A FRIENDLY INTRODUCTION TO TRIANGULAR TRANSPORT

Maximilian Ramgraber
Delft University of Technology
Delft, Netherlands
m.ramgraber@tudelft.nl

Daniel Sharp
Massachusetts Institute of Technology
Cambridge, USA
dannys4@mit.edu

Mathieu Le Provost
Massachusetts Institute of Technology
Cambridge, USA
mleprovo@mit.edu

Youssef Marzouk
Massachusetts Institute of Technology
Cambridge, USA
ymarz@mit.edu

March 28, 2025

ABSTRACT

Decision making under uncertainty is a cross-cutting challenge in science and engineering. Most approaches to this challenge employ probabilistic representations of uncertainty. In complicated systems accessible only via data or black-box models, however, these representations are rarely known. We discuss how to characterize and manipulate such representations using *triangular transport maps*, which approximate any complex probability distribution as a transformation of a simple, well-understood distribution. The particular structure of triangular transport guarantees many desirable mathematical and computational properties that translate well into solving practical problems. Triangular maps are actively used for density estimation, (conditional) generative modelling, Bayesian inference, data assimilation, optimal experimental design, and related tasks. While there is ample literature on the development and theory of triangular transport methods, this manuscript provides a detailed introduction for scientists interested in employing measure transport without assuming a formal mathematical background. We build intuition for the key foundations of triangular transport, discuss many aspects of its practical implementation, and outline the frontiers of this field.

1 Motivation

Who is this tutorial for? This manuscript is an accessible introduction to *triangular transport*, a powerful and versatile method for generative modelling and Bayesian inference. In particular, triangular transport underpins effective algorithms for data assimilation, solving inverse problems, and performing simulation-based inference, with applications across myriad scientific disciplines.

This tutorial targets researchers with an interest in applied statistical methods but without a formal background in mathematics. Consequently, we will focus more on intuition, general concepts, and implementation, referring the reader to other relevant articles for more formal exposition and theory.

How does triangular transport work? Like other measure transport methods, triangular (measure) transport is a framework to transform one probability distribution into another. This operation is highly useful, as it allows us to characterize a complex target distribution π by transforming a simpler, known reference distribution η. Throughout this manuscript, we use orange to denote variables associated with the (problem-specific) target distribution and green to denote variables associated with the (user-defined) reference distribution, e.g., a standard Gaussian. The idea of coupling two distributions is used in a wide range of applications. In *generative modelling*, for example, we are usually interested in creating samples of a target distribution π, such as the distribution of 400×400 pixel images of cats. Measure transport methods approach this challenge by first learning a transport map $\mathbf{S}$ that transforms η to π, and then use $\mathbf{S}$ to convert reference samples $\mathbf{z} \sim \eta$ (here: 400×400 pixel images of Gaussian white noise) into samples from the target distribution $\mathbf{x} = \mathbf{S}(\mathbf{z}) \sim \pi$ (here: 400×400 pixel images of cats).

Some of these methods – among them triangular transport – can also characterize *conditionals* $\pi(\boldsymbol{a}|\boldsymbol{b}^*)$ of the *joint* target distribution $\pi(\boldsymbol{a}, \boldsymbol{b})$ of two random variables $\boldsymbol{a}$ and $\boldsymbol{b}$. Here blue denotes the fact that $\boldsymbol{b}^*$ is a deterministic, fixed value. Conditioning operations often arise as stochastic generalizations of evaluating deterministic processes (see Figure 1). As we will describe in Section 2, conditioning is also central to the Bayesian approach to statistical inference, which is an important tool across many scientific disciplines. We distinguish here between generating from $\pi(\boldsymbol{a}|\boldsymbol{b}^*)$ ("generate an image of a grey cat") and $\pi(\boldsymbol{a}, \boldsymbol{b})$ ("generate a color and a cat of that color") by assuming that $\boldsymbol{b}^*$ is determined outside of our control, either by user or application.

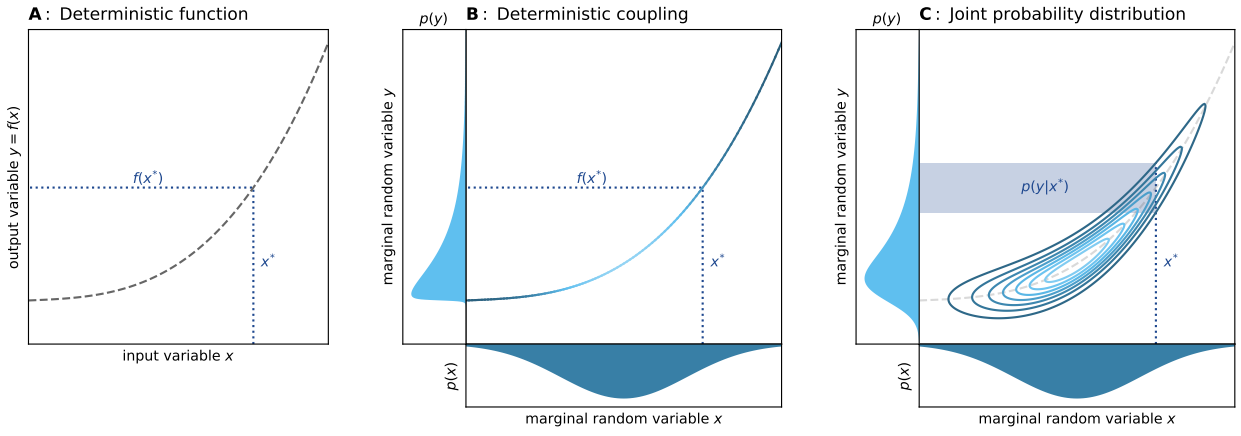

Figure 1: Progression from a fully deterministic to a fully stochastic system. (A) Numerical models are usually represented as deterministic functions. (B) In the presence of input uncertainty, deterministic functions encode a deterministic coupling which yields uncertain output (see Section 2.2). (C) If the function itself is uncertain, this coupling "blurs" into a joint probability distribution. Function evaluation now corresponds to characterizing a conditional distribution. Mind that (A) and (B) can also be parsed as degenerate joint probability distributions.

What makes triangular transport special? At the heart of triangular transport is their eponymous *triangular* structure. This structure sets them apart from other measure transport methods such as normalizing flows (e.g., Rezende and Mohamed, 2015; Kobyzev et al., 2021), which compose together many simpler but somewhat ad hoc transformations, often interleaved with permutations, and even from conditional normalizing flows (e.g., Van Den Oord et al., 2016), which parameterize normalizing flows in order to represent block triangular (rather than strictly triangular) maps. Triangular structure – discussed in greater detail in subsequent sections – has many important practical properties:

- **Parsimony**: The parameterization of the map function is at the user's discretion (see Section 3.1). This means we can adjust the map's overall complexity, down to the complexity with which it resolves individual variables and variable dependencies. This allows us to implement nonlinear maps that are as complex as necessary, and yet as simple as possible (see Section 4.2).
- **Sparsity**: Triangular maps have a natural ability to exploit *conditional independence*. This improves their computational efficiency, which enables these maps to scale to high-dimensional settings. Further, such structure makes them highly robust to spurious correlations and smaller sample sizes (see Section 2.3.3).
- **Numerical convenience**: Constructing triangular maps boils down to parameterizing simple one-dimensional monotone functions, a task with a rich body of supporting literature. Because of this, these maps are easy to optimize and invert, which we investigate in detail.
- **Explainability**: Triangular maps have a clear correspondence between their constituent elements and the statistical features they represent (see Section 2.3). We can then readily describe different factorizations of the target distribution using the elements of a triangular map, with a particular focus on combinations of various conditional and marginals of the target.

In what applications has triangular transport been successful? Triangular transport has been applied to a wide range of statistical problems in many different disciplines. Examples of such applications include:

- **Bayesian inference**: Triangular transport lends itself exceptionally well to the sampling of conditional distributions. As such, it has found application in both variational (El Moselhy and Marzouk, 2012) and simulation-based inference (Marzouk et al., 2017; Rubio et al., 2023; Baptista et al., 2024a), for large-scale inverse problems (Brennan et al., 2020) and in applications with multiscale structure (Parno et al., 2016).
- **Data assimilation**: Triangular transport provides true nonlinear generalizations of popular filtering (Spantini et al., 2022) and smoothing (Ramgraber et al., 2023a,b) algorithms such as the ensemble Kalman filter and smoother, and their many variants (Grange et al., 2024).
- **Density estimation and generative modelling**: The coupling learned by triangular transport is highly useful for the estimation (Wang and Marzouk, 2022; Martinez-Sanchez et al., 2024; López-Marrero et al., 2024) and sampling (Irons et al., 2022) of non-Gaussian probability distributions, even in high dimensions (Katzfuss and Schäfer, 2024).
- **Optimal experimental design**: Due to the close connections between conditional densities and expected information gain or mutual information, triangular transport maps are useful for estimating common objectives in Bayesian optimal experimental design (Huan et al., 2024; Koval et al., 2024; Li et al., 2024).

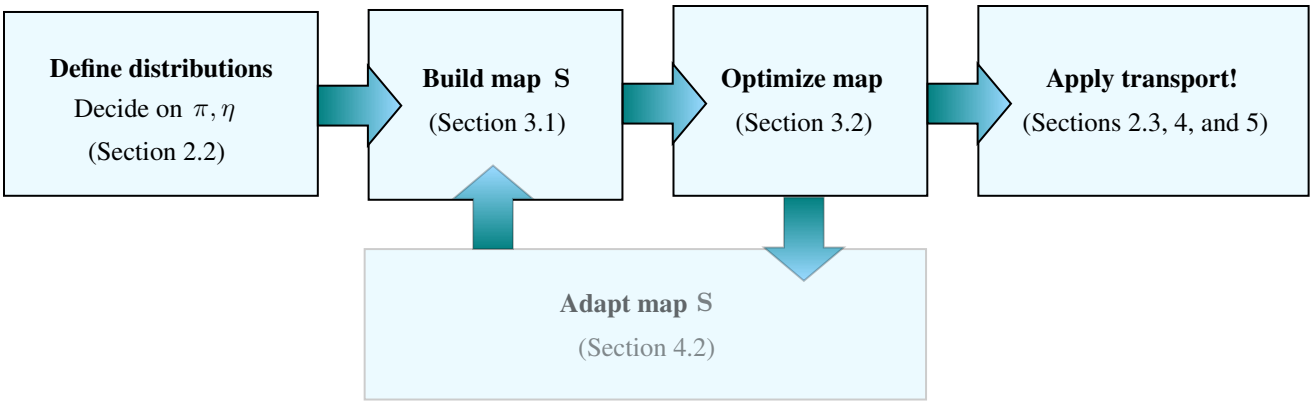

Figure 2: Flowchart of the main steps required to define, build, optimize, and apply triangular maps, with links to the relevant sections.

Further applications of triangular transport include methods for joint state-parameter inference in state-space models (Spantini et al., 2018; Grashorn et al., 2024; Zhao and Cui, 2024), solving Fokker–Planck equations (Zeng et al., 2023), stochastic programming (Backhoff et al., 2017), and even the discovery of causal models from data (Akbari et al., 2023; Xi et al., 2023).

How is this tutorial structured? In the following, we will provide an intuition-focused introduction to the theoretical basics of triangular transport (Section 2), discuss practical aspects related to their implementation in code (Section 3), and conclude with a brief overview of interesting research directions (Section 5). For researchers interested chiefly in practical implementation, a flow chart of the most important steps in the construction and application of triangular transport is provided in Figure 2. First, we define the target π and reference η (Section 2.3). Then, we structure, parameterize (Section 3.1), and optimize (Section 3.2) the triangular map. Finally, we can deploy the map in the application of our choice, with some practical heuristics listed in Section 4. The tutorial will involve several recurring variables, which are summarized in Table 1.

