



doi 10.26089/NumMet.v27r436

Electric currents for discrete data generation

Alexander S. Kolesov

Skolkovo Institute of Science and Technology (Skoltech),
Moscow, Russia

Artificial Intelligence Research Institute (AIRI),
Moscow, Russia

ORCID: 0009-0004-1711-0086, e-mail: a.kolesov@skoltech.ru

Stepan I. Manukhov

Lomonosov Moscow State University, Faculty of Physics,
Moscow, Russia

ORCID: 0000-0001-8915-2198, e-mail: manukhov2000akk@gmail.com

Vladimir V. Palyulin

Skolkovo Institute of Science and Technology (Skoltech),
Moscow, Russia

ORCID: 0000-0002-3047-6937, e-mail: v.palyulin@skoltech.ru

Alexander A. Korotin

Skolkovo Institute of Science and Technology (Skoltech),
Moscow, Russia

Artificial Intelligence Research Institute (AIRI),
Moscow, Russia

ORCID: 0000-0003-3773-342X, e-mail: a.korotin@skoltech.ru

Abstract: In this paper, we propose a new method for data generation in discrete settings called Electric Current Discrete Data Generation (ECD²G). The method establishes an analogy between electric current flow in a circuit and the transfer of probability mass between data distributions. Samples from the source distribution are treated as current input nodes of a circuit, while samples from the target distribution are interpreted as current output nodes. A neural network is used to learn the distribution of electric currents describing the probability flows in the circuit. To map the source distribution to the target one, source samples are transferred through the circuit pathways according to the learned currents. This process ensures the transfer between data distributions without loss of probability in the exact circuit construction. The results of numerical computational experiments are presented to illustrate applicability and effectiveness of the proposed ECD²G method.

Keywords: discrete data generation, electric currents, probability flow, distribution transfer.

For citation: A. S. Kolesov, S. I. Manukhov, V. V. Palyulin and A. A. Korotin, “Electric currents for discrete data generation,” Numerical Methods and Programming. **27** (4), 585–602 (2026). doi 10.26089/NumMet.v27r436.

Электротехнический вычислительный метод для генерации дискретных данных

А. С. Колесов

Сколковский институт науки и технологий (Сколтех),
Москва, Российская Федерация

Институт искусственного интеллекта,
Москва, Российская Федерация

ORCID: 0009-0004-1711-0086, e-mail: a.kolesov@skoltech.ru

С. И. Манухов

Московский государственный университет имени М. В. Ломоносова,
физический факультет,
Москва, Российская Федерация

ORCID: 0000-0001-8915-2198, e-mail: manukhov2000akk@gmail.com

В. В. Палюлин

Сколковский институт науки и технологий (Сколтех),
Москва, Российская Федерация

ORCID: 0000-0002-3047-6937, e-mail: v.palyulin@skoltech.ru

А. А. Коротин

Сколковский институт науки и технологий (Сколтех),
Москва, Российская Федерация

Институт искусственного интеллекта,
Москва, Российская Федерация

ORCID: 0000-0003-3773-342X, e-mail: a.korotin@skoltech.ru

Аннотация: В статье предлагается новый метод ECD²G для генерации дискретных данных, основанный на теории электрических цепей (от англ. Electric Current Discrete Data Generation). Метод устанавливает аналогию между протеканием электрического тока в цепи и переносом вероятностной массы между распределениями данных. Выборки из исходного распределения интерпретируются как узлы входа тока в цепь, а выборки из целевого распределения рассматриваются как узлы вытекания тока. Нейросетевой подход позволяет выучить распределение электрических токов в цепи, описывающих потоки вероятности. Для отображения исходного распределения данных в целевое распределение метод осуществляет перенос данных по цепи в соответствии с токами, обученными нейросетью. При корректном построении электрической цепи данный процесс гарантированно обеспечивает перенос между распределениями данных без потери вероятности. Приводятся данные численных экспериментов, подтверждающие работоспособность метода ECD²G.

Ключевые слова: генерация данных с дискретной структурой, электрические токи, поток вероятности, перенос данных.

Для цитирования: Колесов А.С., Манухов С.И., Палюлин В.В., Коротин А.А. Электротехнический вычислительный метод для генерации дискретных данных // Вычислительные методы и программирование. 2026. 27, № 4. 585–602. doi 10.26089/NumMet.v27r436.

1. Introduction. Generative modeling is a challenging problem whose modern solutions often draw direct inspiration from concepts in physics and the theory of stochastic processes. From the viewpoint of numerical methods, a generative model is not only a probabilistic construction, but also requires computational procedures for approximating the underlying dynamics, estimating transition rules, and performing stable sampling. These issues are especially important for discrete data such as text, graphs, and molecular data, where the state space is finite but combinatorially large.

Recently, Diffusion Models (DM) have become one of the most popular classes of generative methods. In continuous settings, their formulation is closely connected with stochastic or deterministic differential equations, and practical generation depends on numerical approximation of the forward and reverse diffusion processes [1–4]. A number of recent works have adapted this methodology to discrete domains [5–7]. Instead of corrupting data with Gaussian noise, these models define a forward process that gradually corrupts discrete structures (e.g., tokens) through random transitions to other discrete states or a masking token. The ability to approximate and reverse this process has led to impressive results in various applications ranging from text generation [6] to molecule design [8].

At the same time, electrostatic-based generative models have emerged as another physics-inspired framework for continuous data. These approaches interpret data samples as electrostatic charges in an electrostatic field. In pioneering works [9, 10], the field created by data points on a plate inside a hemisphere was studied, which made it possible to generate data from the uniform distribution on the hemisphere. Subsequent works [11, 12] introduced a capacitor-based paradigm, in which the field between charged plates is learned to enable data-to-data mapping. This family of models is also intrinsically computational, as sampling requires approximating of fields, potentials, or particle trajectories.

However, there is a significant gap: despite their high efficiency in continuous settings, electrostatic theory-based methods have not yet been formulated for generative modeling in discrete data spaces. The fundamental concepts of continuous charge and potential have no direct analogues in discrete domains, which presents a novel and open research challenge. In this work, we use a discrete numerical route and replace continuous electrostatic fields by electric currents on a finite graph. This leads to computational problems that are natural for numerical analysis: solving or approximating potentials and currents in a circuit, estimating them from samples, and using the resulting current field to define a sampling procedure.

We introduce Electric Current Discrete Data Generation (ECD²G) method, which enables to generate discrete data grounded in the principles of electrical engineering. We establish a theoretical connection between discrete data generation and current flow in a circuit, and formulate the corresponding numerical approximation procedure for learning and using these currents. The proposed framework provides a new paradigm for both noise-to-data and data-to-data generation. The main idea of the proposed construction is summarized in Figure 1, where probability mass enters the layered circuit at the source distribution and leaves it at the target distribution.

2. Background. Let us recall the basic principles of electrical engineering to better understand the physical motivation behind our method. In this section, we discuss the graph representation of a circuit, Kirchhoff’s and Ohm’s laws, and the superposition principle. Further details on these electrical engineering concepts can be found in any textbook on the subject, for instance, in [13].

An electric circuit can be considered as a graph $\mathcal{G}(V, E)$, where V is the set of vertices (nodes) and E is the set of edges. Each edge is associated with a resistor between two nodes. An edge $e(v, v') \in E$ between vertices $v \in V$ and $v' \in V$ has a resistance $R(v, v')$. We manually select a subset of vertices called *sources* and denote them by $V^{\text{src}} \subset V$. A current is injected into the circuit through these source vertices. We denote the

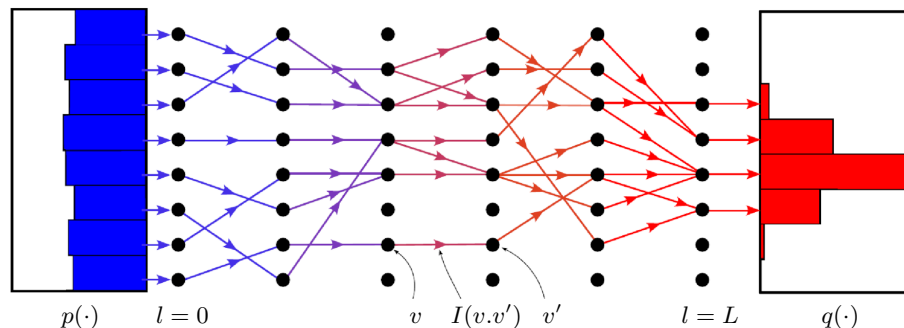


Figure 1. Our ECD²G method places two data distributions $p(x_0)$ and $q(x_L)$ on the input layer $\ell = 0$ and output layer $\ell = L$ of a circuit, respectively. The distribution $p(x_0)$ is interpreted as a current flowing into the circuit, and $q(x_L)$ as a current flowing out. The flow of electric current in the circuit is associated with the transfer of probability mass.

As it is flowing through the circuit, the electric current transforms samples from the source distribution p into samples from the target distribution q

current flowing into each vertex v by the function $I^{\text{src}}(v)$, which equals zero if $v \notin V^{\text{src}}$. Similarly, we define another subset of vertices $V^{\text{snk}} \subset V$ called *sinks*, such that $V^{\text{src}} \cap V^{\text{snk}} = \emptyset$. We denote the currents flowing out of these sink vertices by the function $I^{\text{snk}}(v)$, which equals zero if $v \notin V^{\text{snk}}$.

As currents flow from the sources V^{src} to V^{snk} , they obey the following physical laws:

1. *Kirchhoff's current law.* Consider any vertex $v \in V$. There exists a set of edges $\{e(v, v')\}_{v' \in J(v)}$ connecting v to all its neighboring vertices, where the set of these neighbors is denoted by $J(v)$. Let $J_{\text{in}}(v)$ be the set of neighboring vertices from which current flows into v , and let $J_{\text{out}}(v)$ be the set of neighboring vertices to which current flows out from v . If the current between v and a vertex $v' \in J(v)$ is equal to zero, we assign v' to $J_{\text{in}}(v)$ by default. The sets $J_{\text{in}}(v)$ and $J_{\text{out}}(v)$ are disjoint and their union is $J(v) = J_{\text{in}}(v) \cup J_{\text{out}}(v)$.

