arXiv:2301.05638v1 [hep-ph] 13 Jan 2023

从第一性原理出发的深度学习对称性及其李群、代数与子代数

Roy Forestano, [1], * Konstantin T. Matchev, [1], † Katia Matcheva, [1], ‡Alexander Roman, [1], § Eyup Unlu, [1], ¶ 以及 Sarunas Verner[1], **[1] *Institute for Fundamental Theory, Physics Department, University of Florida, Gainesville, FL 32611, USA*（日期：2023年1月12日）

我们设计了一种深度学习算法，用于发现和识别带标签数据集中存在的连续对称群。我们使用全连接神经网络对对称变换及其相应的生成元进行建模。我们构造了损失函数，以确保所施加的变换是对称变换，并确保相应的生成元集合构成一个闭合的（子）代数。我们通过若干示例验证了这一流程，这些示例展示了对称性所保持的不同类型的守恒量。在推导完整对称性集合的过程中，我们分析了旋转群 $SO(2)$、$SO(3)$ 和$SO(4)$ 的完整子群结构，以及洛伦兹群 $SO(1,3)$。其他示例包括挤压映射、分段不连续标签和 $SO(10)$，展示了我们的方法具有完全普适性，并可广泛应用于物理学和数据科学。我们的研究还为在李群及其性质的数学研究中采用机器学习方法开辟了道路。

目录

I. 引言 1II. 设置与记号 3

III. 深度学习方法 4
A. 不变性 B. 无穷 4
小性 C. 正交性 4
D. 闭合性 5
5

IV. 线性代数方法 5

V. 二维中的保长度变换 6

VI. 三维中的保长度变换 8

VII. 四维中的保长度变换 9
A. 二生成元子代数 9B. 三生成元子代数 10C. 四生成元子代数 10
D. 六生成元代数 12

VIII. 四维闵可夫斯基空间中的洛伦兹变换 12
A. 二生成元子代数 13B. 三生成元子代数 14C. 四生成元子代数 16D. 六生成元代数 16

IX. 二维挤压映射 16

X. 不连续预言机 17
A. 分段线性预言机 17B. 曼哈顿距离预言机 18

XI. 结论 18致谢 19参考文献 19

* roy.forestano@ufl.edu
† matchev@ufl.edu
‡ matcheva@ufl.edu§ alexroman@ufl.edu
¶eyup.unlu@ufl.edu
** verner.s@ufl.edu

I. 引言

对称性在现代物理学中发挥着基础性作用[1]。具有连续对称性的物理系统会呈现守恒定律，这些定律具有普遍适用性，对于理解系统的行为和演化不可或缺。在粒子物理学中，对称性为观测到的粒子动物园及其相互作用提供了组织原则，并指导模型构建者寻找可行的标准模型（SM）扩展[2]。与此同时，对称性的数学研究本身也

颇具趣味，且有着悠久的历史。

过去十年间，人们对机器学习（ML）在高维物理学数据中的应用越来越感兴趣，并将其作为一种灵敏工具，用于事件模拟、数据分析和统计推断 [3–5]。近来，机器学习也被用于辅助完成传统上属于理论物理学家范畴的任务，例如进行符号计算 [6, 7] ，或通过在合成数据上训练符号回归来推导解析公式 [8–20]。

在不同背景下，许多研究团队都开展了机器学习在对称性研究中的应用。一类研究探讨给定对称性如何体现在数据的学习到的表示中 [21, 22]，或体现在机器学习架构本身中，例如神经网络（NN）的嵌入层 [23]。一些方案尝试设计从一开始就内置所需对称性属性的专用机器学习架构（等变神经网络） [24–30]，并测试其性能 [31]。将对称性直接纳入机器学习模型，可以使其更经济（就学习到的表示而言）、更易解释，也更易训练。这种方法还可扩展到离散（置换）对称性 [32, 33]。这些努力为数据驱动的盲搜铺平了道路，以检验数据中是否存在对标准模型（SM）既有对称性的破坏，从而搜寻新物理[34–36]。

近年来，机器学习也被用于探究更形式化的理论问题。例如，深入理解问题中存在的对称性，可以揭示守恒量 [37, 38] 或提示一种更根本的统一图景 [39]。机器学习已被用于发现势的对称性 [23, 40, 41],、判断给定的一对输入是否通过对称性相关联 [42], 、区分尺度不变性与共形对称性 [43], , 以及探索弦景观 [44–46]。近期研究利用生成对抗网络学习能够保持概率分布不变的变换 [47]。机器学习的应用也延伸到了群论，这门学科为对称性提供了抽象的数学语言。例如，近期研究使用机器学习计算李群不可约表示的张量积和分支规则 [48] ，并获得数据中存在的对称性的李群生成元 [49, 50]。

本文的主要目标是设计一种深度学习方法，模仿传统理论学家的思维方式，并能够从给定数据集中发现和分类全部（连续）对称性，并遵循第一性原理，即不作任何先验假设或带有任何成见。我们的方法唯一的输入是一个带标签数据集 $\{\mathbf{x};\ y\}$ 如下面公式（1）所示。预言机 $\varphi(\mathbf{x}) = y$ 随后可以从数据集中学习得到，也可以由外部提供。具备这些要素后，我们将依次完成以下目标:

• **发现对称性**。第一步（见第IIIA节）是学习生成一个对称变换$\mathbf{x} \to \mathbf{x}'$，该变换能够保持预言机值不变。变换$\mathbf{f}$ 由一个在数据集上训练的神经网络编码。通常，可能存在许多不同的对称变换$\mathbf{f}$；从群论的角度研究并分类这些变换，是本文的主要目标。

•**对称生成元的发现**。 学习如何生成任意（有限）对称变换 $\mathbf{f}$ 后，我们接着聚焦于 无穷小 变换 $\delta\mathbf{f}$，进而得到对称生成元$\mathbf{J}$。

• **李子代数的发现**。 通过向损失函数添加适当的项，我们的方法要求学习到的生成元集合 $\{\mathbf{J}_\alpha\}$ 构成一个闭代数。这使我们能够发现对称群的子代数，并根据其结构常数对其进行识别。由于生成元的数量 N_g 是一个自由输入，当训练未能成功时（以损失量化），我们可以排除 N_g维子代数的存在。

• **完整李代数的发现**。 使损失为零的 N_g 最大值表明完整李代数的维数。相应的已学习生成元集合描述了数据集的完整对称群。

• **对称群及其子群的识别**。随后使用前两个步骤中找到的生成元集合，求出相应完整代数及其子代数的结构常数，从而识别相应的对称群及其子群。

我们的研究补充并拓展了 [23, 40, 41, 49, 50]中此前的相关工作。我们注意到，我们的方法完全通用，并不预设数据集中可能存在何种对称性——相反，这些对称性是从头学习得到的。此外，我们的方法是基无关的，因为我们没有为学习到的变换和生成元选择特定的便利基。因此，我们的结果并不总是与群论教材中生成元的优美典范形式相符。不过，正如下文的示例所明确展示的，我们学习到的所有变换和生成元都满足相应对称群和子群的定义性质。

本文结构安排如下。在第II 节中，我们介绍分析所需的设定以及相应的符号和约定。第III节概述深度学习方法的主要步骤，并解释和说明所需的要素

损失函数。对于尚未完全熟悉深度学习方法的读者，第IV 节介绍一种类似的线性代数方法，该方法通常（但并非总是如此——反例见第X 节）能够实现类似的目标。其余各节均提供一个单独且完整的实例，以说明我们的方法。各实例根据预言机的选择和特征空间的维数加以区分。在第V-VII 节中，我们选择一个返回特征空间中欧几里得距离的预言机，而特征空间的维数依次设为 $n = 2$ （见第V节）、 $n = 3$ （见第VI节）和 $n = 4$ （见第VII节）。第VIII 节聚焦于洛伦兹群，即预言机计算四维闵可夫斯基时空中的伪欧几里得距离。第IX 节探讨 $n = 2$ 维空间中的挤压变换，此时预言机返回两个特征的乘积。为展示我们技术的普适性，第X 节给出两个不连续预言机的实例。我们在第XI节总结全文并作出结论。

II. 设置与符号

我们的出发点是一个带标签数据集，其中包含 m个样本，每个样本有 n 个特征，以及目标标签 y:

$$m \text{ samples} \left\{ \begin{array}{ccccc} x_1^{(1)}, & x_1^{(2)}, & \ldots, & x_1^{(n)}; & y_1 \\ x_2^{(1)}, & x_2^{(2)}, & \ldots, & x_2^{(n)}; & y_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_m^{(1)}, & x_m^{(2)}, & \ldots, & x_m^{(n)}; & y_m \end{array} \right. . \tag{1}$$

按照机器学习的术语，数据集（1）是一个包含 $m \times n$ 个特征和 n 个样本的数据框。在下文中，我们使用样本索引的频率远高于特征索引，因此对样本索引显式使用下标，而将特征索引隐含在粗体向量符号中：$\mathbf{x}$:

$$\mathbf{x} \equiv \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}. \tag{2}$$

这样，我们就能以紧凑的形式写出输入特征:

$$\{\mathbf{x}_i\} \equiv \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\} . \tag{3}$$

为了研究数据（1）的对称性，我们需要知道函数 $y(\mathbf{x})$，它可以通过解析方式给出，也可以借助预言机 φ: $\mathbb{R}^n \to \mathbb{R}$该预言机能够计算相应的目标标签输出 y_1、 y_2、……、 y_m:

$$y_i = \varphi(\mathbf{x}_i) , \qquad i = 1, 2, \ldots, m . \tag{4}$$

这引出了两种基本情形:

- The function $y(\mathbf{x})$ is known analytically, and that same function has been used as in (4) to compute

完全按照（1）中的标签。这种情形令理论学家感兴趣，本文也采用了这种方法——对于下面的每个示例，我们都会指定相应的解析预言机 φ（$\mathbf{x}$），并着手研究由此产生的对称性。不过请注意，我们从未利用预言机的解析形式知识，例如，我们不会对函数求导，也不会以其他方式对其进行符号运算 φ（$\mathbf{x}$）。就我们的目的而言，我们只以数值计算的方式使用预言机——我们的方法将其视为一个黑箱，只需给定 $\mathbf{x}$ 的值，它就能生成标签的数值 y。

- 函数关系 $y(\mathbf{x})$事先未知，数据集（1）是我们唯一可用的信息。这是数据科学家经常遇到的典型情况。此时，需要先进行一个初步步骤，创建预言机 φ（通常是一个在数据集（1）上训练的神经网络），它能够计算并向我们报告 $y = \varphi(\mathbf{x})$的取值。这是一个标准的回归任务，与本文无关，因为它可以使用众多成熟的机器学习回归技术之一来完成。因此，我们可以放心地假定这一步初步工作已经完成，并且已经有了这样一个数值预言机 $y = \varphi(\mathbf{x})$。

由于我们只在数值计算中使用预言机，因此在我们看来，上述两种情况并无实质区别。下文中，无论预言机 φ $(\mathbf{x})$ 的来源如何，我们都会以完全相同的方式使用它。

在这一通用设置下，我们的主要任务是推导出对称变换 $\mathbf{f}$: $\mathbb{R}^n \to \mathbb{R}^n$

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}) , \tag{5}$$

，它保留了 φ所诱导的数据集标签（1）。换句话说，我们希望找到满足以下条件的函数 $\mathbf{f}(\mathbf{x})$:

$$\varphi(\mathbf{x}'_i) \equiv \varphi(\mathbf{f}(\mathbf{x}_i)) = \varphi(\mathbf{x}_i), \quad \forall i = 1, 2, \ldots, m . \tag{6}$$

对称性 $\mathbf{f}(\mathbf{x})$ 的具体形式取决于参数的初始化方式，因此，通过采用不同的初始化重复这一过程，原则上我们将得到一整族对称变换。

接下来，我们将重点关注无穷小对称变换，并着手研究相应的生成元集合 $\{\mathbf{J}_\alpha\}$，其中 $\alpha = 1, 2, \ldots, N_g$; 我们使用小写希腊字母标记对称变换的生成元。若满足闭合条件，即所有李括号 $[\,.\,,\,.\,]$ 均可表示为该集合中生成元的线性组合，则给定的生成元集合 $\{\mathbf{J}_\alpha\}$ 构成一个李代数:

$$[\mathbf{J}_\alpha, \mathbf{J}_\beta] = \sum_{\gamma=1}^{N_g} a_{[\alpha\beta]\gamma} \mathbf{J}_\gamma . \tag{7}$$

系数 $a_{[\alpha\beta]\gamma}$ 称为结构常数（由方括号可知，它们关于前两个指标反对称），其取值将揭示我们的数据集中存在的对称群。

原则上，生成元的数量 N_g 是一个必须预先指定的超参数（类似于某些聚类算法（如 K-均值）中簇数量的选择）。因此，当我们在给定的 N_g 取值下找到一个闭代数时，只能确定它是一个子代数；我们还必须尝试更大的 N_g 取值。完整的代数对应于能够找到由不同生成元构成的闭代数时， N_g 所能取到的最大值。

III. 深度学习方法

在我们的方法中，我们使用神经网络（NN） $\mathcal{F}_{\mathcal{W}}$ 对输出函数$\mathbf{f}$ 进行建模，网络的输出层包含 n 个神经元，对应于数据点$\mathbf{x}'$的 n 个变换后的特征。可训练的网络参数（权重和偏置）统称为 $\mathcal{W}$。在训练过程中，这些参数会不断变化并收敛到训练后的网络 $\mathcal{F}_{\widehat{\mathcal{W}}}$对应的参数训练后的 取值 $\mathcal{W}$ ，也就是说，帽号表示训练所得的结果。一旦确定了参数 $\mathcal{W}$，我们就可以使用标准方法求出结构常数。我们选择一个合适的损失函数，以确保训练后的网络具备所需的性质。以下各小节将分别讨论损失函数的各项贡献；在我们的实现中，这些贡献会合并起来并同时进行最小化。

该神经网络 $\mathcal{F}_{\mathcal{W}}$ 在 PyTorch [51]中实现为一个顺序前馈神经网络。第V至VIII节中的示例足够简单，无需隐藏层、偏置或激活函数即可完成，即只需使用线性变换（参见第IV节）。对于后面各节中的示例，我们确实添加了隐藏层。使用 Adam优化器进行优化，学习率介于 0.03 和 0.1 之间。损失函数经过设计，旨在实现快速高效的训练过程，无需大量调整超参数。训练数据（3）通常包含数百个点，并从标准正态分布中采样得到。