2 Theory

2.1 Bayesian inference

To begin, let us briefly revisit some basic concepts of Bayesian inference which will serve to motivate the operations explored in the following sections. In short, Bayesian inference is based on *Bayes' theorem*: given two random variables (RVs) $\mathbf{a}$, $\mathbf{b}$ with joint probability density function (pdf) $p(\mathbf{a}, \mathbf{b})$, we see

$$p\left(\mathbf{a}|\mathbf{b}^*\right) = \frac{p\left(\mathbf{a}\right) p\left(\mathbf{b}^*|\mathbf{a}\right)}{p\left(\mathbf{b}^*\right)}, \tag{1}$$

Equation (1)subsumes three sequential operations (see Figure 3; e.g., Gelman et al., 2013):

1. First, the marginal prior $p\left(\mathbf{a}\right)$ is combined with a conditional observation model (sometimes also called *likelihood model*) $p\left(\mathbf{b}|\mathbf{a}\right)$, yielding a joint probability distribution $p\left(\mathbf{a}, \mathbf{b}\right) = p\left(\mathbf{a}\right) p\left(\mathbf{b}|\mathbf{a}\right)$ over all possible

Table 1: Recurring notation and variables.

bold font	vector-valued variable or function	Roman font	scalar-valued variable or function
$\mathbf{S}$	Target-to-reference map	S_k	k-th map component function
$\mathbf{R}$	Reference-to-target map (see Section 3.2.2)	τ	twice Archimedes' constant, i.e., $6.283185...$
π	target distribution of interest	η	reference distribution, often standard Gaussian
orange	variable associated with π	green	variable associated with η
$\mathbf{x} \sim \pi$	target random variable	$\mathbf{z} \sim \eta$	reference random variable
$\mathbf{X}^i$	ith realization of $\mathbf{x} \sim \pi$	$\mathbf{Z}^i$	$i$the realization of $\mathbf{z} \sim \eta$
$S^\sharp \eta$	pullback distribution	$S_\sharp \pi$	pushforward distribution
K	number of target dimensions	N	ensemble size
p	generic probability density function (pdf)	$\mathbf{a}, \mathbf{b}$	generic random variables
$\mathbf{y}^*$	conditioning variable	$\mathbf{x}^*$	conditioned variable $\mathbf{x}^* \sim p(\mathbf{x}\|\mathbf{y}^*)$
c	basis function coefficient	r	rectifier ($r : \mathbb{R} \to \mathbb{R}^+$)
f	monotone function	g	nonmonotone function

combinations of $\mathbf{a}$ and $\mathbf{b}$. This joint distribution describes how the variable of interest $\mathbf{a}$ and the predicted observations $\mathbf{b}$ relate to each other.

2. Next, this joint distribution is conditioned on a specific observation $\mathbf{b}^*$. In practice, this means evaluating $p(\mathbf{a}, \mathbf{b})$ for all possible values $\mathbf{a}$ while keeping $\mathbf{b}$ fixed at the value of $\mathbf{b}^*$. This extracts a slice $p(\mathbf{a}, \mathbf{b}^*)$ of this joint distribution at $\mathbf{b}^*$ along different values of $\mathbf{a}$.

3. Since the probability densities along this slice do not generally integrate to 1, this slice does not constitute a valid pdf. The final step thus normalizes the probability densities against the slice's probability mass $p(\mathbf{b}^*) = \int p(\mathbf{t}, \mathbf{b}^*)\, d\mathbf{t}$, yielding the posterior pdf $p(\mathbf{a}|\mathbf{b}^*)$.

In summary, Bayes' theorem first constructs a joint distribution $p(\mathbf{a}, \mathbf{b})$ from a prior $p(\mathbf{a})$ and an observation model $p(\mathbf{b}|\mathbf{a})$, then conditions it on a specific observation value $\mathbf{b}^*$ and re-normalizes. In consequence, one could reformulate Equation (1) equivalently as:

$$p(\mathbf{a}|\mathbf{b}^*) = \frac{p(\mathbf{a}, \mathbf{b}^*)}{\int p(\mathbf{t}, \mathbf{b}^*)\, d\mathbf{t}}, \tag{2}$$

where $p(\mathbf{a}, \mathbf{b}^*)$ evaluates the joint pdf $p(\mathbf{a}, \mathbf{b})$ for all possible $\mathbf{a}$ while keeping $\mathbf{b}$ fixed at $\mathbf{b}^*$, and the denominator acts as a normalizing constant. This equation, or reformulation thereof, lie at the heart of all Bayesian inference algorithms. Unfortunately, it is generally impossible to formulate $p(\mathbf{a}, \mathbf{b})$ in closed form, which in turn makes it difficult to evaluate the posterior $p(\mathbf{a}|\mathbf{b}^*)$. To overcome this challenge, different Bayesian inference methods use different strategies. As we shall see in the following, triangular transport solves this challenge by first approximating an almost arbitrary joint pdf $p(\mathbf{a}, \mathbf{b})$ by using the concept of measure transport (Section 2.2.1). Crucially, this transformation then allows us to evaluate any of its conditionals $p(\mathbf{a}|\mathbf{b}^*)$ (Section 2.3.2).

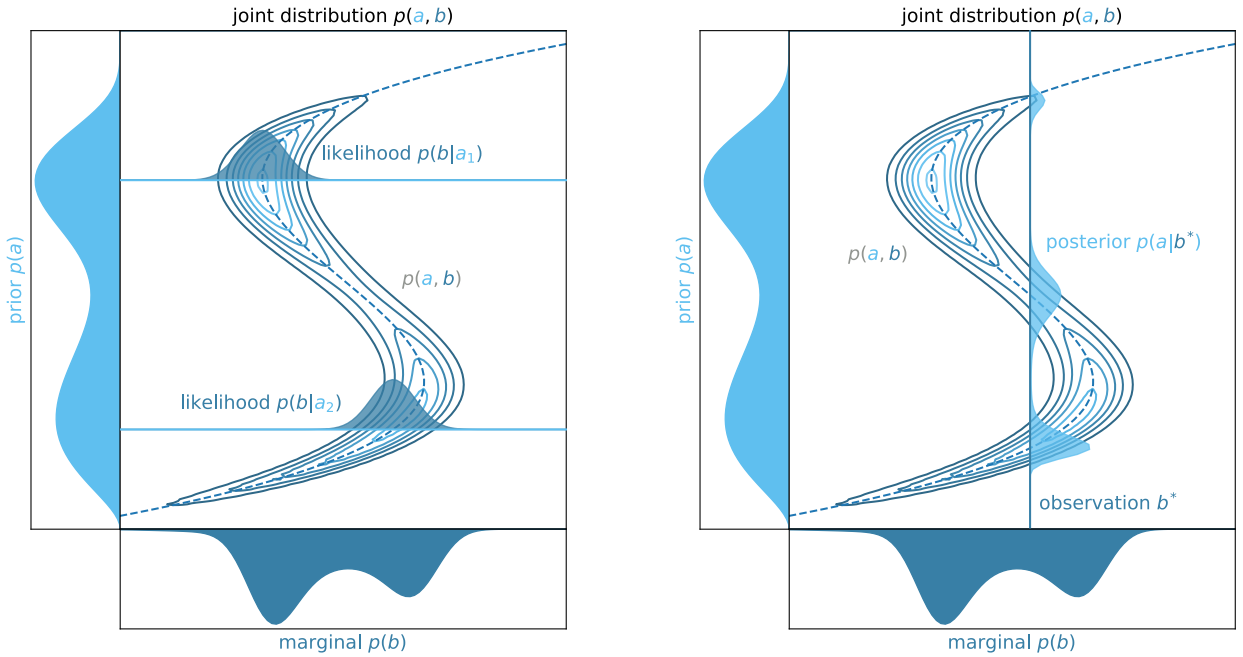

Figure 3: Schematic illustration of Bayes theorem, using a Beta mixture prior $p(a)$ and an observation model $p(b|a) = \mathcal{N}\left(\mu = 2a^3 - a, \sigma = 0.075\right)$. Left: We can create a joint distribution $p(a, b)$ from a prior $p(a)$ and the observation model $p(b|a)$. Right: conditioning this joint distribution on a specific value b^* retrieves a posterior $p(a|b^*)$.

The remainder of this tutorial drops this generic notation for two RVs $\mathbf{a}$ and $\mathbf{b}$ jointly distributed as $p(\mathbf{a}, \mathbf{b})$ in favour of a single RV $\mathbf{x}$ distributed according to a distribution π. You can think of $\mathbf{x}$ as an augmented RV $\mathbf{x} = [\mathbf{b}, \mathbf{a}]$, and of the joint density as $\pi(\mathbf{x}) = p(\mathbf{a}, \mathbf{b})$. Transport methods generally operate on this joint distribution $p(\mathbf{a}, \mathbf{b})$; we focus primarily on the ones that will allow us to characterize the desired conditionals $p(\mathbf{a}|\mathbf{b})$.

2.2 The change-of-variables formula

The key to understand transport methods is the *change-of-variables formula*. This formula allows us to relate a RV $\mathbf{x}$ associated with a complicated target pdf π, known only to proportionality or through samples, to a second RV $\mathbf{z}$, associated with a much simpler, user-specified reference pdf η through an invertible, differentiable transformation $\mathbf{S}$. In essence, the change-of-variables formula describes what happens to pdfs when they are subjected to specific transformations. For scalar-valued RVs x and z, the change-of-variables formula is defined as

$$\begin{aligned} \pi(x) &= S^\sharp \eta(x) = \eta(S(x)) \left| \frac{\partial S(x)}{\partial x} \right|, \\ \eta(z) &= S_\sharp \pi(z) = \pi(S^{-1}(z)) \left| \frac{\partial S^{-1}(z)}{\partial z} \right|, \end{aligned} \tag{3}$$

where $z = S(x)$ and $x = S^{-1}(z)$, and the *pullback density*[1] $S^\sharp \eta(x)$ obtains π by applying the inverse map S^{-1} to the reference distribution η. The alternate form in the second line of Equation (3) reflects the fact that since S is invertible,

[1]The *pullback* density $S^\sharp \eta(\mathbf{x})$ is the result of applying of the *inverse* map S^{-1} to a RV $z \sim \eta$.

each distribution π and η can be expressed in terms of the other through either the (forward) map S or its inverse S^{-1}. Consequently, the *pushforward pdf*[2] $S_\sharp\pi(z)$ likewise approximates η by applying the forward map S to the target[3] π. For multivariate RVs $\mathbf{x}$ and $\mathbf{z}$, the same principle applies. Given a multivariate monotone function $\mathbf{S}$, Equation (3) generalizes to:

$$\begin{aligned}\pi(\mathbf{x}) &= \mathbf{S}^\sharp\eta(\mathbf{x}) = \eta(\mathbf{S}(\mathbf{x}))\,|\det\nabla_{\mathbf{x}}\mathbf{S}(\mathbf{x})|\\ \eta(\mathbf{z}) &= \mathbf{S}_\sharp\pi(\mathbf{z}) = \pi(\mathbf{S}^{-1}(\mathbf{z}))\,\left|\det\nabla_{\mathbf{z}}\mathbf{S}^{-1}(\mathbf{z})\right|\end{aligned} \tag{4}$$

The change-of-variables formula in Equation (3) has a surprisingly intuitive interpretation. It states that the probability density $\eta(z)$ at a point of interest z after the transformation equals the original probability density $\pi(x)$ at the pre-transformation point $x = S^{-1}(z)$, adjusted for any deformation the transformation might have induced at this location $|\partial S(x)/\partial x|$. Equation (4) extends this notion to multi-dimensional systems. Intuitively, the absolute value of the determinant of the Jacobian of $\mathbf{S}$, namely $|\det\nabla_{\mathbf{x}}\mathbf{S}(\mathbf{x})|$, measures the inflation/deflation of an infinitesimal volume centered about $\mathbf{x}$ by the map $\mathbf{S}$. If $|\det\nabla_{\mathbf{x}}\mathbf{S}(\mathbf{x})| = 1$, the map $\mathbf{S}$ preserves infinitesimal volumes about $\mathbf{x}$. The compensation term, similar to 'u-substitution' in calculus, is necessary because transformations $\mathbf{S}$ "stretch" or "squeeze" the spaces they are applied to. Accounting for this spatial distortion ensures that the probability mass is preserved and thus the transformed distribution $\eta(\mathbf{z})$ remains a valid pdf.

The change-of-variables formula allows us to describe how a probability distribution π changes when subjected to a specific transformation $\mathbf{S}$. An example is provided in Figure 4, which shows how a non-Gaussian target pdf π can be related to a Gaussian reference pdf η through an invertible transformation; this invertibility is equivalent to monotonicity in the scalar case.

2.2.1 Connection to transport methods

The change-of-variables formula has three knobs to tune: the original distribution π, the map $\mathbf{S}$, and its transformed output η. When considering the change-of-variables in elementary calculus courses, π and $\mathbf{S}$ are often assumed known, and we seek its transformed output η. Transport methods choose a slightly different approach. We assume π and η to be known (at least partially), and instead seek the specific map $\mathbf{S}$ which relates the two distributions to each other.