Kirchhoff's current law states that the algebraic sum of all currents entering any node must equal zero. Equivalently, the total current entering a node equals the total current leaving it. For any node $v \notin V^{\text{src}} \cup V^{\text{snk}}$:

$$\sum_{v' \in J_{\text{in}}(v)} I(v, v') = \sum_{v' \in J_{\text{out}}(v)} I(v, v'), \quad (1)$$

where $I(v, v')$ denotes the current flowing from v to v' . If a node $v \in V^{\text{src}}$ or $v \in V^{\text{snk}}$ is considered, then equation (1) becomes:

$$I^{\text{src}}(v) = \sum_{v' \in J_{\text{out}}(v)} I(v, v'), \quad I^{\text{snk}}(v) = \sum_{v' \in J_{\text{in}}(v)} I(v', v).$$

A direct consequence of applying the Kirchhoff's law to every node in the circuit is that the total current entering the circuit must equal the total current leaving it. For the circuit with a source vertex set V^{src} and a sink vertex set V^{snk} , the following equality holds:

$$\sum_{v \in V^{\text{src}}} I^{\text{src}}(v) = \sum_{v \in V^{\text{snk}}} I^{\text{snk}}(v), \quad (2)$$

where the left-hand side of (2) represents the total injected current at the source nodes while the right-hand side is the total extracted current at the sink nodes. The graph representation of the circuit and the introduced source/sink notations are illustrated in Figure 2, while Figure 3 shows the local balance of incoming and outgoing currents at a vertex as required by Kirchhoff's current law.

2. *Ohm's law.* In a circuit, there exists a function $\varphi : V \rightarrow \mathbb{R}$, called the electric potential, which is defined up to a constant and fully describes the set of currents in the circuit. For arbitrary neighboring nodes v and v' in the circuit φ satisfies:

$$\frac{\varphi(v) - \varphi(v')}{R(v, v')} = I(v, v'), \quad (3)$$

where $R(v, v')$ and $I(v, v')$ are the resistance and the current between the nodes, respectively. The difference in potentials $\varphi(v) - \varphi(v')$ for neighboring vertices is called the voltage and denoted as $U(v, v')$.

3. *Superposition principle.* Consider two systems of sources and sinks on $\mathcal{G}(V, E)$:

$$\{I_1^{\text{src}}(v)\}_{v \in V_1^{\text{src}}}, \{I_1^{\text{snk}}(v)\}_{v \in V_1^{\text{snk}}}$$

and

$$\{I_2^{\text{src}}(v)\}_{v \in V_2^{\text{src}}}, \{I_2^{\text{snk}}(v)\}_{v \in V_2^{\text{snk}}}.$$

Suppose, the first system establishes potentials $\varphi_1(v)$ and the second system establishes potentials $\varphi_2(v)$, where $v \in V$. Then, the two systems merged together will produce a potential

$$\varphi(v) = \varphi_1(v) + \varphi_2(v), \quad (4)$$

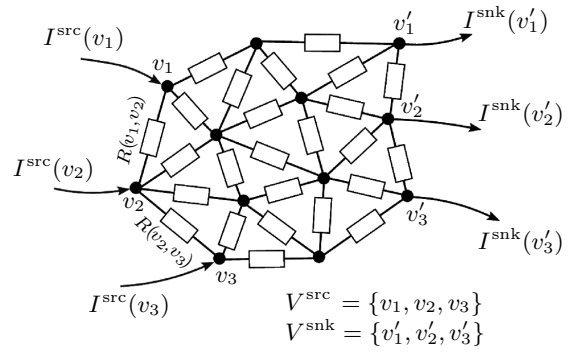


Figure 2. An illustration of an electric circuit

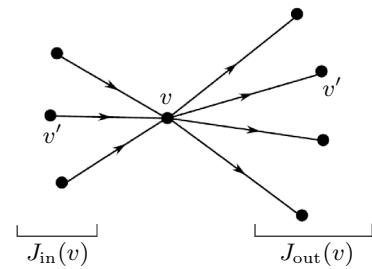


Figure 3. Incoming and outgoing currents at a vertex v



which also means that current in the merged system is additive for all $v, v' \in V$:

$$I(v, v') = I_1(v, v') + I_2(v, v'). \quad (5)$$

3. ECD²G method. This section presents a novel generative modeling paradigm grounded in electrical engineering for discrete data settings. It is called “Electric currents discrete data generation”. In subsection 1 of section 3, we formulate the task of distribution transfer and give an intuitive description of our proposed method. In subsection 2, we provide the theoretical foundation of ECD²G method and the main propositions for a connected graph. In subsections 3 and 4, we specify a graph structure for an electric circuit and formulate the learning and inference algorithms for this graph, respectively.

3.1. Problem setup and intuitive explanation. Let the state space be the discrete product space $\mathcal{X} = \mathbb{S}^D$, where $\mathbb{S}$ is a finite set of categories, e.g., $\mathbb{S} = \{1, 2, \dots, S\}$. A source distribution $p \in \mathcal{P}(\mathbb{S}^D)$ and a target distribution $q \in \mathcal{P}(\mathbb{S}^D)$ are considered. Here $\mathcal{P}$ is the Borel measure. These data distributions are assumed to be available through random empirical samples $z_1, \dots, z_N \sim p$ and $y_1, \dots, y_M \sim q$, where each sample is a D -dimensional vector of categories. In these notations, the goal is to construct a procedure T that transforms a new sample $x^{\text{new}} \sim p$ into a sample distributed as q , i.e., $T(x^{\text{new}}) \sim q$.

Let us turn to the intuitive explanation of our ECD²G approach. To implement transfer in the full discrete state space $\mathcal{X} = \mathbb{S}^D$, we construct a current circuit $\mathcal{G}(V, E)$, whose source layer is a copy $\mathcal{X}_{(0)}$ of $\mathcal{X}$ and whose sink layer is another copy $\mathcal{X}_{(L)}$. We interpret a current $I^{\text{src}}(x_0)$ flowing into a source node $x_0 \in \mathcal{X}_{(0)}$ as the source probability $p(x_0)$. Analogously, a current $I^{\text{snk}}(x_L)$ flowing out of a sink node $x_L \in \mathcal{X}_{(L)}$ is considered as the target probability $q(x_L)$. Thus, the circuit is defined on full states rather than on isolated categories. In the present implementation and experiments, we work with full states and do not use a coordinate-wise factorization inside the sampler. Factorized sampling and normalization are treated as a scalability direction for high-dimensional spaces in section 6. By associating the electric current in the circuit with the transfer of probability mass, we map samples $x^{\text{new}} \sim p$ through the circuit to the target distribution q .

3.2. Main theoretical construction of our ECD²G method. For convenience, we consider a general electric circuit as a connected graph $\mathcal{G}(V, E)$. A particular practical example of realization is given in the next subsection. We assume that each edge $e(v, v') \in E$ has a resistance $R(v, v')$. Recall that from Kirchhoff’s law (2) the total incoming current I^{src} equals the total outgoing current I^{snk} . Without loss of generality, we suppose that the total incoming current and the total outgoing current are each equal to 1:

$$\sum_{v \in V^{\text{src}}} I^{\text{src}}(v) = \sum_{v' \in V^{\text{snk}}} I^{\text{snk}}(v') = 1. \quad (6)$$

This allows us to interpret I^{src} and I^{snk} as some distributions p and q on V^{src} and V^{snk} :

$$\forall v \in V^{\text{src}} : I^{\text{src}}(v) = p(v) \quad \text{and} \quad \forall v' \in V^{\text{snk}} : I^{\text{snk}}(v') = q(v').$$

The transfer of probability mass from source to sink is realized by the stochastic walk on the circuit. We set up the following rule for the steps between neighboring vertices.

Definition 1 (Movement rule). *Let $J_{\text{out}}(v)$ contain the neighbors to which positive current leaves v in the chosen orientation. The probability of transition from vertex $v \in V : v \notin V^{\text{snk}}$ to a neighbor $v' \in J_{\text{out}}(v)$ is defined as:*

$$\text{prob}(v'|v) \stackrel{\text{def}}{=} \frac{I(v, v')}{\sum_{v'' \in J_{\text{out}}(v)} I(v, v'')}. \quad (7)$$

If $v \in V^{\text{snk}}$, then the extracted current $I^{\text{snk}}(v)$ is treated as a terminal outgoing edge from v to an absorbing terminal event associated with this sink. In the layered practical construction used below, sinks are placed only on the last layer and have no ordinary outgoing edges. The following more general notation also covers circuits where an ordinary outgoing current from a sink is present: the flow either terminates at this sink with probability $\text{prob}(\text{term}|v)$ or continues to vertex v' with probability $\text{prob}(v'|v)$ as follows:

$$\text{prob}(\text{term}|v) \stackrel{\text{def}}{=} \frac{I^{\text{snk}}(v)}{I^{\text{snk}}(v) + \sum_{v'' \in J_{\text{out}}(v)} I(v, v'')}, \quad \text{prob}(v'|v) \stackrel{\text{def}}{=} \frac{I(v, v')}{I^{\text{snk}}(v) + \sum_{v'' \in J_{\text{out}}(v)} I(v, v'')}. \quad (8)$$

The denominator represents the total current flowing out of vertex v , including the terminal edge at sink vertices. Therefore, the probabilities (7) and (8) of movement to neighboring vertices or to a terminal event are proportional to the current in the corresponding edges. After determining the one-step transition (7) in the circuit, it should be noted that the following termination property holds for the multi-step transition in the circuit.

