willie Padilla, and Jordan Malof. Benchmarking deep inverse models over time, and the neural-adjoint method. *Advances in Neural Information Processing Systems*, 33:38–48, 2020.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pp. 1530–1538. PMLR, 2015.

Subham Sahoo, Christoph Lampert, and Georg Martius. Learning equations for extrapolation and control. In *International Conference on Machine Learning*, pp. 4442–4450. PMLR, 2018.

Tim Salimans, Diederik Kingma, and Max Welling. Markov chain Monte Carlo and variational inference: Bridging the gap. In *International conference on machine learning*, pp. 1218–1226. PMLR, 2015.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *science*, 324 (5923):81–85, 2009.

Simon J Sheather. Density estimation. *Statistical science*, pp. 588–597, 2004.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Tony Tohme, Kevin Vanslette, and Kamal Youcef-Toumi. A generalized Bayesian approach to model calibration. *Reliability Engineering & System Safety*, 204:107141, 2020.

Tony Tohme, Dehong Liu, and Kamal Youcef-Toumi. GSR: A generalized symbolic regression approach. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL `https://openreview.net/forum?id=lheUXtDNvP`.

Tony Tohme, Mohsen Sadr, Kamal Youcef-Toumi, and Nicolas Hadjiconstantinou. MESSY Estimation: Maximum-entropy based stochastic and symbolic density estimation. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL `https://openreview.net/forum?id=Y2ru0LuQeS`.

Belinda Tzen and Maxim Raginsky. Neural stochastic differential equations: Deep latent Gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019.

Silviu-Marian Udrescu and Max Tegmark. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020.

Silviu-Marian Udrescu, Andrew Tan, Jiahai Feng, Orisvaldo Neto, Tailin Wu, and Max Tegmark. AI Feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity. *Advances in Neural Information Processing Systems*, 33:4860–4871, 2020.

Mojtaba Valipour, Bowen You, Maysum Panju, and Ali Ghodsi. Symbolicgpt: A generative transformer model for symbolic regression. *arXiv preprint arXiv:2106.14131*, 2021.

Kevin Vanslette, Tony Tohme, and Kamal Youcef-Toumi. A general model validation and testing tool. *Reliability Engineering & System Safety*, 195:106684, 2020.

Marco Virgolin and Solon P Pissis. Symbolic regression is NP-hard. *arXiv preprint arXiv:2207.01018*, 2022.

Sven Wang and Youssef Marzouk. On minimax density estimation via measure transport. *arXiv preprint arXiv:2207.10231*, 2022.

Li Wenliang, Danica J Sutherland, Heiko Strathmann, and Arthur Gretton. Learning deep kernels for exponential family densities. In *International Conference on Machine Learning*, pp. 6737–6746. PMLR, 2019.

CJ Wild and GAF Seber. *Nonlinear regression*. New York: Wiley, 1989.

Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E Turner, José Miguel Hernández-Lobato, and Alexander L Gaunt. Deterministic variational inference for robust bayesian neural networks. In *International Conference on Learning Representations*, 2018.

Sihong Wu, Qinghua Huang, and Li Zhao. Fast Bayesian inversion of airborne electromagnetic data based on the invertible neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 61:1–11, 2023.

Wei Wu, Qinwei Fan, Jacek M Zurada, Jian Wang, Dakun Yang, and Yan Liu. Batch gradient method with smoothing l1/2 regularization for training of feedforward neural networks. *Neural Networks*, 50:72–78, 2014.

Zongben Xu, Hai Zhang, Yao Wang, XiangYu Chang, and Yong Liang. L 1/2 regularization. *Science China Information Sciences*, 53:1159–1169, 2010.

Caglar Yardim, Peter Gerstoft, and William S Hodgkiss. Geoacoustic and source tracking using particle filtering: Experimental results. *The Journal of the Acoustical Society of America*, 128(1):75–87, 2010.

Hengzhe Zhang, Aimin Zhou, Hong Qian, and Hu Zhang. PS-Tree: A piecewise symbolic regression tree. *Swarm and Evolutionary Computation*, 71:101061, 2022.

Xin Zhang and Andrew Curtis. Bayesian geophysical inversion using invertible neural networks. *Journal of Geophysical Research: Solid Earth*, 126(7):e2021JB022320, 2021.