一种预测模型参数所对应生成元的替代方法采用相同的损失函数：神经网络 NN， $\mathcal{F}:\backsim\{\mathcal{G}_\alpha,\mathbf{a}\}\to\{\mathcal{G}_\alpha,\widehat{\mathbf{a}}\}$，接收一组随机初始化的$n\times n$生成元 $\{\mathcal{G}_\alpha\}$和一个 $N_b\times N_g$ 结构常数数组 $\{\mathbf{a}\}$，将每个生成元及结构常数数组分别展平，并逐步使$\widehat{\ }$元素收敛到目标生成元集合 $\{\mathcal{G}_\alpha\}$ 和结构常数数组 $\{\widehat{\mathbf{a}}\}$。此处， $N_b=\binom{N_g}{2}=N_g(N_g-1)/2$ 表示给定生成元数量时唯一的单向李括号的数量 $\mathcal{F}$ 。

给定生成元数量 N_g 的集。

这种替代方法可通过模块列表实现，该列表由顺序神经网络层组成：生成元使用两个隐藏层，结构常数则使用一个由两个隐藏层构成的顺序层。每一层都包含偏置，隐藏层则使用修正线性单元（ReLU）激活函数。优化器、平均学习率、模型超参数和训练数据生成方式均与上文所述的替代实现相同。原始方法将数据直接输入网络，而这种方法则将数据输入损失函数，再通过模型预测的生成元对其进行变换。

A. 不变性

通过要求变换（5）前后的标签保持不变，确保了对该变换的不变性。为此，我们在损失函数中加入以下均方误差（MSE）项 L:

$$L_{inv}(\mathcal{W},\{\mathbf{x}_i\})=\frac{1}{m}\sum_{i=1}^{m}\left[\varphi\left(\mathcal{F}_{\mathcal{W}}(\mathbf{x}_i)\right)-\varphi(\mathbf{x}_i)\right]^2 . \quad (8)$$

使用此损失函数训练的神经网络将找到一种任意一般的（有限）对称变换 $\mathcal{F}_{\widehat{\mathcal{W}}}$，该变换由训练网络的参数值 $\mathcal{W}$ 参数化。$\widehat{\ }$

为了找到多个对称性，可以重复这一过程。或者，也可以同时训练多个网络，并修改损失函数以确保得到的变换彼此足够不同（见下文第IIIC 节）。

B. 无穷小性

为了聚焦于可能的对称变换集合的生成元，我们将范围限制在恒等变换$\mathbf{I}$附近的无穷小变换 $\delta\mathcal{F}$:

$$\delta\mathcal{F}\ \equiv\ \mathbf{I}+\varepsilon\,\mathcal{G}_{\mathcal{W}}\,, \quad (9)$$

其中 ε是一个无穷小参数，而新神经网络 $\mathcal{W}$ 的参数 $\mathcal{G}$ 将被限制为有限值，以确保 (9) 是一个无穷小变换。随后，损失函数 (8) 可改写为

$$L_{inf}(\mathcal{W},\{\mathbf{x}_i\})=\frac{1}{m\varepsilon^2}\sum_{i=1}^{m}\left[\varphi(\mathbf{x}_i+\varepsilon\mathcal{G}_{\mathcal{W}}(\mathbf{x}_i))-\varphi(\mathbf{x}_i)\right]^2 , \quad (10)$$

其中，我们在分母中显式引入了因子 ε^2，以考虑到一般变换按 ε [41]进行缩放这一事实。最小化损失函数后，训练得到的神经网络 $\mathcal{G}_{\widehat{\mathcal{W}}}$ 将表示相应的生成元

$$\mathbf{J} \;=\; \mathcal{G}_{\widehat{\mathcal{W}}}\,, \tag{11}$$

其中

$$\widehat{\mathcal{W}} \equiv \arg\min_{\mathcal{W}}\Big(L(\mathcal{W},\{\mathbf{x}_i\})\Big) \tag{12}$$

是使损失函数最小化的神经网络参数的学习所得值。原则上，$\mathcal{W}$的结果，因此$\mathbf{J}$的结果，将取决于初始化时网络参数的起始值 $\mathcal{W}_0$。如果现在在不同的初始条件 $\mathcal{W}_0$（或使用不同的超参数值）下重复训练 N_g 次，我们将得到一组 N_g（通常彼此不同的）生成元 $\{\mathbf{J}_\alpha\}$, $\alpha = 1, 2, \ldots, N_g$。

C. 正交性

为了使上一步得到的生成元集合 $\{\mathbf{J}_\alpha\}$, α =1, 2, … , N_g尽可能彼此不同，我们在损失函数中引入以下额外的正交项

$$L_{ortho}(\mathcal{W},\{\mathbf{x}_i\}) = \sum_{\alpha<\beta}^{N_g}\left(\sum_p \mathcal{W}_\alpha^{(p)}\mathcal{W}_\beta^{(p)}\right)^2, \tag{13}$$

其中，索引 p 遍历各个神经网络参数 $\mathcal{W}_\alpha^{(p)}$。

D. 闭合性

为了检验某组不同的生成元 $\{\mathbf{J}_\alpha\}$, $\alpha = 1, 2, \ldots, N_g$是否生成了一个群，我们需要检查代数（7）的闭合性。这可以通过几种方式实现。最有原则的方法是最小化

$$L_{closure}(a_{[\alpha\beta]\gamma}) = \sum_{\alpha<\beta}\mathrm{Tr}\left(\mathbf{C}^T_{[\alpha\beta]}\mathbf{C}_{[\alpha\beta]}\right), \tag{14}$$

关于候选结构常数参数 $a_{[\alpha\beta]\gamma}$，其中闭合失配定义为

$$\mathbf{C}_{[\alpha\beta]}(a_{[\alpha\beta]\gamma}) \equiv \big[\mathbf{J}_\alpha,\mathbf{J}_\beta\big] - \sum_{\gamma=1}^{N_g} a_{[\alpha\beta]\gamma}\mathbf{J}_\gamma. \tag{15}$$

由于 $L_{closure}$ 是半正定的，$L_{closure} = 0$ 表明代数是闭合的，因此我们处理的是一个真正的（子）群。实际操作中，[1] 我们将式（14）的最小化与前面讨论的损失函数各项同时进行，方法是将式（$\mathbf{J}_\alpha \to \mathcal{G}_{\mathcal{W}_\alpha}$ （15）中的 N_g 替换掉。采用这种同步构造的优点是，对于任意给定的 N_g，我们得到的每一组生成元都已经构成一个“尽可能闭合”的代数。因此，最终得到的训练损失总量大小可以表明，对于该 N_g 值，闭代数是否存在。

找到一组闭合的有效生成元后，我们可以在方便选取的典范基下重新训练神经网络，得到这组生成元的典范形式；其结构常数进而揭示所发现的对称变换背后的群的本质$\mathcal{G}_{\widehat{\mathcal{W}}_\alpha}$, $\alpha = 1, 2, \ldots, N_g$。

第V-X节通过若干复杂程度递增的示例说明上述步骤。

IV. 线性代数方法

通用逼近定理 [52] 保证，上一节的深度学习方法几乎可以处理任何对称变换，包括高度非线性的变换。与此同时，物理学中一大类有趣的对称性是线性变换，用线性代数的常规形式体系便足以处理。此外，对称生成元的分析涉及无穷小变换，这些变换用线性算子表示。出于这些原因，也为了吸引尚未熟悉机器学习技术细节的读者，本节将用线性代数的语言重新表述上一节的分析。我们沿用相同的符号和约定，但将表示神经网络的花体符号替换为相应的黑板粗体符号，表示作用于 n 维特征空间的 $n \times n$ 矩阵。

本节关注变换（5）中的线性子类，它们由一个一般矩阵编码 $\mathbb{F}$

$$\mathbf{x}' = \mathbb{F}\,\mathbf{x}\,, \tag{16}$$

其 n^2 分量 $\mathbb{F}_{ij}$ 由最小化损失函数确定

$$L_{inv}(\mathbb{F},\{\mathbf{x}_i\}) = \frac{1}{m}\sum_{i=1}^{m}\left[\varphi\left(\mathbb{F}\,\mathbf{x}_i\right) - \varphi(\mathbf{x}_i)\right]^2. \tag{17}$$

[1] 另一种可能的方法是，在展平并进行格拉姆–施密特正交归一化后，针对生成元空间，最小化对易子的空间外分量。

如果最小化成功，那么这种线性对称性就存在，并由学习得到的矩阵$\mathbb{F}$表示。

类比于（9），我们可以将相应的无穷小线性变换写为

$$\delta\mathbb{F}(\varepsilon) = \mathbb{I} + \varepsilon\,\mathbb{G}\,, \tag{18}$$

其中 $\mathbb{I}$是单位 $n\times n$矩阵， $\mathbb{G}$是一个 $n\times n$矩阵，其分量 $\mathbb{G}_{ij}$ 有待通过优化确定。为了得到一个单一的生成元矩阵 $\mathbb{J}$，我们可以使用一个类似于（10）的损失函数

$$L_{\mathbb{J}}(\mathbb{G},\{\mathbf{x}_i\}) = \frac{h_{inv}}{m\varepsilon^2}\sum_i^m \Big[\varphi\big(\mathbf{x}_i + \varepsilon\,\mathbb{G}\,\mathbf{x}_i\big) - \varphi(\mathbf{x}_i)\Big]^2 + h_{norm}\Big[\mathrm{Tr}\big(\mathbb{G}^T\mathbb{G}\big) - 2\Big]^2, \tag{19}$$

其中常数 h_{inv} 和 h_{norm} 在（19）中是超参数权重，用于决定损失函数（19）中两项的相对贡献，分别用于约束对称不变性和有限归一化[2] 条件。随后，通过最小化（19）即可得到实际的生成元 $\mathbb{J}$：

$$\mathbb{J} = \arg\min_{\mathbb{G}}\Big(L_{\mathbb{J}}(\mathbb{G},\{\mathbf{x}_i\})\Big)\,. \tag{20}$$

通过使用不同的初始条件重复这一过程多次，每次都会得到一个不同的生成元 $\mathbb{J}$。或者，我们也可以将 N_g 个损失函数（19）的副本相加并同时最小化，从而一次性生成全部生成元 N_g：

$$\sum_{\alpha=1}^{N_g} L_{\mathbb{J}}(\mathbb{G}_\alpha,\{\mathbf{x}_i\})\,, \tag{21}$$

这将得到一组 N_g 生成元 $\mathbb{J}_\alpha$，α =1、2、.....、N_g，以及它们各自的无穷小变换 $\delta\mathbb{F}_\alpha \equiv \mathbb{I} + \varepsilon\,\mathbb{J}_\alpha$。

此时，生成元 $\mathbb{J}_\alpha$ 彼此完全解耦且相互独立。我们可以添加一个类似于（13）的损失项，迫使它们彼此不同：

$$L_{ortho}(\mathbb{G},\{\mathbf{x}_i\}) = \sum_{\alpha<\beta}^{N_g}\big[\mathrm{Tr}(\mathbb{G}_\alpha^T\mathbb{G}_\beta)\big]^2\,. \tag{22}$$

最后，我们可以通过加入一个类似于（14）的损失项来强制满足闭合性质：

$$L_{closure}(a_{[\alpha\beta]\gamma}) = \sum_{\alpha<\beta}\mathrm{Tr}\left(\mathbb{C}^T_{[\alpha\beta]}\mathbb{C}_{[\alpha\beta]}\right), \tag{23}$$

[2] 为了使（18）成为无穷小变换，矩阵 $\mathbb{G}$ 需要是有限的。我们选择将生成元归一化为 $\mathrm{Tr}(\mathbb{G}_\alpha^T\mathbb{G}_\beta) = 2\delta_{\alpha\beta}$，因此（19）第二行中会出现因子 2。

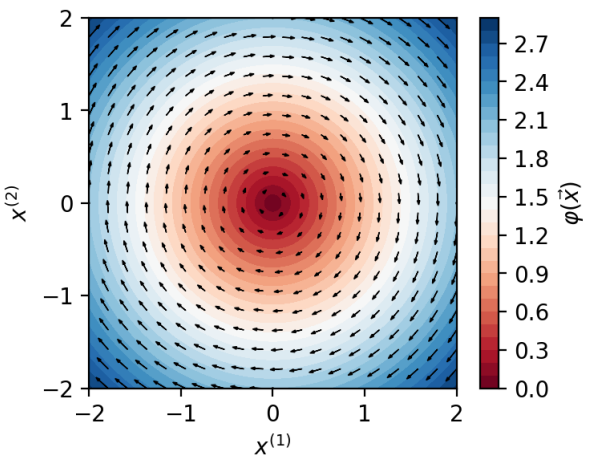


图 1. 一个保持预言机函数（25）不变的代表性对称变换，该变换由我们在第 V 节的二维练习中通过所述过程找到。在本图及随后所有类似图中，箭头表示由对称变换产生的位移 $\mathbf{x}'-\mathbf{x}$。色彩图展示了预言机函数 φ（$\mathbf{x}$）。

其中

$$\mathbb{C}_{[\alpha\beta]}(a_{[\alpha\beta]\gamma}) \equiv \big[\mathbb{G}_\alpha,\mathbb{G}_\beta\big] - \sum_{\gamma=1}^{N_g} a_{[\alpha\beta]\gamma}\mathbb{G}_\gamma\,, \tag{24}$$

并且还要相对于参数 $a_{[\alpha\beta]\gamma}$ 最小化总损失函数。

V. 二维中的保长度变换

在本节及接下来的两节中，我们将重点讨论保持特征向量的欧几里得长度不变的变换，即我们的预言机 φ 将返回

$$\varphi(\mathbf{x}) \equiv |\mathbf{x}| = \left\{\sum_{i=1}^{n}[x^{(i)}]^2\right\}^{1/2}. \tag{25}$$

在这种情况下，我们期望方法能够发现正交群 $O(n)$ 的对称性，其生成元可用克罗内克δ表示为

$$\left(\mathbb{O}_{mn}\right)^{ij} = \delta^i_m\delta^j_n - \delta^j_m\delta^i_n\,, \tag{26}$$