Proposition 1 (Finite-step termination). *If the movement through the circuit starts at vertices V^{src} and obeys the movement rule 1, it reaches a terminal event associated with a node in V^{snk} in a finite number of steps.*

Proof. We add to every sink $v \in V^{\text{snk}}$ an absorbing terminal event connected by an edge carrying current $I^{\text{snk}}(v)$. Since the graph is finite, infinite movement in the extended graph is only possible if there exists a cycle of vertices $v_1, v_2, \dots, v_C$ along edges with positive current (here $v_C = v_1$). However, current flows from higher to lower potential. For such a cycle to sustain a current, the potentials must satisfy $\varphi(v_1) > \varphi(v_2) > \dots > \varphi(v_C) = \varphi(v_1)$, which is a contradiction. Thus, the movement reaches one of the terminal events in a finite number of steps. $\square$

Proposition 1 allows us to define a *stochastic* mapping $T : V^{\text{src}} \rightarrow V^{\text{snk}}$ that transforms sources to sinks according to movement rule 1. The following corollary states the induced distribution-transfer property.

Corollary 1 (ECD²G transfer). *For the circuit described above, the sequential movement by T from $v \in V^{\text{src}}$ terminates at a sink $T(v) \in V^{\text{snk}}$ with probability 1. Moreover, if $v \sim p$ is a random variable, then $T(v) \sim q$, i.e. the movement through the circuit with probabilities proportional to the currents transforms distribution p on V^{src} into distribution q on V^{snk} .*

This result follows directly from classical network flow theory [14, §6] and properties of absorbing Markov chains [15, §11]. The conservation (1) and normalization (6) conditions establish a feasible unit flow from sources V^{src} to the terminal sink events. The movement rule (7) defines an absorbing Markov chain on the extended graph. The total flow through the terminal edge of a sink $v' \in V^{\text{snk}}$ is $I^{\text{snk}}(v') = q(v')$, so the absorption probabilities at the terminal sink events are exactly q . For a rigorous proof of this absorption property for Markov chains defined by unit flows see, e.g., [16, §3].

3.3. Practical realization. In practice, we are interested in transforming distributions on a particular discrete space $\mathcal{X} = \mathbb{S}^D$. We consider a specific structure of an electric circuit as a layered graph $\mathcal{G}(V, E)$ with vertex layers indexed by $\ell = 0, \dots, L$. This graph consists of $|V| = (L + 1)|\mathcal{X}| = (L + 1)S^D$ vertices, where each layer is a copy of $\mathcal{X}$. We denote the set of vertices on the ℓ -th layer as $\mathcal{X}_{(\ell)}$, then $V = \mathcal{X}_{(0)} \cup \dots \cup \mathcal{X}_{(L)}$. In this layered realization, the source vertices are exactly $\mathcal{X}_{(0)}$ and the sink vertices are exactly $\mathcal{X}_{(L)}$; intermediate layers contain no sinks. The corresponding single source-sink layered circuit is sketched in Figure 4. Edges E of this graph can connect only vertices from neighboring layers. Thus, $E = \bigcup_{\ell=0}^{L-1} \mathcal{X}_{(\ell)} \times \mathcal{X}_{(\ell+1)}$.

We denote the potentials, currents, and resistances with an additional layer index ℓ as

$$\varphi_\ell(x_\ell), I_{\ell, \ell+1}(x_\ell, x_{\ell+1}), R_{\ell, \ell+1}(x_\ell, x_{\ell+1}).$$

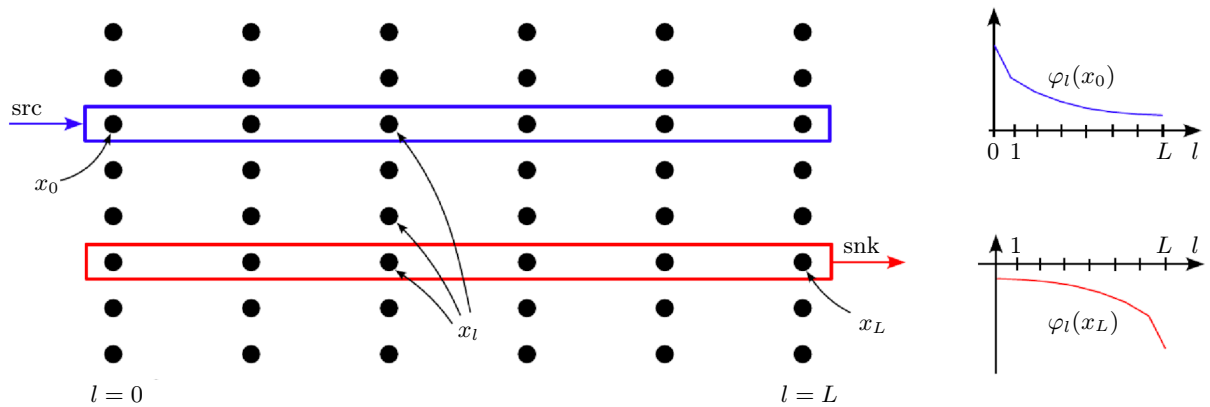


Figure 4. A sketch of single source-sink configuration for electric circuit $\mathcal{G}_L(|\mathcal{X}|, R, r)$

In addition, the resistance is set as follows:

$$R_{\ell,\ell+1}(x_\ell, x_{\ell+1}) = \begin{cases} r, & x_\ell = x_{\ell+1}, \\ R, & x_\ell \neq x_{\ell+1}. \end{cases} \quad (9)$$

In other words, the “forward” resistance is r , and all “lateral” resistances are equal and denoted by R . Thus, using $R > r$ in practice, we encourage the preservation of the state at the next layer rather than its change. By introducing the resistances r and R into the circuit, we can now define the electrical circuit as a layered graph $\mathcal{G}_L(|\mathcal{X}|, R, r)$.

Calculation of potentials φ_ℓ for $\mathcal{G}_L(|\mathcal{X}|, R, r)$ and arbitrary current distributions $I^{\text{src}}, I^{\text{snk}}$ is a challenging task, especially when the distributions are only available through a limited amount of empirical samples (subsection 1). However, using the superposition principle (4), it can be reduced to solving the problem for a single unit input current at $x_0 \in \mathcal{X}_{(0)}$, which flows out at $x_L \in \mathcal{X}_{(L)}$. Because potentials are defined up to an additive constant, all single source-sink potentials below are taken in a shared gauge, for example, by imposing the same zero-mean normalization over the circuit vertices. The potentials themselves are not unique: adding the same constant to all potentials does not change any current. Only potential differences are physically meaningful and uniquely determine the currents. Therefore, the same gauge must be fixed for all single source-sink problems before their potentials are combined by the superposition principle. Let $\varphi_\ell(x_\ell|x_0, x_L)$ define the potential created at point x_ℓ in such a system. Then, the superposition principle yields

$$\varphi_\ell(x_\ell) = \sum_{x_0, x_L} \varphi_\ell(x_\ell|x_0, x_L) \pi(x_0, x_L), \quad (10)$$

where $\pi(x_0, x_L)$ is an arbitrary joint distribution on $\mathcal{X}_{(0)} \times \mathcal{X}_{(L)}$ satisfying the conditions $\sum_{x_L} \pi(x_0, x_L) = I^{\text{src}}(x_0)$ and $\sum_{x_0} \pi(x_0, x_L) = I^{\text{snk}}(x_L)$. Since the right-hand side of (10) is determined (up to a constant) uniquely by the marginal input $I^{\text{src}}(x_0)$ and output $I^{\text{snk}}(x_L)$ current distributions, the following observation arises.

Remark 1 (Transfer plan independence). *Potential $\varphi_\ell(x_\ell)$ is independent of $\pi(x_0, x_L)$ under the shared gauge fixed above. The corresponding currents are gauge-invariant.*

The following result provides a recipe for calculating potentials $\varphi_\ell(x_\ell|x_0, x_L)$ in one source/one sink system. Applying Kirchhoff’s current law to each vertex of the layered circuit and using Ohm’s law to express every edge current as a potential difference divided by the corresponding resistance, one obtains the node-voltage equations for the circuit. Considering the symmetry of the single source-sink setting, the vertices at each layer can be grouped into three types: the source state x_0 , the sink state x_L , and all other states $x' \in \mathcal{X} \setminus \{x_0, x_L\}$. This reduction yields the system of equations established in Lemma 1.

Lemma 1 (Single source-sink potentials). *Let a unit current flow into the electrical circuit $\mathcal{G}_L(|\mathcal{X}|, R, r)$ at position $x_0 \in \mathcal{X}_{(0)}$ and flow out at position $x_L \in \mathcal{X}_{(L)}$. Then the potentials $\varphi_\ell(x_\ell|x_0, x_L)$ at the vertices for each layer ℓ of this graph are determined by a system of $3(L+1) \times 3(L+1)$ equations (A1.2.4.1)–(A1.2.4.3).*

The proof of this lemma is given in Appendix 1.

Finally, using the Ohm’s law (3), Lemma 1 and the superposition principle (4), (5), one may compute the electric current between x_ℓ and $x_{\ell+1}$:

$$I_{\ell,\ell+1}(x_\ell, x_{\ell+1}) = \frac{\varphi_\ell(x_\ell) - \varphi_{\ell+1}(x_{\ell+1})}{R_{\ell,\ell+1}(x_\ell, x_{\ell+1})} = \mathbb{E}_{(x_0, x_L) \sim \pi} \left[\frac{\varphi_\ell(x_\ell|x_0, x_L) - \varphi_{\ell+1}(x_{\ell+1}|x_0, x_L)}{R_{\ell,\ell+1}(x_\ell, x_{\ell+1})} \right], \quad (11)$$

where $\mathbb{E}_\pi$ means the expectation over distribution π . Using the derived closed form for $\varphi_\ell(x_\ell|x_0, x_L)$, we can now estimate the current employing random samples $(x_0, x_L) \sim \pi$ for any coupling π of p and q . This is the key component of our method.