A Invertible symbolic expressions recovered by ISR for the considered distributions

Table 1: Invertible symbolic expressions recovered by our ISR method for density estimation of several distributions (see Figure 4) in Section 4.1. Here, MoGs denotes "Mixture of Gaussians."

Example	Expression
Gaussian	$\mathbf{u} = \mathbf{x}$, i.e. $u_1 = x_1,\ u_2 = x_2$ $s_1(u_2) = 1.16$ $t_1(u_2) = 0$ $v_1 = u_1 \cdot \exp\big(s_1(u_2)\big) + t_1(u_2)$ $v_2 = u_2$ $s_2(v_1) = 1.14$ $t_2(v_1) = -9.39$ $o_1 = v_1$ $o_2 = v_2 \cdot \exp\big(s_2(v_1)\big) + t_2(v_1)$ $\mathbf{z} = \mathbf{o}$, i.e. $z_1 = o_1,\ z_2 = o_2$
Banana	$\mathbf{u} = \mathbf{x}$, i.e. $u_1 = x_1,\ u_2 = x_2$ $s_1(u_2) = 0.52 \sin(1.86\, u_2) + 0.084 \sin(5.5 \sin(1.86\, u_2) + 2.55) + 1.7$ $t_1(u_2) = 0.74 - 0.12 \sin(3.65 \sin(2.45\, u_2) - 0.82)$ $v_1 = u_1 \cdot \exp\big(s_1(u_2)\big) + t_1(u_2)$ $v_2 = u_2$ $s_2(v_1) = 1.72\,(0.025 - 0.47 \sin(0.33\, v_1))\,(0.23 \sin(0.33\, v_1) - 0.29) + 2.24$ $t_2(v_1) = -3.74\,(0.022 \sin(0.62\, v_1) + 0.035 \sin(0.63\, v_1) - 0.76)\,(0.45 \sin(0.62\, v_1) + 0.7 \sin(0.63\, v_1) + 0.38) + 0.027$ $o_1 = v_1$ $o_2 = v_2 \cdot \exp\big(s_2(v_1)\big) + t_2(v_1)$ $\mathbf{z} = \mathbf{o}$, i.e. $z_1 = o_1,\ z_2 = o_2$
Ring	$\mathbf{u} = \mathbf{x}$, i.e. $u_1 = x_1,\ u_2 = x_2$ $s_1(u_2) = -3.14\,\big(-0.15\, u_2^2 - 0.26 \sin(1.26\, u_2) + 0.094 \sin(3.25\, u_2) - 0.3\big)\,\big(0.098\, u_2^2 + 0.39 \sin(1.26\, u_2) + 0.014 \sin(3.25\, u_2) + 0.16\big)$ $+0.09 \sin\big(0.17\, u_2^2 + 0.69 \sin(1.26\, u_2) + 0.76 \sin(3.25\, u_2) + 0.25\big) - 0.30 \sin\big(0.94\, u_2^2 + 2.35 \sin(1.26\, u_2) - 1.31 \sin(3.25\, u_2) + 1.57\big) + 0.053$ $t_1(u_2) = -0.012$ $v_1 = u_1 \cdot \exp\big(s_1(u_2)\big) + t_1(u_2)$ $v_2 = u_2$ $s_2(v_1) = 0.037 \sin(0.22 \sin(0.22\, v_1) + 0.22 \sin(0.22\, v_1) - 0.27) + 0.052 \sin(0.23 \sin(0.22\, v_1) + 0.23 \sin(0.22\, v_1) - 0.39) - 0.11$ $t_2(v_1) = 0.13 \sin(0.13 \sin(0.37\, v_1) + 0.13 \sin(0.59\, v_1) - 1.16) + 0.65 \sin(0.021\, v_1 + 1.34 \sin(0.37\, v_1) + 2.29 \sin(0.59\, v_1) + 1.44) + 0.33$ $o_1 = v_1$ $o_2 = v_2 \cdot \exp\big(s_2(v_1)\big) + t_2(v_1)$ $\mathbf{u} = \mathbf{o}$, i.e. $u_1 = o_1,\ u_2 = o_2$ $s_1(u_2) = -3.65\,(-0.47 \sin(1.38\, u_2) - 0.026 \sin(1.93\, u_2) - 0.1)\,(-0.014 \sin(1.38\, u_2) - 0.39 \sin(1.93\, u_2) - 0.35) - 0.44 \sin(1.74 \sin(1.38\, u_2) + 1.71 \sin(1.93\, u_2) + 5.55)$ $-0.46 \sin(3.31 \sin(1.38\, u_2) + 0.44 \sin(1.93\, u_2) - 0.7) + 0.38$ $t_1(u_2) = 0.11 \sin(0.85 \sin(1.38\, u_2) + 0.86 \sin(1.38\, u_2)) + 0.12 \sin(0.87 \sin(1.38\, u_2) + 0.87 \sin(1.38\, u_2)) + 0.053$ $v_1 = u_1 \cdot \exp\big(s_1(u_2)\big) + t_1(u_2)$ $v_2 = u_2$ $s_2(v_1) = 0.25\, v_1^2 - 0.0071 \sin(1.67\, v_1) - 0.1 \sin(5.29\, v_1) - 0.63 \sin\big(-1.075\, v_1^2 + 0.55 \sin(1.67\, v_1) + 0.77 \sin(5.29\, v_1)\big)$ $+0.46 \sin\big(1.89\, v_1^2 - 0.27 \sin(1.67\, v_1) - 1.69 \sin(5.29\, v_1) + 0.9\big) + 1.22$ $t_2(v_1) = 0.62 \sin(1.56 \sin(0.43\, v_1) + 1.57 \sin(0.43\, v_1) - 1.68) + 0.61 \sin(1.56 \sin(0.43\, v_1) + 1.57 \sin(0.43\, v_1) - 1.68) - 0.48$ $o_1 = v_1$ $o_2 = v_2 \cdot \exp\big(s_2(v_1)\big) + t_2(v_1)$ $\mathbf{z} = \mathbf{o}$, i.e. $z_1 = o_1,\ z_2 = o_2$
MoG	$\mathbf{u} = \mathbf{x}$, i.e. $u_1 = x_1,\ u_2 = x_2$ $s_1(u_2) = -0.039\,(-0.032 \sin(1.44\, u_2) - \sin(1.48\, u_2) + 0.94)^2 - 3.03\,(0.18 \sin(1.44\, u_2) + 0.14 \sin(1.48\, u_2) - 0.63)\,(0.26 \sin(1.44\, u_2) + 0.28 \sin(1.48\, u_2) - 0.31)$ $+0.047 \sin(1.44\, u_2) + 0.13 \sin(1.48\, u_2) - 0.029 \sin(0.15 \sin(1.44\, u_2) + 1.4 \sin(1.48\, u_2) - 3.47) - 0.11 \sin(0.45 \sin(1.44\, u_2) + 1.46 \sin(1.48\, u_2) + 2.65) - 0.17$ $t_1(u_2) = 0.052 - 0.12 \sin(1.13 \sin(3.14\, u_2) + 1.64)$ $v_1 = u_1 \cdot \exp\big(s_1(u_2)\big) + t_1(u_2)$ $v_2 = u_2$ $s_2(v_1) = 0.34\,(0.15 \sin(1.024\, v_1) + 0.13 \sin(2.022\, v_1)) \sin(2.022\, v_1) + 0.014 \sin^2(2.022\, v_1) + 0.15 \sin(0.98\, v_1 \sin(1.024\, v_1) + 1.9 \sin(2.022\, v_1) - 1.41)$ $-0.22 \sin(3.38 \sin(1.024\, v_1) + 1.83 \sin(2.022\, v_1) + 1.5) + 0.39$ $t_2(v_1) = -0.46 \sin(0.89\, v_1 + 0.21 \sin(1.38\, v_1) - 1.56) + 0.33 \sin(1.23\, v_1 + 1.69 \sin(1.38\, v_1) - 0.44 \sin(1.58\, v_1) + 1.75\, v_1) + 0.094$ $o_1 = v_1$ $o_2 = v_2 \cdot \exp\big(s_2(v_1)\big) + t_2(v_1)$ $\mathbf{u} = \mathbf{o}$, i.e. $u_1 = o_1,\ u_2 = o_2$ $s_1(u_2) = 3.36\,(-0.36 \sin(3.1\, u_2) - 0.29)\,(0.26 \sin(3.1\, u_2) + 0.19) - 0.45 \sin(3.88 \sin(3.1\, u_2) - 1.83) - 0.38$ $t_1(u_2) = 0$ $v_1 = u_1 \cdot \exp\big(s_1(u_2)\big) + t_1(u_2)$ $v_2 = u_2$ $s_2(v_1) = 0.0035\, v_1^2 - 2.88\,\big(-0.079\, v_1^2 - 0.33\big)\,\big(0.042\, v_1^2 + 0.4\big) + 2.6\,(-0.34 \sin(1.7\, v_1) - 0.088 \sin(1.71\, v_1))\,(-0.053 \sin(1.7\, v_1) - 0.38 \sin(1.71\, v_1)) + 1.61$ $t_2(v_1) = -0.14$ $o_1 = v_1$ $o_2 = v_2 \cdot \exp\big(s_2(v_1)\big) + t_2(v_1)$ $\mathbf{z} = \mathbf{o}$, i.e. $z_1 = o_1,\ z_2 = o_2$