此时代数为

$$[\mathbb{O}_{mn},\mathbb{O}_{pq}] = \delta_{np}\mathbb{O}_{mq} + \delta_{mq}\mathbb{O}_{np} - \delta_{mp}\mathbb{O}_{nq} - \delta_{nq}\mathbb{O}_{mp}. \tag{27}$$

注意，生成元 $\mathbb{O}_{mn}$ 由两个指标标记，它们表示旋转平面。

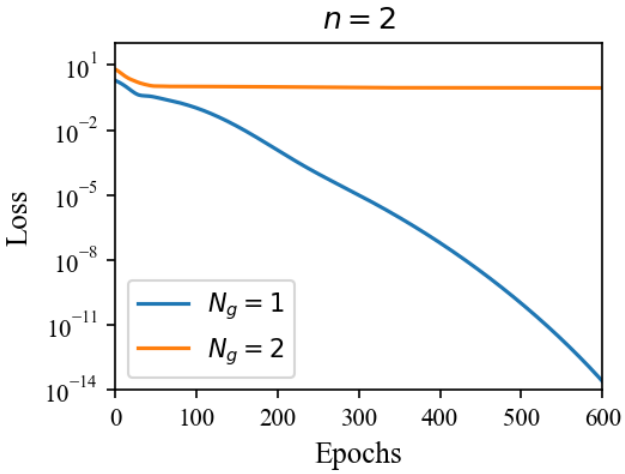


图 2. 第二节二维练习中，训练损失随训练轮次的变化情况：V。我们的算法找到了一个对称生成元（蓝色曲线所示的训练损失稳步下降），但未能找到两个不同的对称生成元（橙色曲线所示的训练损失仍然很大）。

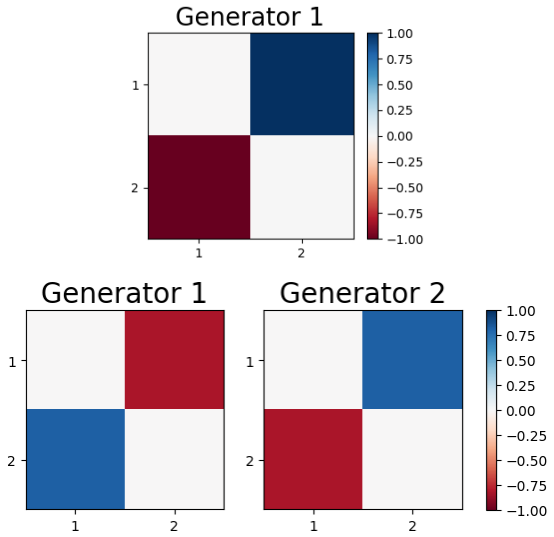


图 3. 顶行：针对 $n = 2$ 和 $N_g = 1$ 成功学习到的生成元。底行：针对 $n = 2$ 和 $N_g = 2$ 未能成功学习到的“生成元”。在本图及后续所有此类图中，每个面板都表示一个以矩阵形式呈现的学习所得生成元 $\mathbb{J}_\alpha$，矩阵各元素的值由颜色条标示。

在本节中，我们从最简单的二维情形 $n = 2$开始，它应对应于单生成元群 $O(2)$。首先，我们尝试生成一个通用的（即不一定是无穷小的）对称变换。为此，我们使用不变性损失（8）训练网络$\mathcal{F}_\mathcal{W}$。这项练习取得了成功；根据初始化的不同，我们找到了各种对称变换。它们都涉及在（$x^{(1)}$，$x^{(2)}$）平面内绕原点旋转。其中一个具有代表性的对称变换如图1所示。

接下来，我们讨论对称生成元代数。先从单个生成元开始，$N_g = 1$，此时无需在损失函数中加入正交性项和闭合项。训练取得成功，损失函数降至零，如图2中的蓝色实线所示。所得生成元的矩阵形式以图示方式呈现在图3的顶行中，我们立即认出这是熟悉的矩阵 $\mathbb{O}_{12}$，它来自式（26），生成 12 平面中的旋转

$$\mathbb{O}_{12} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} . \tag{28}$$

请注意，该生成元具有预期的反对称性质。

找到一个对称生成元后，我们接下来检查是否存在第二个不同的生成元。为此，我们在损失函数中加入正交性项和闭合项，并重复训练。这次训练未能成功，正如图 2中的橙色实线所示，损失在大约 50 个训练轮次后趋于平稳。在图 3的第二行中，我们展示了本例中找到的两个候选生成元的结果。值得注意的是，尽管它们确实具有二维旋转生成元的预期形式，但它们本质上是同一个变换，只相差一个整体符号。这意味着它们不满足正交条件——事实上，我们发现本例中较大的总损失主要来自正交性损失（13）。由于总损失很大，且随着进一步训练并未改善（见图 $N_g = 2$ 中的橙色曲线 2），因此这两个生成元均无效，应予以舍弃。由此我们得出结论，在这种情况下，不存在具有 $N_g = 2$个不同生成元的李代数。

顺带一提，在 $N_g = 2$ 情形下，经过多次训练运行，我们有时会发现，算法选择创建彼此正交、但并非真正对称变换的生成元；也就是说，正交性损失降至零，却是以不变性损失大幅增加为代价（10）。原则上，当机器面对两个相互排斥的要求时，很难预测它会如何选择。例如，在一些训练运行中，两个候选生成元最终变得完全相同，而不是相差一个负号。另请注意，图3 第二行中两个候选生成元的归一化不正确——这是因为程序也选择违反归一化条件，以降低正交性损失；在这种情况下，正交性损失是占主导地位的惩罚项。无论如何，由于失败的训练样例在理论上价值不大，以下我们只展示成功的训练运行所得到的有效（子）代数。

接下来，我们检查 $N_g > 2$ 的情况。同样，我们

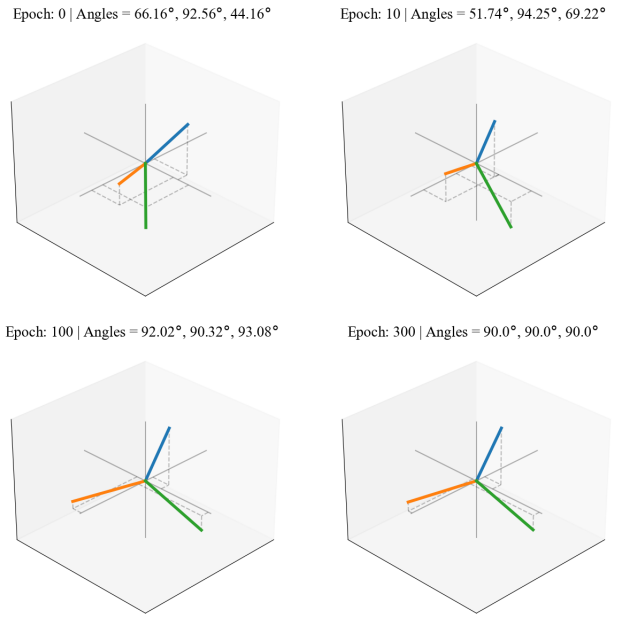


图 4. 训练中间阶段所发现的三个生成元的特征向量坐标轴：训练 1 轮后（左上）、10 轮后（右上）、100 轮后（左下）和 300 轮后（右下）。为方便起见，我们在每个面板顶部列出了每对坐标轴之间的角度（以度为单位）。

发现训练最终产生了较大的损失，而且闭代数不存在一致的解。因此我们得出结论，在这个 $n = 2$ 例子中，只有一个单参数对称群，其生成元为（28）。

虽然本节的例子相当简单，但它确实概述并验证了我们方法的主要步骤。接下来的几节将介绍更复杂的例子。

VI. 三维中的保长度变换

在本节中，我们将研究在 25 维空间中保持预言机不变的对称变换 $n = 3$ 。再次发现，仅使用不变性损失 8 进行训练，总能得到有效的对称变换。我们还观察到，矩阵形式 $\mathbb{F}$ 是反对称的，这与正交群 $O(3)$ 的预期一致。不过，数据现在是三维的，因此很难像图 1 那样直接将对称变换可视化。相反，我们选择绘制由实特征向量所定义的对称轴（在本例中为旋转轴）。图 4 展示了 $N_g = 3$ 个生成元对应的情形下，训练不同阶段得到的变换。每个面板顶部列出了每对坐标轴之间的相对角度（以度为单位）。请注意，对称轴起初的方向是随机的，但正交性损失项 13 会逐渐将它们推向相互正交构型。

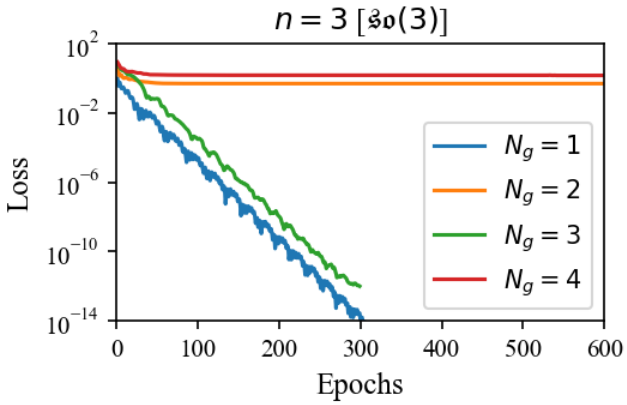


图 5. 与图 2 相同，不过针对的是 $n = 3$ 第 VI 节讨论的练习。训练结果得到了含一个或三个生成元的有效闭代数，但无法得到含两个或四个生成元的闭代数。

为了分析所发现的对称变换的群结构，我们接下来研究无穷小变换的生成元。首先，我们尝试确定完整代数的维数，即能够构成闭代数的线性无关生成元的最大数量。图 5 展示了 N_g 取不同值时的损失曲线：1、2、3 和 4（对于 $N_g \geq 5$，我们不展示结果，因为其损失较大）。我们观察到，只有在 $N_g = 1$ 和 $N_g = 3$ 的情况下，训练才取得成功。这表明完整代数有 3 个生成元，与正交 $n(n-1)/2$ 群具有 $O(n)$ 个生成元的预期相符。同时，在 $N_g = 1$ 处训练成功，揭示了一个仅含单个生成元的子群，它属于 $O(3)$，这正是上一节讨论的 $O(2)$。

典型训练运行的结果 $N_g = 3$ 如图6所示，其中上排展示了成功学习到的三个生成元 $\mathbb{J}_\alpha$、 $\alpha = 1$、2、3，而下方的面板则以图示形式展示了矩阵形式的结构常数，如下所示。每一行（标记为$\alpha\beta = 12$、31、23）代表三种可能的对易子之一，而各列（标记为 $\gamma = 1$、2、3）代表图中上方各面板所示的已发现生成元 $\mathbb{J}_\gamma$ 。然后，每个单元格中的条目表示定义方程（7）中的结构常数 $a_{[\alpha\beta]\gamma}$ 。

仔细观察图 6 最上方一行可以发现，我们的示例中找到的三个生成元近似为

$$\mathbb{J}_1 \approx \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} = -\mathbb{O}_{31}, \tag{29a}$$

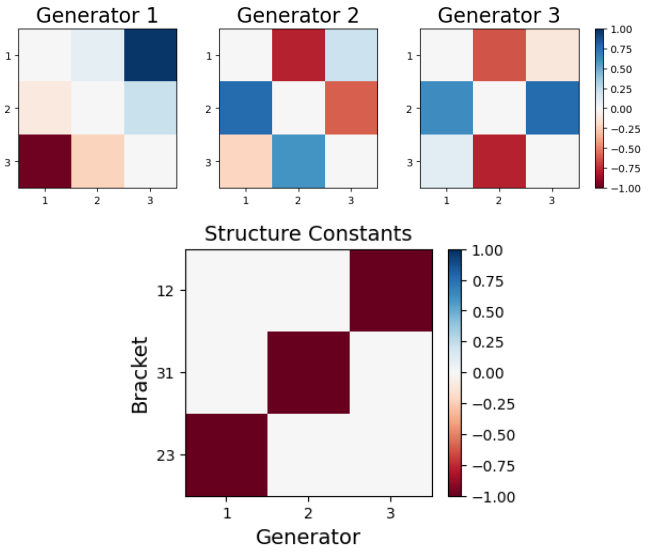


图6。顶行：成功学习得到的生成元，用于$n = 3$，$N_g = 3$第VI节中讨论的练习。底部面板：相应的结构常数的矩阵形式（见正文）。

$$\mathbb{J}_2 \approx \frac{1}{\sqrt{2}}\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} = -\frac{1}{\sqrt{2}}(\mathbb{O}_{12} + \mathbb{O}_{23}), \quad (29b)$$

$$\mathbb{J}_3 \approx \frac{1}{\sqrt{2}}\begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix} = -\frac{1}{\sqrt{2}}(\mathbb{O}_{12} - \mathbb{O}_{23}). \quad (29c)$$

底部面板显示，找到的三个生成元所构成的代数 $\mathbb{J}_\alpha$ 为

$$[\mathbb{J}_\alpha, \mathbb{J}_\beta] = -\epsilon_{\alpha\beta\gamma}\mathbb{J}_\gamma, \quad (30)$$

其中我们识别出通常的 SO(3)代数，其中涉及 Levi-Civita 置换符号$\epsilon_{\alpha\beta\gamma}$。从现在起，对于重复的生成元类型指标 α，β，γ，……，我们将使用爱因斯坦求和约定。

VII. 四维中的保长度变换

在本节中，我们将前两节的讨论推广到 $n = 4$ 维的情形。这里的新特点是存在多个具有不同维数的非平凡子代数。

通过不变性损失（8）发现单个有限对称变换十分简单，而且总能找到四维空间中的某个有限旋转，并以正交矩阵表示。因此，本节仅聚焦于识别（子）代数。与之前一样，我们针对多个不同生成元的数量取不同值（$N_g = 2$、3、4、5、6、7），并开启正交性和