3.4. Learning and inference algorithms. Training (Algorithm 1). Recall that $\mathcal{X} = \mathbb{S}^D$. The algorithm below is written for full-state pair. We do not use coordinate-wise factorization in the reported experiments. To recover currents $I_{\ell,\ell+1}(x_\ell, x_{\ell+1})$ in a circuit, we approximate them using a neural net $I_\theta(\cdot) : \mathbb{S}^D \times \mathbb{S}^D \times \{0, \dots, L-1\} \rightarrow \mathbb{R}$. It takes as input the states (nodes in the circuit) $x_\ell \in \mathcal{X}_{(\ell)}$ and $x_{\ell+1} \in \mathcal{X}_{(\ell+1)}$, and the edge-layer index $\ell \in \{0, \dots, L-1\}$ of the current state as a condition. The output of the network is a value $I_\theta(x_\ell, x_{\ell+1}, \ell) \approx I_{\ell,\ell+1}(x_\ell, x_{\ell+1})$.

Algorithm 1. Training

```

1: input Distributions  $p(x)$  and  $q(y)$  accessible by samples;
   Resistances  $r > 0$  and  $R > 0$ ;
   Layered graph (circuit)  $\mathcal{G}_L(|\mathbb{S}^D|, R, r)$ ;
   NN approximator  $I_\theta$ ;
    $N$  training steps;

2: output The learned currents  $I_\theta$ ;

3: For  $1, \dots, N$  steps:
4:   Sample batches of  $x \sim p$  and  $y \sim q$ ;
5:   Set  $x_0 \equiv x$  and  $x_L \equiv y$  on  $\mathcal{G}_L(|\mathbb{S}^D|, R, r)$ ;
6:   Sample batch of edge-layer indices  $\ell \sim \mathcal{U}(\{0, \dots, L-1\})$ ;
7:   Sample batch of states  $x_\ell, x_{\ell+1} \sim \mathcal{U}(\mathcal{X})$ ;
8:   Approximate  $I_{\ell, \ell+1}(x_\ell, x_{\ell+1})$  with  $(x_0, x_L)$  using (11);
9:   Minimize the MSE between  $I_\theta(x_\ell, x_{\ell+1}, \ell)$  and the estimated current  $I_{\ell, \ell+1}(x_\ell, x_{\ell+1})$ .

10: stop

```

Since the circuit consists of $(L+1)S^D$ nodes, considering all possible pairs of nodes in neighboring layers for current recovery is an infeasible task. Therefore, we randomly select them for training. Namely, we sample the edge-layer index ℓ from the discrete uniform distribution on $\{0, \dots, L-1\}$ and randomly select states $x_\ell \in \mathcal{X}_{(\ell)}$ and $x_{\ell+1} \in \mathcal{X}_{(\ell+1)}$ from $\mathcal{U}(\mathcal{X})$, where $\mathcal{U}$ is the uniform distribution. The target current $I_{\ell, \ell+1}(x_\ell, x_{\ell+1})$ is estimated using (11). We additionally sample a random batch $(x_0, x_L) \sim p \times q$ to obtain a Monte-Carlo estimate $\hat{I}_{\ell, \ell+1}(x_\ell, x_{\ell+1})$ for (11). Then, we train a neural network I_θ to approximate the current. The resulting loss for training is given by the following expression:

$$\mathbb{E}_{\ell \sim \mathcal{U}(\{0, \dots, L-1\})} \mathbb{E}_{x_\ell, x_{\ell+1} \sim \mathcal{U}(\mathcal{X})} \|I_\theta(x_\ell, x_{\ell+1}, \ell) - \hat{I}_{\ell, \ell+1}(x_\ell, x_{\ell+1})\|_2^2 \rightarrow \min_\theta.$$

For modeling the current flow starting at any input $x = x_0 \sim p$, we use the movement rule from Definition 1 in the layered graph. In the reported low-dimensional experiments, the candidate set at step ℓ is the next layer $\mathcal{X}_{(\ell+1)}$, so all terms in the normalizer can be computed explicitly. Since the learned current I_θ is a signed approximation, the sampling rule uses only the forward positive part of this current:

$$w_\theta(x_{\ell+1}|x_\ell, \ell) = \max\{I_\theta(x_\ell, x_{\ell+1}, \ell), 0\}.$$

Thus, currents predicted to flow in the reverse direction are not taken into account during sampling. We set them to zero and renormalize the remaining probabilities:

$$\text{prob}_\theta(x_{\ell+1}|x_\ell) = \frac{w_\theta(x_{\ell+1}|x_\ell, \ell)}{\sum_{z \in \mathcal{X}_{(\ell+1)}} w_\theta(z|x_\ell, \ell)}. \quad (12)$$

This positive-current projection and renormalization is an approximation used by the learned sampler. If the denominator in (12) is zero, we use the self-transition $x_{\ell+1} = x_\ell$, which is available in the layered graph. The movement always stops at the last layer $\ell = L$, whose vertices are the only sinks in our practical construction.

4. Related works. Generative Flow Networks (GFlowNets) [17] form a family of models that learn to generate discrete structures with a probability proportional to a given reward function. Although the problem they consider differs from ours, their approach shares similarities with our work. They also consider the flow of probability mass through a graph. However, GFlowNets learn flows or policies over graph edges, whereas in our method we first predefine the edge resistances of an electric circuit and then compute or approximate the corresponding currents.

Discrete Diffusion Models (DDM) [6, 18, 19] form a powerful class for data generation in discrete data setups. Unlike our method, they use a forward (data destruction) process based on a *predefined* transition matrix. As a result, the forward process of DDM, using either a uniform [5] or an absorbing [7] transition matrix, corrupts clean data either into a uniformly distributed sample or a special token. In contrast, our approach is theoretically able to learn transformations between any discrete data distributions.



Discrete Flow Matching (DFM) [18] learns a flow field between two data distributions. In contrast to our ECD²G method, the flow learned by DFM depends on the data transfer plan π used for training, while our flow is the uniquely determined flow of the electric current (Remark 1). At the estimator level, our current in an arbitrary edge $(x_\ell, x_{\ell+1})$ can be evaluated by Monte-Carlo averaging over samples from π , whereas FM/DFM estimators are usually evaluated at sampled interpolant points [20, §1.2].

Electrostatic-based methods [10–12] interpret data distributions $p(\cdot)$ and $q(\cdot)$ as positive and negative charge densities, respectively. Data transfer and/or generation are performed by moving along the field lines created by these charge distributions. The theoretical justification for these methods relies on the principle of flux conservation in the generated field. The proposed ECD²G model also employs electricity theory, where flux conservation corresponds to the current law (1). However, all existing electrostatic-based models operate in a continuous setting, whereas ECD²G performs discrete data generation.

5. Computational experiments. In this section, we present proof-of-concept experiments for our proposed ECD²G method. We demonstrate a one-dimensional transfer task in subsection 1 as well as a two-dimensional one in subsection 2. Further experimental details are provided in Appendix 2.

5.1. 1D experiment. The first intuitive test to validate our approach is to solve a transfer task between two one-dimensional categorical distributions, whose densities and current flows can be visualized. We assume the source p is uniformly distributed over the 50 categories $\{0, \dots, 49\}$ and the target q is the normalized discretization of Gaussian weights $\exp(-(s - 25)^2/2)$ on the same categorical support. Both distributions use $S = 50$ categories. Their visualizations are shown in Figure 5. We represent the circuit between p and q as a layered graph $\mathcal{G}_L(50, R, r)$ with $L = 9$ and layers $\ell = 0, \dots, 9$, where resistances r and R are 0.1 and 100, respectively.

We train a neural network I_θ using our Algorithm 1 and evaluate it at the inference stage (see subsection 4) stage. The reference current field in the exact circuit model between the input and output distributions, computed according to (11), is shown in Figure 5. Our approach maps to the target density quite well, recovering the target distribution (see Figure 6). We also compare the target current flow obtained from (11) with the

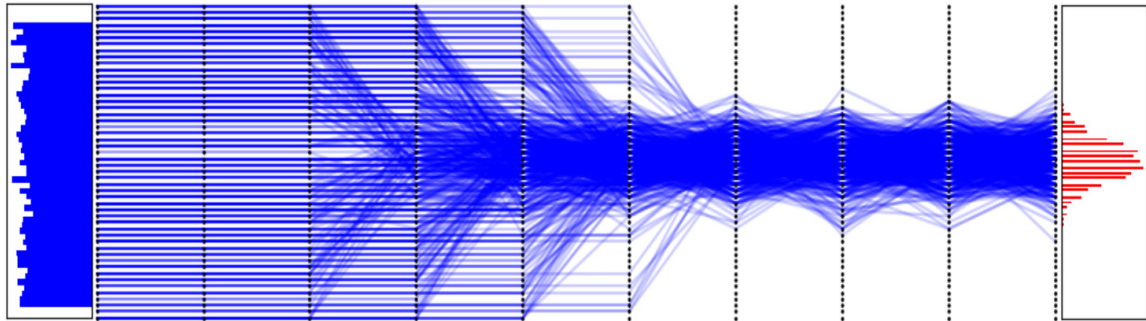


Figure 5. Illustrative 1D Uniform→Gaussian experiment: input and target samples $x_0 \sim p(x_0)$ and $x_9 \sim q(x_9)$ together with the target currents obtained from (11) on the layered graph $\mathcal{G}_9(50, 100, 0.1)$