As discussed in Section 1, we generally do not know the target distribution π in closed form. Often, the target density π is known only partially, either through samples[4] $\mathbf{X} \sim \pi$ or up to proportionality ($\tilde{\pi} = m\pi$, where $m > 0$ is an unknown constant)[5]. On the other hand, the reference distribution η is defined as a simple, well-known distribution, often a standard Gaussian pdf $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ (Figure 5). The map $\mathbf{S}$ is identified by minimizing an objective function over a specified class of functions, see the discussion in Section 3.2. By finding this map, we learn how to construct the unknown target distribution π by transforming a well-defined reference distribution η.

[2]The *pushforward* density $S_\sharp\pi(\mathbf{z})$ is the result of applying the *forward* map S to a RV $x \sim \pi$.

[3]A mnemonic bridge to remember the $\sharp$ notation is that the pull*back* relies on the *inverse* map $\mathbf{S}^{-1}$ to sample, and has the $\sharp$ in the superscript where the -1 would be.

[4]Sample approximations are common if π is only known from a dataset or if the prior can be sampled but not evaluated.

[5]Unnormalized densities $\tilde{\pi}$ can arise in, e.g., Bayesian statistics, when the prior and likelihood can be evaluated at least point-wise, but it is infeasible to quantify the model evidence (see Equation (1)).

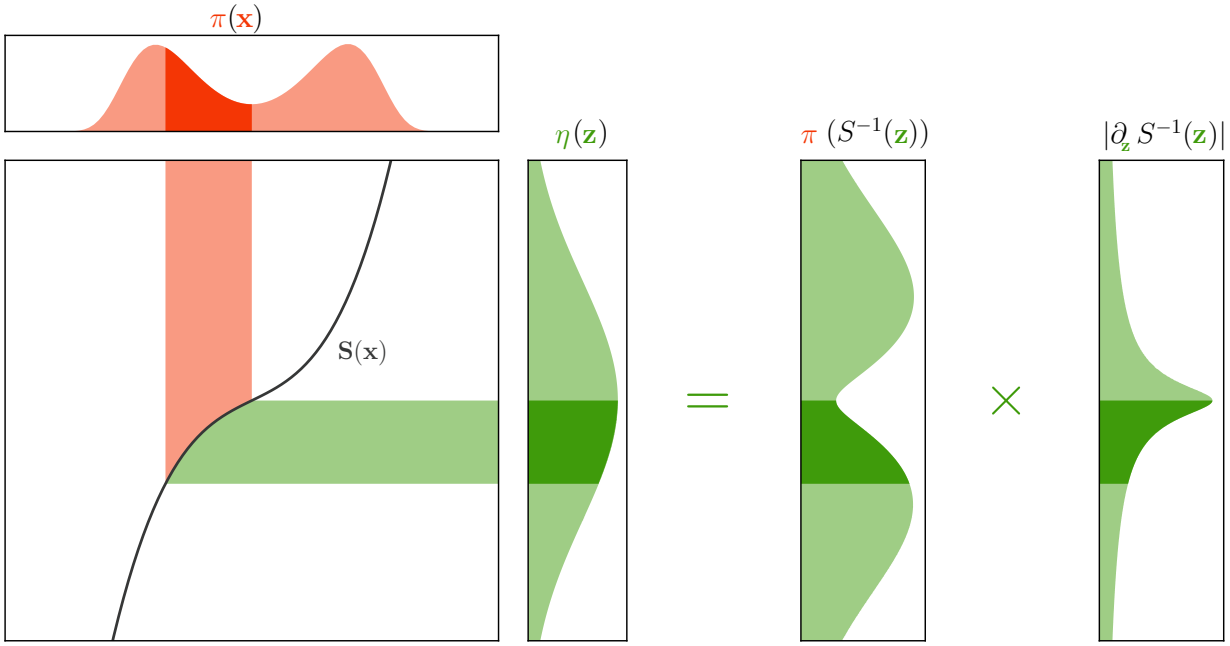

Figure 4: Illustration of the change-of-variables formula. A monotone function S allows us to relate a RV x associated with a pdf π to a RV z associated with a pdf η. We can evaluate $\eta(z)$ by evaluating the target π at the pre-image of z, that is to say $\pi(S^{-1}(z))$, then multiplying it with the absolute inverse map's gradient $|\partial_z S^{-1}(z)|$.

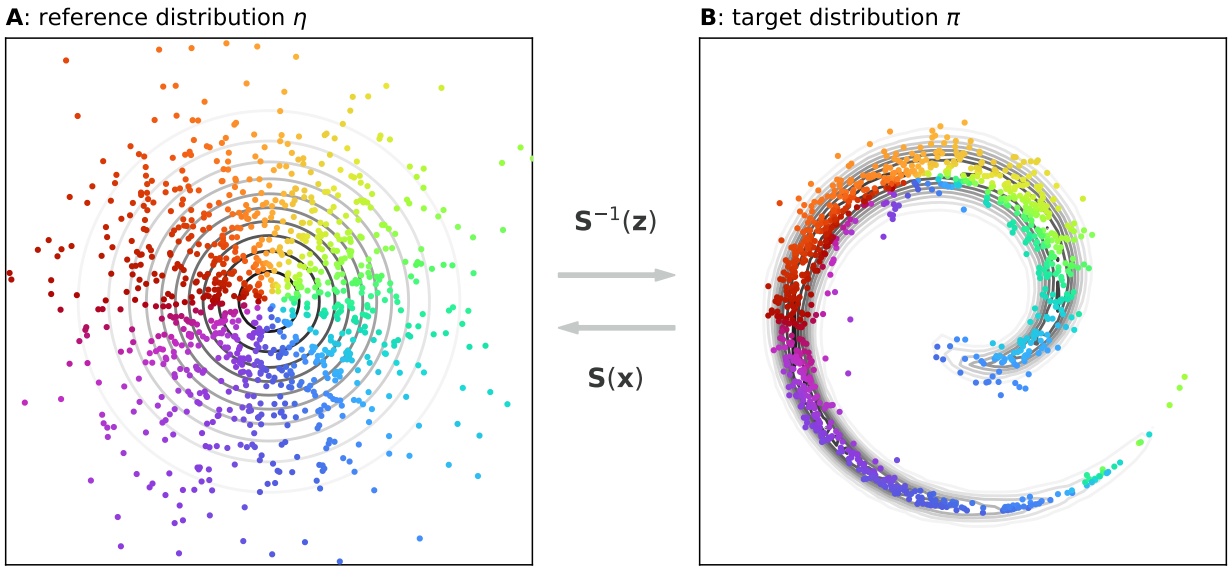

Figure 5: A transport map $\mathbf{S}$ relates a RV $\mathbf{x}$ associated with the target π to a RV $\mathbf{z}$ associated with the reference η. If the map is monotone, it can be applied both ways.

Among its other uses, learning the map $\mathbf{S}$ allows us to cheaply draw new samples from the target distribution π. This is achieved by first sampling the reference η, then applying the inverse map $\mathbf{S}^{-1}$ to the resulting reference samples $\mathbf{z}$. This is especially useful in applications where sampling the target conventionally would involve computationally expensive expensive simulations of, e.g., partial differential equations (*emulation*), or systems in which the sample-generating process is not known exactly (*generative modelling*: e.g., Baptista et al., 2024c).

2.3 Triangular maps and their uses

Many classes of functions are viable choices for the transport map $\mathbf{S}$. Common examples include normalizing flows and GANs. However, an especially useful class among these are *triangular* transport maps. In addition to sampling the target π, triangular maps are unique in allowing us to sample *conditionals* of π, which makes them a flexible and versatile tool for Bayesian inference. Triangular maps are structured as follows:

$$\mathbf{S}(\mathbf{x}) = \begin{bmatrix} S_1(x_1) \\ S_2(x_1, x_2) \\ \vdots \\ S_K(x_1, \dots, x_K) \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_K \end{bmatrix} = \mathbf{z}, \tag{5}$$

where the full map $\mathbf{S} : \mathbb{R}^K \to \mathbb{R}^K$ is comprised of map components $S_k : \mathbb{R}^k \to \mathbb{R}$, $k = 1, \dots, K$, each of which depends only on the first k entries of the target RV $\mathbf{x} = [x_1, \dots, x_K]^\top$ and we *enforce* that $\partial_{x_k} S_k(x_1, \dots, x_k) > 0$ for any feasible choice of $x_1, \dots, x_k$. When all of $S_1, \dots, S_K$ satisfy this, we call the map $\mathbf{S}$ "monotone". The eponymous *triangular* nature of $\mathbf{S}$ refers to the fact that the map's partial derivatives with regards to $\mathbf{x}$ are lower-triangular; that is to say, the Jacobian matrix $\nabla\mathbf{S}$ has all zeros above its diagonal. This structure—also known as a *Knothe–Rosenblatt rearrangement* (Knothe, 1957; Rosenblatt, 1952) or KR map—has a number of highly desirable properties:

1. Over all functions satisfying this structure, there is one that uniquely couples π and η under mild conditions (e.g., Marzouk et al., 2017).

2. This triangular structure allows us to evaluate the **determinant of the map's Jacobian** $\det \nabla\mathbf{S}(\mathbf{x})$ efficiently as the product of its diagonal entries (Marzouk et al., 2017), which proves highly useful for the map's optimization (see Section 3.2), not to mention when performing the density estimation itself for the changed variables:

$$\det \nabla\mathbf{S}(\mathbf{x}) = \prod_{k=1}^{K} \frac{\partial S_k(x_1, \dots, x_k)}{\partial x_k}. \tag{6}$$

3. Triangular maps are **easily invertible**. In particular, we pick a class of functions that are monotone in their last input x_k (with no requirements on the other inputs $x_1, \dots, x_{k-1}$), i.e. $\partial_{x_k} S_k > 0$ everywhere; this guarantees the monotonicity and thus eases our inversion computation. We will discuss ways to guarantee this property in Section 3.1.1.

4. Perhaps most importantly, triangular maps naturally **factorize the target distribution** into a product of marginal conditional pdfs. We will investigate this property in greater detail in the following sections.

2.3.1 Map inversion

In the forward map evaluation (Equation (5)), each of the map's constituent map component functions S_k can be evaluated independently, even in parallel, and then assembled into the full reference vector $\mathbf{z}$. However, the same does not hold for the inverse map:

$$\mathbf{S}^{-1}(\mathbf{z}) = \begin{bmatrix} S_1^{-1}(z_1) \\ S_2^{-1}(z_2; x_1) \\ S_3^{-1}(z_2; x_1, x_2) \\ \vdots \\ S_K^{-1}(z_K; x_1, \dots, x_{K-1}) \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_K \end{bmatrix} = \mathbf{x}. \tag{7}$$

Here, the inverse map's *component functions* S_k^{-1} must be evaluated in sequence and cannot be evaluated independently. This process begins by inverting the first map component $S_1^{-1}(z_1)$, a trivial one-dimensional root finding problem[6], which yields x_1. This output x_1 serves as auxiliary input for the second map component's inversion $S_2^{-1}(z_2; x_1)$, yielding another one-dimensional root-finding problem, which provides x_2. All subsequent map component inversions are similar one-dimensional root-finding problems that likewise depend on the outcomes of previous inversions. This dependence of each inversion S_k^{-1} on each of $x_1, \dots, x_{k-1}$, the outcomes of previous inversions, effectively factorizes the target distribution as a product of marginal conditionals (Villani, 2007):

$$\pi(\mathbf{x}) = \underbrace{\pi(x_1)}_{S_1^{-1}(z_1)} \underbrace{\pi(x_2|x_1)}_{S_2^{-1}(z_2;x_1)} \underbrace{\pi(x_3|x_1,x_2)}_{S_3^{-1}(z_3;x_1,x_2)} \dots \underbrace{\pi(x_K|x_1,\dots,x_{K-1})}_{S_K^{-1}(z_K;x_1,\dots,x_{K-1})}, \tag{8}$$

where each term corresponds to, and is in turn sampled by, one of the inverse map components indicated in the underbraces[7]. In other words, for a particular sample i, each row of Equation (7) can be used to generate a sample X_k^i from a particular marginal distribution conditioned on $\mathsf{X}_1, \dots, \mathsf{X}_{k-1}$:

$$\mathsf{X}_k^i = S_k^{-1}(\mathsf{Z}_k^i; \mathsf{X}_1^i, \dots, \mathsf{X}_{k-1}^i) \sim S_k^\sharp \eta_k = \pi(x_k|x_1, \dots, x_{k-1}), \tag{9}$$

where $S_k^\sharp \eta_k$ is the pullback of the one-dimensional marginal reference η_k.