B Details of network architectures

In our experiments, we train all models using Adam optimizer with a dynamic learning rate decaying from 10^{-2} to 10^{-4}. In addition, for INN, all (hidden) neurons of the subnetworks are followed by Leaky ReLU activations, while for ISR, each neuron in the hidden layers is followed by an activation function from the following library:

$$\left\{1, \mathrm{id}, \bullet^2(\times 4), \sin(2\pi\bullet), \sigma, \bullet_1 \times \bullet_2\right\}$$

where 1 represents the constant function, "id" is the identity operator, σ denotes the sigmoid function, and $\bullet$ denotes placeholder operands, e.g. $\bullet^2$ corresponds to the square operator. Also, $\bullet_1 \times \bullet_2$ denotes the multiplication operator, and each activation function may be duplicated within each layer.

The architecture details are provided below.

Density estimation via normalizing flow. In Section 4.1, we train INN and ISR using a batch size of 64. For the "Gaussian" and "Banana" distributions, we adopt 1 affine coupling block with 2 fully connected (hidden) layers per subnetwork. For the "Ring" and "Mixture of Gaussians (MoGs)" distributions, we use 2 invertible blocks with 2 fully connected layers for each subnetwork.

Inverse Kinematics. In Section 4.2, we train all models using a batch size of 100. For all models, we adopt 6 reversible blocks with 3 fully connected layers per subnetwork.

Geoacoustic Inversion. In Section 4.3, we train all models using a batch size of 200. For all models, we adopt 5 invertible blocks with 4 fully connected layers for each subnetwork.

C Quantitative Evaluation – Inverse Kinematics

Here, we quantitatively evaluate the quality of the estimated posteriors by the different models considered in the inverse kinematics experiment. To ensure a fair comparison among all methods, we use the same training data and train all models for the same number of epochs, and using identical batches and architectures (as provided in the previous section).

As suggested in (Kruse et al., 2021), we evaluate the correctness of the estimated posteriors using two metrics. First, we use the Maximum Mean Discrepancy (MMD) introduced by Gretton et al. (2012), which computes the *posterior mismatch* between the distribution $\hat{p}(\mathbf{x} \,|\, \mathbf{y}^*)$ produced by a model and a ground truth estimate $p_{\mathrm{gt}}(\mathbf{x} \,|\, \mathbf{y}^*)$, which in this case is obtained via rejection sampling (see Section 4.2), i.e.

$$\mathrm{Err}_{\mathrm{post}} = \mathrm{MMD}\big(\hat{p}(\mathbf{x} \,|\, \mathbf{y}^*), p_{\mathrm{gt}}(\mathbf{x} \,|\, \mathbf{y}^*)\big) \tag{C.1}$$

Second, we measure the *re-simulation error*, which applies the true forward process f in Eq. (20) to the generated samples $\mathbf{x}$ and computes the mean squared distance to the target $\mathbf{y}^*$, i.e.

$$\mathrm{Err}_{\mathrm{resim}} = \mathbb{E}_{\mathbf{x} \sim \hat{p}(\mathbf{x} \,|\, \mathbf{y}^*)} \left[||f(\mathbf{x}) - \mathbf{y}^*||_2^2 \right] \tag{C.2}$$

Table 2: Quantitative results for the inverse kinematics benchmark experiment.

Method	$\mathrm{Err}_{\mathrm{post}}$	$\mathrm{Err}_{\mathrm{resim}}$
INN	0.0259	0.0163
cINN	0.0162	0.0087
ISR	0.0286	0.0196
cISR	0.0221	0.0134