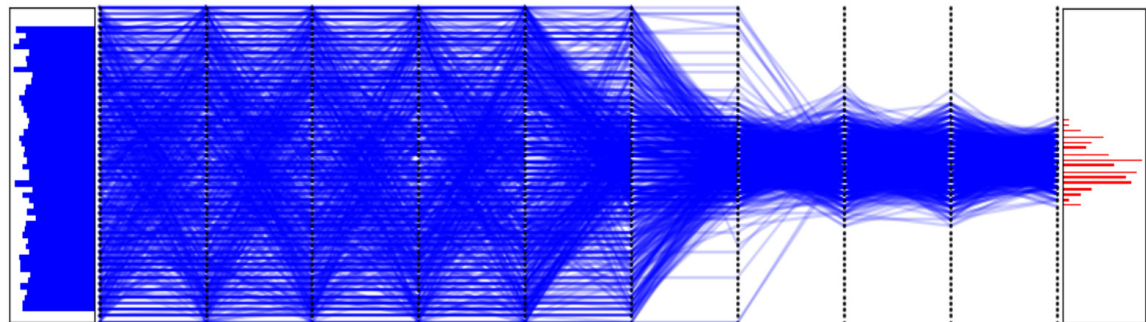


Figure 6. Illustrative 1D Uniform→Gaussian experiment: input samples $x_0 \sim p(x_0)$ and samples $x_9 \sim T(x_0)$ mapped by $T(x)$ together with the currents learned by I_θ on the layered graph $\mathcal{G}_9(50, 100, 0.1)$

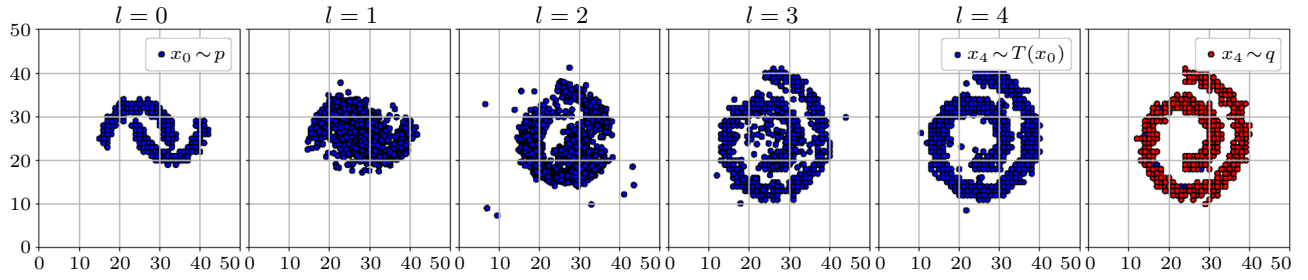


Figure 7. Illustrative 2D Moons-to-Swiss-roll experiment: input and target distributions $p(x_0)$ and $q(x_4)$ together with the intermediate distributions at layers $\ell = 1, \dots, 3$ and samples transformed from $x \sim p$ by $T(x)$ in the layered graph $\mathcal{G}_4(2500, 10, 0.1)$

currents learned by I_θ . The current field learned by the neural network is shown in Figure 6. Despite the more stochastic structure of the learned currents, the transported samples accurately recover the target distribution.

5.2. 2D Moons-to-Swiss-roll experiment. We also provide a proof-of-concept for our method by transferring between two-dimensional data distributions. We consider the 2D Moons dataset [11] to form the source distribution p and a 2D projection of the Swiss roll dataset [11] to form the target q . The main idea of the experiment is to provide interpolation of samples from p to q along a layered graph. Their visualizations are shown in Figure 7. Both datasets are discretized into $S = 50$ categories, resulting in a two-dimensional categorical space of $S^D = 50^2 = 2500$ points. The graph structure for this experiment is $\mathcal{G}_L(2500, R, r)$ with $L = 4$, $R = 10$ and $r = 0.1$. Our approach maps to the target density effectively, which allows to successfully recover the target distribution as demonstrated in Figure 7.

For completeness, we compare the results of applying our method to the 2D Moons-to-Swiss-roll data with the ones obtained using the baselines DDM [5] and DFM [18]. We use the Maximum Mean Discrepancy (MMD) metric with the radial basis function (RBF) kernel [21], where the bandwidth is chosen by the median heuristic $\sigma = \sigma_{\text{median}}$. The MMD is evaluated between 2048 generated and 2048 original samples from the Swiss roll. The Monte-Carlo batch size used to estimate target currents in (11) is 256. For the DFM/FM baseline implementation, we follow the public codebase `facebookresearch/flow_matching` as closely as possible and adapt it to our discrete setting. The central values of MMD with standard deviations for the corresponding methods over 5 runs are presented in Table 1.

Table 1. The values of the MMD metric for the ECD²G, DDM and DFM approaches, calculated over 5 runs with a sample size of 2048

	ECD ² G	DDM	DFM
MMD ($\times 10^{-3}$)	1.24 ± 0.18	1.67 ± 0.15	2.32 ± 0.22

6. Conclusion. Our ECD²G method establishes an analogy between the transfer of samples from one distribution to another and the flow of electric current through a circuit. This method not only ensures the transfer, but also give an opportunity to solve the problem of factorization. Illustrations and comparisons with baseline methods are presented, demonstrating the capability and robustness of our approach. Below, we outline the limitations of our method and directions for future work.

Our training approach involves sampling layers ℓ and states x_ℓ within these layers. We use uniform discrete distributions over edge-layer indices and full states in $\mathcal{X}$ (see subsection 4 of section 3). This allows us to approximate the current in the circuit between states x_ℓ and $x_{\ell+1}$. However, this sampling is not unique, and there are many other possible ways to define a training volume for efficient learning. This is a promising direction for further research.

Circuit topology and edge resistances are important design choices in our approach. The layered construction used in this work is one convenient realization: it encourages a flow to remain at the same state in the next layer, while slightly suppressing flow to other nodes. Further work could explore resistance schemes that assign distinct resistances to each node pair, as well as alternative circuit architectures beyond the layered topology studied here.

We perform low-dimensional experiments, where each sampling step can be carried out explicitly using (7). Thus, the present implementation does not rely on a coordinate-wise factorization during sampling. However,

the denominator in (7) likely becomes computationally infeasible in high-dimensional spaces, particularly when $D > 3$, as the number of categories increases. Following discrete diffusions [6, 22], further development of factorized sampling and normalization schemes is an important direction for improving the method’s scalability.

References

1. J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep Unsupervised Learning Using Nonequilibrium Thermodynamics,” in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, Lille, France, July 6–11, 2015 (JMLR: W&CP, Vol. 37, 2015), pp. 2256–2265. doi [10.48550/arXiv.1503.03585](https://doi.org/10.48550/arXiv.1503.03585).
2. J. Ho, A. Jain, and P. Abbeel, “Denoising Diffusion Probabilistic Models,” *Advances in Neural Information Processing Systems* **33**, 6840–6851 (2020). doi [10.48550/arXiv.2006.11239](https://doi.org/10.48550/arXiv.2006.11239).
3. Y. Song and D. P. Kingma, “How to Train Your Energy-Based Models,” arXiv:2101.03288 (2021). doi [10.48550/arXiv.2101.03288](https://doi.org/10.48550/arXiv.2101.03288).
4. P. Dhariwal and A. Nichol, “Diffusion Models Beat GANs on Image Synthesis,” *Advances in Neural Information Processing Systems* **34**, 8780–8794 (2021). doi [10.48550/arXiv.2105.05233](https://doi.org/10.48550/arXiv.2105.05233).
5. J. Austin, D. D. Johnson, J. Ho, et al., “Structured Denoising Diffusion Models in Discrete State-Spaces,” *Advances in Neural Information Processing Systems* **34**, 17981–17993 (2021). doi [10.48550/arXiv.2107.03006](https://doi.org/10.48550/arXiv.2107.03006).
6. A. Lou, C. Meng, and S. Ermon, “Discrete Diffusion Modeling by Estimating the Ratios of the Data Distribution,” in *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, Vienna, Austria, July 21–27, 2024 (PMLR, Vol. 235, 2024), pp. 32819–32848. doi [10.48550/arXiv.2310.16834](https://doi.org/10.48550/arXiv.2310.16834).
7. S. S. Sahoo, M. Arriola, Y. Schiff, et al., “Simple and Effective Masked Diffusion Language Models,” *Advances in Neural Information Processing Systems* **37**, 130136–130184 (2024). doi [10.52202/079017-4135](https://doi.org/10.52202/079017-4135).
8. X. Zhang and Q. Zhang, “DDMG: A New Discrete Diffusion Model Developed for Molecular Graph Generation,” in *6th International Conference on Computer Engineering and Application (ICCEA)*, Hangzhou, China, April 25–27, 2025 (IEEE, 2025), pp. 1489–1493. doi [10.1109/ICCEA65460.2025.11102178](https://doi.org/10.1109/ICCEA65460.2025.11102178).
9. Y. Xu, Z. Liu, M. Tegmark, and T. Jaakkola, “Poisson Flow Generative Models,” *Advances in Neural Information Processing Systems* **35**, 16782–16795 (2022). doi [10.48550/arXiv.2209.11178](https://doi.org/10.48550/arXiv.2209.11178).
10. Y. Xu, Z. Liu, Y. Tian, et al., “PFGM++: Unlocking the Potential of Physics-Inspired Generative Models,” in *Proceedings of the 40th International Conference on Machine Learning (ICML)*, Honolulu, Hawaii, USA, July 23–29, 2023 (PMLR, 2023), pp. 38566–38591. doi [10.48550/arXiv.2302.04265](https://doi.org/10.48550/arXiv.2302.04265).
11. A. Kolesov, S. I. Manukhov, V. V. Palyulin, and A. Korotin, “Field Matching: An Electrostatic Paradigm to Generate and Transfer Data,” in *Proceedings of the Forty-second International Conference on Machine Learning*, Vancouver, Canada, July 13–19, 2025 (PMLR, Vol. 267, 2025), pp. 31202–31222. doi [10.48550/arXiv.2502.02367](https://doi.org/10.48550/arXiv.2502.02367).
12. S. I. Manukhov, A. Kolesov, V. V. Palyulin, and A. Korotin, “Interaction Field Matching: Overcoming Limitations of Electrostatic Models,” arXiv:2506.02950 (2025). doi [10.48550/arXiv.2506.02950](https://doi.org/10.48550/arXiv.2506.02950).
13. W. K. Chen, *The Electrical Engineering Handbook* (Elsevier, 2004). <https://shop.elsevier.com/books/the-electrical-engineering-handbook/chen/978-0-12-170960-0>. Cited September 28, 2026.
14. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications* (Prentice-Hall, 1993).
15. C. M. Grinstead and J. L. Snell, *Introduction to Probability* (American Mathematical Society, 1997).
16. J. G. Kemeny and J. L. Snell, *Finite Markov Chains* (Van Nostrand, Princeton, N.J, 1960).
17. E. S. Whitammer, S. Lahlou, T. Deleu, et al., “GFlowNets and Variational Inference,” arxiv:2210.00580 (2022). doi [10.48550/arXiv.2210.00580](https://doi.org/10.48550/arXiv.2210.00580).
18. I. Gat, T. Remez, N. Shaul, et al., “Discrete Flow Matching,” *Advances in Neural Information Processing Systems* **37**, 133345–133385 (2024). doi [10.52202/079017-4239](https://doi.org/10.52202/079017-4239).
19. E. Hoogeboom, D. Nielsen, P. Jaini, et al., “Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions,” *Advances in Neural Information Processing Systems* **34**, 12454–12465 (2021). doi [10.48550/arXiv.2102.05379](https://doi.org/10.48550/arXiv.2102.05379).
20. G. Ryzhakov, S. Pavlova, E. Sevriugov, and I. Oseledets, “Explicit Flow Matching: On the Theory of Flow Matching Algorithms with Applications,” arXiv:2402.03232 (2024). doi [10.48550/arXiv.2402.03232](https://doi.org/10.48550/arXiv.2402.03232).