2.3.2 Sampling conditionals

As it turns out, the factorization of the target distribution π in Equations (8) and (9) also allows us to sample *conditionals* of π, including the Bayesian posterior $p(\mathbf{a}|\mathbf{b})$ (assuming $p := \pi$ and $[\mathbf{b}, \mathbf{a}] := [\mathbf{x}_{1:k}, \mathbf{x}_{k+1:K}]$). This can be achieved by manipulating the inversion process. First, observe that the factorization in Equation (8) can be aggregated into two blocks:

[6] This root finding problem has a single unique solution within the domain of S_k^{-1}, since we require S_k be monotone in x_k.

[7] This is only the case if the reference distribution η has no dependence between any of its marginals (Spantini et al., 2022), i.e., $\eta(\mathbf{z}) = \eta_1(z_1)\eta_2(z_2)\dots\eta_K(z_K)$; the standard Gaussian fulfills this property.

$$\pi(\mathbf{x}) = \underbrace{\pi(\mathbf{x}_{1:k})}_{\mathbf{S}_{1:k}^{-1}(\mathbf{z}_{1:k})} \underbrace{\pi(\mathbf{x}_{k+1:K}|\mathbf{x}_{1:k})}_{\mathbf{S}_{k+1:K}^{-1}(\mathbf{z}_{k+1:K};\mathbf{x}_{1:k})}. \tag{10}$$

Similarly, we can aggregate the map component functions into two blocks:

$$\mathbf{S}^{-1}(\mathbf{z}) = \begin{bmatrix} \mathbf{S}_{1:k}^{-1}(\mathbf{z}_{1:k}) \\ \mathbf{S}_{k+1:K}^{-1}(\mathbf{z}_{k+1:K};\mathbf{x}_{1:k}) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{1:k} \\ \mathbf{x}_{k+1:K} \end{bmatrix} = \mathbf{x}. \tag{11}$$

Instead of evaluating of Equation (7) from top to bottom (S_1^{-1} to S_K^{-1}), if we are interested in sampling conditionals, we may skip the upper map block $\mathbf{S}_{1:k}^{-1}$ and replace its corresponding output $\mathbf{x}_{1:k} = [x_1, \dots, x_k]$ with arbitrary, user-specified values $\mathbf{x}_{1:k}^* = [x_1^*, \dots, x_k^*]$. This results in the following truncated inversion for the lower map block $\mathbf{S}_{k+1:K}^{-1}$,

$$\mathbf{S}_{k+1:K}^{-1}(\mathbf{z}_{k+1:K};\mathbf{x}_{1:k}^*) = \begin{bmatrix} S_{k+1}^{-1}(z_{k+1};\mathbf{x}_{1:k}^*) \\ S_{k+2}^{-1}(z_{k+2};\mathbf{x}_{1:k}^*, x_{k+1}^*) \\ \vdots \\ S_K^{-1}(z_K;\mathbf{x}_{1:k}^*, x_{k+1}^*, \dots, x_{K-1}^*) \end{bmatrix} = \begin{bmatrix} x_{k+1}^* \\ x_{k+2}^* \\ \vdots \\ x_K^* \end{bmatrix} = \mathbf{x}_{k+1:K}^*. \tag{12}$$

Resuming the inversion starting with the map component inverse S_{k+1}^{-1} thus yields samples $\mathbf{x}_{k+1:K}^*$ from the conditional $\pi(\mathbf{x}_{k+1:K}|\mathbf{x}_{1:k}^*)$ instead of the target $\pi(\mathbf{x})$. Equivalent to Equation (8), the corresponding conditional distribution now factorizes as:

$$\pi(\mathbf{x}_{k+1:K}|\mathbf{x}_{1:k}^*) = \underbrace{\pi(x_{k+1}|\mathbf{x}_{1:k}^*)}_{S_{k+1}^{-1}(z_{k+1};\mathbf{x}_{1:k}^*)} \underbrace{\pi(x_{k+2}|\mathbf{x}_{1:k}^*, x_{k+1})}_{S_{k+2}^{-1}(z_{k+2};\mathbf{x}_{1:k}^*, x_{k+1}^*)} \cdots \underbrace{\pi(x_K|\mathbf{x}_{1:k}^*, \mathbf{x}_{k+1:K-1})}_{S_K^{-1}(z_K;\mathbf{x}_{1:k}^*, x_{k+1}^*, \dots, x_{K-1}^*)}, \tag{13}$$

This means that the manipulated triangular map inversion in Equation (7) allows us to sample conditionals of the target distribution π for arbitrary $\mathbf{x}_{1:k}^*$, or to estimate the density of this conditional distribution. Recalling Section 2.1, it is plain to see why this operation proves extremely useful in Bayesian inference: If we define our target pdf π as the joint distribution $p(\mathbf{a}, \mathbf{b})$ (using the notation of Section 2.1) between the RV of interest $\mathbf{a}$ and the observation predictions $\mathbf{b}$, and consider the manipulated samples $\mathbf{x}_{1:k}^*$ to be the observations $\mathbf{b}^*$ of $\mathbf{b}$, then the manipulated inversion in Equation (12) samples the posterior $p(\mathbf{a}|\mathbf{b}^*)$. An illustration of the forward, inverse, and conditional mapping operations is provided in Figure 6.

2.3.3 Conditional independence

A related, very useful property of triangular transport maps is that they naturally allow for the exploitation of *conditional independence*[8] by construction. Recalling the map's generic factorization of the conditionals of target π in Equation (8),

[8] By default, many statistical methods assume that all RVs directly influence each other. *Conditional independence* arises whenever two RVs $\boldsymbol{a}$ and $\boldsymbol{c}$ only affect each other indirectly via a third RV $\boldsymbol{b}$. In this case, we say that $\boldsymbol{a}$ is conditionally independent of $\boldsymbol{c}$ (and vice versa) given $\boldsymbol{b}$, represented symbolically as $\boldsymbol{a} \perp\!\!\!\perp \boldsymbol{c} \mid \boldsymbol{b}$. As an example, consider a as the chance of rain, b as the chance of wet ground (which can, but does not have to be caused by rain), and c as the chance of slipping. As rain (a) only affects the chance of slipping (c) via wet ground (b), we have $a \perp\!\!\!\perp c \mid b$.

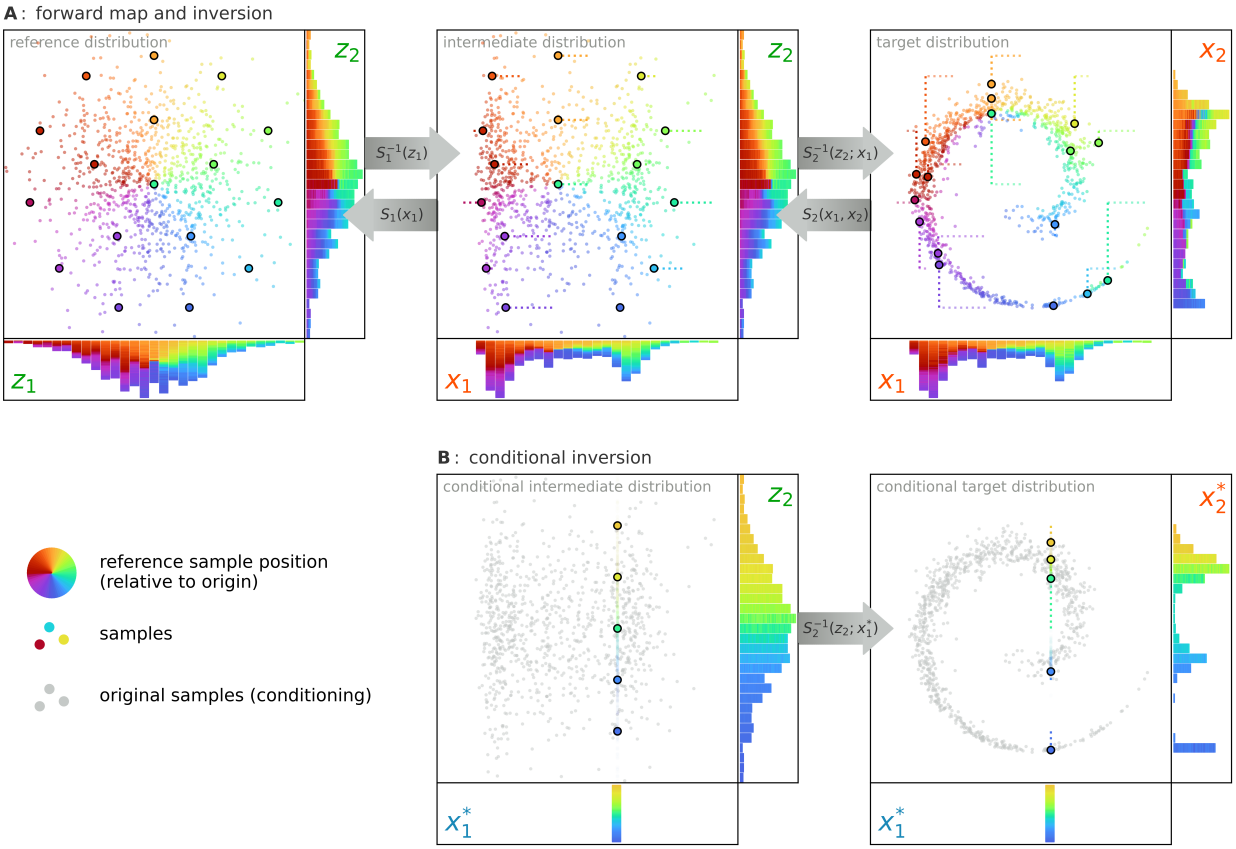

Figure 6: (A) The forward map (top row, from right) and its inverse (top row, from left) operate via implicit intermediate distributions (center), as the map components transform the distributions one marginal at a time. (B) Supplying the inverse map with a manipulated intermediate distribution (bottom row, center) as a starting point will instead sample conditionals of the target distribution.

we might ask ourselves what happens in systems in which we can exploit conditional independence. For example, if we have a target distribution $\pi(\mathbf{x}_{1:4})$ and conditional independence properties $x_3 \perp\!\!\!\perp x_1 | x_2$ and $x_4 \perp\!\!\!\perp x_1, x_2 | x_3$ (corresponding to *Markov structure*), the map's factorization could be reduced as follows:

$$\begin{aligned}\pi(\mathbf{x}_{1:4}) &= \pi(x_1)\,\pi(x_2|x_1)\,\pi(x_3|\cancel{x_1}, x_2)\,\pi(x_4|\cancel{x_1}, \cancel{x_2}, x_3) \\ &= \pi(x_1)\,\pi(x_2|x_1)\,\pi(x_3|x_2)\,\pi(x_4|x_3)\,.\end{aligned} \tag{14}$$

We refer to this reduction as *sparsification*. Triangular transport maps allows us to leverage these conditional independence properties by simply dropping the corresponding arguments from the map components S_k:

$$\mathbf{S}(\mathbf{x}_{1:4}) = \begin{bmatrix} S_1(x_1) \\ S_2(x_1, x_2) \\ S_3(\cancel{x_1}, x_2, x_3) \\ S_4(\cancel{x_1}, \cancel{x_2}, x_3, x_4) \end{bmatrix} = \begin{bmatrix} S_1(x_1) \\ S_2(x_1, x_2) \\ S_3(x_2, x_3) \\ S_4(x_3, x_4) \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix} = \mathbf{z}_{1:4}. \tag{15}$$

Equivalently, its inverse map would be:

$$\mathbf{S}^{-1}(\mathbf{z}_{1:4}) = \begin{bmatrix} S_1^{-1}(z_1) \\ S_2^{-1}(z_2; x_1) \\ S_3^{-1}(z_3; \cancel{x_1}, x_2) \\ S_4^{-1}(z_4; \cancel{x_1}, \cancel{x_2}, x_3) \end{bmatrix} = \begin{bmatrix} S_1^{-1}(z_1) \\ S_2^{-1}(z_2; x_1) \\ S_3^{-1}(z_3; x_2) \\ S_4^{-1}(z_4; x_3) \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \mathbf{x}_{1:4}. \tag{16}$$

Making use of conditional independence properties in this way is useful for two reasons:

1. **Robustness**: Any conditional independence we can enforce by construction is statistical information the map does not have to pry from the samples or the model, improving the overall fidelity and robustness of the approximation to π in settings with finite ensemble size.
2. **Efficiency**: The removal of superfluous dependencies reduces the number of input arguments to many of the map components S_k, decreasing the evaluation and inversion complexity (e.g., see Section 3.1.1 for evaluation complexity). This property is often called sparsification as it turns the Jacobian $\nabla\mathbf{S}$ into a sparse matrix.