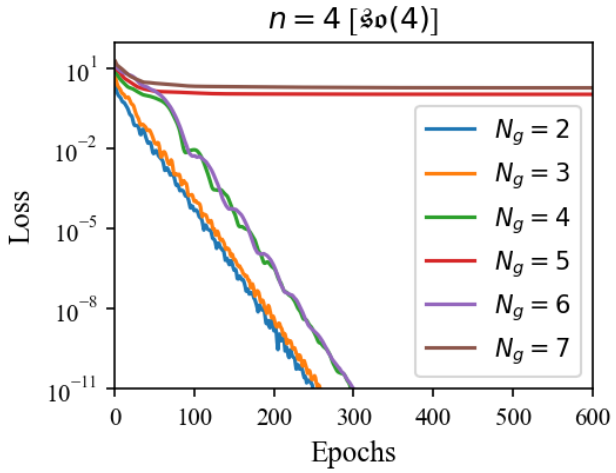


图7。与图5相同，但针对 $n = 4$ 第VII节中讨论的练习。训练结果得到包含一个（未显示）、两个、三个、四个和六个生成元的有效闭代数，但不包含五个或七个生成元。

闭合性损失。图7展示了具有代表性的损失曲线。我们观察到，在其中四种情况下找到了闭代数，即 $N_g = 2$、3、4、6。下面将逐一讨论这些情况（当然，$N_g = 1$这一平凡情形始终可行，因此从现在起不再专门讨论）。

A. 两生成元子代数

我们发现的一个封闭子代数示例如图 $N_g = 2$ 所示，该子代数的生成元8。上方各图表明，发现的这两个生成元可近似表示为

$$\mathbb{J}_1 \approx \begin{pmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \mathbb{O}_{12}, \quad (31a)$$

$$\mathbb{J}_2 \approx \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{pmatrix} = \mathbb{O}_{34}, \quad (31b)$$

其中我们辨认出旋转生成元 $\mathbb{O}_{12}$ 和$\mathbb{O}_{34}$，它们来自式（26）。从式（31）的矩阵形式可见，这两个生成元彼此对易，因为这两次旋转完全发生在不同的平面内。因此，由 $\mathbb{J}_1$ 和 $\mathbb{J}_2$ 构成的代数是阿贝尔的，这一点也由图 8下方的图独立验证。

需要注意的是，我们无法控制通过该过程找到的生成元的整体方向。图 8 中的示例经过精心挑选，因此可以根据典型生成元轻松辨认。

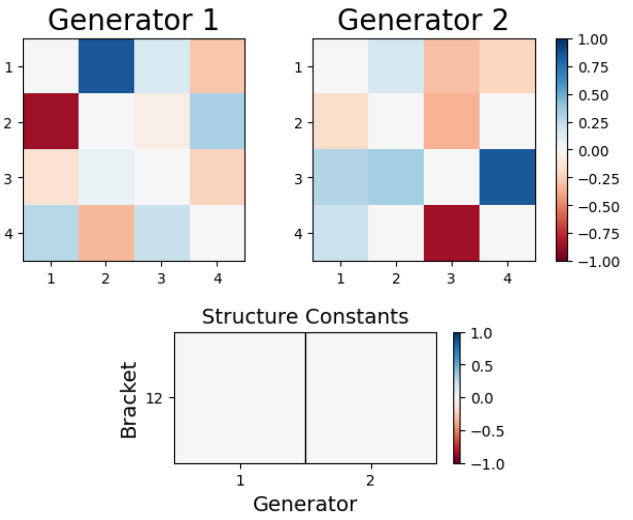


图8。上排：$n = 4$， $N_g = 2$ 在第VIIA节讨论的练习中成功学习到的生成元。下方：对应的结构常数。结构常数为零表明这是一个阿贝尔子群。

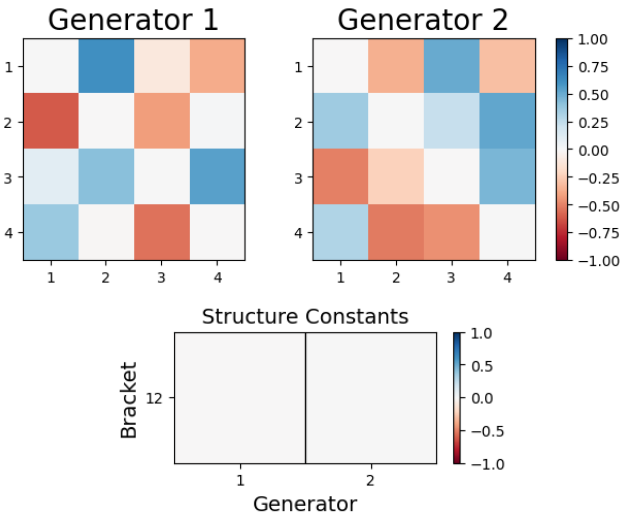


图9。与图8相同，但采用了不同的训练运行，$N_g = 2$仍然如此。

（26）。一次典型的训练运行通常会以某种随机方向返回生成元集合，但仍保留其对易性质。图 9展示了这样一个典型示例。这一次，找到的两个生成元 $\mathbb{J}_1$ 和 $\mathbb{J}_2$ 是六个典型生成元的更一般的线性组合： $\mathbb{O}_{12}$、 $\mathbb{O}_{13}$、 $\mathbb{O}_{14}$、 $\mathbb{O}_{23}$、 $\mathbb{O}_{24}$和 $\mathbb{O}_{34}$ ，它们属于 $O\,(4)$ 群。尽管如此，找到的生成元 $\mathbb{J}_1$ 和 $\mathbb{J}_2$ 仍然对易，并构成完整对称群的阿贝尔二维子代数。

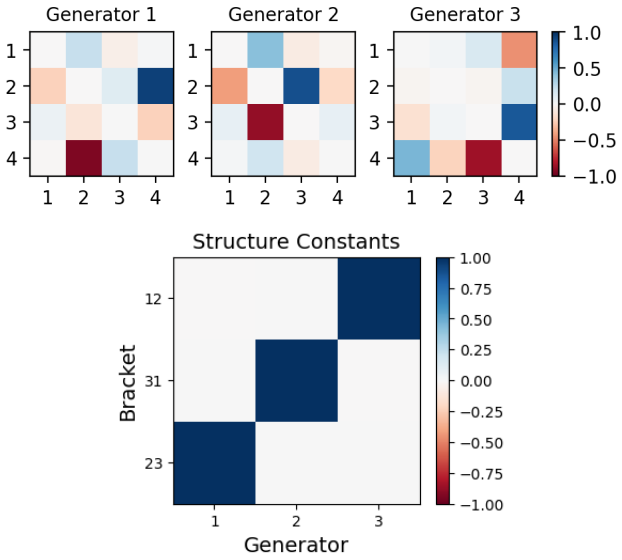


图10。上排：$n = 4$， $N_g = 3$ 在第VIIB节讨论的练习中成功学习到的生成元。下方：对应的结构常数，可识别为 SO(3)代数的结构常数（见正文）。

B. 三生成元子代数

接下来讨论发现的具有三个生成元的子代数（ $N_g = 3$）。由于 $SO(4)$包含 $SO(3)$作为子群，[3] 我们知道这样的子代数应当存在，而且确实找到了这样的解，如图7所示。如上所述，一次典型的训练运行会以随机取向生成这三个生成元。为便于解读，图10 展示了一个精心挑选的例子，其中上方面板所示的生成元可以辨认为近似

$$\mathbb{J}_1 \approx \mathbb{O}_{24}, \qquad \mathbb{J}_2 \approx \mathbb{O}_{23}, \qquad \mathbb{J}_3 \approx \mathbb{O}_{34}. \tag{32}$$

该代数主要涉及最后三个特征维度上的旋转，而第一个特征基本不受对称性的影响。图 10的下方图版证实这是式（ $SO30$）中的 (3) 代数。当然，更常规的训练运行会产生一组三个生成元，它们涉及全部四个特征维度，但仍具有相同的对易关系。

C. 四生成元子代数

图 7 表明，我们的方法也能找到一个有四个生成元的代数。要从理论上探究其起源，

[3] 由于 $SO(4) = SO(3) \otimes SO(3)$， $SO(4)$ 代数是两个独立 $SO(3)$ 子代数的直和。

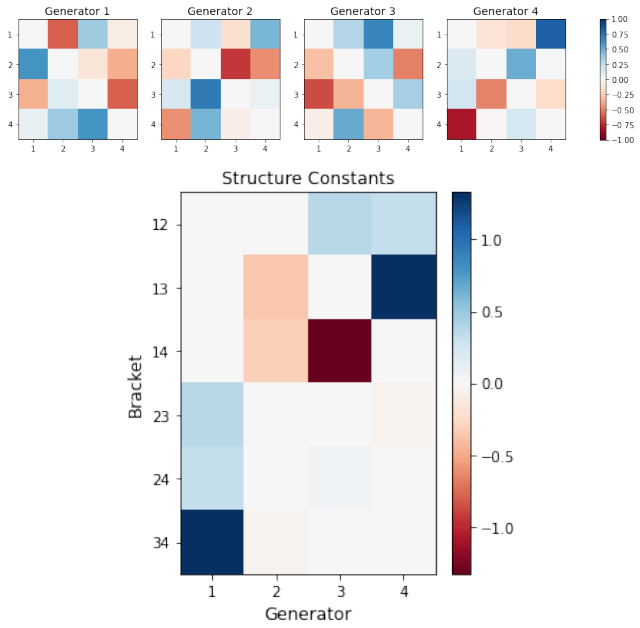


图 11. 上排：成功学得的生成元，对应于 $n = 4$， $N_g = 4$ 练习，该练习见第 VIIC 节。下图：相应的结构常数。

根据原基中的对易生成元对的和与差定义一个新的生成元基（26） [53]

$$\mathbb{S}_1 \equiv \tfrac{1}{2}\left(\mathbb{O}_{34} + \mathbb{O}_{12}\right), \quad \mathbb{D}_1 \equiv \frac{1}{2}\left(\mathbb{O}_{34} - \mathbb{O}_{12}\right), \tag{33a}$$

$$\mathbb{S}_2 \equiv \tfrac{1}{2}\left(\mathbb{O}_{42} + \mathbb{O}_{13}\right), \quad \mathbb{D}_2 \equiv \frac{1}{2}\left(\mathbb{O}_{42} - \mathbb{O}_{13}\right), \tag{33b}$$

$$\mathbb{S}_3 \equiv \tfrac{1}{2}\left(\mathbb{O}_{23} + \mathbb{O}_{14}\right), \quad \mathbb{D}_3 \equiv \frac{1}{2}\left(\mathbb{O}_{23} - \mathbb{O}_{14}\right). \tag{33c}$$

这个基变换将原始生成元的代数（27）解耦如下

$$[\mathbb{S}_i, \mathbb{S}_j] = -\epsilon_{ijk}\,\mathbb{S}_k, \tag{34a}$$

$$[\mathbb{D}_i, \mathbb{D}_j] = -\epsilon_{ijk}\,\mathbb{D}_k, \tag{34b}$$

$$[\mathbb{S}_i, \mathbb{D}_j] = 0. \tag{34c}$$

因此，四个生成元构成的闭代数可以由三项 $\mathbb{S}$ 加任意一项 $\mathbb{D}$ 构成，也可以由三项 $\mathbb{D}$ 加任意一项 $\mathbb{S}$ 构成。无论哪种情况，三个生成元都满足SO 型对易关系，而第四个生成元与其他所有生成元对易。图11展示了通常采用的表示形式，呈现了一次训练运行中一个代表性结果所找到的生成元及其代数，运行参数为 $N_g = 4$。我们观察到，找到的生成元近似为

$$\mathbb{J}_1 \equiv \frac{1}{2}\left(\mathbb{O}_{21} + \mathbb{O}_{43}\right) = -\,\mathbb{S}_1, \tag{35a}$$

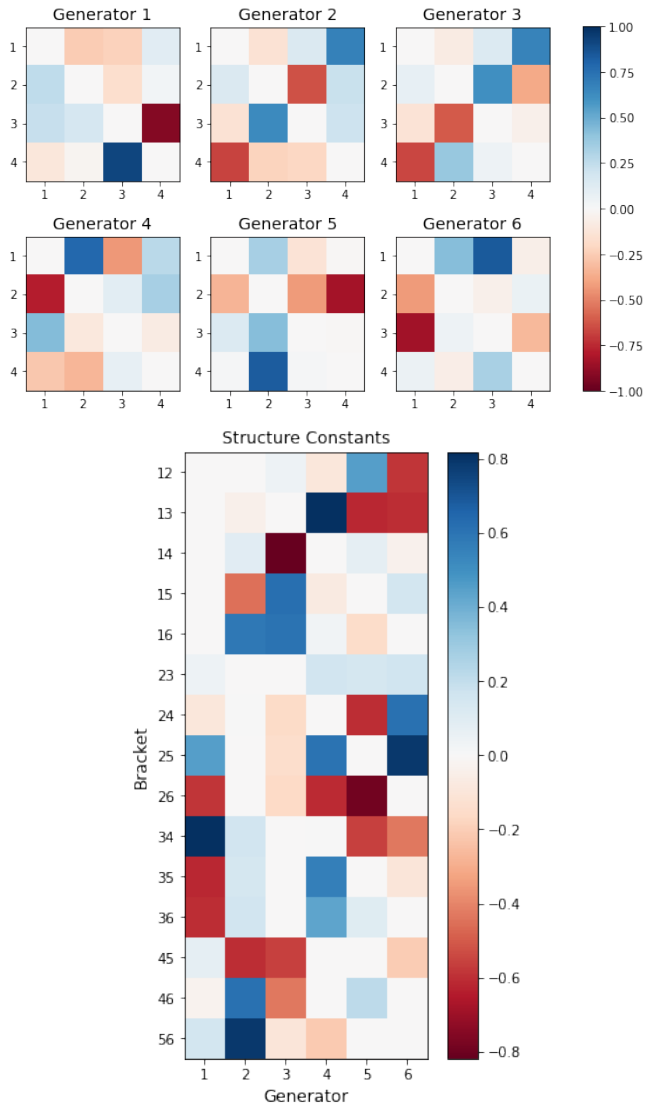


图 12. 上排：成功学得的生成元，对应于 $n = 4$， $N_g = 6$ 练习，该练习见第 VIID 节。下图：相应的结构常数。

$$\mathbb{J}_2 \equiv \frac{1}{2}\left(\mathbb{O}_{32} + \mathbb{O}_{14}\right) = -\,\mathbb{D}_3, \tag{35b}$$

$$\mathbb{J}_3 \equiv \frac{1}{2}\left(\mathbb{O}_{13} + \mathbb{O}_{42}\right) = \mathbb{S}_2, \tag{35c}$$

$$\mathbb{J}_4 \equiv \frac{1}{2}\left(\mathbb{O}_{14} + \mathbb{O}_{23}\right) = \mathbb{S}_3. \tag{35d}$$