21. A. Gretton, K. M. Borgwardt, M. J. Rasch, et al., “A Kernel Two-Sample Test,” *Journal of Machine Learning Research* **13**, 723–773 (2012). <https://jmlr.org/papers/v13/gretton12a.html>. Cited September 29, 2026.
22. A. Campbell, J. Benton, V. De Bortoli, et al., “A Continuous Time Framework for Discrete Denoising Models,” *Advances in Neural Information Processing Systems* **35**, 28266–28279 (2022). doi 10.48550/arXiv.2205.14987.

Received
June 22, 2026

Accepted
August 31, 2026

Published
October 5, 2026

Information about the authors

Alexander S. Kolesov — PhD student, Researcher; 1) Skolkovo Institute of Science and Technology (Skoltech), Skoltech, Bolshoy Boulevard, 30, building 1, 121205, Moscow, Russia; 2) Artificial Intelligence Research Institute (AIRI), Presnenskaya naberezhnaya, 6, bulding 2, 123112, Moscow, Russia.

Stepan I. Manukhov — PhD student, Researcher; Lomonosov Moscow State University, Faculty of Physics, Leninskie Gory, 1, 119991, Moscow, Russia.

Vladimir V. Palyulin — Ph. D., Associate Professor; Skolkovo Institute of Science and Technology (Skoltech), Skoltech, Bolshoy Boulevard, 30, building 1, 121205, Moscow, Russia.

Alexander A. Korotin — Ph. D., Head of Laboratory; 1) Skolkovo Institute of Science and Technology (Skoltech), Skoltech, Bolshoy Boulevard, 30, building 1, 121205, Moscow, Russia; 2) Artificial Intelligence Research Institute (AIRI), Presnenskaya naberezhnaya, 6, bulding 2, 123112, Moscow, Russia.

Appendix 1

Single source-sink setting

A1.1. Conductance matrix and the node-voltage equation in a matrix form. To solve the formulated problem, we need to define the conductance matrix and write down the node-voltage equation in a matrix form.

Definition A1.1 (Conductance matrix). *The conductance matrix $\hat{G}$, defined for an electrical circuit represented by a graph $\mathcal{G}(V, E)$, is a matrix of size $|V| \times |V|$ such that*

$$\hat{G}_{ik} = \begin{cases} \sum_{v_j \in J(v_i)} \frac{1}{R(v_i, v_j)}, & i = k, \\ -\frac{1}{R(v_i, v_k)}, & i \neq k \text{ and } v_k \in J(v_i), \\ 0, & i \neq k \text{ and } v_k \notin J(v_i). \end{cases}$$

Let us define the vector containing all node potentials

$$\varphi = \begin{pmatrix} \varphi(v_1) \\ \varphi(v_2) \\ \vdots \\ \varphi(v_{|V|}) \end{pmatrix}$$

and the vector of currents

$$\mathbf{i} = \begin{pmatrix} i(v_1) \\ i(v_2) \\ \vdots \\ i(v_{|V|}) \end{pmatrix},$$



where $i(v_k) = \sum_{v \in J_{\text{in}}(v_k)} I(v, v_k)$. If a current $I^{\text{src}}(v_k)$ flows into the node v_k , then $i(v_k) = I^{\text{src}}(v_k)$. If a current $I^{\text{snk}}(v_k)$ flows out of the node v_k , then $i(v_k) = -I^{\text{snk}}(v_k)$. In all other cases, Kirchhoff's current law gives $i(v_k) = 0$, since the sum of currents flowing into a node equals the sum of currents flowing out of it.

Thus, the successive application of Ohm's and Kirchhoff's laws leads to the equation for determining the node potentials, known as the *node-voltage equation*:

$$\hat{G} \cdot \varphi = i. \quad (\text{A1.1.1})$$

A1.2. Analytical solution for $\mathcal{G}_L(|\mathcal{X}|, R, r)$. Consider an electrical circuit $\mathcal{G}_L(|\mathcal{X}|, R, r)$ with a unit current entering the system at $x_0 \in \mathcal{X}_{(0)}$ and leaving at $x_L \in \mathcal{X}_{(L)}$. Assume there are no other sources or sinks in the circuit besides x_0 and x_L . Each layer $\mathcal{X}_\ell$ of the graph $\mathcal{G}_L(|\mathcal{X}|, R, r)$ is a copy of $\mathcal{X} = \mathbb{S}^D$. Consequently, two distinct situations may arise: $x_0 = x_L$ or $x_0 \neq x_L$. These cases must be analyzed separately.

A1.2.1. Structure of the potential φ and current i vectors. The graph $\mathcal{G}_L(|\mathcal{X}|, R, r)$ consists of $L + 1$ layers. Each layer contains $n = |\mathcal{X}|$ vertices. Therefore, we will express the potential vector φ and the current vector i in a block structure:

$$\varphi = \begin{pmatrix} \varphi_0 \\ \varphi_1 \\ \vdots \\ \varphi_L \end{pmatrix}, \quad i = \begin{pmatrix} i_0 \\ i_1 \\ \vdots \\ i_L \end{pmatrix}, \quad (\text{A1.2.1.1})$$

where each φ_ℓ and i_ℓ is a column vector of height n :

$$\forall \ell = 0, \dots, L: \quad \varphi_\ell = \begin{pmatrix} \varphi_\ell(x_{\ell 1}) \\ \varphi_\ell(x_{\ell 2}) \\ \vdots \\ \varphi_\ell(x_{\ell n}) \end{pmatrix}, \quad i_\ell = \begin{pmatrix} i_\ell(x_{\ell 1}) \\ i_\ell(x_{\ell 2}) \\ \vdots \\ i_\ell(x_{\ell n}) \end{pmatrix}. \quad (\text{A1.2.1.2})$$

A1.2.2. Symmetry of the problem. The problem possesses a high degree of symmetry. In formulas (A1.2.1.1) and (A1.2.1.2), we considered “vertical” cross-sections of the graph, i.e., layers. Let us now consider a “horizontal” cross-section, namely, the sets of values:

$$(\varphi_0(x'), \varphi_1(x'), \varphi_2(x'), \dots, \varphi_L(x')), \quad \text{where } x' \in \mathcal{X} \text{ is fixed.}$$

These sets of values are entirely equivalent for all $x' \neq x_0$ and $x' \neq x_L$. Therefore, the problem of finding $n(L + 1)$ unknown potentials reduces to finding only $3(L + 1)$ potentials at the following points:

$$\begin{cases} \varphi_\ell(x_0), & \ell = 0, \dots, L; \\ \varphi_\ell(x_L), & \ell = 0, \dots, L; \\ \varphi_\ell(x'), & \ell = 0, \dots, L, \text{ where } x' \neq x_0, x_L. \end{cases}$$

A1.2.3. Structure of the conductance matrix and the node-voltage equation. For the resistances defined in (9), the conductance matrix takes the block-tridiagonal form:

$$\hat{G} = \begin{pmatrix} D & B & 0 & 0 & \dots & 0 \\ B^T & 2D & B & 0 & \dots & 0 \\ 0 & B^T & 2D & B & \dots & 0 \\ 0 & 0 & B^T & 2D & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & B \\ 0 & 0 & 0 & \dots & B^T & D \end{pmatrix}.$$

Here B and D are $n \times n$ matrices, $n = |\mathcal{X}|$. They are expressed through resistances by the following way:

$$D = \left(\frac{1}{r} + \frac{n-1}{R} \right) \cdot \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{pmatrix}, \quad B = \begin{pmatrix} -\frac{1}{r} & -\frac{1}{R} & -\frac{1}{R} & \dots & -\frac{1}{R} \\ -\frac{1}{R} & -\frac{1}{r} & -\frac{1}{R} & \dots & -\frac{1}{R} \\ -\frac{1}{R} & -\frac{1}{R} & -\frac{1}{r} & \dots & -\frac{1}{R} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\frac{1}{R} & -\frac{1}{R} & -\frac{1}{R} & \dots & -\frac{1}{r} \end{pmatrix}.$$