This second property is the key to applying transport methods in high-dimensional systems. As each map component function S_k generically depends on all previous arguments $\mathbf{x}_{1:k-1}$, the computational demand explodes with the dimension of the target π when optimizing or evaluating the map. With sufficient conditional independence, however, sparse maps can overcome this dramatic increase in complexity. For instance, the Markov structure in Equations (15) and (16) results in a sparse map $\mathbf{S}$ with numerical complexity scaling linearly in the target dimension K.

A comprehensive account of the link between conditional independence and the sparsity of triangular maps is given in Spantini et al. (2018), through two main lines of results. First, given a sparse undirected probabilistic graphical model that encodes Markov properties of the target distribution π, it is shown how to predict the sparsity pattern of the triangular map $\mathbf{S}$. This process relies on an ordered graph elimination algorithm, and can thus be performed before learning the map itself. But the resulting sparsity pattern depends on the chosen ordering of the random variables, which underscores the fact that triangular maps are intrinsically *anisotropic* objects: a good ordering is necessary in order to maximize sparsity. We comment further on methods for finding such orderings in Section 5. Second, Spantini et al. (2018) show that a property somehow dual to sparsity is *decomposability*: given the Markov structure of some distribution π on $\mathbb{R}^K$, the inverse of the map $\mathbf{S}$ defined by $\mathbf{S}_\sharp\pi = \eta$ can be represented as the composition of finitely many low-dimensional triangular transport maps, where low dimensionality here means that each component map is a function only of a small number of inputs. The exact structure of this decomposition follows from, again, an ordered decomposition of the original graph. These results allow very high-dimensional problems to be broken into many smaller and more manageable parts, given some conditional independence.

2.3.4 Block-triangular maps

Recalling the block structure in Section 2.3.2, we have so far assumed that the map component "blocks" in Equation (11) are internally triangular; i.e., $\mathbf{S}_{1:k}$ has output $[S_1(x_1), \ldots, S_k(\mathbf{x}_{1:k})]$ and similar for $\mathbf{S}_{k+1:K}$. We may instead use *any* invertible functions $\mathbf{S}_{1:k} : \mathbb{R}^k \to \mathbb{R}^k$ and $\mathbf{S}_{k+1:K} : \mathbb{R}^K \to \mathbb{R}^{K-k}$, for example certain neural networks (e.g., Baptista et al., 2024c). This still permits sampling conditionals and can be numerically advantageous in high-dimensional

systems, but can compromise the ability to exploit conditional independence within the map blocks. In the following, we will assume that the transport map is *fully triangular*, even where we adopt the block-triangular structure for ease of notation.

3 Implementation

In the preceding section, we have discussed the theoretical foundations of triangular transport, but stopped short of defining the precise method to evaluate S_k, or the optimization problem which identifies the specific map $\mathbf{S}$ from π to η. In this section, we will fill these gaps, and provide guidance on the practical implementation of triangular transport methods in code. First, though, a slight interlude about the "work" a map performs to transform π and η into one another. Without a loss of generality, suppose π and η have the same mean and covariance. In the case that both are Gaussian, this implies that they are the same distribution, and the transport map $\mathbf{S}$ is trivially the identity. Once π becomes non-Gaussian, the transport map must be strictly nonlinear, for any linear map merely transforms these first two moments! The nonlinearity of a map is therefore directly correlated with how "different" our reference and target distributions are, measured in one way or another (e.g., the Kullback–Leibler divergence, discussed below). This theoretical idea underpins many practical considerations; for example, we will assume that π and η approximated have the same tails (i.e., $\pi(x) \approx C\eta(x)$ for some $C \in \mathbb{R}$ when x is sufficiently large), implying the map *must* become close to linear sufficiently far from the origin.

3.1 Structuring the map components S_k

A core component of any triangular transport map $\mathbf{S}$ is the definition of the constituent map components S_k. In this subsection, we will introduce and discuss important features in the definition of these map components S_k.

3.1.1 Monotonicity

Let us recall from Section 2.2 that each map component function must be monotone in its last argument x_k, that is to say, $\partial_{x_k} S_k > 0$ on the *entire* domain (i.e., for any choice $\mathbf{x}_{1:k-1}$). Together with the triangular structure and linearity in the tails, this property guarantees that the resulting map $\mathbf{S}$ is bijective.

This may not be immediately intuitive, so an illustration of this effect is provided in Figure 7, which shows an example in $\mathbb{R}^2$. Here, invertibility ensures that every tuple $(\mathsf{X}_1, \mathsf{X}_2)$ maps to a unique tuple $(\mathsf{Z}_1, \mathsf{Z}_2)$. Graphically, this means that if we overlay Figure 7C and Figure 7D on top of each other, any pair of contour lines between the subplots must not intersect more than once. This is achieved due to triangularity and monotonicity:

- The **triangular structure** ensures that the contours of every map component $\mathbf{S}_{1:k-1}(\mathbf{x}_{1:k-1})$ independent of x_k and are thus constant along x_k. In consequence, the contours of S_1 in Figure 7C are constant along x_2, which aligns them parallel to x_2 (see Figure 7C).
- The **monotonicity** of $S_1(x_1)$ relates each X_1 to a unique Z_1. We then obtain a unique tuple $(\mathsf{Z}_1, \mathsf{Z}_2)$ when we ensure that the second set of contours from $S_2(\mathsf{X}_1, x_2)$ increases monotonously along x_2 (see Figure 7D). This is achieved through the **monotonicity requirement** $\partial_{x_k} S_k > 0$.

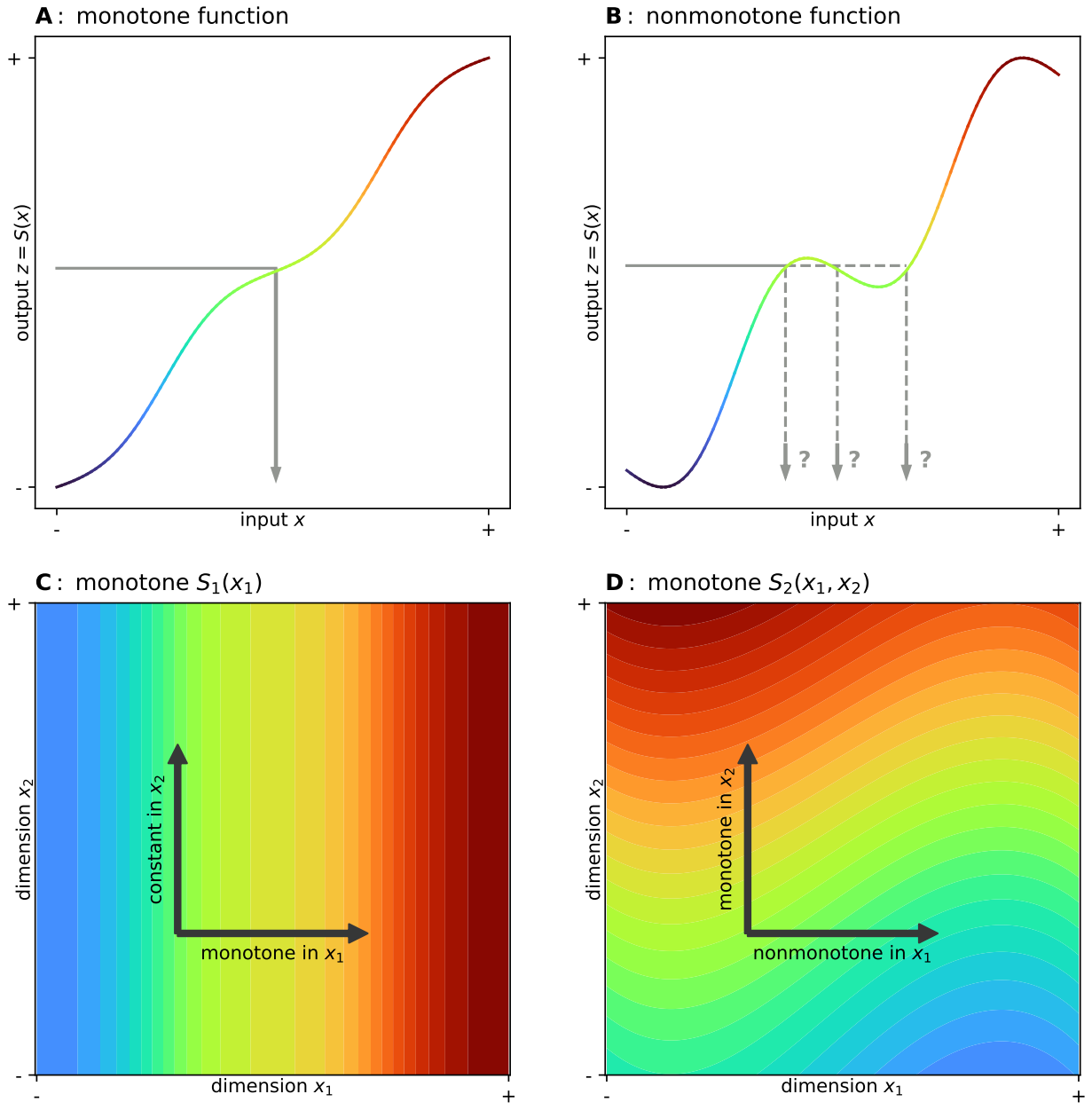

Figure 7: (A) Monotonicity guarantees that a function remains invertible. (B) Non-monotone map component functions have non-unique inverses. How do multivariate maps $\mathbf{S}$ remain monotone? (C) The map component function $S_1(x_1) = z_1$ is monotone in x_1 but constant in x_2, as it does not depend on this input. (D) The map component function $S_2(x_1, x_2) = z_2$ is monotone in x_2 but can be nonmonotone in x_1. Color indicates the magnitude of the map component's output in all subplots.

The same principle extends to higher dimensions: the triangular structure means that the $(k-1)$-dimensional contours of $\mathbf{S}_{1:k-1}(\mathbf{x}_{1:k-1})$ are aligned along x_k. Monotonicity of S_k along x_k then ensures that for $(\mathsf{X}_1, \ldots, \mathsf{X}_{k-1})$ and any x_k, we obtain a unique tuple $(\mathsf{Z}_1, \ldots, \mathsf{Z}_k)$. There are several strategies that can ensure each map component satisfies these monotonicity requirements. We will explore these strategies in the following.

Monotonicity through integration A general, powerful, but also computationally demanding method to enforce monotonicity makes use of a *rectifier* and *integration* (e.g., Baptista et al., 2023). In this formulation, our starting point is a smooth, differentiable, but nonmonotone function $S_k^{\text{non}} : \mathbb{R}^k \to \mathbb{R}$. An example of such a nonmonotone function is:

$$S_3^{\text{non}}(x_1, x_2, x_3) = \underbrace{c_0 + c_1 x_1 + c_2 x_2 + c_3 x_1 x_2 + c_4 {x_1}^2}_{\text{nonmonotone part } g(x_1, x_2)} + \underbrace{c_5 x_3 + c_6 x_1 {x_3}^2 + c_7 x_2 x_3}_{\text{pre-monotone part } \hat{g}(x_1, x_2, x_3)}, \tag{17}$$

where the coefficients $\{c_i\}$ parameterize S_k^{non}. In Equation (17),we have conceptually separated the terms of S_3^{non} into a nonmonotone part g, which does not depend on x_3, and a "pre-monotone" part $\hat{g}$, which does. In general, $\hat{g}$ will not be monotone in x_3, and consequently neither will be S_3^{non}. However, we can monotonize it by first applying a rectifier $r : \mathbb{R} \to \mathbb{R}^+$ to $\hat{g}$, then integrating the rectified output over x_k:

$$\begin{aligned} S_k(x_1, \ldots, x_k) &= \underbrace{g(x_1, \ldots, x_{k-1})}_{\text{nonmonotone part}} + \underbrace{f(x_1, \ldots, x_k)}_{\text{monotone part}}, \\ \text{where } \underbrace{f(x_1, \ldots, x_k)}_{\text{monotone function}} &= \int_0^{x_k} \overbrace{r(\underbrace{\hat{g}(x_1, \ldots, x_{k-1}, t)}_{\text{nonmonotone function}})}^{\text{positive function}} \, dt. \end{aligned} \tag{18}$$

The two steps combined in Equation (18) are as follows:

1. **Rectification**: First, $\hat{g}$ is sent through the rectifier r, a function which maps its output to strictly positive values. Examples of useful rectifiers include exponential functions, the softplus function (i.e., $\log(1 + \exp(x))$) or Exponential Linear Units (Clevert et al., 2016).
2. **Integration**: Numerically or analytically integrating the output of the resulting strictly positive function over x_k yields a function f monotone in x_k, which turn also guarantees the monotonicity of S_k in x_k.