因此， $\mathbb{J}_1$、 $\mathbb{J}_3$ 和 $\mathbb{J}_4$ 构成一个 SO(3) 代数，这由公式 (34a所示；此外，它们三者都与 $\mathbb{J}_2$对易，这由公式 (34c 所示。这一模式正是我们在图11下方面板中观察到的。

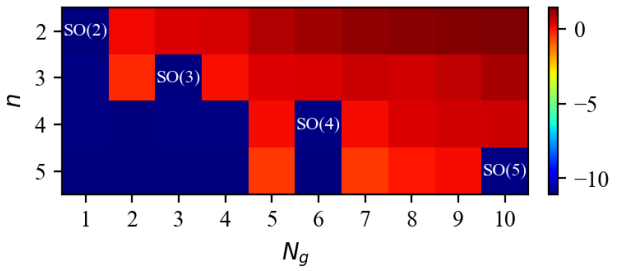


图 13. 我们搜索保欧几里得预言机的变换李代数（25），在 $n = 2$、3、4、5 维空间中，以及不同生成元数量 N_g下所得结果。单元格按训练期间达到的最低损失值的以 10 为底的对数进行颜色编码。

D. 六生成元代数

最后，我们来看 N_g 的情况，它将揭示 SO（4）的完整代数。图12 展示了一次具有代表性的训练运行结果，该运行旨在寻找一个由$N_g = 6$ 个生成元构成的闭代数。在学得的生成元中，我们可以大致辨认出 $\mathbb{J}_1 \approx \mathbb{O}_{43}$、$\mathbb{J}_2 \approx -\mathbb{D}_3$、$\mathbb{J}_3 \approx \mathbb{S}_3$、$\mathbb{J}_4 \approx \mathbb{O}_{12}$、$\mathbb{J}_5 \approx \mathbb{O}_{42}$ 和 $\mathbb{J}_6 \approx \mathbb{O}_{13}$。

前面三节的分析可以很容易地扩展到更高维度（$n \geq 5$）。我们又检查了 n 的几个值，每次都能奏效——我们得到了保留预言机（25）的有效有限对称变换，找到了完整 $n(n-1)/2$ 个正交 $O(n)$ 群生成元所构成的闭代数，以及有效的子代数。作为消遣，我们在图 26 中展示了推导出的 $SO(10)$ 群的 45 个生成元。

在总结我们对正交群的讨论时，图13汇总了我们为 $n \leq 5$ 以及不同生成元数量 N_g下找到的闭代数和子代数的结果。表格中的每个单元格都代表一个维度数固定 n 且生成元数量固定为 N_g的独立练习。单元格按训练期间达到的最低损失值的以 10 为底的对数着色。损失达到 10^{-12}时训练即终止，因此蓝色单元格表示成功的训练运行，最终得到了闭代数。每一行最右侧的蓝色单元格对应完整代数，在本例中为 $SO(n)$，如图所示。前面的蓝色单元格对应各种子代数。事实上，我们原本没有预料到在 $N_g = 4$ 的情况下会存在这些子代数 $n = 4$ 和 $n = 5$，但我们的模型给了我们惊喜。

VIII. 四维闵可夫斯基空间中的洛伦兹变换

本节考虑四维闵可夫斯基时空（t，x，y，z）（采用自然单位制，其中$c = 1$）。通常的洛伦兹变换保持二次型

$$\varphi(t,x,y,z) = t^2 - x^2 - y^2 - z^2, \quad (t,x,y,z) \in \mathbb{R}^4, \tag{36}$$

因此，这将是本节中的预言机。四个输入特征为

$$x^{(0)} = t, \quad x^{(1)} = x, \quad x^{(2)} = y, \quad x^{(3)} = z, \tag{37}$$

其中，为遵循标准物理学记号，我们从 0 开始为特征编号。

我们的分析照常进行。首先，我们找出对称变换，它们通常是推动和旋转的组合。经过检查，我们确认这些变换具有所需的对称性

$$\mathbb{F}_{0i} = \mathbb{F}_{i0}, \quad \mathbb{F}_{ij} = -\mathbb{F}_{ji}, \quad \mathbb{F}_{00} = \mathbb{F}_{ii} = 0, \quad \forall i = 1,2,3.$$

接下来，我们分析生成元代数。在给出数值结果之前，为方便读者，我们先总结有关洛伦兹群 $O(1,3)$ 数学结构的一些相关信息。它有六个生成元：三个推动生成元 $\mathbb{K}_i$，

$$\mathbb{K}_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \tag{38a}$$

$$\mathbb{K}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \tag{38b}$$

$$\mathbb{K}_3 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}, \tag{38c}$$

以及旋转的三个生成元 $\mathbb{L}_i$，由下式给出

$$\mathbb{L}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{39a}$$

$$\mathbb{L}_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{pmatrix}, \tag{39b}$$

$$\mathbb{L}_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}, \tag{39c}$$

	$\mathbb{K}_1-\mathbb{L}_2$	$\mathbb{K}_2+\mathbb{L}_1$	$\mathbb{K}_3$	$\mathbb{L}_3$	$\mathbb{L}_2$	$\mathbb{L}_1$
	$\mathbb{X}_1$	$\mathbb{X}_2$	$\mathbb{X}_3$	$\mathbb{X}_4$	$\mathbb{X}_5$	$\mathbb{X}_6$
$\mathbb{X}_1$	0	0	$-\mathbb{X}_1$	$-\mathbb{X}_2$	$+\mathbb{X}_3$	$+\mathbb{X}_4$
$\mathbb{X}_2$	0	0	$-\mathbb{X}_2$	$+\mathbb{X}_1$	$+\mathbb{X}_4$	$-\mathbb{X}_3$
$\mathbb{X}_3$	$+\mathbb{X}_1$	$+\mathbb{X}_2$	0	0	$-\mathbb{X}_1-\mathbb{X}_5$	$\mathbb{X}_2-\mathbb{X}_6$
$\mathbb{X}_4$	$+\mathbb{X}_2$	$-\mathbb{X}_1$	0	0	$-\mathbb{X}_6$	$+\mathbb{X}_5$
$\mathbb{X}_5$	$-\mathbb{X}_3$	$-\mathbb{X}_4$	$+\mathbb{X}_1+\mathbb{X}_5$	$+\mathbb{X}_6$	0	$-\mathbb{X}_4$
$\mathbb{X}_6$	$-\mathbb{X}_4$	$+\mathbb{X}_3$	$-\mathbb{X}_2+\mathbb{X}_6$	$-\mathbb{X}_5$	$+\mathbb{X}_4$	0

表I. 以生成元表示的洛伦兹代数 (43). 单元格显示了将最左列中的一个生成元与顶行中的一个生成元进行对易所得的结果在顶行。

完整的洛伦兹代数可概括如下

$$[\mathbb{L}_i,\mathbb{L}_j]=\epsilon_{ijk}\,\mathbb{L}_k, \tag{40}$$

$$[\mathbb{L}_i,\mathbb{K}_j]=\epsilon_{ijk}\,\mathbb{K}_k, \tag{41}$$

$$[\mathbb{K}_i,\mathbb{K}_j]=-\epsilon_{ijk}\,\mathbb{L}_k. \tag{42}$$

在不同的基底下，子群结构最容易看出，

$$\mathbb{X}_1=\mathbb{K}_1-\mathbb{L}_2, \tag{43a}$$

$$\mathbb{X}_2=\mathbb{K}_2+\mathbb{L}_1, \tag{43b}$$

$$\mathbb{X}_3=\mathbb{K}_3, \tag{43c}$$

$$\mathbb{X}_4=\mathbb{L}_3, \tag{43d}$$

$$\mathbb{X}_5=\mathbb{L}_2, \tag{43e}$$

$$\mathbb{X}_6=\mathbb{L}_1. \tag{43f}$$

所得代数在 $\mathbb{X}_i$ 基底下的总结见表I。通过观察可知，洛伦兹代数有几个非平凡子代数。

A. 两个生成元子代数

存在若干二维子代数：

- 阿贝尔子代数。 有两个阿贝尔子代数：第一个由生成元集合 $\{\mathbb{X}_1,\mathbb{X}_2\}$生成，第二个则由$\{\mathbb{X}_3,\mathbb{X}_4\}$生成。前一种情况的一个代表性示例如图 14所示。我们将这两个

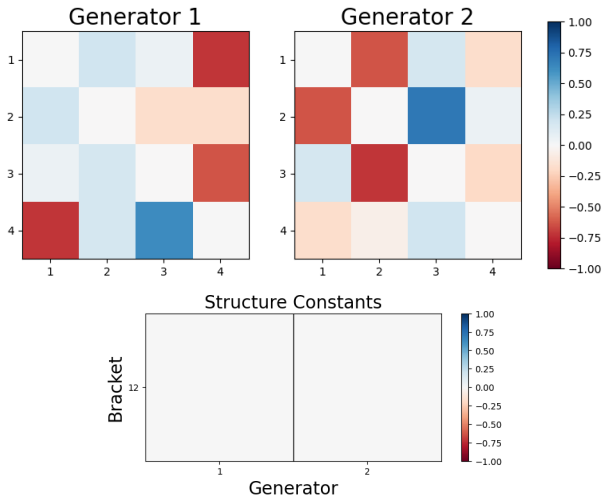


图 14. 上排：使用伪欧几里得预言机（$n=4,\ \ N_g=2$ 成功学习到的生成元，对应于第 VIIIA 节中讨论的练习（36）。下方面板：相应的结构常数。此例展示了一个阿贝尔子代数。

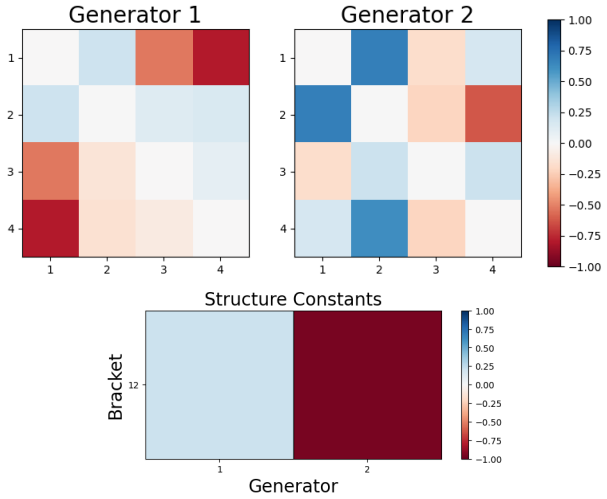


图 15. 与图 14 相同，但展示了我们用该方法找到的一个非阿贝尔子代数。

发现的生成元近似为

$$\mathbb{J}_1\approx-\left(\mathbb{K}_3-\mathbb{L}_1\right), \tag{44a}$$

$$\mathbb{J}_2\approx-\left(\mathbb{K}_1+\mathbb{L}_3\right). \tag{44b}$$

这一结果与公式（43a）和（43b）一致——这些变换是沿两个坐标轴的推动与绕其中两个坐标轴的旋转的组合；在本例中，这两个坐标轴为 x 和 z。

- 非阿贝尔子代数。如表I所示，洛伦兹群的李代数有两个非阿贝尔子代数，分别由 $\{\mathbb{X}_1,\mathbb{X}_3\}$ 和 $\{\mathbb{X}_2,\mathbb{X}_3\}$生成，

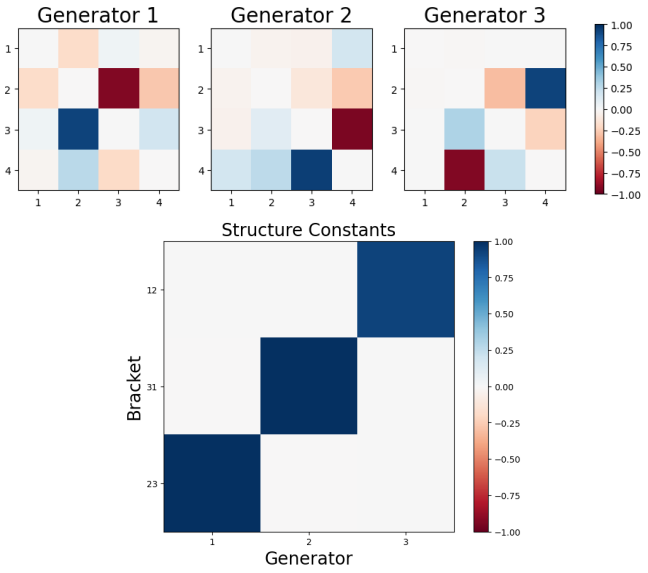


图16。顶部：针对第 $n = 4$节 $N_g = 3$ 中所述的练习，使用伪欧几里得预言机（VIIIB，36）成功学习到的生成元。底部面板：相应的结构常数，可识别为 $SO(3)$ 型子代数的结构常数。

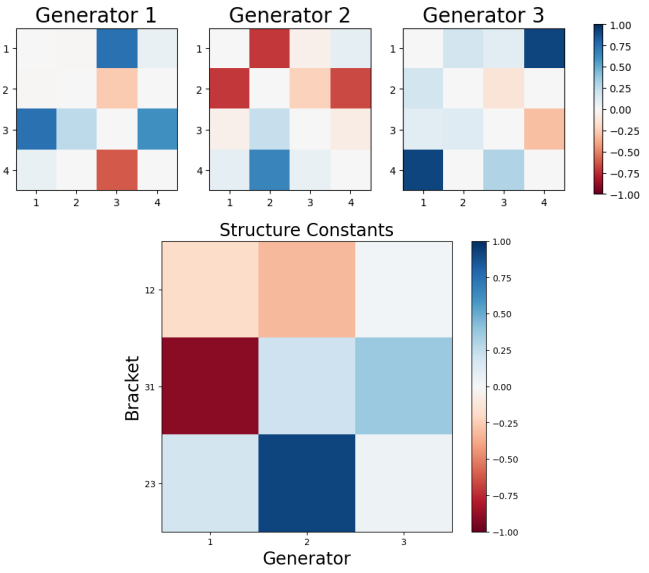


图17。与图16相同，但展示了一个$Hom(2)$ 子代数的示例（47）。

分别。图15展示了这一非阿贝尔情形的一个代表性训练示例。顶部面板显示，找到的两个生成元近似为

$$\mathbb{J}_1 \approx -\mathbb{K}_3 = -\mathbb{X}_3, \tag{45a}$$

$$\mathbb{J}_2 \approx \mathbb{K}_1 - \mathbb{L}_2 = \mathbb{X}_1, \tag{45b}$$

而底部面板证实，它们的李括号近似为 $[\mathbb{J}_1, \mathbb{J}_2] \approx [-\mathbb{X}_3, \mathbb{X}_1] = -\mathbb{X}_1 = -\mathbb{J}_2$，正如表I所预期的那样。

B. 三生成元子代数

表I揭示了若干个三维子代数：

• *SO(3)*。集合 $\{\mathbb{X}_4, \mathbb{X}_5, \mathbb{X}_6\}$ 生成了一个子代数，该子代数同构于 SO（3）三维旋转群的李代数，见第VI节。图16展示了一个具有代表性的训练示例。生成元可以识别为

$$\mathbb{J}_1 \approx \mathbb{L}_3, \tag{46a}$$

$$\mathbb{J}_2 \approx \mathbb{L}_1, \tag{46b}$$

$$\mathbb{J}_3 \approx \mathbb{L}_2. \tag{46c}$$

此外，我们还发现了一些例子，其子代数与 so(3)同构，由一个旋转和两个推动，或一个推动和两个旋转组成。

• *Hom(2)*。集合 $\{\mathbb{X}_1, \mathbb{X}_2, \mathbb{X}_3\}$ 生成的子代数与Hom(2)（欧几里得相似变换群）的李代数同构：

$$[\mathbb{X}_1, \mathbb{X}_2] = 0, \ [\mathbb{X}_3, \mathbb{X}_1] = \mathbb{X}_1, \ [\mathbb{X}_2, \mathbb{X}_3] = -\mathbb{X}_2. \tag{47}$$

一个具有代表性的训练示例如图所示 17。从顶排的图中，我们可以辨认出所找到的三个生成元为

$$\mathbb{J}_1 \approx +\mathbb{K}_2 - \mathbb{L}_1, \tag{48a}$$

$$\mathbb{J}_2 \approx -\mathbb{K}_1 - \mathbb{L}_2, \tag{48b}$$

$$\mathbb{J}_3 \approx +\mathbb{K}_3. \tag{48c}$$

图 17 中的底部图确认，它们的代数近似符合式（47）。

• *E(2)*。 $\{\mathbb{X}_1, \mathbb{X}_2, \mathbb{X}_4\}$ 生成的子代数与 E(2)（欧几里得群）的李代数同构：

$$[\mathbb{X}_1, \mathbb{X}_2] = 0, \ [\mathbb{X}_4, \mathbb{X}_1] = \mathbb{X}_2, \ [\mathbb{X}_2, \mathbb{X}_4] = \mathbb{X}_1. \tag{49}$$

一个具有代表性的示例如图 18所示，我们可以由此辨认出所找到的三个生成元为

$$\mathbb{J}_1 \approx -\mathbb{K}_3 - \mathbb{L}_2, \tag{50a}$$

$$\mathbb{J}_2 \approx -\mathbb{L}_1, \tag{50b}$$

$$\mathbb{J}_3 \approx -\mathbb{K}_2 + \mathbb{L}_3. \tag{50c}$$

如预期的那样，其中两个生成元包含沿着两个坐标轴推动并绕其旋转的组合，

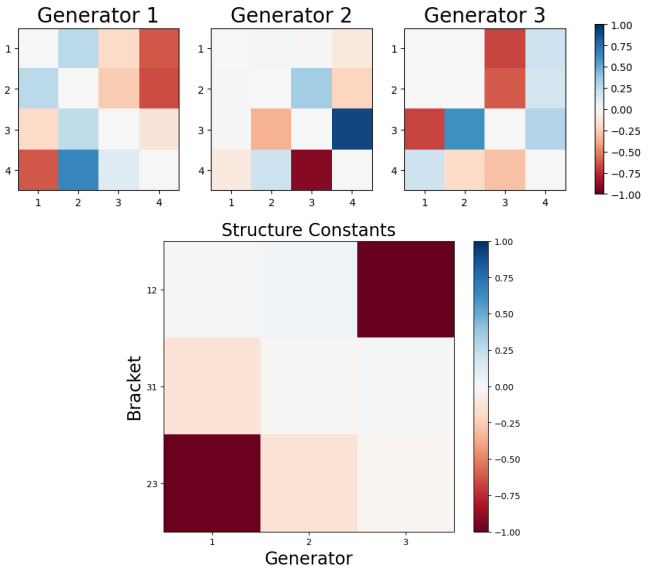


图18。与图16相同，但展示了一个$E(2)$ 子代数（49）。

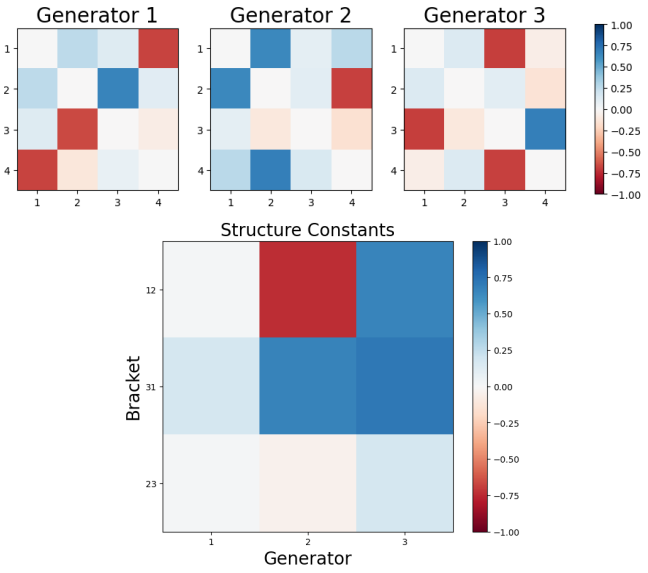


图19。与图16相同，但展示了一个$Bianchi\ VII_a$ 子代数示例。

在本例中， y 和 z；第三个生成元则是绕剩余坐标轴x旋转。图18底部面板显示，所得代数近似为式（49）所示的代数。

• $Bianchi\ VII_a$。集合 $\{\mathbb{X}_1, \mathbb{X}_2, \mathbb{X}_3 + a\mathbb{X}_4\}$ 与$a \neq 0$ 生成了一个 Bianchi VII$_a$ 子代数。图19展示了一个具有代表性的训练示例，从中我们识别出所找到的三个生成元为

$$\mathbb{J}_1 \approx -\mathbb{K}_3 - \mathbb{L}_3 = -(\mathbb{X}_3 + \mathbb{X}_4), \tag{51a}$$

$$\mathbb{J}_2 \approx +\mathbb{K}_1 - \mathbb{L}_2 = \mathbb{X}_1, \tag{51b}$$

$$\mathbb{J}_3 \approx -\mathbb{K}_2 - \mathbb{L}_1 = -\mathbb{X}_2. \tag{51c}$$

该集合对应于一个 Bianchi VII$_a$ 子代数，其中 $a = 1$，其结构常数确实与图19下方面板中的结果一致。

• *SL(2,R)*。集合 $\{\mathbb{X}_1, \mathbb{X}_3, \mathbb{X}_5\}$ 生成了一个同构于 $SL(2, R)$ 的李代数的子代数； $SL(2, R)$ 是双曲平面的等距变换群：

$$[\mathbb{X}_1, \mathbb{X}_3] = -\mathbb{X}_1, \tag{52a}$$

$$[\mathbb{X}_5, \mathbb{X}_1] = -\mathbb{X}_3, \tag{52b}$$

$$[\mathbb{X}_3, \mathbb{X}_5] = -\mathbb{X}_1 - \mathbb{X}_5. \tag{52c}$$

图20展示了一个具有代表性的训练示例，从中我们识别出所找到的三个生成元为

$$\mathbb{J}_1 \approx \mathbb{K}_1 - \mathbb{L}_3, \tag{53a}$$

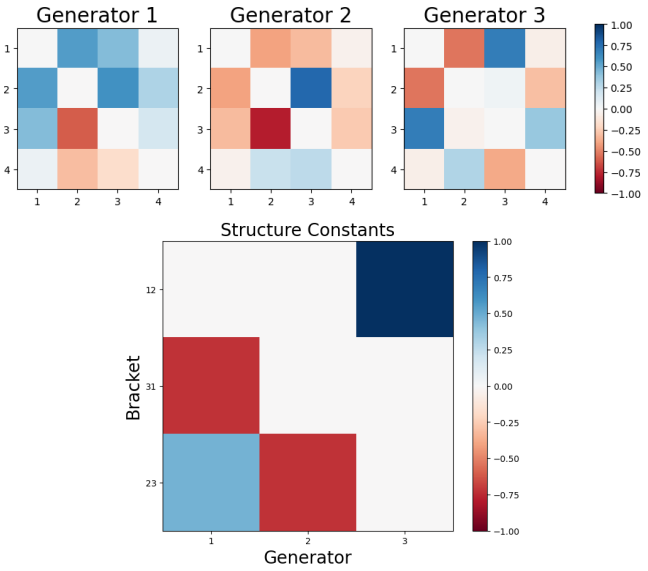


图20。与图16相同，但展示了一个 $SL(2,R)$ 子代数(52)的示例。

$$\mathbb{J}_2 \approx -\mathbb{L}_3, \tag{53b}$$

$$\mathbb{J}_3 \approx \mathbb{K}_2. \tag{53c}$$

很容易验证，图20 下半部分给出的结构常数与式（52）一致。

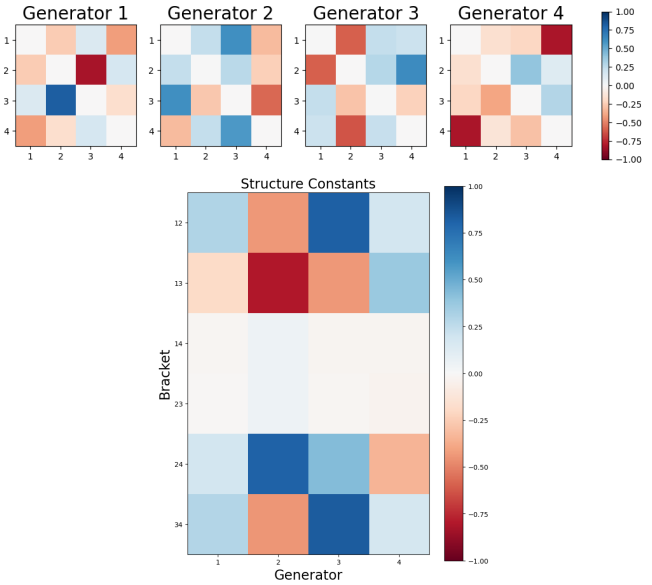


图 21. 上排：使用伪欧几里得预言机（ $n = 4$， $N_g = 4$ ）成功学习到的生成元，对应第 VIIIC 节中讨论的练习（36）。下方：对应的结构常数。

C. 四生成元子代数

唯一的四维子代数由以下生成$\{\mathbb{X}_1, \mathbb{X}_2, \mathbb{X}_3, \mathbb{X}_4\}$。图21展示了一个代表性示例。找到的四个生成元如下：

$$\mathbb{J}_1 \approx \mathbb{L}_3 = \mathbb{X}_4, \tag{54a}$$

$$\mathbb{J}_2 \approx \mathbb{K}_2 + \mathbb{L}_1 = \mathbb{X}_2, \tag{54b}$$

$$\mathbb{J}_3 \approx -\mathbb{K}_1 + \mathbb{L}_2 = -\mathbb{X}_1, \tag{54c}$$

$$\mathbb{J}_4 \approx -\mathbb{K}_3 = -\mathbb{X}_3. \tag{54d}$$

图 21 的下方展示了对应的结构常数。结果与表格 I 一致。例如， $\mathbb{J}_1$ 和 $\mathbb{J}_4$ 对易，因为 $[\mathbb{X}_3, \mathbb{X}_4] = [\mathbb{K}_3, \mathbb{L}_3] = 0$。类似地， $\mathbb{J}_2$ 和 $\mathbb{J}_3$ 对易，这是由于$[\mathbb{X}_1, \mathbb{X}_2] = 0$。图 21 下方其余各行所示的结果，也可以借助表格I进行类似验证。

D. 六生成元代数

我们的方法也能够找出完整的六维代数。结果如图22所示，可大致识别为

$$\mathbb{J}_1 \approx -\mathbb{L}_1, \quad \mathbb{J}_2 \approx -\mathbb{K}_2, \quad \mathbb{J}_3 \approx -\mathbb{L}_3, \tag{55a}$$

$$\mathbb{J}_4 \approx +\mathbb{K}_1, \quad \mathbb{J}_5 \approx -\mathbb{K}_3, \quad \mathbb{J}_6 \approx -\mathbb{L}_2. \tag{55b}$$

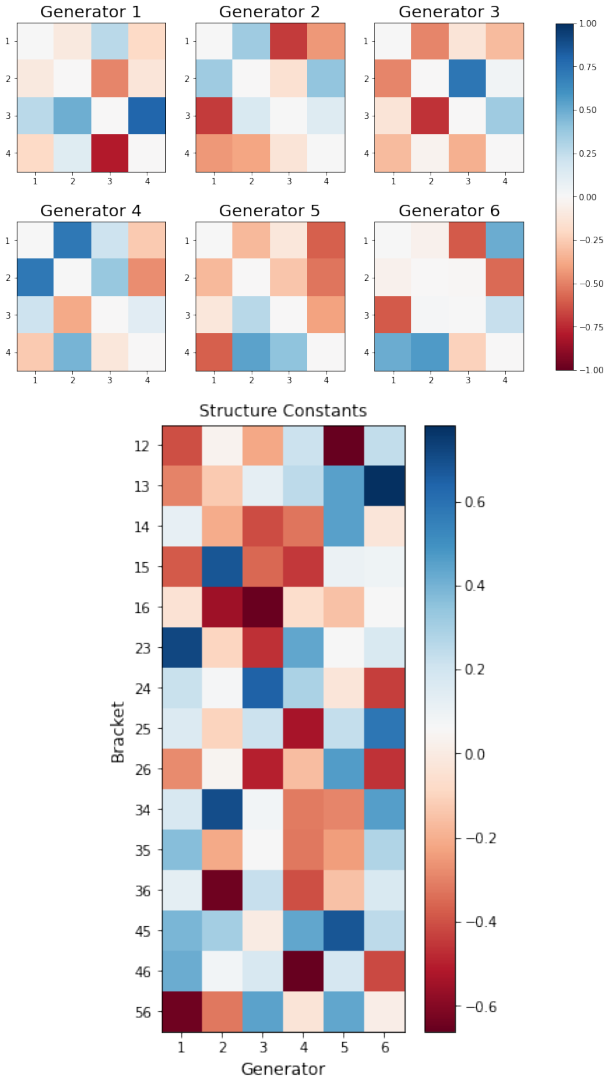


图 22. 上排：使用伪欧几里得预言机（ $n = 4$， $N_g = 6$ ）成功学习到的生成元，对应第 VIIID 节中讨论的练习（36）。下方：对应的结构常数。

相应的结构常数显示在图22的下方图版中。

IX. 二维挤压映射

再次考虑二维情形，但现在令预言机返回两个输入特征的乘积：

$$\varphi(\mathbf{x}) = x^{(1)}\, x^{(2)}. \tag{56}$$

这个预言机函数如图 23 所示，以彩色热力图呈现了 $(x^{(1)},\ x^{(2)})$ 笛卡尔平面上的预言机值。我们可以看到，具有相同预言机值的点构成双曲线。

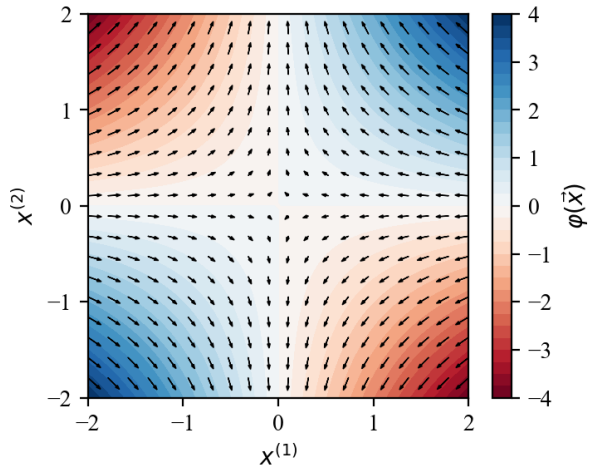


图 23. 彩色热力图显示了预言机 (56) 在（$x^{(1)}$, $x^{(2)}$）笛卡尔平面上的取值。叠加的向量场表示我们通过方法发现的对称性。

与之前一样，我们的方法找到了图 23中向量场所示的对称变换。相应的生成元为

$$\mathbb{J} = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{57}$$

我们验证了，在此示例中，该方法无法找到多于一个生成元。

这些结果完全符合预期。保持预言机 (56) 不变的对称性是挤压映射

$$\mathbb{F} = \begin{pmatrix} \frac{1}{\ell} & 0 \\ 0 & \ell \end{pmatrix}, \tag{58}$$

其中 ℓ 是一个比例因子。考虑带有 18 的无穷小变换（$\ell = 1 + \varepsilon$），立即可得生成元 (57)。

X. 不连续预言机

A. 分段线性预言机

我们的设置并不局限于前面各节所讨论的连续预言机函数。我们的方法也适用于分段定义函数，例如

$$\varphi(\mathbf{x}) = \begin{cases} -x^{(2)}, & \text{for } x^{(1)} < 0, \\ +x^{(1)}, & \text{for } x^{(1)} \geq 0. \end{cases} \tag{59}$$

得到的预言机函数以色彩图的形式展示在图24中。由于预言机现在是一个函数

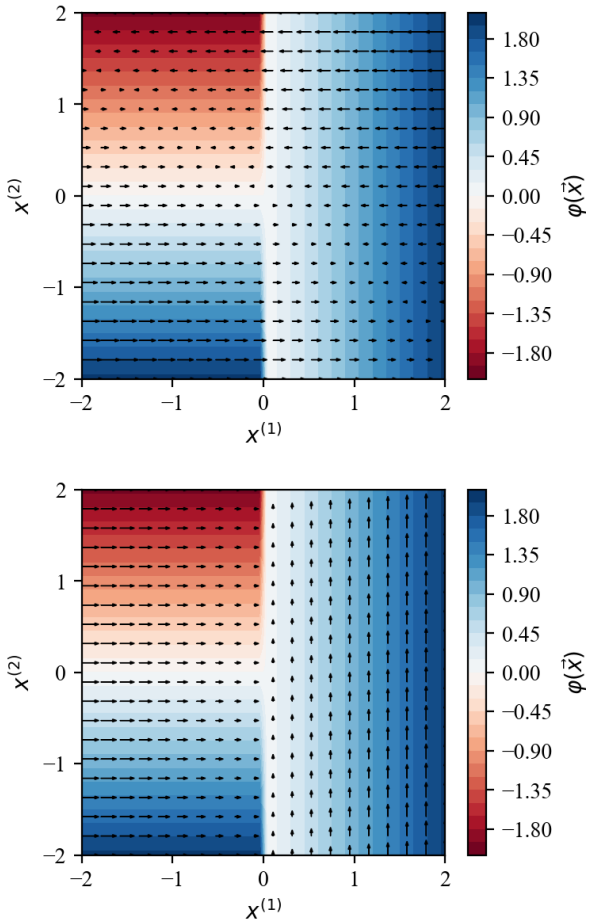


图 24. 预言机（59）生成的对称性：采用无隐藏层且无偏置的浅层方法（上图），或采用具有三个隐藏层和偏置的深层方法（下图）。

该函数不可连续微分，因此我们需要对对称变换的参数化进行适当的推广。

图中凸显了将神经网络用作通用函数逼近器的优势 24。上图未使用隐藏层，下图则采用了具有三个隐藏层的深度学习架构。随后，我们分别展示了两种情况下找到的对称变换，并将其向量场叠加在预言机色彩图上。结果差异明显，尤其是在预言机函数不连续点附近。下图中的深度学习方法正确识别出了一种变换，能够在所考虑定义域的所有位置保持预言机值不变。相比之下，上图中的浅层方法无法在不连续点附近调整变换，导致边界附近出现这样的区域$x^{(1)} = 0$：箭头穿过等势线，违反了守恒定律。

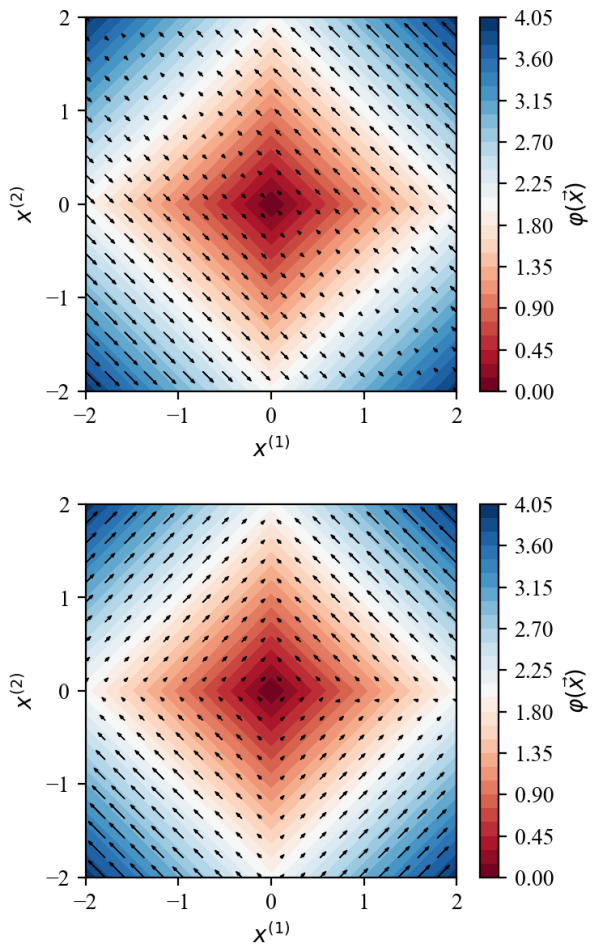


图 25. L1 预言机（60）生成的对称性：采用不含隐藏层和偏置的浅层方法（上图），或采用含三个隐藏层和偏置的深层方法（下图）。

B. 曼哈顿距离预言机

在本小节中，我们再来看一个二维预言机示例（$n = 2$）。该预言机虽然连续，但并非连续可微：

$$\varphi(\mathbf{x}) = |x^{(1)}| + |x^{(2)}|. \tag{60}$$

我们的方法所得结果如图25所示，与图24完全类似。上图中我们未使用隐藏层，下图中则采用了具有三个隐藏层和偏置的深度学习架构。我们观察到，深度学习方法再次能够正确处理不连续性，生成的变换始终沿着、但从不跨越预言机函数值相等的等值线。

XI. 结论

本文研究了数据科学中一个在许多领域都常见的基本问题：带标签数据集的对称性是什么，以及如何识别其群结构？为此，我们设计了一种深度学习方法，利用全连接神经网络对一般的对称变换及其生成元进行建模。随后，我们构造了合适的损失函数，以确保所施加的变换是对称变换，并且相应的生成元集合构成一个闭合的（子）代数。我们方法的一个重要优势是，无需预先了解数据中可能存在何种对称性；也就是说，我们不是从预定义的可能性列表中检验对称性，而是直接学习对称性。

我们的方法非常通用，普遍适用于各种情况。分析的核心是定义守恒量的预言机 $\varphi(\mathbf{x})$。正如本文通过若干示例所展示的，预言机可以完全一般化。例如，它可以是连续双线性函数，如欧几里得距离、闵可夫斯基距离和挤压映射的情形；相应的对称性已在第 V-IX节讨论。它也可以是不连续函数（第 XA节），或不光滑且不可连续微分的函数（第 XB节）。

在推导完整对称性集合的过程中，该方法还使我们能够分析对称群的完整子群结构。本文明确推导了旋转群 $SO(2)$（第 V节）、$SO(3)$（第 VI节）和 $SO(4)$（第VII节），以及洛伦兹群$SO(1,3)$（第 VIII节）的子群结构。最后，作为一个极具挑战性的示例，我们成功地将该方法应用于 维欧几里得长度保持旋转的情形。大统一中常用的相应群 $SO(10)$的 45 个生成元 [54]见图26。

本文讨论的对称性对基于模拟的推断，尤其是根据观测进行参数恢复，具有重要意义。在典型的逆问题情境中，标签 y（可能是多维的）充当观测变量，特征 $\mathbf{x}$ 充当输入参数。参数恢复任务是给定 y 推断$\mathbf{x}$。存在对称性时，这种反演并非唯一，而是会得到一整族有效的输入参数，它们都由对称性联系起来。因此，预先知道数据中存在对称性，对参数恢复算法会非常有帮助，尤其是在数据维度很高、肉眼难以察觉对称性的情况下。

请注意，同样的方法可以扩展或修改，用于解决群

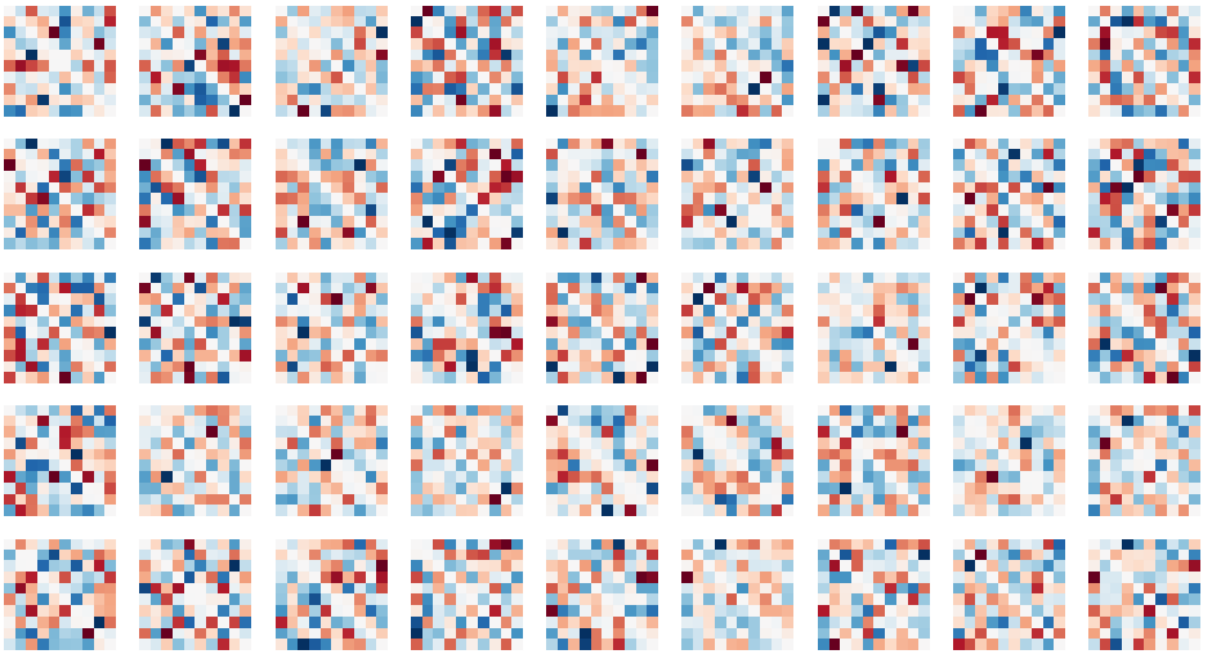


图 26. 在 SO(10) 群的情形下（$n = 10$ 维空间中的保长度旋转），我们的方法找到的 $N_g = 45$ 生成元集合。由于问题的复杂性，本例将学习率降至 0.01。

论和/或本文未讨论的超越标准模型模型构建中的其他常见问题和任务。

- 卡当子代数的推导。在我们的方法中，只需将 $a_{[\alpha\beta]\gamma} = 0$ 在（15）中设为此值，即可轻松实现这一点，这会迫使所有学习到的生成元彼此对易。

- 直积分解。我们还可以寻找其代数可表示为两个独立子代数直和的对称性——只需将两个子代数各自的闭合项纳入损失函数即可。

- 内部对称性。请注意，我们的方法可以很容易地推广到寻找内部对称性 [41], ，这在高能理论的模型构建中可能很有用。

- 生成元的典范基。由于我们的方法是基无关的，因此生成元是在一般基下获得的，在这种情况下，结构常数可能难以辨认。不过，我们可以通过在损失函数中加入一项，鼓励结构常数稀疏，从而帮助程序找到生成元的典范基。

本文中，我们重点研究了在标签判定中不存在噪声或统计涨落的理想化示例。将我们的方法用于含噪数据集（包括真实实验数据）进行测试，将很有启发性。我们将这项研究留待未来发表。

总之，我们的研究为在李群及其性质的研究中采用机器学习方法打开了大门，并进一步促进了形式数学与理论计算机科学这两个领域之间的交融。

致谢

我们感谢 A. Davis、S. Gleyzer、K. Kong、S. Mrenna、H. Prosper 和 P. Shyamsundar 进行了有益的讨论。我们感谢 P. Ramond 分享群论见解并给予启发。本研究部分得到了美国能源部项目资助，项目编号为 DE-SC0022148。

[1] David J. Gross, 《对称性在基础物理学中的作用》, 《美国国家科学院院刊》**93**, 14256–14259 (1996)。[2] Csaba Cs´aki、Salvator Lombardo 和 Ofri Telem, 《TASI 非超对称标准模型讲座》, 载于 *Pro-ceedings, Theoretical Advanced Study Institute in Elementary Particle Physics : Anticipating the Next Discoveries in Particle Physics (TASI 2016): Boulder, CO, USA, June 6-July 1, 2016*, 由 Rouven Essig 和 Ian Low 编（WSP, 2018 501 570, 1811 04279）第 – 页 [ ] arXiv:.hep-ph。

[3] Dimitri Bourilkov, 《粒子物理学中的机器学习与深度学习应用》, 《国际现代物理学杂志A辑》34, 1930019（2020）1912, arXiv: .08245 [physics.data-an].[4] Paolo Calafiura、David Rousseau 和 Kazuhiro Terao, *Artificial Intelligence for High Energy Physics* （World-Scientific, 2022）。[5] Tilman Plehn、Anja Butter、Barry Dillon 和 Claudius Krause, 《面向大型强子对撞机物理学家的现代机器学习》（2022）, arXiv:2211.01421 [高能物理—现象学]。[6] Aur´elien Dersy、Matthew D. Schwartz 和 Xiaoyuan Zhang, 《用机器学习简化多重对数》（2022）, arXiv:2206.04115 [计算机科学与机器学习]。[7] Abdulhakim Alnuqaydan、Sergei Gleyzer 和 Harrison Prosper, 《SYMBA：利用机器学习对高能物理中的平方振幅进行符号计算》（2022）, arXiv:2206.08901 [高能物理—现象学]。[8] Suyong Choi, 《使用符号回归构造对 $H \to WW^* \to l^+\nu l^-\bar{\nu}$ 标准模型希格斯玻色子质量敏感的运动学变量》, 《高能物理学杂志》08, 110（2011）, arXiv:1006.4998 [高能物理—现象学]。[9] Silviu-Marian Udrescu 和 Max Tegmark, 《AI Feynman：一种受物理学启发的符号回归方法》, 《科学进展》**6**, 2631 2020 1905 11481 eaay（）, [arXiv: . 计算物理 .[10] Guillaume Lample 和 Fran˛cois Charton, 《用于符号数学的深度学习》（ 2019 1912 01412 ）, arXiv: . [ ] cs.SC .[11]Miles Cranmer、Alvaro Sanchez-Gonzalez、Peter Battaglia、Rui Xu、Kyle Cranmer、David Spergel 和 Shirley Ho, 《利用归纳偏置从深度学习中发现符号模型》（2020）, arXiv:2006.11287 [计算机科学与机器学习]。[12] Anja Butter、Tilman Plehn、Nathalie Soybelman 和 Johann Brehmer, 《回归公式——大型强子对撞机版》（2021）, arXiv:2109.10414 [高能物理—现象学]。[13] Nikos Arechiga、Francine Chen、Yan-Ying Chen、Yanxia Zhang、Rumen Iliev、Heishiro Toyoda 和 Kent Lyons, 《端到端符号回归加速对科学实验的理解》（2021）, arXiv:2112.04023 [计算机科学与机器学习]。[14] Konstantin T. Matchev、Katia Matcheva 和 Alexan-der Roman, 《利用量纲分析和符号回归对系外行星凌日光谱进行解析建模》, 《天体物理学杂志》930, 33（2022）, arXiv:2112.11600 [astro-ph.EP]。[15]St´ephane d'Ascoli、Pierre-Alexandre Kamienny、Guil-laume Lample 和 Fran˛cois Charton, 《面向递归序列的深度符号回归》（2022）, arXiv:2201.04600 [计算机科学与机器学习]。[16] Pablo Lemos、Niall Jeffrey、Miles Cranmer、Shirley Ho 和 Peter Battaglia, 《利用机器学习重新发现轨道力学》（2022）, arXiv:2202.02306 [astro-ph.EP]。[17] Pierre-Alexandre Kamienny、St´ephane d'Ascoli、Guil-laume Lample 和 Fran˛cois Charton, 《利用Transformer模型进行端到端符号回归》（2022）, arXiv:2204.10532 [计算机科学与机器学习]。[18] Jiachen Li、Ye Yuan 和 Hong-Bin Shen, 《符号表达式Transformer：一种用于符号回归的计算机视觉方法》（2022）, arXiv:2205.11798 [cs.CV]。[19] Yoshitomo Matsubara、Naoya Chiba、Ryo Igarashi、Tat-sunori Taniai 和 Yoshitaka Ushiku, 《重新思考用于科学发现的符号回归数据集与基准测试

[20]Zhongtian Dong、Kyoungchul Kong、Konstantin T. Matchev 和 Katia Matcheva, 《机器比理论学家更聪明吗：用符号回归推导粒子运动学公式》, （2022）, arXiv:2211.08420 [hep-ph].[21] Raban Iten、Tony Metger、Henrik Wilming、L´ıdia del Rio 和 Renato Renner, 《用神经网络发现物理概念》, 《物理评论快报》124, 010508（2020）, arXiv:1807.10300 [quant-ph]。[22] Barry M. Dillon、Gregor Kasieczka、Hans Olischlager、Tilman Plehn、Peter Sorrenson 和 Lorenz Vogel, 《对称性、安全性与自监督》, 《SciPost物理学》第12卷, 188（2022）, arXiv:2108.04253 [hep-ph]。[23] Sven Krippendorf 和 Marc Syvaeri, 《用神经网络检测对称性》, （ 2020 2003 13679 ）, ar [ ] Xiv: . 计算物理 .[24]Anja Butter、Gregor Kasieczka、Tilman Plehn 和 Michael Russell, 《利用洛伦兹层进行深度学习顶夸克标记》, 《SciPost物理学》第5卷, 028（2018）, arXiv:1707.08966 [hep-ph]。[25] Gurtej Kanwar、Michael S. Albergo、Denis Boyda、Kyle Cranmer、Daniel C. Hackett、S´ebastien Racani`ere、Danilo Jimenez Rezende 和 Phiala E. Shana-han, 《用于格点规范理论的等变流采样》, 《物理评论快报》第125卷, 121601（2020）, arXiv:2003.06413 [hep-lat]。[26] Alexander Bogatskiy、Brandon Anderson、Jan T. Of-fermann、Marwah Roussi、David W. Miller 和 Risi Kondor, 《面向粒子物理学的洛伦兹群等变神经网络》, （2020）, arXiv:2006.04780 [hep-ph]。[27]Shiqi Gong、Qi Meng、Jue Zhang、Huilin Qu、Congqiao Li、Sitian Qian、Weitao Du、Zhi-Ming Ma 和 Tie-Yan Liu, 《用于喷注标记的高效洛伦兹等变图神经网络》, 《高能物理学杂志》07, 030（2022）, arXiv:2201.08187 [hep-ph]。[28] Alexander Bogatskiy *et al.*, 《面向物理学的对称群等变架构》, 载于 *2022 Snowmass Summer Study* 2022 2203 06153 [ ] ( ) arXiv: . 计算机科学与机器学习 .[29]Congqiao Li、Huilin Qu、Sitian Qian、Qi Meng、Shiqi Gong、Jue Zhang、Tie-Yan Liu 和 Qiang Li, 《洛伦兹对称设计能否提升喷注物理中的网络性能？》（2022）, arXiv:2208.07814 [hep-ph]。[30]Zichun Hao、Raghav Kansal、Javier Duarte 和 Nadezda Chernyavskaya, 《洛伦兹群等变自编码器》, （2022）, arXiv:2212.07347 [hep-ex]。[31]Nate Gruver、Marc Finzi、Micah Goldblum 和 Andrew Gordon Wilson, 《用于衡量学习到的等变性的李导数》, （2022）, arXiv:2210.02984 [计算机科学与机器学习]。[32] Michael James Fenton、Alexander Shmakov、Ta-Wei Ho、Shih-Chieh Hsu、Daniel Whiteson 和 Pierre Baldi, 《利用保持对称性的注意力网络进行无排列多喷注事件重建》, 《物理评论D辑》105, 112008（2022）, arXiv:2010.09206 [hep-ex]。[33] Alexander Shmakov、Michael James Fenton、Ta-Wei Ho、Shih-Chieh Hsu、Daniel Whiteson 和 Pierre Baldi, 《SPANet：利用保持对称性的注意力实现面向粒子物理学的广义无排列集合指派》, 《SciPost物理学》**12**, 178 2022 2106 03898 ( ), ar [ ] Xiv: . hep-ex .[34] Rupert Tombs 和 Christopher G. Lester, 《一种通过自监督学习挑战数据中对称性的方法》, 《仪器科学与技术杂志》17, P08024（2022）, arXiv:2111.05442 [hep- ph]。

," (2022), arXiv:2206.10540 [计算机科学与机器学习].

[35]Christopher G. Lester 和 Rupert Tombs，《利用无监督学习检测破缺对称性，并探讨其与自然界宇称破坏搜索的关联。（此前标题为："Stressed GANs snag desserts"）》，arXiv电子印本，arXiv:2111.00616（2021），arXiv:2111.00616 [hep-ph]。[36] Mattias Birman、Benjamin Nachman、Raphael Seb-bah、Gal Sela、Ophir Turetz 和 Shikma Bressler，《基于标准模型对称性、由数据引导的新物理搜索》，Eur. Phys. J. C 82，508（2022），arXiv:2203.07529 [hep-ph]。[37] Miles Cranmer、Sam Greydanus、Stephan Hoyer、Peter Battaglia、David Spergel 和 Shirley Ho，《拉格朗日神经网络》，（2020），arXiv:2003.04630 [计算机科学与机器学习]。[38] Ziming Liu 和 Max Tegmark，《从轨迹中用机器学习发现守恒定律》，《物理评论快报》126，180604 2021 2011（），arXiv: .04698 [计算机科学与机器学习]。[39] Tailin Wu 和 Max Tegmark，《迈向用于无监督学习的人工智能物理学家》，《Physical Review E》100（2019），10.1103/physreve.100.033311，arXiv:1810.10525 [计算物理]。[40] Gabriela Barenboim、Johannes Hirn 和 Veronica Sanz，《对称性遇上人工智能》，《SciPost物理学》11，014（2021），arXiv:2103.06115 [计算机科学与机器学习]。[41] Sean Craven、Djuna Croon、Daniel Cutting 和 Rachel Houtz，《机器学习流形》，《物理评论D辑》105，096030（2022），arXiv:2112.07673 [hep-ph]。[42] Sebastian J. Wetzel、Roger G. Melko、Joseph Scott、Maysum Panju 和 Vijay Ganesh，《通过解读孪生神经网络发现对称不变量和守恒量》，《Phys. Rev. Res.》2，033499（2020），arXiv:2003.04299 [计算物理]。[43] Heng-Yu Chen、Yang-Hui He、Shailesh Lal 和 M. Zaid Zaz，《共形场论中的机器学习习作》，（2020），arXiv:2006.16114 [hep-th]。[44] Yang-Hui He，《机器学习弦景观》，Phys. Lett. B 774，564–568（2017）。

[45] Jonathan Carifio、James Halverson、Dmitri Krioukov 和 Brent D. Nelson，《弦景观中的机器学习》，《高能物理学杂志》**09**，157 2017 1707 00655（），[ ] arXiv:。hep-th。[46] Fabian Ruehle，《弦理论中的数据科学应用》，Phys. Rept. 839，1–117（2020）。[47] Krish Desai、Benjamin Nachman 和 Jesse Thaler，《利用深度学习发现对称性》，《物理评论D辑》105，096031（2022），arXiv:2112.05722 [高能物理—现象学]。[48] Heng-Yu Chen、Yang-Hui He、Shailesh Lal 和 Suva-jit Majumder，《李结构的机器学习及其在物理学中的应用》，Phys. Lett. B 817，136297（2021），arXiv:2011.00871 [hep-th]。[49] Ziming Liu 和 Max Tegmark，《机器学习中的隐藏对称性》，《物理评论快报》128，180201（2022），arXiv:2109.09721 [计算机科学与机器学习]。[50] Artem Moskalev、Anna Sepliarskaia、Ivan Sosnovik 和 Arnold Smeulders，《Liegg：研究学习到的李群生成元》，（2022），arXiv:2210.04345 [计算机科学与机器学习]。[51] Adam Paszke、Sam Gross、Francisco Massa、Adam Lerer、James Bradbury、Gregory Chanan、Trevor Killeen、Zem- ing Lin、Natalia Gimelshein、Luca Antiga、Alban Des- maison、Andreas Kopf、Edward Yang、Zachary DeVito、Martin Raison、Alykhan Tejani、Sasank Chilamkurthy、Benoit Steiner、Lu Fang、Junjie Bai 和 Soumith Chin- tala，《PyTorch：一种命令式风格、高性能的深度学习库》，载于 *Advances in Neural Information Processing Systems 32* 2019（Curran Associates, Inc.，）第8024–8035页。[52] Kurt Hornik、Maxwell Stinchcombe 和 Halbert White，《多层前馈网络是通用逼近器》，《神经网络》2，359–366（1989）。[53] J. Hladik 和 J.M. Cole，*Spinors in Physics*，《当代物理学研究生教材》（施普林格纽约，1999）。[54] Murray Gell-Mann、Pierre Ramond 和 Richard Slan- sky，《复旋量与统一理论》，Conf. Proc. C 790927，315–321（1979），arXiv:1306.4669 [高能物理—现象学]。