Given that $i_1, i_2, \dots, i_{L-1}$ are identically equal to zero (since the sum of incoming and outgoing currents is balanced at the vertices of the intermediate layers), equation (A1.1.1) takes the following form:

$$\begin{cases} D\varphi_0 + B\varphi_1 & = i_0, \\ B^T\varphi_{\ell-1} + 2D\varphi_\ell + B\varphi_{\ell+1} & = 0, \quad \forall \ell = 1, \dots, L-1, \\ B^T\varphi_{L-1} + D\varphi_L & = i_L. \end{cases} \quad (\text{A1.2.3.1})$$

A1.2.4. Case $x_0 \neq x_L$. Let us now consider the case of non-coinciding source and sink. Equations in (A1.2.3.1) in this case can be rewritten in the following form:

1. The upper equality corresponding to $D\varphi_0 + B\varphi_1 = i_0$ splits into

$$\begin{cases} \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_0(x_0) - \frac{1}{r} \varphi_1(x_0) - \frac{n-2}{R} \varphi_1(x') - \frac{1}{R} \varphi_1(x_L) = 1, \\ \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_0(x') - \frac{1}{r} \varphi_1(x_0) - \left(\frac{1}{r} + \frac{n-3}{R} \right) \varphi_1(x') - \frac{1}{R} \varphi_1(x_L) = 0, \\ \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_0(x_L) - \frac{1}{r} \varphi_1(x_L) - \frac{n-2}{R} \varphi_1(x') - \frac{1}{R} \varphi_1(x_0) = 0. \end{cases} \quad (\text{A1.2.4.1})$$

2. The middle equality corresponding to $B^T\varphi_{\ell-1} + 2D\varphi_\ell + B\varphi_{\ell+1} = 0$ reduces to:

$$\begin{cases} -\frac{1}{r} \varphi_{\ell-1}(x_0) - \frac{1}{R} \varphi_{\ell-1}(x_L) - \frac{n-2}{R} \varphi_{\ell-1}(x') \\ + 2 \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_\ell(x_0) - \frac{1}{r} \varphi_{\ell+1}(x_0) - \frac{1}{R} \varphi_{\ell+1}(x_L) - \frac{n-2}{R} \varphi_{\ell+1}(x') = 0, \\ -\frac{1}{r} \varphi_{\ell-1}(x_L) - \frac{1}{R} \varphi_{\ell-1}(x_0) - \frac{n-2}{R} \varphi_{\ell-1}(x') \\ + 2 \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_\ell(x_L) - \frac{1}{r} \varphi_{\ell+1}(x_L) - \frac{1}{R} \varphi_{\ell+1}(x_0) - \frac{n-2}{R} \varphi_{\ell+1}(x') = 0, \\ -\frac{1}{R} (\varphi_{\ell-1}(x_0) + \varphi_{\ell-1}(x_L)) - \left(\frac{1}{r} + \frac{n-3}{R} \right) \varphi_{\ell-1}(x') \\ + 2 \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_\ell(x') - \frac{1}{R} (\varphi_{\ell+1}(x_0) + \varphi_{\ell+1}(x_L)) - \left(\frac{1}{r} + \frac{n-3}{R} \right) \varphi_{\ell+1}(x') = 0. \end{cases} \quad (\text{A1.2.4.2})$$

3. The last equality corresponding to $B^T\varphi_{L-1} + D\varphi_L = i_L$ is:

$$\begin{cases} -\frac{1}{r} \varphi_{L-1}(x_0) - \frac{1}{R} \varphi_{L-1}(x_L) - \frac{n-2}{R} \varphi_{L-1}(x') + \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_L(x_0) = 0, \\ -\frac{1}{r} \varphi_{L-1}(x_L) - \frac{1}{R} \varphi_{L-1}(x_0) - \frac{n-2}{R} \varphi_{L-1}(x') + \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_L(x_L) = -1, \\ -\left(\frac{1}{r} + \frac{n-3}{R} \right) \varphi_{L-1}(x') - \frac{1}{R} (\varphi_{L-1}(x_0) + \varphi_{L-1}(x_L)) + \left(\frac{1}{r} + \frac{n-1}{R} \right) \varphi_L(x') = 0. \end{cases} \quad (\text{A1.2.4.3})$$



These equations can be significantly simplified if one takes into account that $n = |\mathcal{X}| = S^D \gg 1$. In this case, $1/n \rightarrow 0$, and $n \approx n - 1 \approx n - 2 \approx n - 3$. Then, multiplying the equations by R/n and introducing $\tilde{r} = r \cdot n/R$, we obtain the following modifications of (A1.2.4.1–A1.2.4.3):

$$\begin{cases} \left(\frac{1}{\tilde{r}} + 1 \right) \varphi_0(x_0) - \frac{1}{\tilde{r}} \varphi_1(x_0) - \varphi_1(x') = \frac{r}{\tilde{r}}, \\ \varphi_0(x') = \varphi_1(x'), \\ \left(\frac{1}{\tilde{r}} + 1 \right) \varphi_0(x_L) - \frac{1}{\tilde{r}} \varphi_1(x_L) - \varphi_1(x') = 0; \\ \left\{ \begin{aligned} -\frac{1}{\tilde{r}} \varphi_{\ell-1}(x_0) - \varphi_{\ell-1}(x') + 2 \left(\frac{1}{\tilde{r}} + 1 \right) \varphi_{\ell}(x_0) - \frac{1}{\tilde{r}} \varphi_{\ell+1}(x_0) - \varphi_{\ell+1}(x') &= 0, \\ -\frac{1}{\tilde{r}} \varphi_{\ell-1}(x_L) - \varphi_{\ell-1}(x') + 2 \left(\frac{1}{\tilde{r}} + 1 \right) \varphi_{\ell}(x_L) - \frac{1}{\tilde{r}} \varphi_{\ell+1}(x_L) - \varphi_{\ell+1}(x') &= 0, \\ -\varphi_{\ell-1}(x') + 2\varphi_{\ell}(x') - \varphi_{\ell+1}(x') &= 0; \end{aligned} \right. \\ \left\{ \begin{aligned} -\frac{1}{\tilde{r}} \varphi_{L-1}(x_0) - \varphi_{L-1}(x') + \left(\frac{1}{\tilde{r}} + 1 \right) \varphi_L(x_0) &= 0, \\ -\frac{1}{\tilde{r}} \varphi_{L-1}(x_L) - \varphi_{L-1}(x') + \left(\frac{1}{\tilde{r}} + 1 \right) \varphi_L(x_L) &= -1, \\ \varphi_{L-1}(x') &= \varphi_L(x'). \end{aligned} \right. \end{cases}$$

Thus, in this approximation

$$\varphi_0(x') = \varphi_1(x') = \dots = \varphi_L(x').$$

Recall that the potential is defined up to an additive constant. Therefore, the reference point from which it is measured can be chosen arbitrarily. Let us set:

$$\varphi_{\ell}(x') = 0 \quad \forall \ell = 0, 1, \dots, L.$$

Consequently, the system of equations simplifies further, decoupling into two independent parts:

1. System for x_0 is

$$\begin{cases} \gamma \varphi_0(x_0) - \varphi_1(x_0) = r, \\ -\varphi_{\ell-1}(x_0) + 2\gamma \varphi_{\ell}(x_0) - \varphi_{\ell+1}(x_0) = 0, \quad \forall \ell = 1, \dots, L-1, \\ -\varphi_{L-1}(x_0) + \gamma \varphi_L(x_0) = 0. \end{cases} \quad (\text{A1.2.4.4})$$

2. System for x_L is

$$\begin{cases} \gamma \varphi_0(x_L) - \varphi_1(x_L) = 0, \\ -\varphi_{\ell-1}(x_L) + 2\gamma \varphi_{\ell}(x_L) - \varphi_{\ell+1}(x_L) = 0, \quad \forall \ell = 1, \dots, L-1, \\ -\varphi_{L-1}(x_L) + \gamma \varphi_L(x_L) = -r. \end{cases} \quad (\text{A1.2.4.5})$$

Here $\gamma = 1 + \tilde{r} = 1 + \frac{r \cdot n}{R}$. From the solution of (A1.2.4.4), it is easy to obtain the solution of (A1.2.4.5), namely $\varphi_{\ell}(x_L) = -\varphi_{L-\ell}(x_0)$.

Suppose that $\varphi_{\ell}(x_L)$ is proportional to $\varphi_0(x_L)$ via some polynomial in γ :

$$\varphi_{\ell}(x_L) = P_{\ell}(\gamma) \varphi_0(x_L). \quad (\text{A1.2.4.6})$$

Substituting (A1.2.4.6) into (A1.2.4.5), we find that the polynomial $P_{\ell}(\gamma)$ has to satisfy the conditions:

$$\begin{aligned} P_0(\gamma) &= 1, \\ P_1(\gamma) &= \gamma, \\ P_{\ell+1}(\gamma) &= 2\gamma P_{\ell}(\gamma) - P_{\ell-1}(\gamma). \end{aligned}$$

These are the recurrence relations for the Chebyshev polynomials of the first kind:

$$P_{\ell}(\gamma) \equiv T_{\ell}(\gamma).$$

Substituting (A1.2.4.6) into the last equation of (A1.2.4.5), we obtain:

$$-\varphi_{L-1}(x_L) + \gamma\varphi_L(x_L) = (-T_{L-1}(\gamma) + \gamma T_L(\gamma))\varphi_0(x_L) = -r.$$

Thus, we get

$$\varphi_0(x_L) = -\frac{r}{-T_{L-1}(\gamma) + \gamma T_L(\gamma)}$$

and

$$\varphi_\ell(x_L) = -\frac{T_\ell(\gamma)r}{-T_{L-1}(\gamma) + \gamma T_L(\gamma)}, \quad \varphi_\ell(x_0) = \frac{T_{L-\ell}(\gamma)r}{-T_{L-1}(\gamma) + \gamma T_L(\gamma)}.$$

The corresponding tridiagonal boundary-value problem has a unique solution (given that the additive constant has been fixed by choosing $\varphi_\ell(x') = 0$). Hence, the potentials above are uniquely determined in the chosen gauge.

A1.2.5. Case $x_0 = x_L$. In this case, similarly, $\varphi_\ell(x') = 0 \forall \ell = 0, 1, \dots, L$, and the system of equations for $\varphi_\ell(x_0) \equiv \varphi_\ell(x_L)$ takes the form:

$$\begin{cases} \gamma\varphi_0(x_0) - \varphi_1(x_0) = r, \\ -\varphi_{\ell-1}(x_0) + 2\gamma\varphi_\ell(x_0) - \varphi_{\ell+1}(x_0) = 0, \\ -\varphi_{L-1}(x_0) + \gamma\varphi_L(x_0) = -r. \end{cases} \quad (\text{A1.2.5.1})$$

Let us suppose that the solution has the form:

$$\varphi_\ell(x_0) = P_\ell(\gamma)\varphi_0(x_0) - rF_\ell(\gamma). \quad (\text{A1.2.5.2})$$

Then, the polynomials $P_\ell(\gamma)$ and $F_\ell(\gamma)$ satisfy the recurrence relations

$$\begin{aligned} P_{\ell+1}(\gamma) &= 2\gamma P_\ell(\gamma) - P_{\ell-1}(\gamma), \\ F_{\ell+1}(\gamma) &= 2\gamma F_\ell(\gamma) - F_{\ell-1}(\gamma) \end{aligned}$$

with the initial conditions

$$P_0(\gamma) = 1, P_1(\gamma) = \gamma, \quad F_0(\gamma) = 0, F_1(\gamma) = 1, F_2(\gamma) = 2\gamma.$$

This allows us to identify $P_\ell(\gamma)$ and $F_\ell(\gamma)$ with the Chebyshev polynomials of the first and second kind, respectively:

$$P_\ell(\gamma) = T_\ell(\gamma), \quad F_\ell(\gamma) = U_{\ell-1}(\gamma).$$

Note that the polynomial $U_\ell(\gamma)$ incorporates an additional condition $U_{-1}(\gamma) \equiv 0$, which is not usually implied in the standard definition.

Substituting (A1.2.5.2) into the last equation of (A1.2.5.1), we obtain:

$$\varphi_0(x_0) = r \frac{\gamma U_{L-1}(\gamma) - U_{L-2}(\gamma) - 1}{\gamma T_L(\gamma) - T_{L-1}(\gamma)}.$$

Therefore, the final expression for the potentials is:

$$\varphi_\ell(x_0) = r \left(T_\ell(\gamma) \frac{\gamma U_{L-1}(\gamma) - U_{L-2}(\gamma) - 1}{\gamma T_L(\gamma) - T_{L-1}(\gamma)} - U_{\ell-1}(\gamma) \right). \quad (\text{A1.2.5.3})$$

This completes the proof of the Lemma 1.

After imposing the same zero reference for the all other states potentials, no remaining additive degree of freedom is left. Therefore, the solution of this reduced system is unique in the fixed gauge.

Appendix 2

Experimental details

In the 1D illustrative example of subsection 1 of section 5, we use a simple Multi-Layer Perceptron (MLP) neural net with Rectified Linear Unit (ReLU) activations and hidden layers of size [128, 128, 128]. Thus, the architecture of our model obtains a one-dimensional vector (a sample from a layered graph), then processes it to

128-dimensional vector three times and outputs a one-dimensional sample from a layered graph. To condition on the layer number, we use a lookup table whose embedding dimension is two. The number of embeddings is chosen according to the number of edge-layer indices in the corresponding experiment. The network is trained for 5 000 steps with learning rate of $2 \cdot 10^{-4}$, weight decay of 10^{-4} , batch size of 256, and the Stochastic Gradient Descent (SGD) optimizer without a learning rate schedule. The Monte-Carlo batch size for estimating the current target in (11) is also 256.

The network for the experiment in subsection 2 of section 5 uses the same architecture and optimization settings. For the MMD comparison, we apply an RBF kernel with bandwidth selected by the median heuristic, denoted by σ_{median} . For DFM/FM, we follow the public implementation `facebookresearch/flow_matching` as closely as possible. Training the 1D and 2D experiments takes 2 minutes on a single A100 GPU (30 GB VRAM). In the 2D experiment, sampling of generated data with a batch size of 256 takes 0.13 s.

Appendix 3 Ablation study

This appendix provides an empirical analysis of the key hyperparameter governing our method, namely the resistance ratio $\hat{\gamma} = r/R$. This ratio is not merely a numerical hyperparameter but stems directly from the physical analogy underpinning our framework, where it controls the effective conductance of the optimization pathway. The ratio $\hat{\gamma} = r/R$ used in this ablation is distinct from the Chebyshev parameter $\gamma = 1 + rn/R$ used in Appendix 1.

The parameter $\hat{\gamma}$ acts as a convergence rate modulator. We consider three different values of $\hat{\gamma}$ and plot $\varphi_\ell(x_\ell)$ as a function of the layer ℓ (Figure A3.1) and the corresponding marginal distributions at each ℓ -th layer of pathway (Figure A3.2).

In the high-conductance regime ($\hat{\gamma} \rightarrow 1$, Figure A3.1 a), the lower relative lateral resistance facilitates

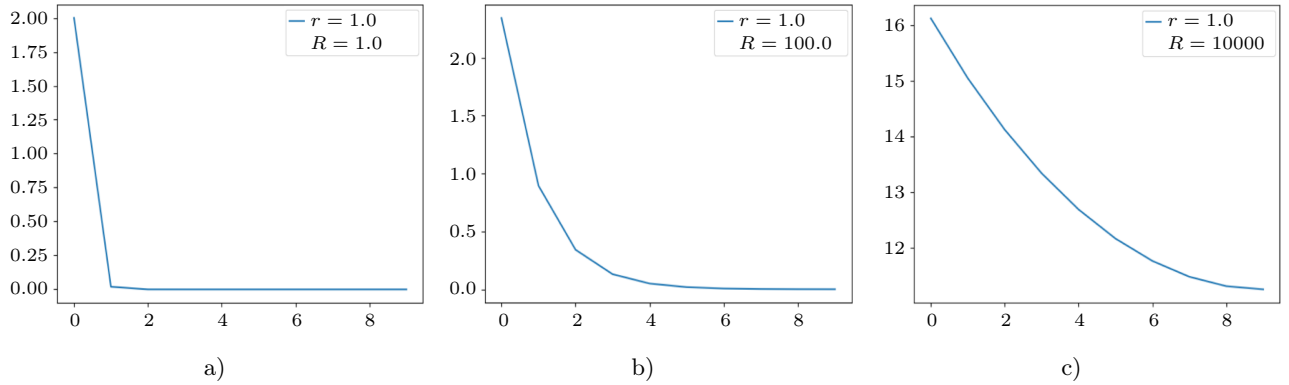


Figure A3.1. The dependence of potential $\varphi_\ell(x_\ell)$ on the layer ℓ for three values of $\hat{\gamma}$:
 a) $\hat{\gamma} = 1$; b) $\hat{\gamma} = 0.01$; c) $\hat{\gamma} = 0.0001$

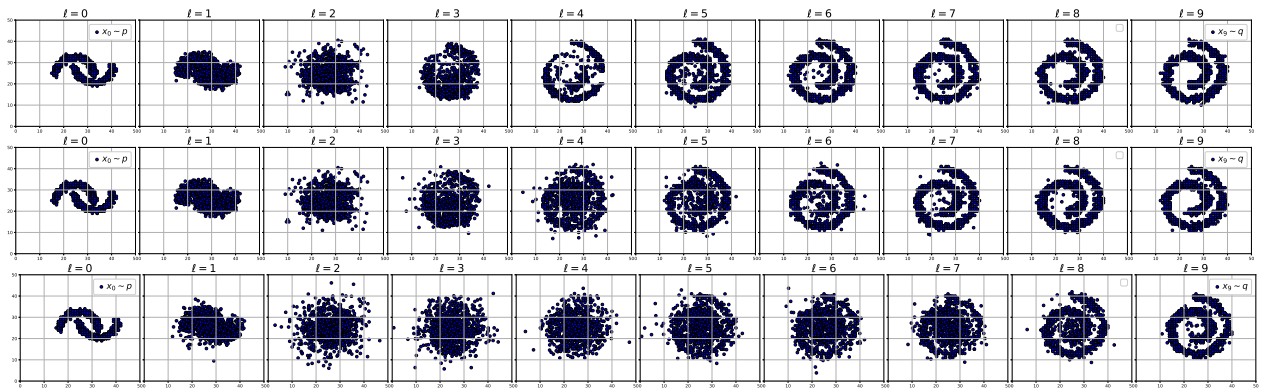


Figure A3.2. Illustrative 2D Moons-to-Swiss-roll experiment: input distribution $p(x_0)$ and intermediate distributions at layers $\ell = 1, \dots, 9$ and samples transformed from $x \sim p$ by $T(x)$ in the layered graph $\mathcal{G}_{10}(2500, 1, 1)$ (top image), $\mathcal{G}_{10}(2500, 100, 1)$ (middle image) and $\mathcal{G}_{10}(2500, 10000, 1)$ (bottom image)

faster flow and leads to rapid convergence in the initial iterations (top image in Figure A3.2).

In the low-conductance regime ($\hat{\gamma} \rightarrow 0$, Figure A3.1 c), the process results in a slower flow compared to the previous example. Consequently, the method requires more iterations to converge to the target distribution (bottom image in Figure A3.2).

Setting $\hat{\gamma} = 0.01$ leads to the intermediate behavior of our model (middle image in Figure A3.2). Thus, the resistance ratio $\hat{\gamma}$ defines the rate of probability flow through the circuit.