This process is visualized in Figure 8, and has a number of important advantages. First off, the use of an integrated rectifier places very few limitations on the structure of S_k^{non}. Furthermore, it also permits the introduction of *cross-terms* such as $x_1 x_k$ between the last argument x_k and previous dimensions $x_{1:k-1}$, which are required for many of the more complex mapping operations (see Section 3.1.2). The drawback of this approach is that in practice evaluating Equation (18) often demands one-dimensional numerical integration via, e.g., quadrature, increasing the computational demand of the map's evaluation, optimization, and inversion.

An important practical consideration when using the integrated map formulation is to ensure that the pre-monotone term $\hat{g}$ reverts to a *constant* in the tails of x_k. This can be ensured by defining $\hat{g}$ as a combination of a constant term and Hermite functions (see Section 3.1.3), which revert to zero in the tails. To understand why this is important, consider the process illustrated in Figure 8. Through rectification and integration, increasingly negative values of $\hat{g}$ become flat stretches in S_k. If one or both tails of the monotone term f become near-flat, it is possible for the root finding during the map's inversion (see Section 2.3.1) to fail for outlying values. However, if $\hat{g}$ instead reverts to a positive constant in the tails, the resulting integrated monotone term f will extrapolate linearly and thus generally remain more robust to outliers during the inversion. This is an artifact from the discussion above where, while π and η may "look" very different for the bulk of the pdf, we expect the tails of both of them to behave similarly (Baptista et al., 2023).

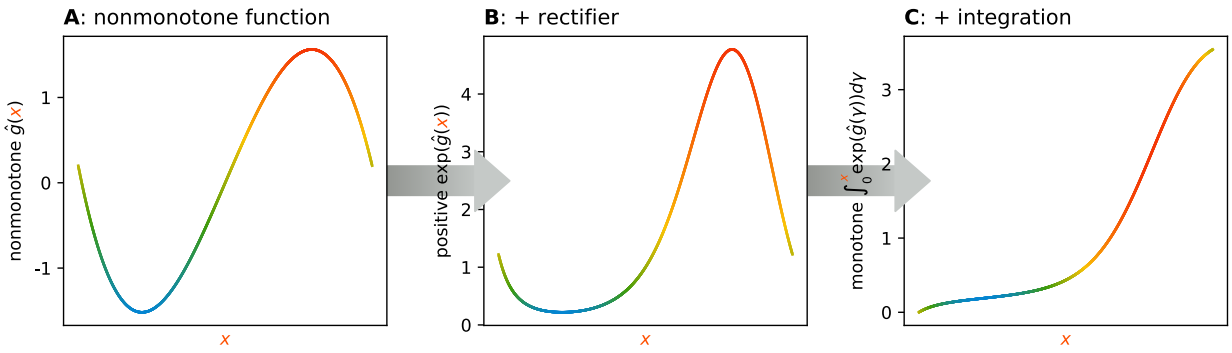

Figure 8: Integrated rectifiers create monotonicity. (A) We start with an arbitrary smooth, non-monotone function $\hat{g}(x)$. (B) Applying a monotone rectifier, for example $\exp(\hat{g}(x))$, yields strictly positive function output. (C) Integrating a strictly positive function yields a monotone function f. Color indicates the magnitude of the nonmonotone function in (A).

Monotonicity through variable separation A more computationally efficient way to ensure monotonicity is to formulate a map component function S_k which is both *separable in* x_k and *linear in the coefficients*, though not generally linear in the inputs $\mathbf{x}$. An example for such a map component function is provided below:

$$S_3(x_1, x_2, x_3) = \underbrace{c_0 + c_1 x_1 + c_2 x_2 + c_3 x_1 x_2 + c_4 {x_1}^2}_{\text{nonmonotone part } g(x_1, x_2)} + \underbrace{c_5 x_3 + c_6 \operatorname{erf}(x_3)}_{\text{monotone part } f(x_3)}, \tag{19}$$

where c_i once more are the map component's coefficients, which are optimized to find the map from π to η, and $c_5, c_6 > 0$. The key aspects in Equation (19) are a clear separation into a nonmonotone term g, which may depend on all arguments except the last $\mathbf{x}_{1:k-1}$, and a monotone term f, which may *only* depend on the last argument x_k. To ensure that S_k remains monotone in x_k, all terms in f must likewise be monotone basis functions (e.g., x_k and $\operatorname{erf}(x_k)$).

The advantages of this separable formulation are two-fold: First, Equation (19) has no need for numerical integration, which reduces computational demand substantially. Second, as we shall see in Section 3.2, this separable formulation allows for extremely efficient map optimization. The price for this efficiency is that variable separation does not allow for cross-terms between the last argument and previous arguments (e.g., $x_1 x_k$), which limits the map's expressivity. We will explore the consequences of this limitation in the next section.

3.1.2 Parameterization

Closely related to the concern of monotonicity is how each map component S_k relates x_k, its last argument, to $x_1, \ldots, x_{k-1}$. This decision leads to three different kinds of map parameterizations, which permit maps of different complexity. In ascending order of complexity, these parameterizations are:

Marginal maps If both the nonmonotone part g and the monotone part f depend exclusively on input x_k, such map component functions S_k result in a *marginal* (or *diagonal*) map. As the name implies, such maps only transform the marginals of $\mathbf{x}$ without capturing any dependencies between its dimensions. An example of a marginal map would be:

$$S_3\left(x_3\right) = \underbrace{c_0}_{\text{nonmonotone part } g(-)} + \underbrace{c_1 x_3 + c_2 {x_3}^3}_{\text{monotone part } f(x_3)} . \tag{20}$$

Recalling the implications of removing dependencies on earlier arguments $\mathbf{x}_{1:k-1}$ from the map component functions (see Section 2.3.3), marginal maps are of no direct use for the conditioning operations in Section 2.3.2. This is because marginal maps implicitly assume all entries of $\mathbf{x}$ are marginally independent, which in turn implies that $\pi(\mathbf{x}_{k+1:K}|\mathbf{x}_{1:k}) = \pi(\mathbf{x}_{k+1:K})$, that is to say, nothing can be learned from $\mathbf{x}_{1:k}$ about $\mathbf{x}_{k+1:K}$. As marginal Gaussianization schemes, however, they can find limited use in Gaussian anamorphosis (Schöniger et al., 2012; Zhou et al., 2011) or as preconditioning tools in certain graph detection methods (Liu et al., 2009). Marginal maps are only included here for completeness' sake, and we use $g(-)$ to denote that the nonmonotone part is constant in $\mathbf{x}$, connecting better to the more complex parameterizations.

Separable maps Closely related to the monotonicity scheme of the same name in the previous section, *separable* maps separate the map component S_k into the sum of a function $g : \mathbb{R}^{k-1} \to \mathbb{R}$ that takes in $\mathbf{x}_{1:k-1}$ and a univariate monotone function $f : \mathbb{R} \to \mathbb{R}$ that just takes in x_k

$$S_2\left(x_1, x_2, x_3\right) = \underbrace{c_0 + c_1 x_1 + c_2 {x_1}^2 + c_3 x_1 x_2}_{\text{nonmonotone part } g(x_1, x_2)} + \underbrace{c_4 x_3 + c_5 {x_3}^3}_{\text{monotone part } f(x_3)} . \tag{21}$$

Separable maps are one level of complexity above marginal maps. Such maps can consider nonlinear dependencies between the different dimensions, but do not permit cross-terms with the last argument x_k, which in turn limits the complexity of the target distributions π they can recover[9]. This limitation is subtle, and will be discussed shortly. An important advantage of separable maps is that, when linear in the coefficients, their optimization can be partially solved in closed-form, which can improve computational efficiency substantially. More detail on this is provided in Appendix A.

Cross-term maps The most versatile map parameterization are *cross-term* maps, which permit the greatest expressiveness at the cost of increased computational demand. As mentioned previously, cross-terms are basis functions which depend both on the last argument x_k and preceding arguments $\mathbf{x}_{1:k-1}$. The presence of these terms requires that such maps must use integrated rectifiers (Section 3.1.1) to ensure the monotonicity of S_k:

$$S_2\left(x_1, x_2\right) = \underbrace{c_0 + c_1 x_1 + c_2 {x_1}^2}_{\text{nonmonotone part } g(x_1)} + \underbrace{\int_0^{x_2} \exp\left(\overbrace{c_3 {x_1}^2}^{\text{cross-term}} t + c_4 t + \overbrace{c_5 x_1 t^2}^{\text{cross-term}} \right) dt}_{\text{monotone part } f(x_1, x_2)} . \tag{22}$$

The presence of cross-terms permits increased control over the local details in the transformations from π to η. In principle, it is also possible to make Equation (22) separable in x_k by removing the dependencies of f on $\mathbf{x}_{1:k-1}$. Since the resulting S_k would not be linear in the coefficients c, however, we cannot make use of a more efficient optimization scheme (Appendix A). In the following subsection, we will develop some intuition about the strengths and limitations of different map parameterizations.

[9] Cross-terms and coefficient linearity are not mutually exclusive. However, one must carefully ensure monotonicity in the final argument for any choice of other inputs $\mathbf{x}_{1:k-1}$

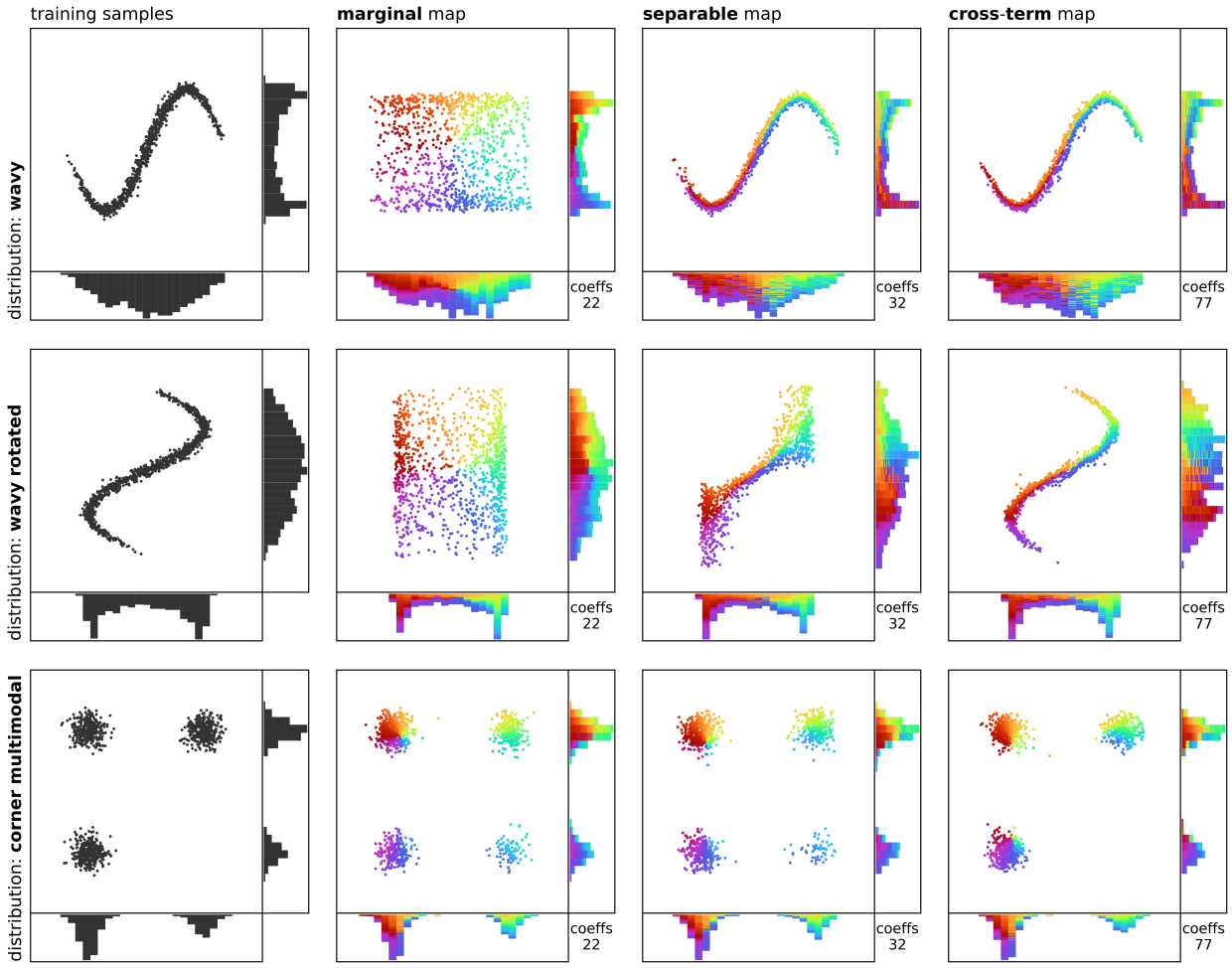

Figure 9: Pullback approximations $\mathbf{S}^\sharp\eta$ to different target distributions π using nonlinear map components S_k for marginal maps, separable maps, and cross-term maps. Some distributions require cross-term maps, for others simpler parameterizations may suffice. The variable ordering is $[x_1, x_2]$, with x_1 plotted on the horizontal axis and x_2 on the vertical axis. Color represents the position of the reference samples $\mathbf{z}$ relative to the mean of η. The number of coefficients (i.e., parameters) for each map is plotted in the bottom-right corner.

Choosing a parameterization To provide more insight into the effects and limitations of each parameterization choice, we have illustrated the pullback $\mathbf{S}^\sharp\eta$ of a transport map approximated using these parameterizations for each of three different target pdfs π in Figure 9. In every case, we begin by sampling the target distributions, proceed to optimize the different maps, then apply the inverse map (Section 2.3.1) to transform reference samples $\mathbf{z} \sim \eta$ into samples from the pullback $\mathbf{x} \sim \mathbf{S}^\sharp\eta$. We may make the following observations:

- **Marginal** maps only reproduce the marginal densities of all three target distributions. As their structure implies independence between the marginals (Section 2.3.3), however, the pullback approximation does not approximate the true target pdfs well. This effect can be subtle: in the corner-multimodal distribution, it is only noticeable through the emergence of a fourth phantom mode.

- **Separable** maps provide much better approximations to the target pdfs, but have some interesting caveats: while they successfully recover the wavy target distribution, they struggle with the distribution once it is rotated by $90°$, and they yield a slight – if less pronounced – fourth phantom mode for the corner-multimodal target. We will discuss the reason for this below.
- **Cross-term** maps prove most versatile, recovering all three target distributions well at the cost of a larger parameter space.

In practice, cross-terms provide greater control over local features. To understand why, let us take a closer look at a single map component inversion S_k^{-1}, which forms the basis of the map inversion (see Section 2.3.1) that ultimately yields the pushforward samples in Figure 9. Consider a generic map component function comprised of a nonmonotone part g and a monotone part f:

$$z_k = S_k(\mathbf{x}_{1:k-1}, x_k) = g(\mathbf{x}_{1:k-1}) + f(\mathbf{x}_{1:k-1}, x_k). \tag{23}$$

As discussed in Section 2.3.1, the inversion of S_k relies on one-dimensional root finding. Reformulating this expression yields the root finding objective for S_k^{-1}, conditioned on the outcomes $\mathbf{x}_{1:k-1}$ of previous map component inversions:

$$z_k - g(\mathbf{x}_{1:k-1}) = f(\mathbf{x}_{1:k-1}, x_k). \tag{24}$$

If we now apply the simplifications of the three map parameterizations above, we obtain three different objectives for the root finding problem:

$$\begin{aligned} z_k - g(-) &= f(x_k) && \text{(marginal maps)} \\ z_k - g(\mathbf{x}_{1:k-1}) &= f(x_k) && \text{(separable maps)} \\ z_k - g(\mathbf{x}_{1:k-1}) &= f(\mathbf{x}_{1:k-1}, x_k) && \text{(cross-term maps)} \end{aligned} \tag{25}$$

These equations provide some insight into the differences between the pullback densities $\mathbf{S}^\sharp\eta$ we have observed in Figure 9. Each term above takes on a different role during the conditional inversion: the reference samples z_k are an independent input, unaffected by the map parameterization choice, and define a *random initial offset* for the root finding. The nonmonotone term g acts as an additional *dynamic offset* for the inversion based on previous values, and the monotone term f defines the *shape of the inverse function* for the one-dimensional root finding over the unknown x_k. From this perspective, the differences between the map parameterizations emerge from how each handles the dependence on previous values $\mathbf{x}_{1:k-1}$:

1. **Marginal** maps (Figure 10, top row) permit only a constant nonmonotone term $g(-)$, which results in no dynamic offset. Their monotone term $f(x_k)$ likewise only depends on x_k. In consequence, marginal maps extract the same x_k from a specific z_k for any given value of $\mathbf{x}_{1:k-1}$.
2. **Separable** maps (Figure 10, center row) also feature monotone term $f(x_k)$ that depends only on x_k, which keeps the inverse function's shape constant for all values of $\mathbf{x}_{1:k-1}$. However, their off-diagonal term $g(\mathbf{x}_{1:k-1})$ can induce varying offsets for different $\mathbf{x}_{1:k-1}$, which introduces a simple dependence on previous values.

3. **Cross-term** maps (Figure 10, bottom row) can vary both the dynamic offset (nonmonotone term $g(\mathbf{x}_{1:k-1})$) and the inverse function's shape (monotone term $f(\mathbf{x}_{1:k-1}, x_k)$) with different $\mathbf{x}_{1:k-1}$, and thus provide the most expressive maps, but often at higher computational cost.

These insights reveal why separable maps succeeded to recover the wavy distribution, but failed to capture its features once it is rotated (Figure 9). For the original wavy distribution, its (vertical) conditionals at different horizontal positions are mostly of the same shape, just shifted vertically, which makes them a perfect fit for recovery via separable maps. Once the distribution is rotated, however, the true vertical conditionals become more challenging to recover. Note that if we were to change the order of the target vector to $[x_2, x_1]$ instead of $[x_1, x_2]$, separable maps would succeed in recovering the rotated wavy distribution and instead fail for the standard one. This illustrates the subtle influence of input variable ordering on a map's approximation ability.

3.1.3 Basis functions

A very important part of the parameterization of the map component function S_k is its construction: what basis functions should be used to represent S_k and how much nonlinearity should they permit? In a nutshell, simpler – perhaps separable – parameterizations are more computationally efficient, but can struggle to recover features of the target pdf π that do not resemble the reference η, as discussed at the beginning of this section. By contrast, more complex parameterizations add the flexibility required to capture "localized" features of π (e.g., skewness, multi-modality, ...), but risk unfavourable bias-variance trade-offs. Depending on the properties of the problem of interest, different parameterization choices promise different advantages:

Polynomial basis functions In function approximation, polynomials are a canonical choice for forming an approximation; in particular, harkening to polynomial chaos expansion and other traditional stochastic function approximation literature (Le Maître and Knio, 2010; Ernst et al., 2012), we can choose orthogonal polynomials for our approximation function class. Since this work focuses on unbounded $\mathbf{x}$, we will consider the probabilists' Hermite polynomial family $\{\mathrm{He}_j\}_{j=1}^{\infty}$, which has the orthogonality property

$$\int_{-\infty}^{\infty} \mathrm{He}_i(x)\mathrm{He}_j(x) \underbrace{\frac{1}{\sqrt{\tau}}\exp(-x^2/2)}_{\text{Gaussian weight}} dx = j!\delta_{ij} = \begin{cases} j! & j = i \\ 0 & j \neq i \end{cases} \tag{26}$$

where $\tau = 6.283185\ldots$ is twice Archimedes' constant. Broadly, Equation (26) shows what it means for polynomials of different orders (e.g., linear, quadratic, cubic, etc.) to be orthogonal under the Gaussian weight. Similar to orthogonal vectors in linear algebra, orthogonal polynomials (more generally, orthogonal functions) are often chosen as a basis for function approximation, as their elements contain no "overlapping information" with respect to a particular weight (which may or may not match η). As these polynomials are dependent on the pdf, a few other orthogonal polynomial families with their respective weights and support include Legendre polynomials (respecting a constant weight on $(-1, 1)$) and Laguerre polynomials (respecting weight e^{-x} on $(0, \infty)$) (Szegő, 1939). These polynomial families may be suited for different problems depending on characteristics of η and π (Wang and Marzouk, 2022)

Generally, if a polynomial family $\{\psi_j\}_{j=1}^{\infty}$ is orthogonal with respect to a weight ρ (e.g., the probabilist Hermite polynomials take ρ as the Gaussian weight), one can approximate a given well-behaved function $f(x)$ with $\hat{f}(x) = c_1\psi_1(x) + c_2\psi_2(x) + \cdots + c_P\psi_P(x)$ using a finite number P of these polynomials. Guarantees for this approximation

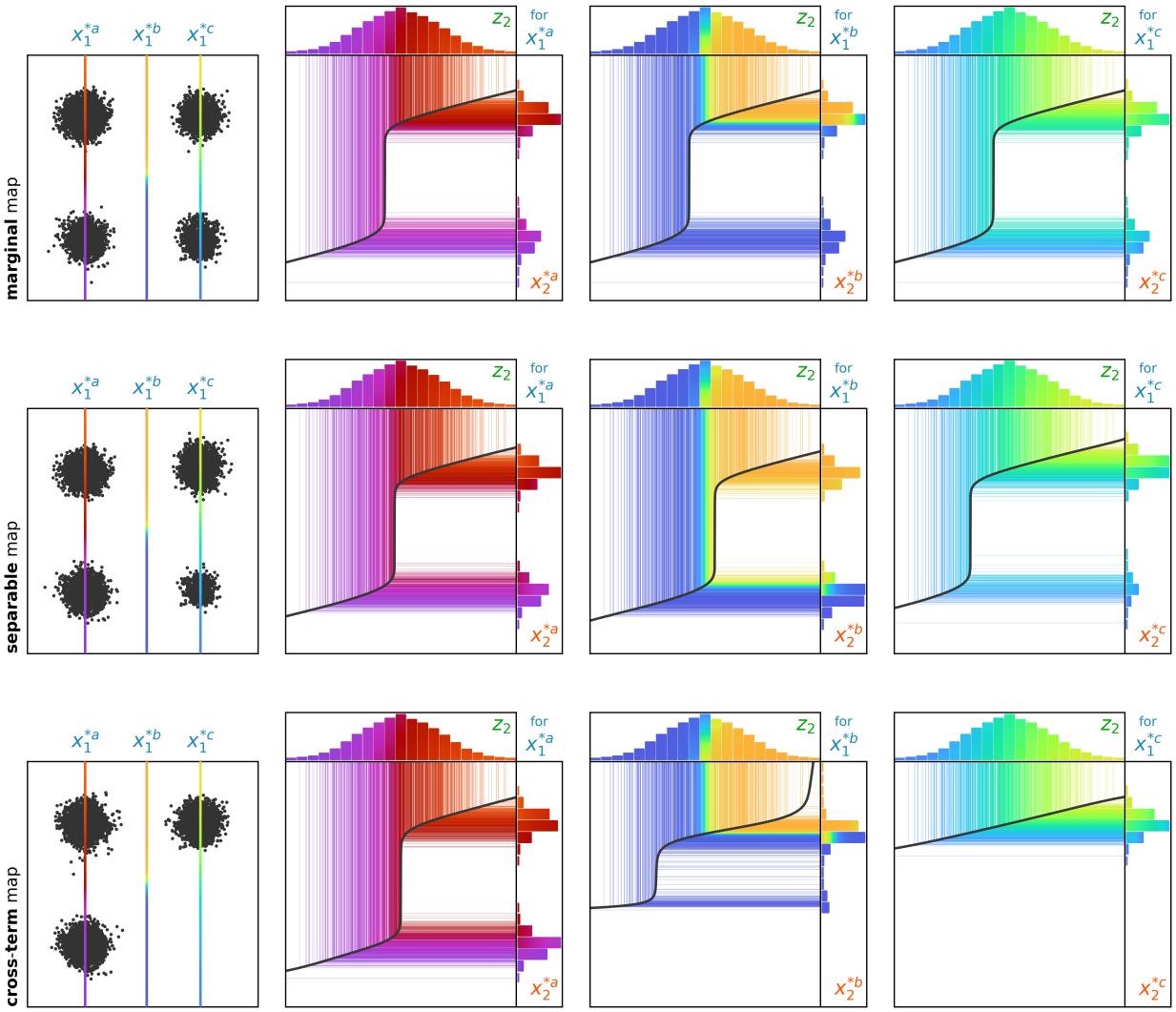

Figure 10: Conditional inversion $x_2 = S_2^{-1}(z_2; x_1)$ for marginal (top row), separable (center row), and cross-term (bottom row) maps. Each column corresponds to a conditional distribution for a different fixed x_1^*. Each term in Equation (25) has a different effect: z_2 defines the horizontal starting position for the inversion, $f(x_1, x_2)$ defines the shape of coupling, and $g(x_1)$ defines its horizontal offset. **Marginal** maps keep f and g fixed, and thus yield the same inverse x_2 for any previous x_1. **Separable** maps also keep f constant, but can adjust the offset g. Finally, **cross-term** maps can adjust both f and g for different x_1, providing the greatest degree of flexibility.

are often given when measuring the error according to our weight ρ via $\int (f(x) - \hat{f}(x))^2 \, d\rho(x)$ (Ernst et al., 2012). If we believe that η and π are very similar, choosing polynomials orthogonal with respect to our reference η (which we know the orthogonal polynomial family of) will also perform well when the input is distributed according to π (which we generally won't know the orthogonal polynomial family of). On the other hand, if η and π are not very similar, such guarantees are not particularly helpful; practitioners see this manifest as a need for more complex map parameterizations (see Section 3.1.2).

As they have unbounded support, Hermite polynomials exert global influence and often require high-order terms to capture fine distributional features. While versatile, the use of polynomial basis functions has a few pitfalls which demand caution:

1. **Importance of standardization**: In response to the above, we would like to make π resemble η more closely by pre-processing the data we have from π. When working with a Gaussian reference η, it is thus generally advised to standardize $\mathbf{x}$ by trying to match the mean and variance of η. We do this by (i) subtracting the empirical mean $\bar{\mathbf{X}} = 1/N \sum_{i=1}^{N} \mathbf{X}^i$ and (ii) scaling each marginal to unit variance by dividing it through the unbiased estimator of the empirical standard deviation σ_k where $\sigma_k^2 = (N-1)^{-1} \sum_{i=1}^{N} (\mathrm{X}_k^i - \bar{\mathrm{X}}_k)^2$ before formulating and applying a transport map[10] $\mathbf{S}$. All subsequent map operations are then implemented in this standardized space. Finally, the map's outputs are transformed back into target space by reversing the standardization. In effect, standardization wraps a separate linear transformation around the transport map, and thereby does not affect the validity of the operations inside this wrapper. In the subsequent sections, we will assume by default that the samples have been standardized. As pointed out in Morrison et al. (2022), invertible diagonal transformations retain the conditional independence structure of the target distribution. In other words, there is no loss of information while working in the marginally rescaled space.

2. **Edge control**: A practical challenge of polynomials is their growth in the tails for higher-order terms. This often leads to volatility if the map is evaluated or inverted far from zero. To address this issue, one useful option is the use of *edge-controlled terms* such as *Hermite functions* $\mathcal{H}_j$. This variant of basis functions are defined as probabilist's Hermite polynomials He_j of order j multiplied with a Gaussian weight, which reverts the polynomial's output to zero far from zero:

$$\mathcal{H}_j(x) = \mathrm{He}_j(x) \exp\left(-\frac{x^2}{4}\right). \tag{27}$$

A nice feature of this form is that $\mathcal{H}_i(x)\mathcal{H}_j(x) = \mathrm{He}_i(x)\mathrm{He}_j(x)\exp(-x^2/2)$, and so $\{\mathcal{H}_j\}_{j=1}^{\infty}$ will inherit the same error properties as $\{\mathrm{He}_j\}$ when measuring error without any weight ρ, i.e., $\int (f(x) - \hat{f}(x))^2 \, dx$.

Illustrations of the first few orders of Hermite functions are provided in Figure 11A. Note that expressing the map exclusively in terms of Hermite functions causes the map component S_k to revert to zero in the tails, which can be undesirable whenever extrapolation may be required. In practice, limiting the Gaussian weight term to polynomials of order two or larger is a good compromise, thus we recommend retaining the linear terms without a weight.

[10] Alternatively, one may standardize the samples by forming the empirical cdf $\tilde{F}_{\pi_k}$ along the marginal for x_k, then standardize the samples by composing $\tilde{F}_{\pi_k}$ with the inverse standard Gaussian cdf $F_{\eta_k}^{-1}$, that is to say, $F_{\eta_k}^{-1} \circ \tilde{F}_{\pi_k}(x_k)$, ensuring each marginal of the standardized samples has a Gaussian distribution.

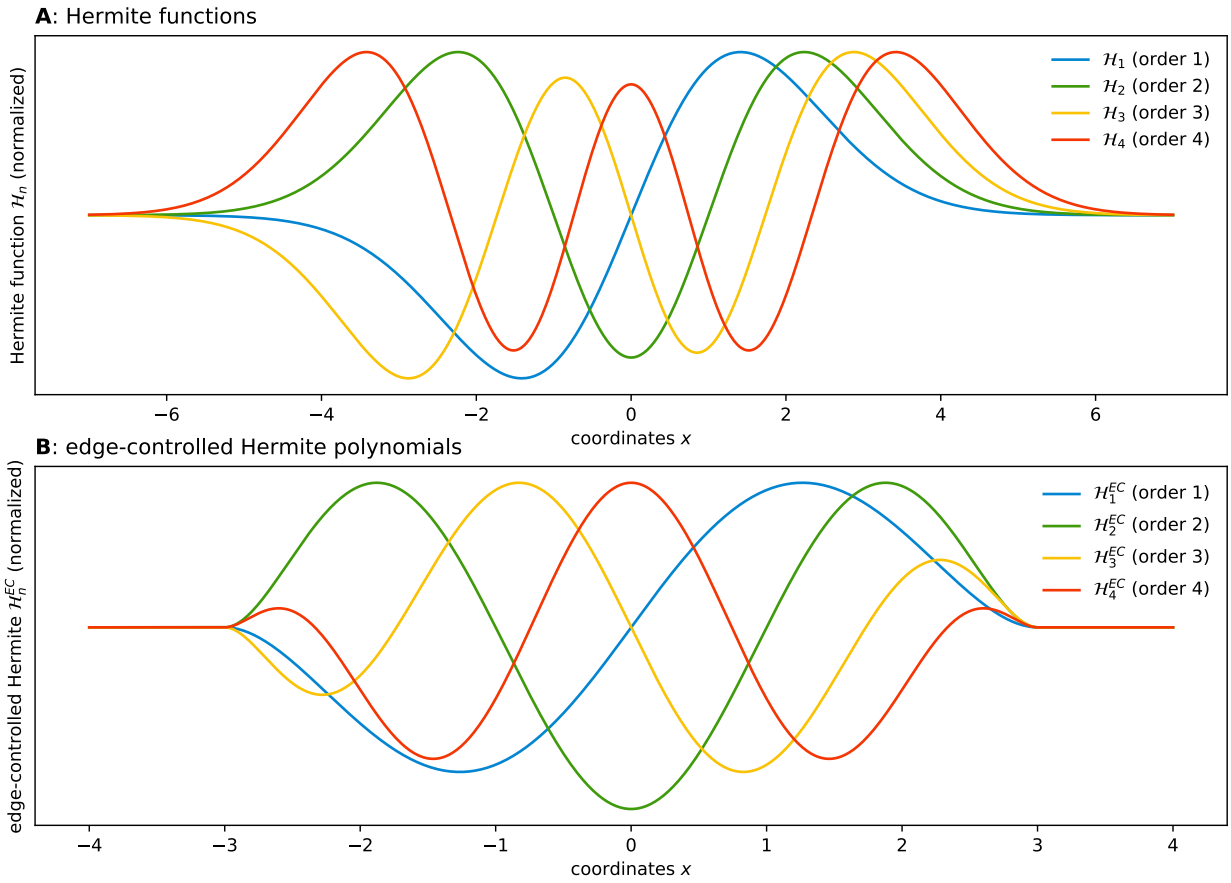

Figure 11: Two types of edge-controlled basis functions based on Hermite polynomials. (A) Hermite functions are multiplied with a Gaussian weight, causing them to approach zero in the tails. (B) Replacing the Gaussian weight with a scaled cubic spline leads to a different edge-controlled basis function. This enforces limited support, ensuring the function reverts to zero at a finite distance r.

A practical issue with Hermite functions can emerge for target distributions π which include sharp features. From (Szegő, 1939), the largest maximizer x_j^* of He_j grows at a rate of $x_j^* = \sqrt{j} + \mathcal{O}(j^{-1/6})$, which means that if $\mathcal{H}_j$ has nontrivial values on $(-r, r)$, we will have that $\mathcal{H}_{4j}$ has nontrivial values on $(-2r, 2r)$; e.g., it is reasonable to estimate from Figure 11(A) that $r > 4$ for $j = 2$ and so $\mathcal{H}_8$ has nontrivial values on $(-8, 8)$. Due to this "global" nature of polynomials influencing the edges of the Hermite functions, maps for such distributions often involve high-order terms with high-magnitude coefficients of opposing signs partially cancelling each other. While this is often unproblematic within the support of the training samples, the resulting high-magnitude coefficients can extend the influence of some Hermite functions farther into the tails, resulting in undesirable tail effects. As an alternative, we might prefer to employ a weight term that reverts the weights to zero at a finite distance. For instance, using a cubic spline weight yields a different *edge-controlled* Hermite polynomial

$$\mathcal{H}_j^{\mathrm{EC}}(x) = \mathrm{He}_j(x)\left(2\min(1, |x|/r)^3 - 3\min(1, |x|/r)^2 + 1\right), \tag{28}$$

where r is a finite edge-control radius specified by the user. Examples of such basis functions are illustrated in Figure 11B. When using these functions in the integrated-rectified formulation, this will create $\mathbf{S}$ that is linear outside of the hypercube $(-r, r)^d$.

Radial basis functions A second useful class of basis functions is *radial basis functions* (RBFs). RBFs exert more local influence than combinations of polynomial basis functions, and generally require two additional parameters: a position parameter μ and a scale (or bandwidth) parameter σ. To avoid the need to optimize these parameters along with the map's coefficients c_i, one can estimate μ and σ from the empirical quantiles along the marginals of the training samples $\{\mathbf{X}^i\}_{i=1}^N$ (Spantini et al., 2022). For the positions μ_i, they propose to place each RBF at the empirical quantiles $q_{i/(j+1)}\,(x_k)$ for $i = 1, \ldots, j$, where j is the total number of RBFs along the k-th marginal. Based on the resulting μ_i, the corresponding scales σ_i are determined by averaging the distances to neighbouring RBFs. To simplify notation, it can be useful to define local coordinates for each RBF:

$$x_k^{\text{loc},i} = \frac{x_k - \mu_i}{\sqrt{\tau}\sigma_i}, \tag{29}$$

where the τ is again twice the Archimedes' constant. Radial basis functions are particularly useful for multimodal distributions because they render the map selectively expressive wherever the target distribution's samples are located. Where a superposition of many high-order polynomials may be required to recover isolated modes, often only a few RBFs may suffice. Besides conventional RBFs, three useful related basis functions are *left edge terms* (LET), *integrated RBFs* (iRBF), and *right edge terms* (RET):

$$\begin{aligned}
\text{RBF}\left(x_k^{\text{loc},i}\right) &= \frac{1}{\sqrt{\tau}} \exp\left(-{x_k^{\text{loc},i}}^2\right), \\
\text{iRBF}\left(x_k^{\text{loc},i}\right) &= \frac{1}{2}\left(1 + \text{erf}\left(x_k^{\text{loc},i}\right)\right), \\
\text{LET}\left(x_k^{\text{loc},i}; \sigma_i\right) &= \frac{1}{2}\left(\sigma_i\sqrt{\tau}x_k^{\text{loc},i}\left(1 - \text{erf}\left(x_k^{\text{loc},i}\right)\right) - \frac{4\sigma_i}{\sqrt{\tau}}\exp\left(-{x_k^{\text{loc},i}}^2\right)\right), \\
\text{RET}\left(x_k^{\text{loc},i}; \sigma_i\right) &= \frac{1}{2}\left(\sigma_i\sqrt{\tau}x_k^{\text{loc},i}\left(1 + \text{erf}\left(x_k^{\text{loc},i}\right)\right) + \frac{4\sigma_i}{\sqrt{\tau}}\exp\left(-{x_k^{\text{loc},i}}^2\right)\right).
\end{aligned} \tag{30}$$

Illustrations of these functions are provided in Figure 12. Note that iRBFs, LETs, and RETs are monotone functions, and thus an excellent choice for a *linear separable* map (see Section 3.1.2). Other sigmoid-like functions and monotone "edge terms" that revert to linear in the tails include logistic and softplus functions, which may be computationally faster.

3.2 Optimization

With the structure and parameterization of the map component functions S_k defined, we may now address the question of how to identify the specific map $\mathbf{S}$ that relates the target distribution π to the reference distribution η. In general, we have two different ways to estimate the map, depending on the type of information available: *maps from samples*, and *maps from densities*. Both follow a very similar approach, but differ fundamentally in the direction in which they define the transport map: