拓扑正则化数据嵌入

Edith Heiter[†] EDITH.HEITER@UGENT.BE
Robin Vandaele[†,§,◊] ROBIN.VNDAELE@GMAIL.COM
Tijl De Bie[†] TIJL.DEBIE@UGENT.BE
Yvan Saeys[§,◊] YVAN.SAEYS@UGENT.BE
Jefrey Lijffijt[†] JEFREY.LIJFFIJT@UGENT.BE
[†]*IDLab*，根特大学电子与信息系统系，比利时根特[§]根特大学应用数学、计算机科学与统计系，比利时根特[◊]生物医学数据挖掘与建模（*DaMBi*），*VIB* 炎症研究中心，比利时根特

arXiv:2301.03338v2 [cs.LG] 7 Nov 2023

摘要

无监督表示学习方法广泛用于深入了解高维、非结构化或结构化数据。在某些情况下，用户可能掌握有关数据的拓扑先验知识，例如已知的簇结构，或已知数据沿着树形或图结构拓扑分布。然而，目前缺少通用方法来确保此类结构在低维表示中得到凸显。这会损害低维嵌入的可解释性，并可能影响下游学习任务。为解决这一问题，我们提出拓扑正则化：一种基于代数拓扑、将拓扑先验知识融入低维嵌入的通用方法。我们提出一类拓扑损失函数，并表明，将嵌入损失与作为正则项的此类拓扑损失函数联合优化，可以得到既反映局部邻近关系、又体现所需拓扑结构的嵌入。我们还全面概述了所需的代数拓扑基础概念，并直观介绍如何针对簇、圈和分岔等多种形状设计拓扑损失函数。我们结合线性和非线性降维方法以及图嵌入方法，从计算效率、鲁棒性和通用性等方面对所提出的方法进行了实证评估。

关键词：拓扑数据分析、持久同调、表征学习、降维、图嵌入、拓扑正则化

1. 引言

数据嵌入方法常用于数据科学中，将原始输入数据转换为更适合学习和后续推断的形式。例如，主成分分析（PCA；Jolliffe，1986年）和线性判别分析（LDA；McLachlan，2005年）等线性降维方法，常用于对数据进行预处理，以供无监督或监督学习使用。扩散映射（Coifman 等，2005年）、统一流形逼近与投影（UMAP；McInnes 等，2018年）以及 *t*-随机邻域嵌入（*t*-SNE；Van der Maaten 和 Hinton，2008年）等非线性降维方法，常用于预处理或可视化高维

数据。对于关系型输入数据，DeepWalk（Perozzi 等，2014）等图嵌入方法能够将数据表示为矩阵结构，便于可视化，或作为常见机器学习算法的输入。

使用数据嵌入的主要理由之一，是它们能够提高数据中的信噪比，使数据中的有意义信息更加突出。这有助于从数据中进行稳健的自动化推断，也有助于人们在二维（2D）或三维（3D）空间中查看嵌入可视化结果时理解数据并与之交互。因此，数据嵌入对于概括数据中的重要高层结构至关重要，能够为各种应用领域和科学领域中的自动化学习与交互式学习提供助益。

遗憾的是，数据嵌入本身也容易受到其通常旨在降低的噪声影响（Vandaele 等人，2022a）。例如，设想一个包含环形拓扑这一有意义结构的高维噪声数据集。将这些数据投影到其二维主成分分析平面上时，每个维度中的噪声都可能使该平面偏离其无噪声时的最佳方向，从而随着维数增加，降低环的显著性，如图 1 所示。

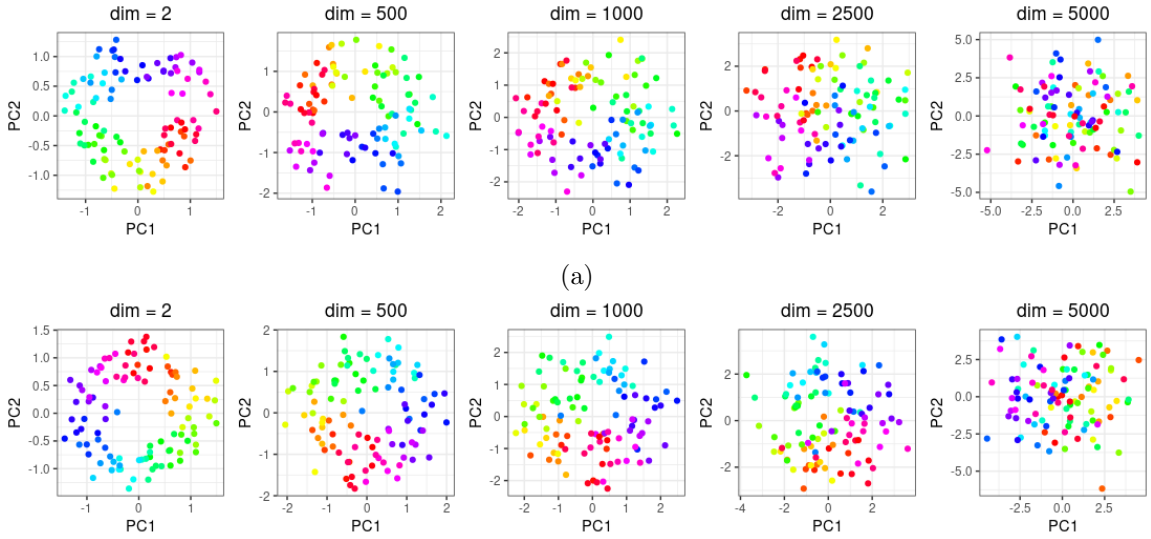


图 1：单位圆上 100 个点的数据集在高维空间中分别加入 (a) 均匀噪声和 (b) 高斯独立同分布噪声后得到的二维 PCA 投影；每个维度上的均值为$\mu = 0$，标准差为 $\sigma = 0.25$。点的颜色标示其未受噪声影响时的圆周角，每幅图的标题表示输入维数。随着维数增加，PCA 越来越难以捕捉圆形拓扑。

对于图而言，相关信息可能包含特定的社区结构。然而，社群之间的随机连边可能导致图嵌入无法有效地将它们分开，从而影响数据可视化或连续学习任务。

在其他情况下，嵌入可能如实捕捉了数据中的高阶结构，但用户可能会对更细或更粗的结构感兴趣，而这些结构无法通过全自动方式得到很好的捕捉，例如嵌入中主簇的子簇。这些情况表明，用户需要或希望使用拓扑先验知识。

这些例子说明，我们需要一种能够将拓扑信息融入数据嵌入的方法。因此，本文提出拓扑正则化这一概念，并给出一种通用方法，以实现拓扑正则化数据嵌入。从总体上看，所提出的方法是，为一个数据集（点云、图等） $\mathbb{X}$求得一个（矩阵）嵌入 $\boldsymbol{E}$，其中$\boldsymbol{E}$ 的每一行都对应数据集中相应元素（数据点、图中的顶点等）的嵌入，并使其最小化一个总损失函数

$$\mathcal{L}_{\mathrm{tot}}(\boldsymbol{E}, \mathbb{X}) \coloneqq \mathcal{L}_{\mathrm{emb}}(\boldsymbol{E}, \mathbb{X}) + \lambda_{\mathrm{top}} \mathcal{L}_{\mathrm{top}}(\boldsymbol{E}), \tag{1}$$

其中，嵌入损失函数 $\mathcal{L}_{\mathrm{emb}}$ 通常旨在保留原始数据中数据元素之间的邻近性概念，而 $\lambda_{\mathrm{top}} > 0$ 则控制拓扑正则化的强度。正如我们将要展示的，各种类型的拓扑先验知识都可以通过拓扑损失函数 $\mathcal{L}_{\mathrm{top}}$直接编码。这得益于拓扑数据分析（TDA）领域的最新进展，尤其是基于持久同调的优化方法（Br¨uel-Gabrielsson 等，2020年；Solomon 等人，2021年；Carriere 等，2021年）。

持久同调背后的理论——第2节将对此进行详细阐述——建立在代数拓扑领域的基本概念之上，而代数拓扑并非数据科学家的常用工具。这可能会使我们的工作难以被专家以外的读者理解，从而限制其在实践中的应用。因此，本文以通俗易懂且自成一体的方式概述拓扑优化的相关概念，从单纯同调的基本概念讲起。随后，我们介绍拓扑正则化，并通过示例说明如何设计拓扑损失函数。之后，我们对拓扑正则化进行广泛的计算与实验分析，研究并探讨其计算成本、鲁棒性、参数敏感性、不同正则化强度下的表现、拓扑损失函数及其应用和未来面临的挑战，同时明确讨论现有局限。

1.1 拓扑正则化示例

为了从非技术层面阐明拓扑正则化的目的及其可能产生的影响，我们首先讨论一个简单但具有说明性的例子，展示其在单细胞组学分析中的应用。（第 4.2.1 节将对此进行深入讨论。）

单细胞组学包含在细胞层面采集的各种类型的数据，例如转录组学、蛋白质组学和表观基因组学。研究数据背后的拓扑模型，有助于更好地理解生物细胞的动态过程及其中涉及的调控相互作用。此类动态过程可以通过轨迹推断方法（也称为伪时间分析）进行建模；这些方法根据细胞表达模式的相似性，沿一条轨迹对细胞进行排序（Saelens 等人，2019年）。

例如，细胞周期是一种广为人知的生物分化模型，发生于细胞生长和分裂的过程中。细胞周期包含不同阶段，分别是

生长（G1）、DNA 合成（S）、生长并为有丝分裂做准备（G2），以及有丝分裂（M）。后两个阶段通常合称为 G2M 阶段。通过研究参与分化模型的细胞的表达数据，可以确定细胞周期特定阶段中及阶段之间涉及的基因（Liu 等人，2017年）。伪时间分析通过为每个细胞分配其在分化过程中所处的时间，从而确定所有细胞在细胞周期模型中的相对位置。

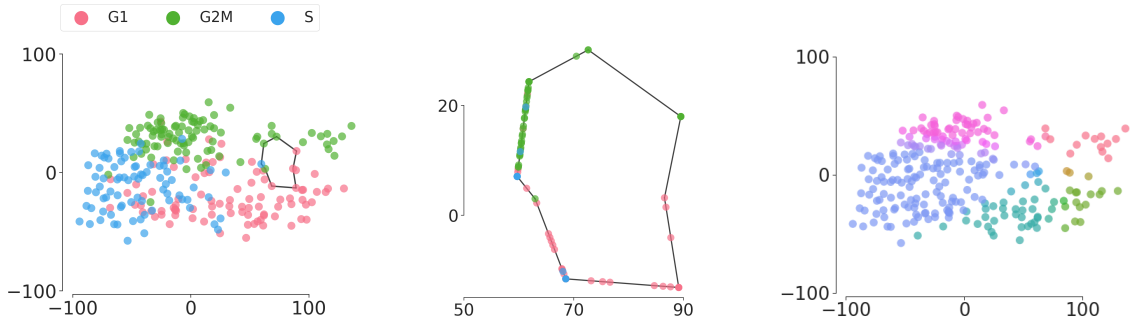
因此，单细胞周期数据的分析构成了一个可利用先验拓扑信息的问题。由于高维表达数据的信噪比通常较低（Libralon 等人，2009年；Zhang 等人，2021年），在进行自动伪时间推断之前，通常会先用降维方法对这些数据进行预处理（Can- noodt 等人，2016年；Saelens 等人，2019年）。拓扑正则化提供了一种工具，可在嵌入过程中增强预期的拓扑信号，从而促进依赖该信号的自动推断。

为说明这一点，我们对（Buettner 等人，2015年）中呈现的真实细胞周期数据应用了一种基于持久同调的自动（细胞）周期和伪时间推断方法（详细信息见第 4.2.1 节），分别在不进行（图 2）和进行（图 3）我们提出的数据主成分分析嵌入拓扑正则化的情况下进行。为此，我们设计了拓扑损失函数，使嵌入偏向圆形模型。与图 1 中的嵌入类似，普通 PCA 嵌入难以捕捉圆形拓扑（图 2 a）。因此，自动捕捉到的环并不可靠，且主要是线性地区分 G1 和 G2M 细胞。将细胞投影到该环的边上——这是从离散拓扑表示中导出连续伪时间的中间步骤，正如 Saelens 等人（2019年）先前所述——会使大多数细胞落在同一条边上（图 2 b）。投影到这条边上的细胞，其伪时间大多是连续的；而其他所有细胞的伪时间则更为离散（图 2 c）。通过将先验拓扑知识纳入主成分分析嵌入，自动捕捉到的拓扑正则化嵌入中的环能够更好地刻画细胞周期模型中 G1、G2M 和 S 期之间的转换（图 3 a）。自动伪时间推断过程也体现了细胞阶段之间更为连续的转换（图 3 b 和 3 c）。

1.2 贡献

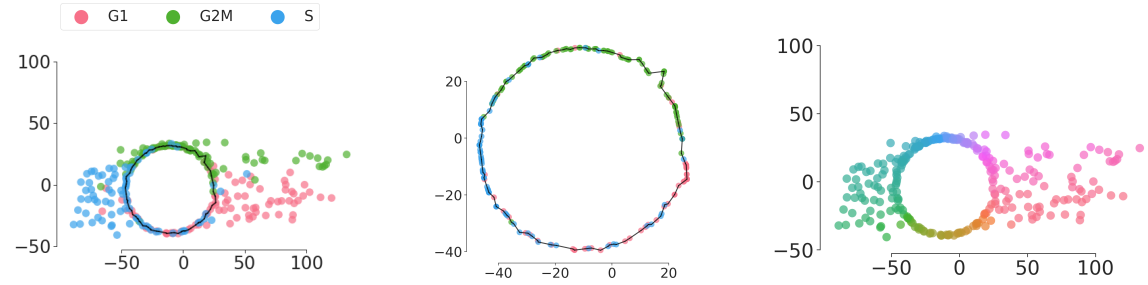
我们提出了拓扑正则化的概念，以有效地引导数据嵌入趋向于先验拓扑信息。

- 在第2节中，我们从单纯同调的基本概念出发，全面、系统且直观地介绍点云（或嵌入）的拓扑优化。
- 在第3节中，我们提出了一类拓扑损失函数，并说明如何针对多种拓扑先验设计这些函数，包括簇、环、分岔等，以及它们的任意组合。我们的方案还包含一种新颖的采样方法，以确保拓扑损失函数稳健且有意义，同时降低计算成本。
- 在第4节中，我们对拓扑正则化进行了广泛的计算与实验分析。我们研究了其计算成本、鲁棒性、参数



(a) 通过持久同调得到的最显著环的表示。(b) 将嵌入数据正交投影到环表示上。(c) 根据 (b) 中的投影推断出的伪时间，以连续色标量化。

图 2：从数据的 PCA 嵌入出发，通过持久同调对真实细胞周期数据进行自动伪时间推断。



(a) 通过持久同调得到的最显著环的表示。(b) 将嵌入数据正交投影到环表示上。(c) 根据 (b) 中的投影推断出的伪时间，以连续色标量化。

图 3：从数据的拓扑正则化PCA嵌入出发，通过持久同调对真实细胞周期数据进行自动伪时间推断。

敏感性、不同正则化强度下的表现、拓扑损失函数或拓扑先验，以及其应用和未来挑战。在此过程中，我们还讨论并通过实证分析了所提出方法目前的局限性。

- 我们在第5节总结本研究，并在第6节讨论拓扑正则化的潜在应用和未解决的挑战。

本文是对早期会议论文（Vandaele 等，2022年b）的重要扩展，补充了大量背景内容、改进了方法，并增加了实验和分析。

1.3 相关工作

本文将讨论主要的相关研究，以及它们与当前论文的关系。

1.3.1 Foundations of Topological Optimization

最直接而言，拓扑正则化建立在一系列近期论文的基础上（Br¨uel- Gabrielsson 等人，2020年；Solomon 等人，2021年；Carriere 等，2021），这些论文表明，在各种情形下，拓扑优化——即对式（1）中的 $\mathcal{L}_{\text{top}}$ 进行优化——是可行的，并且为此奠定了数学基础。拓扑优化本质上颇具难度，因为通过持久同调从点云中量化拓扑信息具有组合性质；输入数据到其持久图的映射高度非线性，且没有显式的解析表示。为应对这一问题，拓扑优化采用 Clarke 次导数（Clarke，1990年）；其适用于持久同调的依据来自 o-极小几何方面的论证（van den Dries，1998年；Carriere 等，2021）。得益于这些近期研究（Br¨uel-Gabrielsson 等人，2020年；Carriere 等，2021），研究人员已为 PyTorch 和 TensorFlow 等软件库开发出强大的拓扑优化工具。借助这些工具，即使不深入了解持久同调的数学基础，也能加以应用。

1.3.2 Incorporating Topological Optimization into Embeddings

拓扑自编码器（Moor 等人，2020年）、DIPOLE（分布式持久性优化局部嵌入，Wagner 等人，2021年）和交错维度约简（Nelson 和 Luo，2022年）已经将拓扑优化与数据嵌入过程相结合。与我们的方法相比，主要区别在于，用于优化的拓扑信息是从原始高维数据中获取的，而不是像本文中那样作为先验传入。虽然这种方法可能有用，但获取此类拓扑信息很大程度上依赖观测之间的距离，而这些距离在高维空间中往往毫无意义且不稳定（Vandaele 等人，2022a；Aggarwal 等人，2001年）。此外，某些构造，例如从弱 *Alpha* 过滤 中获得的*Delaunay*三角剖分——我们将在拓扑正则化中大量使用，并在第2节中详细讨论——从高维数据中计算的成本很高（Cignoni 等人，1998年），因此最好从低维嵌入中计算。

1.3.3 Incorporating Topological Information into Embeddings

除了拓扑正则化之外，也有其他方法将拓扑信息融入数据嵌入。例如，深度嵌入聚类（Xie 等人，2016年）利用深度神经网络同时学习特征表示和簇分配。Bai 等人（2015年）也研究了欧几里得数据在球面上的约束嵌入；此外，自组织星形结构被用于根据具有指定分支数的星形拓扑嵌入数据（Cˆome 等人，2010年）。这些方法的共同点是，它们都需要针对某一种特定输入数据和某一种特定拓扑模型进行大量开发。相比之下，将拓扑优化融入表征学习，提供了一种统一而灵活的方法，用于结合数据嵌入方法与拓扑先验；只要输出为点云，该方法就能很好地泛化到任意输入结构。

1.3.4 Other Forms of Topological Regularization in Machine Learning

通过改变对象的拓扑性质来改进连续学习和推理，近年来已应用于图像分割（Vandaele 等人，2020a年）和监督式机器学习（Chen 等人，2019年；Br¨uel-Gabrielsson 等人，2020年）。例如，基本的图像平滑（也可以通过拓扑优化实现；Br¨uel-Gabrielsson 等人，2020年）会降低伪拓扑特征的显著性，从而改进皮肤病变的后续无监督分割（Vandaele 等人，2020a年）。在监督学习中，拓扑正则化已用于防止分类边界中出现伪拓扑分量（Chen 等人，2019年），或减少机器学习模型权重中的局部极值（Br¨uel-Gabrielsson 等人，2020年），以降低过拟合。与通过（1）进行的拓扑正则化类似，这些方法将损失函数分解为常规训练损失函数（交叉熵损失、二次损失、合页损失等）和一个在所得模型上计算的拓扑惩罚项。

然而，在数据嵌入的拓扑正则化过程中，我们关注的是增强拓扑信号，而非降低虚假拓扑分量的显著性；拓扑损失函数 $\mathcal{L}_{\text{top}}$ 正是为此设计的。由于$\mathcal{L}_{\text{top}}$ 必须针对当前具体的领域应用进行设计，因此这种方法并不具有通用性。然而，在本文中我们表明，拓扑先验可以产生类似于机器学习中常规正则化的效果，即所学习的嵌入对噪声的敏感性降低。

2. 从单纯同调到拓扑优化

本节旨在以说明性的方式介绍拓扑优化如何作用于点云——也就是嵌入。第2.1节将介绍单纯复形和过滤。这些是基本的组合结构，持久同调正是基于它们对拓扑信息进行量化的，具体内容将在第2.4节中说明。第2.2节将介绍两种常用的单纯复形和过滤类型，即 *Vietoris-Rips* 复形及其过滤，以及弱 *Alpha* 复形及其过滤；后者的规模受到限制，因此更适合执行低维拓扑优化（例如在嵌入空间中进行优化）。随后，在第2.5节中，我们将说明如何利用定义在点云（嵌入）的持久图上的拓扑损失函数来进行拓扑优化。最后，在第2.6节中，我们将讨论与持久同调和拓扑优化相关的计算成本。

尽管本节中的许多定义可以推广到更一般的度量空间，为简洁起见，我们将范围限定为点云，即某个欧几里得空间 $\mathbb{R}^k$ 中由互不相同的点组成的有限集。为了阐释持久同调和拓扑优化的概念，我们将以图 4a 所示、形似皮卡丘的二维点云数据集 $\boldsymbol{X}$ 为工作示例。此外，我们还将使用一个更简单的示例来说明（持久）单纯同调计算（图5）。在本节以更广泛的背景介绍相关基础知识时，我们使用符号‘$\boldsymbol{X}$’表示点云数据集。在拓扑正则化的语境下，$\boldsymbol{X}$ 是嵌入 $\boldsymbol{E}$。

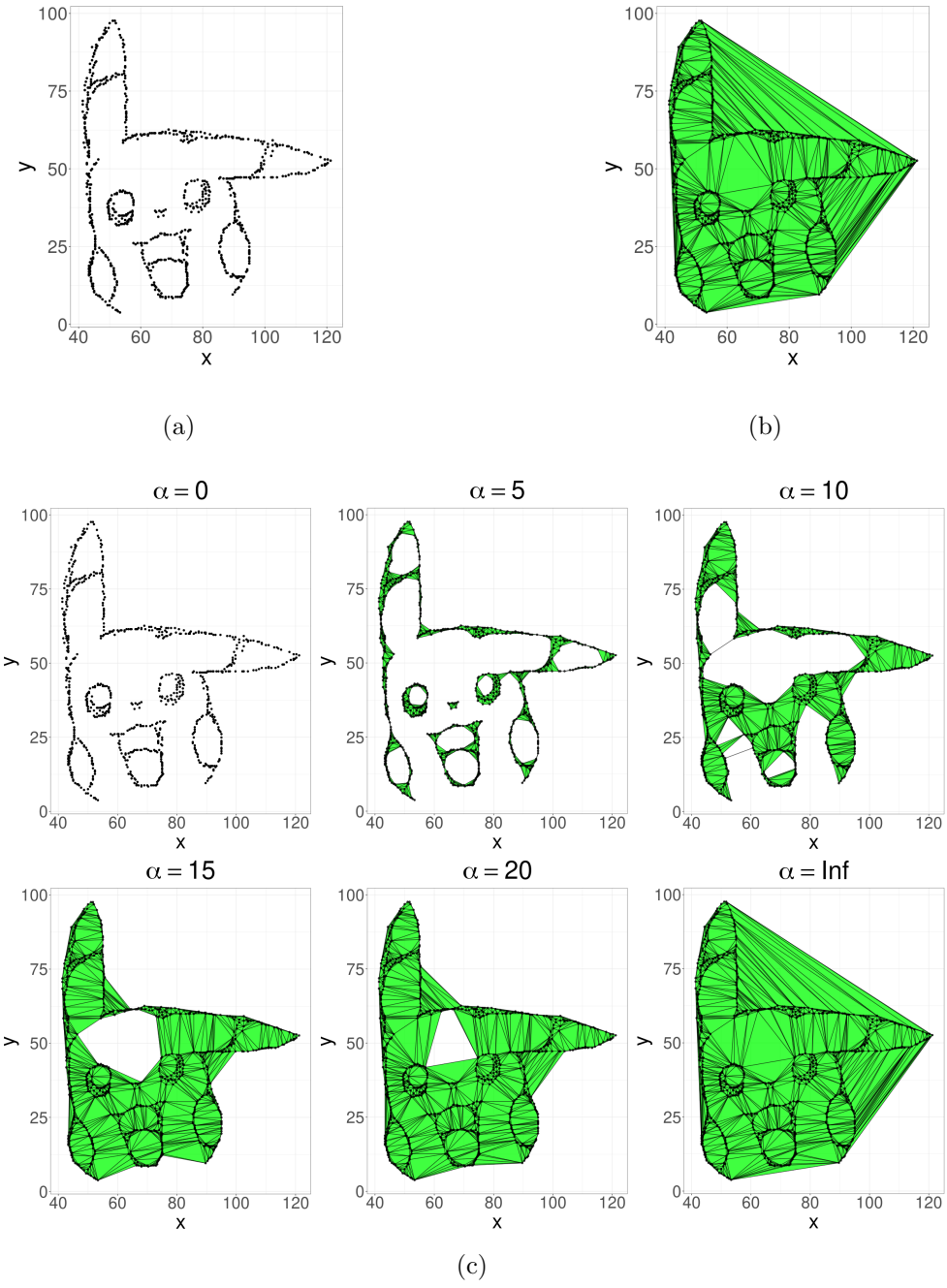


图4：(a) 一组形似皮卡丘的点云数据集 $\boldsymbol{X}$ 。(b) 由 $\boldsymbol{X}$构建的 Delaunay 三角剖分。(c) $\boldsymbol{X}$ 的弱 Alpha 过滤中的各种单纯复形——这是由 Delaunay 三角剖分构成、按包含关系排序的一系列子复形。时间参数 $\alpha = \infty$ 用于非正式地表示任何使弱 Alpha 复形等于 Delaunay 三角剖分的时间参数。

2.1 单纯复形与过滤

如上所述，单纯复形和过滤是基本的组合结构，通过持久同调（第2.4节）可以量化其中的拓扑信息。一般而言，单纯复形可视为图的推广：除了节点或顶点（0-单纯形）和边（1-单纯形）外，还可以包含三角形（2-单纯形）、四面体（3-单纯形）等。

定义 1（抽象单纯复形）。抽象单纯复形 $\mathcal{K}$ 是一个对取子集运算封闭的集合族。更具体地说，它满足以下两个定义性质：

1. $\mathcal{K}$ 是给定集合 $\mathbb{S}$的有限子集构成的集合；
2. 若 $\Delta' \subseteq \Delta \in \mathcal{K}$，则 $\Delta' \in \mathcal{K}$。

$\mathcal{K}$ 中的每个元素 Δ 都称为一个面，其维数等于 $|\Delta| - 1$。$\mathcal{K}$ 的维数是 $\mathcal{K}$中所有面的最大维数。

宽泛地说，单纯复形是带有相应几何结构的抽象单纯复形。单纯复形由单纯形组成，它将三角形和四面体的概念推广到任意维度。

定义 2（单形）。单形 Δ 是 $k+1$ 个仿射无关点 $\boldsymbol{v}_0$，…， $\boldsymbol{v}_k \in \mathbb{R}^k$的凸包，在这种情况下， Δ 也称为 k-单形。$\{\boldsymbol{v}_0, \ldots, \boldsymbol{v}_k\}$ 任意非空子集的凸包称为 Δ的面。因此，面本身也是单纯形。

在图论中，顶点和线通常以特定方式连接在一起，形成图在平面中的绘图，亦即其可视化。从形式上说，这就是图的几何实现（Vandaele 等人，2021b）。类似地，单纯形以特定方式连接在一起，形成单纯复形。

定义 3（单纯复形）。单纯复形 $\mathcal{K}$ 是满足以下条件的一组单纯形

1. $\mathcal{K}$ 中的每个面也包含在 $\mathcal{K}$中；
2. 两个面 Δ、$\Delta' \in \mathcal{K}$ 的非空交集既是 Δ 的面，也是 Δ'的面。

因此，定义 3 中的第一条定义性质类似于定义 1 中抽象单纯复形的第二条定义性质。定义 3 中的第二条性质类似于图 G的平面图绘制：表示两条边 e， e' 的两条线不允许相交于一点，除非该点表示 G 的一个顶点，且该顶点同时与 e 和 e'关联。

就本文而言，混淆与“抽象单纯复形”和“单纯复形”相关的术语与记号并无大碍，也就是说，可以将单纯复形与其离散对应物等同起来。最需要注意的是，（持久）同调（第2.4节）关注的是单纯复形“连续版本”（几何实现）的拓扑性质，而这些性质是通过其“离散”（抽象）对应物

计算得到的。可以将其与图的连通性进行比较：图的连通性（作为图）可以通过其纯粹组合性的有限表示来计算，但这决定了其任意几何实现（作为包含不可数多个点的拓扑空间）的连通性。

图4中的每个独立图示都展示了一个单纯复形。此外，图4c中的每个单纯复形都是下一个单纯复形的子复形，也就是说，其中的所有单纯形（在本例中为顶点、边和三角形）也都是下一个单纯复形的单纯形。这就是过滤的正式定义特征。

定义 4 （过滤）。过滤 $\mathcal{F}$ 是一个单纯复形序列 s

$$\mathcal{F} = (\mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \ldots \subseteq \mathcal{K}_n = \mathcal{K}).$$

虽然过滤不必采用定义 4（Oudot，2015年）中所示的有限序列形式，但在实践中，它们始终如此。

2.2 Vietoris-Rips 过滤与 Alpha 过滤

在实际应用中，点云数据上最常用的过滤之一是*Vietoris-Rips* 过滤，其正式定义如下。

定义 5 （*Vietoris-Rips* 复形及过滤）。设 $\boldsymbol{X}$ 为一个点云，且 $\epsilon \in \mathbb{R}_{\geq 0}$。 $\boldsymbol{X}$ 在时刻 ϵ 的 Vietoris-Rips 复形 定义为

$$\mathrm{VR}_\epsilon(\boldsymbol{X}) \coloneqq \{\Delta \subseteq \boldsymbol{X} : \mathrm{diam}(\Delta) \leq \epsilon\},$$

其中 $\mathrm{diam}(\Delta) \coloneqq \max_{\boldsymbol{x},\boldsymbol{y} \in \Delta} \|x - y\|$ 是 Δ的直径。Vietoris-Rips 过滤 是单纯复形的参数化序列$(\mathrm{VR}_\epsilon)^{\epsilon \in \mathbb{R}_{\geq 0}}$。

因此，时刻 ϵ 的 Vietoris-Rips 复形包含一个单形，对应于每个直径至多为 ϵ的点子集；Vietoris-Rips 过滤则通过增大参数 ϵ，使所得单纯复形逐渐变化。在实际应用中，当 $\boldsymbol{X} \subseteq \mathbb{R}^k$时，维数大于 $k+1$ 的单形会从过滤中排除，因为——正如下文将讨论的——无法通过这些单形来表征拓扑信息。即便如此，Vietoris-Rips 过滤仍可能难以放入（本地）内存。因此，尤其对于低维点云，使用弱 *Alpha* 复形和过滤可能更适合实际应用。图 4b 和 4c 中的单纯复形就是这类例子。形式上，弱 Alpha 复形是某个特定单纯复形的子复形，该单纯复形称为*Delaunay*三角剖分 （Delaunay 等人，1934年）。

定义 6 （*Delaunay*三角剖分）。设 $\boldsymbol{X}$ 为一个点云。 $\boldsymbol{X}$的三角剖分 $\mathcal{K}$ 是一个覆盖 $\boldsymbol{X}$的凸包的单纯复形。 $\boldsymbol{X}$的Delaunay三角剖分 是 $\boldsymbol{X}$ 的一个三角剖分 $\mathcal{K}$，满足 $\boldsymbol{X}$ 中的任何点都不在 $\mathcal{K}$中任何k-单形的外接超球面内。

直观而言，Delaunay三角剖分会使剖分中所有三角形内角的最小角尽可能大，也就是说，它们倾向于避免出现“瘦长三角形”。已知

若 $\mathbb{R}^k$ 中的点云 $\boldsymbol{X}$ 处于一般位置，则其 Delaunay三角剖分唯一（Delaunay 等人，1934年）；也就是说，$\boldsymbol{X}$的仿射包（即包含$\boldsymbol{X}$的最小仿射空间）的维数为 k，并且 $\boldsymbol{X}$ 中没有 $k+2$ 个点位于某个球的边界上，且该球的内部与 $\boldsymbol{X}$不相交。当处理来源于连续分布的有限欧几里得数据时，这一条件通常以概率 1 成立。图 4b 展示了图 4a 中数据 $\boldsymbol{X}$ 的 Delaunay三角剖分。

直观地说，弱 Alpha 过滤现在可以看作定义在 Delaunay 三角剖分上的 Vietoris-Rips 过滤。

定义 7（弱 *Alpha* 复形与过滤）。设 $\boldsymbol{X}$ 是一个点云，其 *Delaunay* 三角剖分为 $\mathcal{K}$ 和 $\alpha \in \mathbb{R}_{\geq 0}$。$\boldsymbol{X}$ 在时刻 α 的弱 Alpha 复形 定义为：

$$\mathrm{WA}_\alpha(\boldsymbol{X}) \coloneqq \{\Delta \in \mathcal{K} : \mathrm{diam}(\Delta) \leq \alpha\},$$

弱 *Al* pha 过滤 是单纯复形 $(\mathrm{WA}_\alpha)^{\alpha \in \mathbb{R}_{\geq 0}}$ 的参数化序列 .

请注意，尽管定义 5 和 7 中的过滤是以不可数集合为参数的，但在实际应用中，它们总等价于有限序列形式的过滤，例如定义 4 中的形式。这是因为对于点云 $\boldsymbol{X}$，所定义的过滤只会在有限个时间值处发生变化。

2.3 单纯同调

在代数拓扑和计算拓扑中，同调 关注研究并量化与（抽象）单纯复形相关的代数对象。它通过抽象单纯复形的边界映射 来定义。为简化本节中的定义，我们假设抽象单纯复形的所有顶点都具有固定的顺序，例如，该顺序由定义复形的数据点的排列所诱导，并且所有计算都在某个域上进行（例如模 2）。

定义 8（边界映射）。设 $\mathcal{K}$ 是一个抽象单纯复形，其顶$(v_0, \ldots, v_m)$ 点按顺序排列，且 $\mathbb{F}$ 是一个域。$\mathcal{K}$ 中顶点的排列自然地为每个单形$\Delta = \{v_{i_0}, \ldots, v_{i_k}\} \in \mathcal{K}$确定一个有序元组$(v_{i_0}, \ldots, v_{i_k})$，$i_0 < \ldots < i_k$，我们称 Δ 是定向的。一个单纯 k- 链 是一个有限形式和

$$\lambda_1 \Delta_1 + \ldots + \lambda_N \Delta_N,$$

其中，$N \in \mathbb{N}$、λ_i 是 $\mathbb{F}$中的系数，而 Δ_i 是 $\mathcal{K}$ 中的定向 k-单形（视为基“向量”），其中 $1 \leq i \leq N$。所有单纯形 k-链构成一个 $\mathbb{F}$向量空间 $\mathcal{K}_k$，并配备向量空间的常规运算。边界映射$\partial_k : \Delta_k \to \Delta_{k-1}$ 在 $\mathbb{F}$向量空间 Δ_k 和 Δ_{k-1} 之间定义如下：对于每个单形 $\Delta = (v_{i_0}, \ldots, v_{i_k})$，

$$\partial_k(\Delta) \coloneqq \sum_{j=0}^{k} (-1)^j (v_{i_0}, \ldots, v_{i_{j-1}}, v_{i_{j+1}} \ldots, v_{i_k}). \tag{2}$$

按照约定，$\partial_0 \equiv 0$，且 $\Delta_{-1} = \{0\}$。边界映射线性扩展至 $\mathcal{K}_k$ 中所有的 k-链。若 C 是 Δ_k 中的一个 k-链，我们也将 $\partial_k(C)$ 称为 C 的边界。

可以证明（Hatcher，2002），$\partial_k \circ \partial_{k+1} \equiv 0$。因此，像 Im($\partial_{k+1}$) 是核 Ker($\partial_k$) 的一个向量子空间。由此，我们可以定义抽象单纯复形 $\mathcal{K}$ 的同调模。“模”这一术语指的是，这些结构可以在更一般的代数结构（即环）上定义（Hatcher，2002）。不过，对于域而言，它们只是向量空间。

定义9（同调模）。设 $\mathcal{K}$ 是一个抽象单纯复形，$\mathbb{F}$ 是一个域，其关联的边界映射为 ∂_k。核 $\ker(\partial_k)$ 称为 k阶循环模，像 $\mathrm{Im}(\partial_k)$ 称为 k阶边界模。$\mathcal{K}$ 的 k阶同调模 是一个$\mathbb{F}$-向量空间

$$H_k \coloneqq \mathrm{Ker}(\partial_k) \,/\, \mathrm{Im}(\partial_{k+1}).$$

关系 $\mathrm{Im}\,\partial_{k+1} \subseteq \ker\,\partial_k$ 具有几何解释。这意味着，每个（$k+1$）阶边界——即一个 k-链，它是某个（$k+1$）-链在 ∂_{k+1}作用下的像——在 ∂_k作用下边界为零，因此是一个 k阶环。然而，反向包含一般并不成立。因此，可能存在一些 k阶环，它们不是任何（$k+1$）-链的边界。这些环对应于单纯复形中的“孔洞”，如图5所示。

第 k阶贝蒂数 β_k 量化了逆包含关系“Ker(∂_k) $\subseteq$Im(∂_{k+1})”未成立的程度。

定义10（贝蒂数）。设 $\mathcal{K}$ 为一个抽象单纯复形，$\mathbb{F}$ 为一个域，并具有相应的同调模 H_k。$\mathcal{K}$的第 k阶贝蒂数记为 β_k，其定义为H_k的秩，即其生成元的数量。

第 k阶贝蒂数刻画了抽象单纯复形（的几何实现）中 k维孔洞的数量。例如，图5（左）中的单纯复形具有一个0维孔洞（一个连通分支）、一个一维孔洞（一个环），且没有高维孔洞。然而，贝蒂数可能因所选域的不同而变化。例如，对于<style id='4'>克莱因瓶的三角剖分 $\mathcal{K}$，可以发现 $\beta_1(\mathcal{K})$ =在 $\mathbb{F} = \mathbb{F}_2$上为2，而对于任意素数 $p > 2$ (Maria，2014)，$\beta_1(\mathcal{K}) = 1$ 在 $\mathbb{F} = \mathbb{F}_p$ 上为 $= 1$。这与克莱因瓶并非无挠这一事实有关。不过，对于低维度中更直观的形状——它们可能在拓扑正则化过程中尤其受到关注——这通常很少成为问题。除非另有说明，本文中的所有（持久）同调计算均在 $\mathbb{F} = \mathbb{F}_2$上进行。

计算单纯同调对于给定的单纯复形 $\mathcal{K}$，我们可以通过对 $\mathcal{K}$的边界矩阵进行线性运算来计算贝蒂数。这些矩阵是边界算子 ∂_k的矩阵表示。对于 $\mathbb{F} = \mathbb{F}_2$，它们就是稀疏关联矩阵 B_k：若第 i个（$k-1$）维单形 Δ_i是第 j个 k维单形 Δ_j的一个面，则 $B_{k_{ij}} = 1$；否则为 $B_{k_{ij}} = 0$。图6展示了图5中左侧单纯复形的边界矩阵 B_k 汇集而成的“完整”边界矩阵 B，该矩阵为单纯复形中的每个单形都设置了一行和一列。随后，根据秩-零化度定理，可由矩阵 B_k 的秩推导出同调模的秩，也就是贝蒂数。

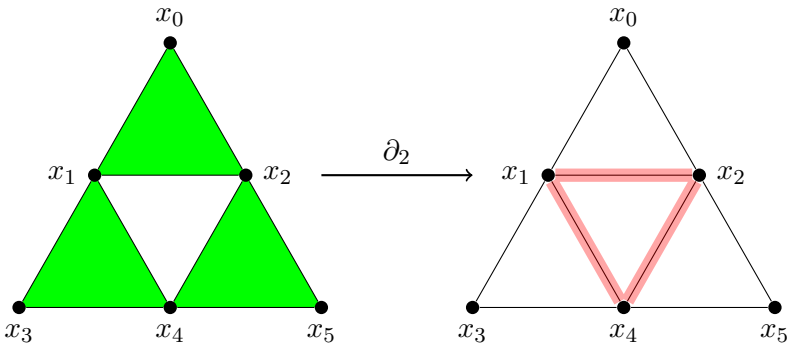


图5：边界算子的几何解释。 ∂_{k+1} 将一个（$k+1$）-链映射为一个 k-环。某些 k-环（例如红色标记的那个）并非任何（$k+1$）-链的边界。它们对应于单纯复形中的“孔洞”。

	x_0	x_1	x_2	x_3	x_4	x_5	x_{01}	x_{02}	x_{12}	x_{13}	x_{14}	x_{24}	x_{25}	x_{34}	x_{45}	x_{012}	x_{134}	x_{245}
x_0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
x_1	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	0
x_2	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0	0
x_3	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
x_4	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	0
x_5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0
x_{01}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
x_{02}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
x_{12}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
x_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
x_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
x_{24}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x_{25}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x_{34}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
x_{45}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x_{012}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x_{134}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x_{245}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$\partial_1 \sim B_1$

$\partial_2 \sim B_2$

图6：图5中左侧单纯复形在 $\mathbb{F}=\mathbb{F}_2$上的完整边界矩阵 B，其中包含所有边界算子 ∂_k的矩阵表示。标签 x_i、 x_{ij}和x_{ijk}分别是单纯形 $\{x_i\}$、 $\{x_i, x_j\}$和 $\{x_i, x_j, x_k\}$的简称。

例如，对于当前的例子（图6），在 $\mathbb{F} = \mathbb{F}_2$ 上，我们有

$$\begin{cases} \operatorname{rank}(B_0) = \operatorname{rank}(\operatorname{Im}(\partial_0)) = 0 \text{ (Definition 8)}, \\ \operatorname{rank}(B_1) = \operatorname{rank}(\operatorname{Im}(\partial_1)) = 5, \\ \operatorname{rank}(B_2) = \operatorname{rank}(\operatorname{Im}(\partial_2)) = 3. \end{cases}$$

请注意，这些秩是在 SageMath 中计算的（SageMath 开发者，2022年）。根据秩-零化度定理，我们得到

$$\begin{cases} \operatorname{rank}(\operatorname{Im}(\partial_0)) + \operatorname{rank}(\operatorname{Ker}(\partial_0)) = 6 \iff \operatorname{rank}(\operatorname{Ker}(\partial_0)) = 6, \\ \operatorname{rank}(\operatorname{Im}(\partial_1)) + \operatorname{rank}(\operatorname{Ker}(\partial_1)) = 9 \iff \operatorname{rank}(\operatorname{Ker}(\partial_1)) = 4. \end{cases}$$

因此，我们发现

$$\begin{cases} \beta_0 = \operatorname{rank}(H_0) = \operatorname{rank}(\operatorname{Ker}(\partial_0)) - \operatorname{rank}(\operatorname{Im}(\partial_1)) = 6 - 5 = 1, \\ \beta_1 = \operatorname{rank}(H_1) = \operatorname{rank}(\operatorname{Ker}(\partial_1)) - \operatorname{rank}(\operatorname{Im}(\partial_2)) = 4 - 3 = 1. \end{cases}$$

因此，我们发现图5左侧的单纯复形有 $\beta_0 = 1$ 个连通分支和 $\beta_1 = 1$ 个环，这与我们从图中观察到的情况一致。

2.4 持久同调与持久图

给定一个由时间参数 t 参数化的过滤，例如弱 Alpha 过滤中的 $t = \alpha$，当 t 增加时，我们可以观察到（抽象）单纯复形中的拓扑孔洞发生变化。例如，如图4c所示，不同的连通分支可能通过边连接起来，或者环可能出现或消失。当一个拓扑孔洞在过滤过程中出现时，我们称其为诞生。反之，当一个拓扑孔洞消失时，我们称其消亡 。这些事件发生时所对应的过滤时间分别称为出生时间 b 和死亡时间 d。简而言之，持久同调追踪拓扑孔洞在过滤过程中的出生与死亡时间。

持久同调通常通过持久图 （图7）进行量化。形式上，一个 k 维持久图是一个集合 $\mathcal{D}_k \subseteq \mathbb{R}^2$，可分解为

$$\underbrace{\{(a_i, \infty) : 1 \le i \le N\}}_{=:\mathcal{D}_k^{\text{ess}}} \cup \underbrace{\{(b_j, d_j) : 1 \le j \le M \wedge b_j < d_j\}}_{=:\mathcal{D}_k^{\text{reg}}} \cup \{(x, x) : x \in \mathbb{R}\},$$

其中， a_i， b_j， d_j、 $1 \le i \le N$、 $1 \le j \le M$分别对应过滤过程中k维孔洞的出生与死亡时间。点 (a_i, ∞) 通常显示在图的上方。它们对应过滤过程中始终不会消亡的孔洞，构成持久图的本质部分 。对于第2.2节中定义的过滤，总有 $\mathcal{D}_0^{\text{ess}} = \{(0, \infty)\}$，且 $\mathcal{D}_k^{\text{ess}} = \emptyset$ 对于 $k \ge 1$成立。这是因为最终，过滤中的单纯复形会由一个永不消亡的连通分支构成，而任何更高维孔洞都会被“填满”，如图4c所示。 $\mathcal{D}_k$ 中的点 (t_{b_j}, t_{d_j})，其坐标$t_{b_j} < t_{d_j}$均为有限值，构成持久图的常规部分 D_k^{reg}。最后，持久图中还包含对角线，以便定义持久图之间一个良好的距离度量，称为瓶颈距离。

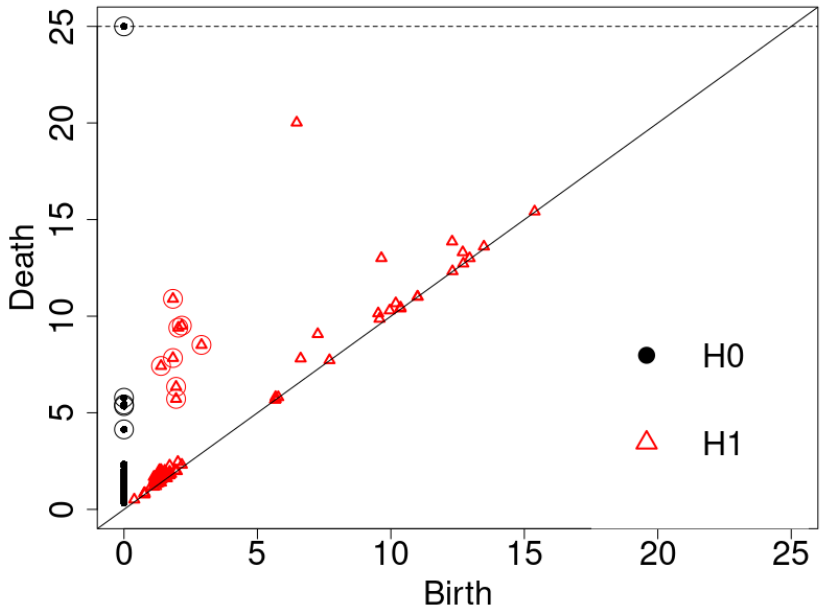


图7：将图4c中弱 Alpha 过滤的持久图 $\mathcal{D}_0$ 和 $\mathcal{D}_1$ 叠加可视化。 H_k 表示同调维数 k。圈出的13个点代表皮卡丘眼睛、嘴巴、鼻子、脸颊和耳朵的特征形状。

定义 11 （瓶颈距离）。设 $\mathcal{D}$ 和 $\mathcal{D}'$ 是两个持久图。它们之间的瓶颈距离 定义为

$$d_{\mathrm{b}}\left(\mathcal{D}, \mathcal{D}'\right) := \inf_{\varphi} \sup_{x} \|x - \varphi(x)\|_{\infty} \in \mathbb{R} \cup \{\infty\},$$

其中， φ 遍历从 $\mathcal{D}$ 到 $\mathcal{D}'$的所有双射，而 x 遍历 $\mathcal{D}$中的所有点。按照惯例，计算两个图中点之间的距离时，我们令 $\infty - \infty = 0$ 。由于持久图按定义包含对角线，因此 $|\mathcal{D}| = |\mathcal{D}'| = |\mathbb{R}|$。这样，$d_{\mathrm{b}}\left(\mathcal{D}, \mathcal{D}'\right)$ 便是良定义的，因为图中未匹配的点总能与对角线上的点匹配。

瓶颈距离通常用于说明为何将持久同调用作一种稳定的方法来量化数据中的拓扑信息，即该方法对微小的数据扰动具有鲁棒性（Oudot，2015年）。

图7展示了图4c中弱 Alpha 过滤的 0 维和 1 维持久图。过滤过程中显著的孔洞，是那些持续时间较长的孔洞 $d - b$；若 $d = \infty$，甚至会无限持续。在持久图中，这些孔洞对应于远高于对角线的点。点（b， ∞）通常绘制在图的上方，例如图7。在图7中叠加绘制的持久图 $\mathcal{D}_0$ 和 $\mathcal{D}_1$ 里，可以观察到13个明显高出的点，它们的出生时间较早 b，这些点在图7中用圆圈标出。 $\mathcal{D}_0$ 中圈出的5个点捕捉了

皮卡丘面部的典型连通分支：两只眼睛、嘴巴、鼻子和轮廓。 $\mathcal{D}_0$ 中圈出的8个点捕捉了皮卡丘面部的典型环：两只眼睛、嘴巴中的两个圈、两边脸颊以及两只耳朵的顶端。虽然弱 Alpha 过滤中连通分支的出生时间自然为零，但这8个典型环在过滤中的出生时间较早但不为零，因为它们出现在较小的尺度上，也就是说，只有连接彼此距离较近的不同数据点时才能定义它们（图4c）。不过，在 $\mathcal{D}_1$ 中还可以观察到出生时间较大的明显高出点。这些点对应于出现在较大尺度上的环，只有连接点云数据中距离更远的点才能定义它们。例如，皮卡丘额头周围的环（在图4c中时间 $\alpha = 10$ 和 $\alpha = 20$ 时存在）是点云中持续时间最长的圈。

计算持久同调从过滤$\mathcal{F} = (\mathcal{K}_0 \subseteq \mathcal{K}_1 \subseteq \ldots \subseteq \mathcal{K}_n = \mathcal{K})$计算持久同调，与计算常规单纯同调类似，都是在最终单纯复形$\mathcal{K}$ 的（此时完整的）边界矩阵 B 上进行（图6）。然而，必须先根据过滤 $\mathcal{F}$中相应单纯形的出现时间，对 B 的行和列进行排序。例如，图6中的边界矩阵 B 就是过滤的矩阵表示

$$\mathcal{F} = (\{\{x_0\}\} \subseteq \{\{x_0\}, \{x_1\}\} \subseteq \ldots \subseteq \{\{x_0\}, \{x_1\}, \ldots, \{x_2, x_4, x_5\}\} = \mathcal{K}), \tag{3}$$

其中 $\mathcal{K}$ 是图5左侧的单纯复形。因此， $\mathcal{F}$中出现的单纯形首先按大小排序，然后按其所含顶点的索引的字典序排序，如图 8 所示。请注意，我们没有指定与这些单纯复形相关的确切出现时间，也没有指定由此产生的 $\mathcal{K}$中的单纯形的确切出现时间。

对于 B 的第 j 列，我们将 low(j) 定义为满足 $B_{ij} \neq 0$ 的最大索引 i 。用于计算持久性的标准算法（有时也称为列算法）（Otter 等，2017），通过算法1在选定的域（例如模 2 ）上约化边界矩阵 B 。随后可以从约化矩阵中读出生灭对，方法如下（ Otter 等，2017）。

- 如果 low(j) $= i$，那么单形 Δ_j 与 Δ_i 配对，而 Δ_i 进入过滤时会导致一个拓扑孔洞诞生，该孔洞在 Δ_j进入时消亡。
- 如果 low(j) 未定义，即约化边界矩阵的第 j 列仅包含零，那么单形 Δ_j 进入过滤时会导致一个拓扑孔洞诞生。如果存在某个 k 使得 low(k) $= j$，那么单形 Δ_j 与 Δ_k 配对；后者进入过滤时，会导致与 Δ_j一同诞生的拓扑孔洞消亡。如果不存在这样的 k ，则 Δ_j 未配对，与之同时诞生的拓扑孔洞将无限期持久存在。

算法1 计算持久性的标准算法。

输入 单纯复形 $\mathcal{K}$上的过滤 $\mathcal{F}$ 的边界矩阵 B 。

对于 $j = 1$ 至 $|\mathcal{K}|$ **，执行**
  当 存在 $i < j$ 使得 low(i) $=$ low(i) **时执行**
    将列 i 加到列 j。

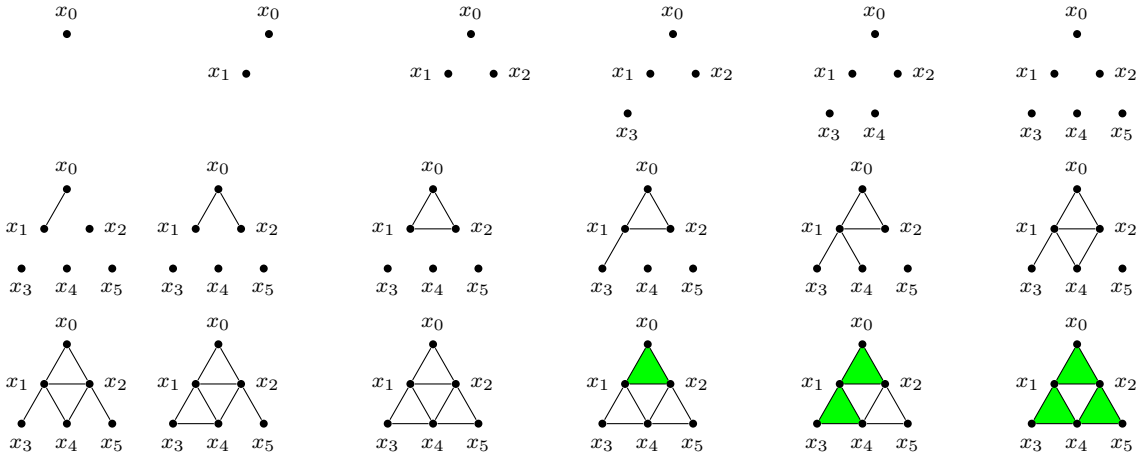


图 8：图5中左侧单纯复形上的过滤，其边界矩阵为图6中的矩阵 B。

	x_0	x_1	x_2	x_3	x_4	x_5	x_{01}	x_{02}	x_{12}	x_{13}	x_{14}	x_{24}	x_{25}	x_{34}	x_{45}	x_{012}	x_{134}	x_{245}
x_0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
x_1	0	0	0	0	0	0	1	0	0	1	1	0	0	0	0	0	0	0
x_2	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0
x_3	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
x_4	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
x_5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
x_{01}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
x_{02}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
x_{12}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
x_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
x_{14}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
x_{24}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x_{25}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x_{34}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
x_{45}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
x_{012}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x_{134}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
x_{245}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

图9：通过算法1得到的图6中的约化边界矩阵 B 。

图9展示了矩阵在 B 中的约化形式，对应于式 (3) 中的过滤。作为示例，我们读出一些生灭对。

- low(x_i) 对每个 $i = 1, \ldots, 5$ 都未定义：每个顶点都会诞生一个连通分支。我们发现，只有对于 x_0，不存在这样的列 x_{ij}：约化矩阵中 low(x_{ij}) $= x_0$。因此，只有通过 x_0 诞生的连通分支会无限期持续。

- low(x_{01}) $= x_1$：随着边 $\{x_0, x_1\}$的加入，随 $\{x_1\}$ 诞生的连通分支消亡。

- low(x_{12}) 未定义，但 low(x_{012}) $= x_{12}$：边 $\{x_1, x_2\}$的加入诞生了一个环，它随着单形 $\{x_0, x_1, x_2\}$的加入而消亡。
- low(x_{24}) 未定义，并且约化矩阵中不存在满足 low(x_{ijk}) $= x_{24}$ 的列 x_{ijk}：边 $\{x_2, x_4\}$ 的加入导致一个无限期持续的环诞生。

请注意，图 8 中也可以观察到所有这些现象。此外，无限期持续的拓扑孔洞对应于过滤中最终单纯复形 $\mathcal{K}$ 的常规单纯同调，该同调是在第 2.3 节末尾计算得出的。

将配对单形映射到它们出现的时间（在实际应用中可以获得这些时间），即可得到出生与死亡元组 (b, d)，也就是持久图。请注意，可以根据配对单形的维度，将出生与死亡元组归入持久图的特定维度。例如，我们知道，边的加入只能导致一个环诞生，或一个连通分支消亡。对于下文将讨论的拓扑优化，从这个例子中最重要的一点是，持久同调和持久图的计算能够追溯一个出生与死亡元组 (b, d) 所对应的特定拓扑孔洞，并将其关联回过滤中导致该孔洞诞生和消亡的两个单纯形。

2.5 拓扑损失函数与拓扑优化

持久同调能够量化数据中从最精细到最粗略的所有拓扑信息。因此，持久图通常被视为数据的拓扑签名，可将其转换为（拓扑）特征，以纳入各种机器学习模型。尽管从持久同调中学习的方法如今已相当成熟且丰富多样（Pun 等人，2018年），但针对持久同调优化数据表示则是最近才受到关注（Br¨uel-Gabrielsson 等人，2020年；Solomon 等人，2021年；Carriere 等，2021年）。如第 2.3 节和第 2.4 节所述，持久同调在代数拓扑中具有相当抽象的数学基础，其计算本质上是组合性的。这使得在优化中使用常规导数变得复杂。为此，拓扑优化采用 Clarke 次导数（Clarke，1990年），而其对持久性的适用性则建立在 o-极小几何的论证之上（van den Dries，1998年；Carriere 等，2021年）。幸运的是，得益于

近期的研究（Br¨uel-Gabrielsson 等人，2020年；Carriere 等，2021年），人们已为 PyTorch 和 TensorFlow 等软件库开发出强大的拓扑优化工具，使用户无需深入了解这些主题也能使用这些工具。

在本节中，我们将重点介绍点云的拓扑优化。请注意，拓扑优化也可应用于其他输入数据结构，例如优化图像中的像素值以进行去噪（Br¨uel-Gabrielsson 等，2020年）。然而，本文旨在对点云嵌入进行拓扑优化。在第2.5.1节中，我们将介绍本文采用的拓扑损失函数的基本形式，该形式沿用（Br¨uel-Gabrielsson 等，2020年）的方法，并给出一些初步示例。在第2.5.2节中，我们将概述点云拓扑优化背后的技术背景，并讨论其对次梯度方法的依赖。

2.5.1 Topological Optimization of Point Clouds: an Example

点云的拓扑优化 $\boldsymbol{X}$ 会根据一个或多个持久图$\mathcal{D}$ 所概括的拓扑信息，优化 $\boldsymbol{X}$ 中各点的位置。本文中的这些持久图将由 $\boldsymbol{X}$ 的弱 Alpha 过滤得到（图 4b 和图 7）。我们将采用（Br¨uel-Gabrielsson 等，2020年）的方法：首先按持久性 $d_k - b_k$降序排列 $\mathcal{D}$ 中的（诞生，消亡）元组（$b_1,\ d_1$），（$b_2,\ d_2$）$\ldots$，（$b_{|\mathcal{D}|},\ d_{|\mathcal{D}|}$）。对于点云，始终只有一个拓扑孔洞，即在时间 $\alpha = 0$诞生的一个连通分支，会无限期地持续存在。其他间隙和孔洞最终都会被填满（图 4b 和图 7）。因此，本文仅优化持久图的常规部分，也就是坐标有限的图中各点。具体做法是使用一个拓扑损失函数；对于给定的 $i \leq j$、拓扑孔洞的维数 k 以及函数 g：$\mathbb{R}^2 \to \mathbb{R}$，该函数定义为

$$\mathcal{L}_{\text{top}}(\mathcal{D}_k) \coloneqq \sum_{l=i, d_l<\infty}^{j} g(b_l, d_l), \qquad \text{where } d_1 - b_1 \geq d_2 - b_2 \geq \ldots, \tag{4}$$

其中 $(b_l, d_l) \in \mathcal{D}_k$, $l = 1, \ldots |\mathcal{D}_k|$。首先按持久性降序排列 $\mathcal{D}_k$ 中的点，(4) 就是一个持久性函数 （Carriere 等，2021），这意味着它对于持久图 $\mathcal{D}_k$中各点的排列保持不变。

例如，考虑通过弱 Alpha 过滤得到的皮卡丘点云表示 $\boldsymbol{X}$ 的 1 维持久图 $\mathcal{D}_1$ （图 4 b 和 7）。如第2.4节所述，按照 (4) 中的排序，持续时间最长的环——由点 $(b_1, d_1) \in \mathcal{D}_1$表示——捕捉到了皮卡丘额头的轮廓。以下三个函数便是拓扑损失函数的示例，可用于增加该环的持久性 $d_1 - b_1$，其中 $i = j = 1$ 取自 (4)。

- $\mathcal{L}_{\text{顶部}_1}(\mathcal{D}_k) = -d_1$,
- $\mathcal{L}_{\text{顶部}_2}(\mathcal{D}_k) = b_1$,
- $\mathcal{L}_{\text{顶部}_3}(\mathcal{D}_k) = -(d_1 - b_1)$。

图 10 展示了针对这些损失函数分别进行优化的结果。

针对 $\mathcal{L}_{\text{top}_1}$进行优化时，我们看到环扩大了。然而，环中的间隙被忽略，甚至还会出现新的间隙（图 10，左）。相反，针对 $\mathcal{L}_{\text{top}_2}$ 进行优化可以填补这些间隙，并确保环在数据中以更小的尺度出现，或者更形式化地说，

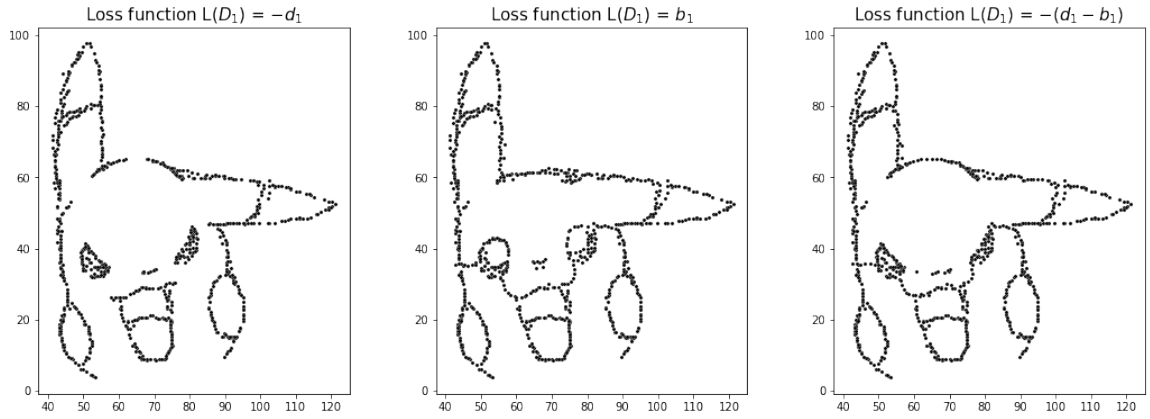


图 10：针对各种拓扑损失函数优化后的点云 $\boldsymbol{X}$，这些损失函数均旨在提高对应于皮卡丘额头轮廓的最显著环的持久性 $d_1 - b_1$。

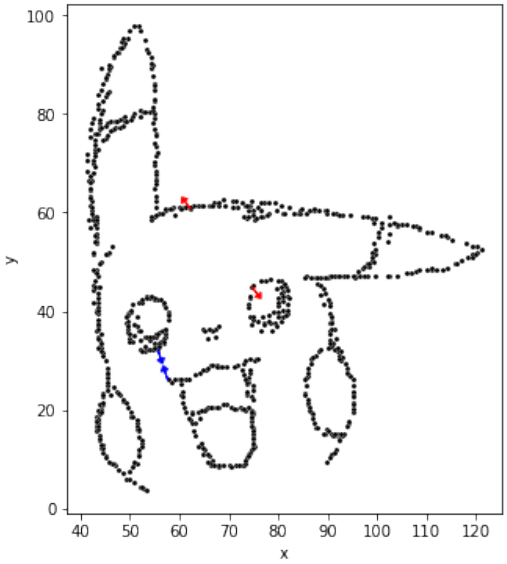


图11：拓扑损失函数 $\mathcal{L}_{\text{top}}(\mathcal{D}_1) = b_1 - d_1$ 的负梯度显示了在 $\boldsymbol{X}$ 的首次拓扑优化迭代期间点的移动方向。梯度仅在四个点上非零。在弱 Alpha 过滤过程中，其中两点的连接导致环诞生（蓝色箭头）。这两点相互靠近，以缩短该环的出生时间。在弱 Alpha 过滤过程中，另外两点的连接导致环消亡（红色箭头）。这两点彼此远离，以延长该环的死亡时间。

对于更早的过滤时间（图 10，中），但不会扩大该环。自然地，在针对 $\mathcal{L}_{\text{top}_3}$ 进行优化时，这两种效果都会实现（图 10，右）。

图11展示了拓扑损失函数 $\mathcal{L}_{\text{top}_3}(\mathcal{D}_k)$ 的负梯度方向，即点移动方向；这是对 $\boldsymbol{X}$进行的第一次拓扑优化迭代。我们观察到， $\boldsymbol{X}$中只有四个点的梯度非零。这些点要么是导致持续时间最长的圈诞生的两个点（图11中的蓝色箭头），要么是导致该圈消亡的两个点（图11中的红色箭头）。这是因为，点（b_1， d_1） $\in \mathcal{D}_1$ 可追溯到那些进入时导致该圈诞生与消亡的配对单形（第2.4节）。当图11中以蓝色梯度标出的点由一条边连接时，该环便在时间 b_1 诞生。当图11中以红色梯度标出的点由一条边连接时，该环便在时间 d_1 消亡；这条边进入过滤后，会与第三个未标记的点构成一个三角形，从而闭合该环。由于这些边的过滤时间，即它们在弱 Alpha 过滤中首次出现的时间，恰好等于其端点之间的距离，因此梯度方向相反，旨在降低出生时间 b_1 或提高死亡时间 d_1，从而增加总持续时间 $d_1 - b_1$。

2.5.2 Technical Background of Topological Optimization of Point Clouds

拓扑优化依赖（随机）次梯度方法，根据持久图所概括的拓扑性质来优化数据表示。为说明这一点，最简单的方式是从基于 Vietoris-Rips 过滤（定义 5）的拓扑优化入手。

对于给定的点云 $\boldsymbol{X} \subseteq \mathbb{R}^d$，Vietoris-Rips 过滤是在单纯复形 $\mathcal{K}$ 上定义的一种过滤，其中包含由 $\boldsymbol{X}$ 中每个点子集构成的单形，并且可以受到维度限制。如果现在 $\boldsymbol{X}$ 由 n 个点组成，我们可以将 Vietoris-Rips 过滤的构造视为一个映射 VR : $A = \left(\mathbb{R}^d\right)^n \to \mathbb{R}^{|\mathcal{K}|}$，其中 A 中的一个元素等于由 n 个 $\mathbb{R}^d$ 中的点构成的点云，经过 VR 映射后得到一个（Vietoris-Rips）过滤；该过滤可以表示为对 $\mathcal{K}$ 中每个单形赋予一个实数值（过滤时间）。

持久同调会在对 $\mathcal{K}$ 进行过滤的过程中，将导致同一个拓扑孔洞诞生与消亡的单形配对。为简化记号，我们考虑一个完整的持久图 $\mathcal{D}$，它是所有维度上的持久图的并集。设过滤过程中配对单形的数量为 p，未配对单形的数量为 q （第2.4节）。那么 $|\mathcal{K}| = 2p + q$。此外，特别是对于 Vietoris-Rips 过滤，数量 p 和 q 仅取决于数据规模 n 和 $\mathcal{K}$ 的维度。对于 $\mathbb{R} \times$ 采用字典序（$\mathbb{R} \cup \{\infty\}$）后，我们可以将持久图 $\mathcal{D}$ 视为 $\mathbb{R}^{2p+q} = \mathbb{R}^{|\mathcal{K}|}$ 中的一个向量。因此，我们将持久同调运算视为一个映射 Pers : $\mathbb{R}^{|\mathcal{K}|} \to \mathbb{R}^{|\mathcal{K}|}$。由于出生与死亡时间必然是单形的过滤值，因此持久同调运算 Pers 只是对输入过滤的坐标进行排列；该输入过滤被视为 $\mathbb{R}^{|\mathcal{K}|}$ 中的一个向量。

如式 (4) 中所定义的拓扑损失函数 $\mathcal{L}_{\text{top}}$，拓扑优化的目标因此是最小化该函数

$$\mathcal{L}_{\text{top}} \circ \text{Pers} \circ \text{VR} : A = \left(\mathbb{R}^d\right)^n \to \mathbb{R}, \tag{5}$$

关于点云 $\boldsymbol{X} \in A$。

o-极小几何 如今为描述参数化的过滤族（此处由点云 $\boldsymbol{X} \in A$ 给出），以及展示这些过滤族与持久性映射和拓扑损失函数复合后的有趣可微性，提供了一个非常合适的框架（Carriere 等，2021）。

定义 12 （*O*-极小结构（*Carriere* 等，*2021*））。实数域 $\mathbb{R}$ 上的O-极小结构是一个集合族（$\mathbb{S}_n$）$_{n\in\mathbb{N}}$，其中每个 $\mathbb{S}_n$ 都是 $\mathbb{R}^n$的一个子集，且满足：

1. $\mathbb{S}_1$ 恰好是由有限个点和区间构成的有限并的集合；

2. $\mathbb{R}^n$ 的所有代数子集（多项式的零水平集）都属于 $\mathbb{S}_n$；

3. 对任意 $n \in \mathbb{N}$， $\mathbb{S}_n$ 都是 $\mathbb{R}^n$ 的布尔子代数；

4. 若 $A \in \mathbb{S}_n$ 且 $B \in \mathbb{S}_m$，则 $A \times B \in \mathbb{S}_{n+m}$；

5. 若 $\pi:\ \mathbb{R}^{n+1} \to \mathbb{R}^n$ 是到前 n 个坐标的线性投影，且 $A \in \mathbb{S}_{n+1}$，则 $\pi(A) \in \mathbb{S}_n$。

对于某个 $n \in \mathbb{N}$，元素 $A \in \mathbb{S}_n$ 称为该 *O*-极小结构中的可定义集。对于可定义集 $A \subseteq \mathbb{R}^n$，若映射 $f:\ A \to \mathbb{R}^m$ 的图是 $\mathbb{R}^{n+m}$ 中的可定义集，则称该映射为可定义的。

可定义集在各种几何运算下保持稳定（Coste，2000；Carriere 等，2021）。尽管其背后的确切细节超出了本文的讨论范围，Carriere 等（2021）证明，若持久性函数 $\mathcal{L}_{\text{top}}$ 在某个 O-极小结构中局部利普希茨且可定义，则 $\mathcal{L}_{\text{top}} \circ$Pers 也局部利普希茨，且复合函数（5）是可定义的。因此，复合函数（5）具有定义良好的克拉克次微分（Clarke，1990），因为它几乎处处可微（Carriere 等，2021）。随后可使用标准随机次梯度算法，针对参数化（Vietoris-Rips）过滤的点云优化（5）（Carriere 等，2021）。

映射的拓扑优化理论

$$\mathcal{L}_{\text{top}} \circ \text{Pers} \circ \text{WA} : A \to \mathbb{R}, \tag{6}$$

其中，WA 将点云映射为其弱 Alpha 过滤，其理论与 Vietoris-Rips 过滤的理论类似。然而，德劳内三角剖分（作为抽象单纯复形）可能会随输入点云 $\boldsymbol{X} \in (\mathbb{R}^d)^n$ 而变化。如果 $\boldsymbol{X}$ 中的点处于一般位置（第 2.2 节），我们可以将 A 限定在 $(\mathbb{R}^d)^n$ 的连通开子集中；该子集具有这样的性质：对于其中的每个 $\boldsymbol{Y} \in A$，其 Delaunay 三角剖分 $\mathcal{K}_{\boldsymbol{Y}}$ 除了相关几何形状外，都与 $\boldsymbol{X}$ 的 Delaunay 三角剖分 $\mathcal{K}$ 相同（Carriere 等，2021）。这意味着，尽管它们的单纯形在 $\mathbb{R}^d$ 中的确切位置可能不同，但定义 $\mathcal{K}_{\boldsymbol{Y}}$ 单纯形的点集（例如，以索引 1，……，n 标记）与定义 $\mathcal{K}$ 单纯形的点集相同，因此复合映射 (6) 中的每个映射也都可以被良好定义。请注意，这并不意味着在整个拓扑优化过程中，Delaunay 三角剖分必然保持不变。

2.6 计算分析

目前，拓扑优化的每次迭代都需要重新计算持久图。因此，拓扑优化的计算分析归结为持久同调的计算分析，而这至今仍然颇具挑战性。尽管持久同调是一种无与伦比的工具，能够量化数据中从最细微到最粗大的所有拓扑孔洞，但它依赖于代数计算（算法1）；对于规模较大的单纯复形，这些计算可能代价高昂。最坏情况下，计算持久同调的复杂度与单纯形数量的三次方成正比（Otter 等，2017）。

对于包含 n 个点的数据集 $\boldsymbol{X} \subseteq \mathbb{R}^d$，弱 Alpha 复形的规模和计算复杂度为 $\mathcal{O}\left(n^{\lceil \frac{d}{2} \rceil}\right)$（Otter 等，2017年；Toth 等，2017年；Somasundaram 等，2021年），因此持久同调的计算复杂度为 $\mathcal{O}\left(n^{3\lceil \frac{d}{2} \rceil}\right)$。然而，在实际应用中，这一复杂度可能显著降低，例如，当点几乎均匀地分布在多面体上时（Amenta 等，2007）。Vietoris-Rips 过滤的规模和计算复杂度为 $\mathcal{O}(\min(2^n, n^{k+1}))$，其中 $d \geq k$ 是感兴趣的同调维数（Otter 等，2017年；Somasundaram 等，2021年），因此持久同调的计算复杂度为 $\mathcal{O}(\min(2^{3n}, n^{3(k+1)}))$。在实践中，选择 k 也会显著减小弱 Alpha 复形的规模，从而降低持久同调计算的复杂度。然而，仍需先构造完整复形，即 Delaunay 三角剖分（Boissonnat 等，2018年）。处理高维数据的弱 Alpha 复形时，这通常是主要的瓶颈。

实践中，对于低维点云，弱 Alpha 复形适合用于计算持久同调；而对于高维情形，点数较少时则更适合使用 Vietoris-Rips 复形（Somasundaram 等，2021年）。在拓扑正则化和数据嵌入的背景下，我们的目标是使点云嵌入的维数 d 较低。这也说明了选择弱 Alpha 过滤的合理性。

请注意，目前已有许多针对持久同调的计算改进和近似算法（Otter 等，2017年）。不过，将它们整合到拓扑优化中仍有待进一步研究。

3. 拓扑正则化数据嵌入

本节提出一系列实用的拓扑损失函数，用于优化点云嵌入中的拓扑先验。我们基于弱 α-过滤的持久同调，展示这些损失对二维点云数据的影响。第3.1节介绍用于优化 k 维孔洞的基本构件。第3.2节提出的采样策略可改善运行时间以及特定结构的表示。第3.3节描述一种更复杂的拓扑损失函数，用于优化喇叭状结构。最后，我们指出如何将拓扑正则化融入降维方法。

3.1 将拓扑结构描述为 k 维孔洞

拓扑损失根据二维点云嵌入上的过滤所对应的持久图计算得出。如第2.4节所述，持久图记录了 k 维拓扑孔洞的诞生 和消亡 时刻。这些特征或它们之间的差异，可用于定义损失函数；经过优化后，这些损失函数会使生成的嵌入偏向各种拓扑结构。我们通过测量一个 k 维孔洞的持久性，将公式（4）中的一般损失实例化为

$$\mathcal{L}_{\text{top}}(\mathcal{D}_k) = \mu \sum_{l=i, d_l<\infty}^{j} d_l - b_l \,. \tag{7}$$

我们假设生灭对已按如下方式排序：(b_1, d_1) 是持久性最大的生灭对。为了最大化或最小化，我们在所有实验中都使用 $\mu \in \{-1, 1\}$ 。由于我们在二维空间中计算低维嵌入，因此只能在 $\mathcal{D}_0$ 和 $\mathcal{D}_1$ 上定义损失函数。对于三维嵌入，按照与 $\mathcal{D}_1$ 和 $\mathcal{D}_2$ 类似的思路，也可以定义 $\mathcal{L}_{\text{top}}(\mathcal{D}_2)$。在第4.3.1节中，我们将介绍公式（4）的一个更灵活版本，但目前仍采用这一基本的持久性度量。

0 维孔洞（即连通分支）对于基于 α-过滤的 0 维孔洞，损失函数可简化为 $\mathcal{L}_{\text{top}}(\mathcal{D}_0) = \mu \sum_{l=i,\ d_l<\infty}^{j} d_l$，因为所有生灭对的诞生时间均为零。在过滤开始时（时间为 $\epsilon = 0$），每个点各自构成一个连通分支；随着 ϵ 增大，这些分支会通过边彼此连通并合并。我们在表 1 中展示了 μ、i 和 j 取不同值时的效果。点云 $(n = 20, d = 2)$ 从各向同性高斯分布中采样得到，我们针对该点云优化拓扑损失，共训练 500个训练轮次。

对于大小为 n 的点云，0 维同调始终包含 n 对，其中有一对具有无限寿命（$d_1 = \infty$）。因此，我们从 $i = j = 2$ 开始，它作用于有限持久性最大的持久对。根据 μ，这种损失会使最后连接的点之间的距离增大或减小。使用 $i = j = 3$ 时，倒数第二次连接的点之间的距离会增大或减小。请注意，当 $\mu = 1$ 时，这会导致最后连接的两个簇之间出现较大的间隔（其中一个簇在这种情况下仅包含一个点），因为这两个簇之间的距离不会影响损失。然而，当 $\mu = -1$ 时，点云会分裂成三个簇，其中最大的两个距离相近（在这种情况下，三个簇恰好彼此等距）；这是因为推迟倒数第二次合并最终会使其成为过滤中的最后一次合并，反之亦然。

使用 $i = j = n$ 最小化持久性，会使彼此距离最近的两个点重合；而最大化持久性，则会使所有点分布到其最近邻距离相等的位置，然后增大所有点之间的间距。使用 $i = 2$ 和 $j = \infty$ （或 $j = n$），我们可以最小化或最大化所有有限死亡时间（注意这两种嵌入的尺度）。对于 $\mu = -1$，我们观察到一种圆形模式，其中有些点位于中心；只有在增加训练轮次进行优化时，该模式的尺度才会增大。

一维洞（即循环拓扑）通过一维洞，我们在二维嵌入上施加循环拓扑。与 0 维拓扑损失不同，公式（7）中的出生时间 b_l 非零，因此会对损失产生影响。在

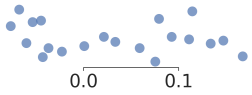
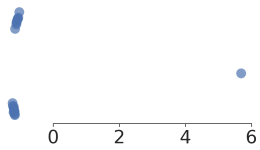
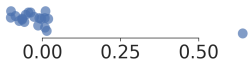
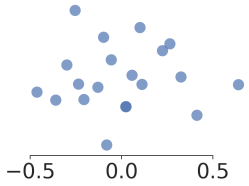
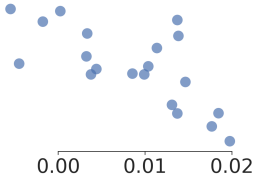
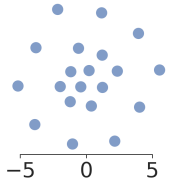
i,j	描述	$\mu = 1$ (最小值)	$\mu = -1$ (最大值)
2,2	点之间的距离 其中最长的边位于 最小生成树 （最小生 成树）		
3,3	点之间的距离 带有第二长边 在最小生成树中		
n,n	点之间的距离，与 最小生成树中的最短边		
2,∞	最小生成树的所有距离		

表 1：使用 $\mathcal{D}_0$ 在合成数据上优化拓扑损失函数，训练 500 个训练轮次。

表2展示了在 $\mathcal{D}_1$ 上针对不同的 i，j 和 μ 优化这一损失的结果。首先要指出的是，最小化一维拓扑（$\mu = 1$）会使不同 i 和 j 值对应的嵌入高度相似。这些嵌入的持久图 $\mathcal{D}_1$ 中都不存在一维拓扑。没有环诞生，因为任何潜在环的半径都小于该环所包含的邻近点之间的距离。

使用 $i = j = 1$ 和 $\mu = -1$优化最显著孔洞的持久性，会得到一个圆，点均匀分布在边界上。请注意，圆外的点不会改变仅用于衡量最大持久性的损失。使用 $i = j = 2$ 进行最大化，会得到两个大小相同的圆，因为优化过程会在两者之间交替。优化所有圈的持久性（$i = 1$，$j = \infty$）会得到一个较大的圆和几个较小的圆。

借助这些基础示例中的直观理解，我们可以在 $\mathcal{D}_0$ 和 $\mathcal{D}_1$ 上构建损失函数的线性组合；例如，如果我们希望所表示的模型既连通又包含一个圆，或者已知存在两个彼此不连通的圆。

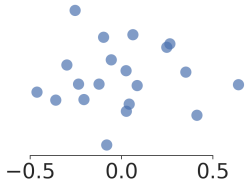
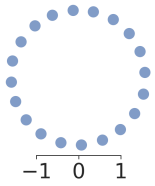
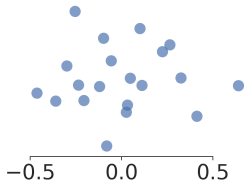
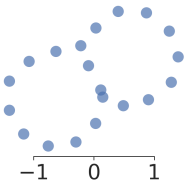
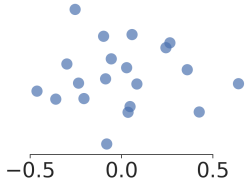
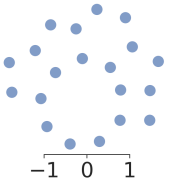
i,j	描述	$\mu = 1$ (最小值)	$\mu = -1$ (最大值)
1,1	最持久的循环的持久性 循环		
2,2	第二持久的循环的持久性 最持久的循环		
1,n	所有循环的持久性		

表2：使用 $\mathcal{D}_1$ 在合成数据上优化拓扑损失函数，训练 500 个训练轮次。

3.2 通过子采样改进运行时间和表示

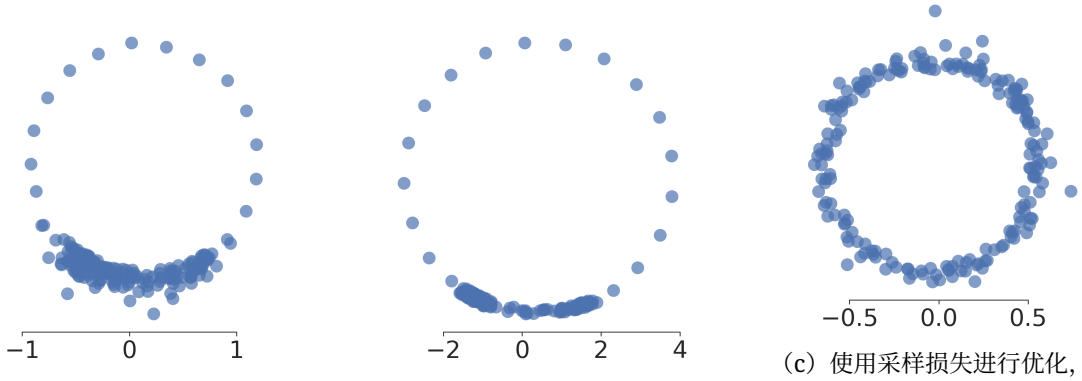
表 1 和表 2 中的示例表明，持久同调能够有效衡量拓扑孔洞的重要程度。然而，它通常无法以自然的方式表示这类孔洞。为说明这一点，我们在一个更大的二维数据集上优化最持久的循环，该数据集包含从各向同性高斯分布中采样的 $n = 200$ 个点，并在图 12a 中展示结果。尽管有些点是该循环的关键部分，移动或移除它们会影响损失，但其他点位于边界之外，不会影响拓扑损失。更重要的是，在这些点成为圆的一部分之前，它们的位置不会发生变化，因为每次梯度更新只会影响促成该循环诞生与消亡的四 个点（见图11）。当优化更多训练轮次时，表示可能会略有改善（图 12b），但运行时间很快就会变得难以接受。

为了通过更多的点来表示拓扑孔洞，我们建议优化损失

$$\widetilde{\mathcal{L}}_{\text{top}}(\boldsymbol{E}) \colon = \mathbb{E}\left[\mathcal{L}_{\text{top}}(\{\boldsymbol{x} \in \mathbf{S} \colon \mathbf{S} \text{是从 } \boldsymbol{E} \text{中随机采样得到的样本，采样比例为 } f_{\mathcal{S}}\})\right], \tag{8}$$

其中 $\mathcal{L}_{\text{top}}$ 的定义与式 (7) 相同。实际操作中，在每次优化迭代期间，$\mathcal{L}_{\text{top}}$ 近似为在点云 $\boldsymbol{E}$的 $n_{\mathcal{S}}$ 个随机样本上计算得到的 $\mathcal{L}_{\text{top}}$ 的均值。

在图12c中，我们展示了使用 $f_{\mathcal{S}} = 0.2$ 和 $n_{\mathcal{S}} = 1$优化最持久的循环的拓扑采样损失后的结果。在数据的一个子集上评估拓扑损失，可以使点在圆形周围的分布更加均衡。另一个



(a) 优化 for 500 训练轮次（17秒）

(b) 优化 for 2k 训练轮次（1分7秒）

（c）使用采样损失进行优化，训练500个训练轮次（5秒），使用 $f_{\mathcal{S}} = 0.2$

图12：使用 $n = 200$ 个点、在 $\mathcal{D}_1$ 上并结合 $i = j = 1$ 和 $\mu = -1$，对二维数据集进行拓扑优化。

基于采样的公式（8）损失的优点是，由于在较小的样本上评估持久同调，优化速度可以显著加快。

带采样的拓扑损失函数计算成本 当采用采样比例 $0 < f_{\mathcal{S}} \leq 1$ 并进行 $n_{\mathcal{S}}$ 次重复时，使用弱α-过滤时，拓扑损失函数的计算复杂度可降至 $\mathcal{O}\left(n_{\mathcal{S}}\left((f_{\mathcal{S}}n)^{3\lceil\frac{d}{2}\rceil}\right)\right)$，其中 d 为嵌入维数。然而，正如第 2.6 节所述，实际计算复杂度通常可能显著更低，这是因为我们会限制所优化的同调维数，而且自然形状中常见的分布可降低德劳内三角剖分的规模和时间复杂度（Dwyer，1991；Amenta 等，2007；Devillers 和 Dum´enil，2019年）。

3.3 Flare结构优化

持久同调对某些拓扑变化具有不变性。例如，线性“I”形结构模型和分叉“Y”形结构模型都由一个连通分量组成，且没有高维孔洞。尽管这两种模型在奇异点方面的拓扑结构不同，但依据它们的（持久）同调无法区分二者。

通过一种名为函数持久性的方法，对持久同调进行改进，可以捕捉这类额外的拓扑现象；Carlsson（2014）也对此进行了充分讨论和阐释。我们不再在数据矩阵$\boldsymbol{E}$上计算持久同调，而是在子集 $\{\boldsymbol{x} \in \boldsymbol{E}$上计算：$f(\boldsymbol{x}) \leq \tau\}$，其中函数 $f\colon \boldsymbol{E} \to \mathbb{R}$和超参数 τ经过精心选择。受此方法启发，对于点云 $\boldsymbol{E}$的图 $\mathcal{D}$，我们提出拓扑损失

$$\widetilde{\mathcal{L}}_{\text{top}}(\boldsymbol{E}) \coloneqq \mathcal{L}_{\text{top}}\left(\{\boldsymbol{x} \in \boldsymbol{E} : f_{\boldsymbol{E}}(\boldsymbol{x}) \leq \tau\}\right), \text{ informally denoted } \left[\mathcal{L}_{\text{top}}(\mathcal{D})\right]_{f_{\boldsymbol{E}}^{-1}]-\infty,\tau]}, \tag{9}$$

其中，f 是 $\boldsymbol{E}$上的实值函数，可能取决于 $\boldsymbol{E}$（它本身也会在优化过程中发生变化）；τ 是一个超参数，而 $\mathcal{L}_{\text{top}}$ 是公式（7）所定义的常规拓扑损失。

在实验中，我们将使用函数持久性的思想来优化星形结构，即星形结构。具体而言，我们将重点关注 f 等于 $\boldsymbol{E}$ 上的缩放中心性度量的情况：

$$f_{\boldsymbol{E}} \equiv \mathcal{E}_{\boldsymbol{E}} \coloneqq 1 - \frac{g_{\boldsymbol{E}}}{\max g_{\boldsymbol{E}}}, \text{ where } g_{\boldsymbol{E}}(\boldsymbol{x}) \coloneqq \left\| \boldsymbol{x} - \frac{1}{|\boldsymbol{E}|} \sum_{\boldsymbol{y} \in \boldsymbol{E}} \boldsymbol{y} \right\|. \tag{10}$$

~ ~有了 $\tau \geq 1$，我们得到 $\mathcal{L}_{\text{top}}(\boldsymbol{E})$ $= \mathcal{L}_{\text{top}}(\boldsymbol{E})$。当 $\tau > 0$ 足够小时，$\mathcal{L}_{\text{top}}$ 会在 $\boldsymbol{E}$ 中心处那些“远离”均值的点上计算 $\mathcal{L}_{\text{top}}$。在第4节中，我们将把这一思想与零维持久性相结合，以优化星形结构。

3.4 在降维方法中融入拓扑正则化

为了对点云嵌入 $\boldsymbol{E}$ 进行正则化，我们将嵌入损失与表示数据先验知识的拓扑损失结合起来。目标是找到一个使总损失函数最小化的嵌入

$$\mathcal{L}_{\text{tot}}(\boldsymbol{E}, \mathbb{X}) \coloneqq \mathcal{L}_{\text{emb}}(\boldsymbol{E}, \mathbb{X}) + \lambda_{\text{top}} \mathcal{L}_{\text{top}}(\boldsymbol{E}), \tag{11}$$

其中，$\mathcal{L}_{\text{emb}}$ 旨在保留原始数据的结构属性，而 $\lambda_{\text{top}} > 0$ 控制拓扑正则化的强度。请注意，$\mathbb{X}$ 本身不必是点云，也不必与 $\boldsymbol{E}$ 位于同一空间。这对于图的表征学习尤其有用。因此，我们可以使用拓扑正则化来嵌入图 G，从而学习 G 的节点在 $\mathbb{R}^d$ 中的表示，并充分保留 G 的属性。为嵌入该图，我们使用了一个改编自 Dagar 等人（2020年）的 DeepWalk 变体。为了通过 DeepWalk 对图嵌入进行正则化，或通过 UMAP 对表格数据嵌入进行正则化，我们只需将它们的目标函数与公式（11）中的拓扑损失 $\mathcal{L}_{\text{top}}$ 相结合。

除了 DeepWalk 和 UMAP，我们还展示了正则化 *PCA* 投影的实验。对 PCA 投影进行拓扑正则化更为复杂，因为使用

$$\mathcal{L}_{\text{emb}}(\boldsymbol{W}, \boldsymbol{X}) \coloneqq \text{MSE}\left(\boldsymbol{X}\boldsymbol{W}\boldsymbol{W}^T, \boldsymbol{X}\right)$$

并不能确保矩阵 $\boldsymbol{W}$ 的正交归一性。为了解决这个问题，我们使用 Pymanopt（Townsend 等人，2016年）在正交归一矩阵构成的斯蒂费尔流形上进行优化。然而，用于计算优化每次迭代最优步长的线搜索算法不适用于我们的拓扑采样损失。因此，我们移除了线搜索中的回溯部分。请注意，在本文的会议版本（Vandaele等，2022b）实验中，我们在权重 $\boldsymbol{W} \in \mathbb{R}^{n \times 2}$ 上使用了正交损失 $\|\boldsymbol{W}^T\boldsymbol{W} - \boldsymbol{I}\|_F$，这对优化产生了不利影响。[1]

1. 在合成周期的实验中，正交损失将 $\boldsymbol{W}$ 的权重转移到了前两个数据维度上，而拓扑损失正则化并未如此。我们怀疑这是正交损失与过高学习率共同造成的数值伪影。

4. 实验与应用

本节中，我们将展示拓扑正则化对合成数据和真实世界数据的影响。在第4.2节中，我们将展示如何利用真实值先验优化嵌入，并量化其对后续预测任务的影响。在第4.3节中，我们将探讨采用不同损失函数参数化时拓扑正则化的鲁棒性。我们围绕六个研究问题设计了实验，这些问题在本节开头列出。我们的代码可在 https://github.com/aida-ugent/TopoEmbedding 获取。

4.1 数据集

我们在一个合成数据集、两个生物数据集和两个网络数据集上评估拓扑正则化的有效性和鲁棒性。表3列出了这些数据集的规模和真实模型，图 13 展示了相应的可视化图。

合成周期 我们从 $\mathbb{R}^2$ 中的单位圆上均匀采样 50 个点。随后，我们在每个维度上从 $[-0.45, 0.45]$ 均匀采样并添加 500 维噪声。

细胞周期 我们研究了一个单细胞轨迹数据集，其中包含 264 个细胞，这些细胞位于 6812 维基因表达空间中（Cannoodt 等人，2018年；Saelens 等人，2019年）。这些数据可视为固定时间点的细胞快照。其真实值是一个圆形模型，通过细胞分化连接三个细胞群（另见第4节.2.1）。

细胞分叉 我们研究了第二个细胞轨迹数据集，其中包含 154 个细胞，这些细胞位于 1770 维表达空间中（Cannoodt 等人，2018年）。这里的真实值是一个分叉模型，通过细胞分化连接四个不同的细胞群。

空手道空手道网络（Zachary，1977）是图挖掘领域中一个广为人知且经过深入研究的网络，由两个不同的社群组成。如图 13d 所示，这两个社群分别以两位关键人物（John A. 和 Mr. Hi）为代表。

《**哈利·波特**》该图（https://github.com/hzjken/character-network）中的节点是《哈利·波特》小说中的人物，边表示人物之间的友好关系（图 13e）。在实验中，我们仅使用最大的连通分支。Vandaele 等人（2020b）此前分析过《哈利·波特》图，并在其中识别出一个圆形模型，该模型展现了小说中“善良”与“邪恶”人物之间的转换。

4.2 拓扑正则化的有效性

本节设计并分析实验，旨在回答以下有关拓扑正则化有效性的问题：

问题1. 拓扑正则化能否成功地将特定的拓扑结构施加到低维嵌入中？

Q2. 拓扑正则化如何影响下游预测任务？

Name	n	d	真实模型
合成周期	50	500	第 1 和第 2 维中的圆
细胞周期	264	6812	圆
细胞分叉	154	1770	分叉
空手道	34	78	两个簇
哈利·波特	58	217	圆

表3：数据规模及其真实模型摘要。

Data	方法	lr	训练轮次	λ_{top}	t 不含拓扑	t 含拓扑
合成周期	PCA	0.01	≤ 1000	0.0005	<1s	12s
细胞周期	PCA	0.01	≤ 1000	0.01	<1s	3分29秒
细胞分叉	UMAP	0.1	100	10	<1s	4s
空手道	DeepWalk	1e-2	50	50	19s	20s
哈利·波特	内积	1e-1	100	1e-1	1分21秒	1分24秒

表4：各数据集优化参数汇总。

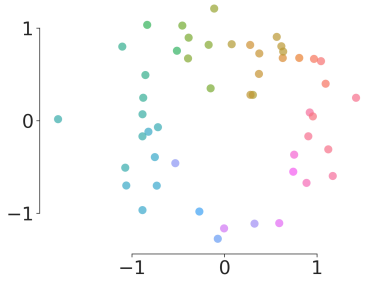
问题3. 拓扑正则化的可扩展性如何?

降维方法及其优化参数的概览见表4，拓扑损失函数见表5。

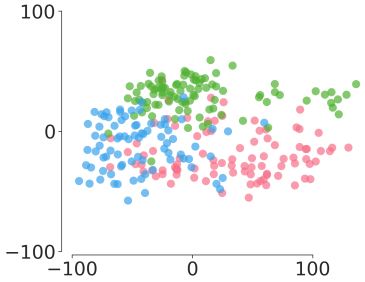
4.2.1 Qualitative Evaluation

本节探讨问题1，通过比较五个数据集各自的拓扑优化嵌入和标准嵌入，说明对嵌入进行拓扑正则化的效果。我们使用拓扑损失函数来模拟数据的真实结构（见表3）。

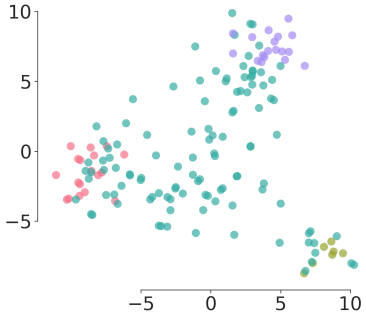
我们将“优化嵌入”用于指仅优化拓扑损失、训练轮次预先设定，并以 PCA、UMAP 或 DeepWalk 嵌入作为初始化的情况。对于优化后的 PCA 嵌入，我们不使用重构损失，但仍在斯蒂费尔流形上进行优化，从而得到正交投影。正则化嵌入 是最小化嵌入损失与拓扑损失的组合后得到的结果。我们将这两种损失之间的权衡 λ_{top} 设置得尽可能低，但也足够高，以确保正则化嵌入与优化嵌入有明显差异。使用 DeepWalk 和 UMAP 时，优化嵌入与正则化嵌入采用相同的训练轮次（见表4）。对于 PCA，我们最多使用 1000 个训练轮次，但当拓扑损失开始停滞时便停止优化（通过最近 100 轮的平均拓扑损失与最近 50 轮平均值之比来衡量）。



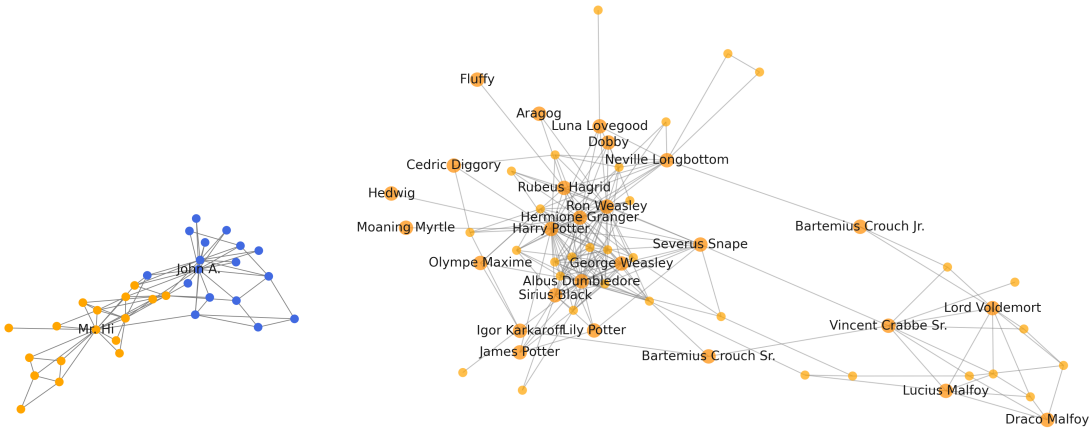
(a) 500 维合成周期数据的前两个数据坐标。



(b) Cell 环 data with 颜色表示 细胞状态 (主成分分



(c) 以颜色表示细胞组的细胞分叉数据（UMAP）。



(d) 空手道网络（NetworkX（e）《哈利·波特》图中的主要连通分支弹簧布局）。(NetworkX 弹簧布局)。边表示人物之间的友好关系。

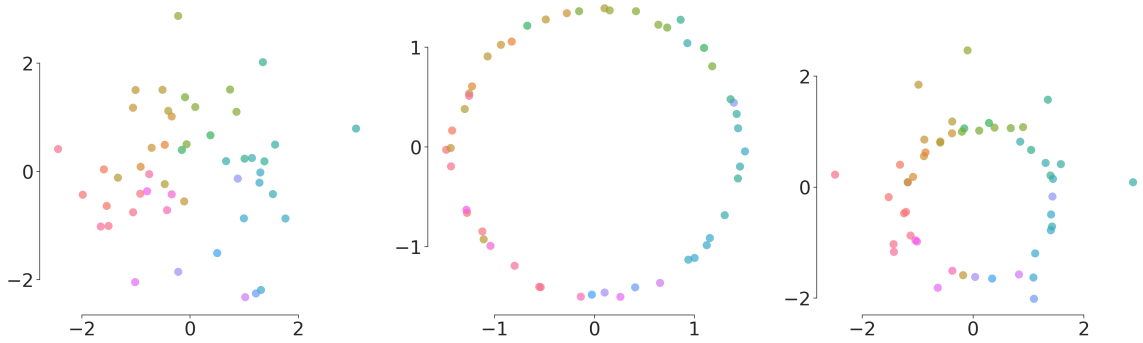
图 13：实验中使用的数据集的可视化图。获取二维嵌入的方法标注在括号中。

合成数据 对于 500 维的合成圆，额外添加的 498 个带噪特征与拓扑（圆形）模型无关。理想的投影嵌入应仅取前两个特征（见图 13a）。然而，在对数据进行 PCA 嵌入时，无关特征的贡献恰好为零的概率很低。直观而言，每添加一个特征，投影平面都会略微偏离由前两个特征张成的平面。在图 14a 中，我们观察到，PCA 嵌入中的圆形孔洞确实不那么明显。请注意，我们观察到，这对于“小型 n 大型 p”数据集是一个显著问题；与其他机器学习模型类似，而且 Vandaele 等人（2022a）最近也对此进行了研究，更多数据可以显著抵消噪声的影响，并使嵌入模型本身得到改善。

为了改善圆的视觉表示，我们使用拓扑损失函数$\mathcal{L}_{\text{top}}(\mathcal{D}_1) = -(d_1 - b_1)$，用于衡量嵌入中最显著的一维孔洞的持久性

Data	图	拓扑损失函数	dim	$f_{\mathcal{S}}$	$n_{\mathcal{S}}$
合成周期	14b,14c, 27	$-(d_1 - b_1)$	1	0.4	5
细胞周期	15b、15c	$-(d_1 - b_1)$	1	0.25	10
细胞分叉	18a、18b、18c	$\sum_{k=2}^{\infty}(d_k - b_k) - [d_3 - b_3]_{\mathcal{E}_{\boldsymbol{B}}^{-1}]-\infty,0.75]}$	0 - 0	N/A	N/A
空手道	19b, 19c	$-(d_2 - b_2)$	0	0.25	10
哈利·波特	20b、20c	$-(d_1 - b_1)$	1	N/A	N/A

表 5：根据持久图计算的拓扑损失汇总 $\mathcal{D}$，其中点 (b_k, d_k) 按持久性排序 $d_k - b_k$。请注意，对于零维同调图 $d_1 = \infty$。



(a) 普通 PCA 嵌入。 (b) 拓扑优化嵌入。(c) 拓扑正则化嵌入。

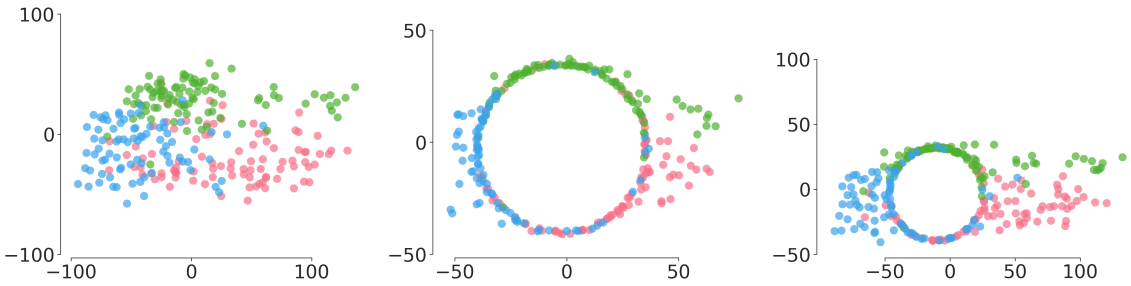
图 1 4：合成数据的表示，按真实坐标着色。

。由于我们的目标是让所有点都位于圆的边界上，因此我们在数据的子集上评估此损失（$f_{\mathcal{S}} = 0.4$, $n_{\mathcal{S}} = 5$）。所得嵌入如图 14c 所示，它更好地呈现了圆形孔洞。作为对比，图 14b 展示了未经重构损失$\mathcal{L}_{\text{emb}}$ 优化嵌入（以主成分分析为初始值），其中的孔洞更为显著。

圆形细胞轨迹数据 为了改进单细胞轨迹中圆形模型的表示，我们使用与合成数据相同的损失，但采用不同的采样参数（$f_{\mathcal{S}} = 0.25$ 和 $n_{\mathcal{S}} = 10$），因为单细胞数据集的规模更大。

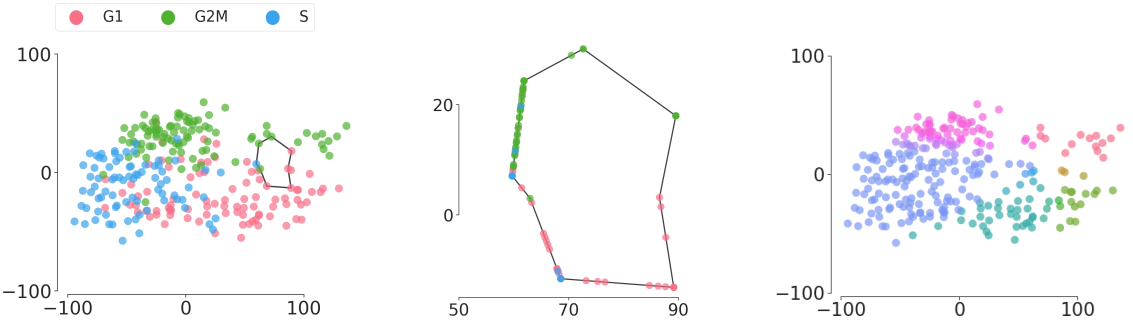
从图 15a 可以看出，普通 PCA 嵌入虽然在一定程度上保留了细胞群的位置关系（以颜色标记），但确实难以将数据嵌入成能够呈现我们已知存在于数据中的环的形式。按照图 15c 所示对嵌入进行拓扑正则化后，我们能够以圆形方式嵌入数据。与图 15b 中的优化嵌入相比，落在圆外的细胞更多。

细胞轨迹数据中的伪时间推断 单细胞组学涵盖在细胞层面收集的各种类型的数据，例如转录组学、蛋白质组学和表观基因组学。



(a) 普通 PCA 嵌入。 (b) Top. 优化嵌入。(c) Top. 正则化嵌入。

图 15：周期性细胞数据的表示，颜色表示细胞组。



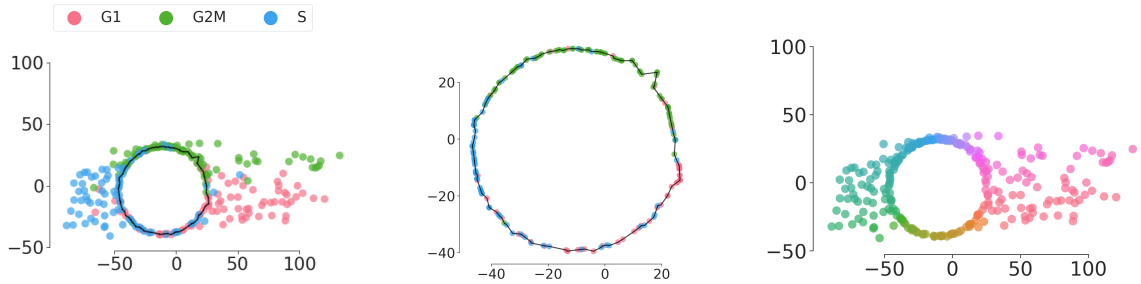
(a) 通过持久同调获得的显著环的表示。(b) 将嵌入数据正交投影到周期表示上。(c) 以连续色标量化显示 (b) 中投影所推断的伪时间。

图 16：通过持久同调，从数据的 PCA嵌入中自动推断真实细胞周期数据的伪时间。

研究数据背后的拓扑模型，或有助于更好地理解生物细胞的动态过程及其中涉及的调控相互作用。这些动态过程可以通过轨迹推断（TI）方法建模，这类方法也称为伪时间分析，它们根据细胞表达模式的相似性，将细胞沿轨迹排序（Saelens 等人，2019年）。

例如，细胞周期是一种广为人知的生物分化模型，描述细胞生长并分裂的过程。细胞周期由不同阶段组成，分别为生长（G1）、DNA 合成（S）、生长和有丝分裂准备（G2）以及有丝分裂（M）。后两个阶段通常合称为 G2M 阶段。因此，通过研究参与细胞周期分化模型的细胞的表达数据，可以识别参与细胞周期特定阶段及阶段间过程的基因（Liu 等人，2017年）。伪时间分析通过为每个细胞分配其在分化过程中所处的时间来实现这类研究，从而确定所有细胞在细胞周期模型中的相对位置。

分析单细胞周期数据是一种可利用先验拓扑信息的应用场景。由于高维表达数据中的信噪比通常较低（Libralon 等人，2009年；Zhang 等人，2021年），因此通常会先通过降维方法对这些数据进行预处理，再进行自



(a) 通过持续同调获得的最显著环的表示。(b) 将嵌入数据正交投影到环表示上。(c) 将 (b) 中投影的伪时间以连续色标量化。

图 17：从数据的拓扑正则化主成分分析嵌入中，通过持续同调自动推断真实细胞周期数据的伪时间。

动伪时间推断（Cannoodt 等人，2016年；Saelens 等人，2019年）。拓扑正则化可在嵌入过程中增强所需的拓扑信号，从而促进依赖于该信噪比的自动推断。

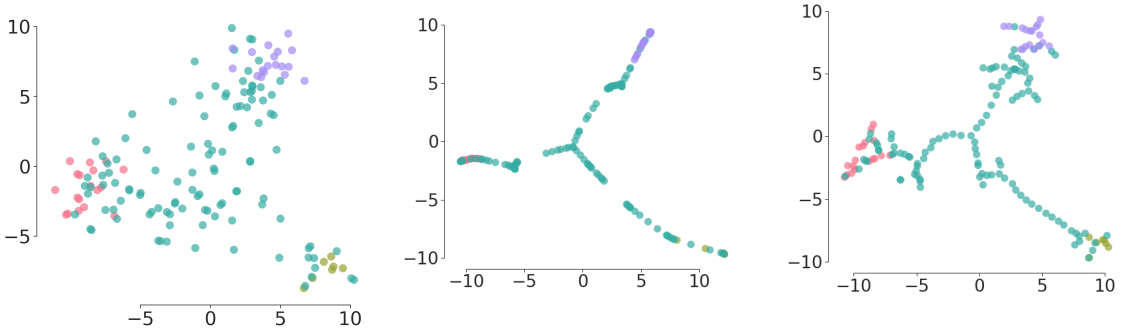
为说明这一点，我们使用持续同调开展自动（细胞）周期和伪时间推断，并比较了在数据的主成分分析嵌入过程中使用和不使用拓扑正则化的情况。在此实验中，我们使用了 Buettner 等人（2015年）也分析过的细胞周期数据的主成分分析嵌入和拓扑正则化嵌入。我们的自动伪时间推断方法包括以下步骤。

1. 首先，使用 Python 软件库 Dionysus（https://pypi.org/project/dionysus/），通过 α-过滤从嵌入中提取持续时间最长的圈的表示。它可以看作一种圆形表示——由数据点之间的边离散化而成——对应于一维持久图中持续时间最长的圈的点（图 16a 和 17a）。

2. 将嵌入数据正交投影到代表性环上。这是从离散化的拓扑表示推导连续伪时间的中间步骤，正如 Saelens 等人（2019年）先前所述（图 16b 和 17b）。

3. 利用正交投影上相邻点之间的距离，得到介于 0 和 2π 之间的环形伪时间（图 16c 和 17c）。

需要注意的是，在本文的范围内，我们并不主张将其视为一种遵循细胞周期模型、适用于单细胞数据的新型通用自动伪时间推断方法。然而，我们选择这一特定方法，是因为它很好地展示了持久同调能够在数据中识别出什么，以及我们旨在通过拓扑正则化优化什么。



(a) 普通 UMAP 嵌入。(b) 拓扑优化嵌入。(c) 拓扑正则化嵌入。

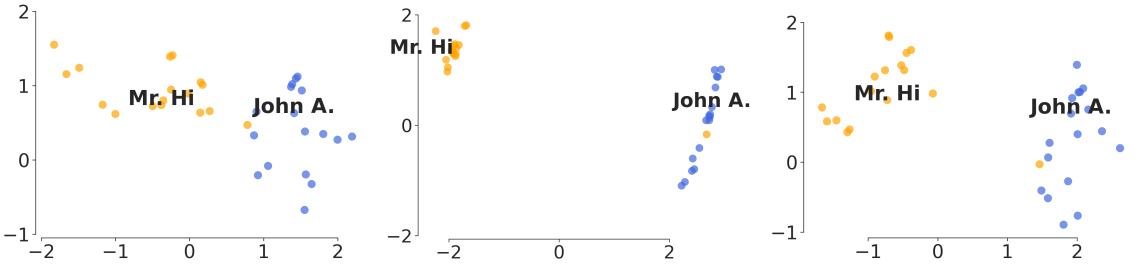
图 18: 以颜色表示细胞群的分叉细胞数据嵌入 up.

例如，我们观察到，普通 PCA 嵌入中最显著的环（的表示）相当虚假，主要以线性方式将 G1 和 G2M 细胞分开。将细胞投影到该环的边上，会使大多数细胞落在同一条边上（图 16b）。投影到这条边上的细胞，其所得伪时间是连续的，而对于所有其他细胞，伪时间则主要是离散的（图 16c）。然而，通过将拓扑先验知识纳入 PCA 嵌入，嵌入中最显著的环（的表示）如今能够更好地刻画细胞周期模型中 G1、G2M 和 S 期之间的转变（图 17a）。用于伪时间推断的自动化过程也反映出细胞阶段之间更为连续的转变（图 17b 和图 17c）。

分叉细胞轨迹数据 我们使用 UMAP 损失来说明拓扑正则化对该单细胞数据集的分叉模型的影响。拓扑损失为 $\mathcal{L}_{\text{top}} \equiv \mathcal{L}_{\text{conn}} + \mathcal{L}_{\text{flare}}$，其中 $\mathcal{L}_{\text{conn}}(\mathcal{D}_0) = \sum_{k=2}^{\infty}(d_k - b_k)$ 衡量嵌入中的有限零维持久性的总量（总和），以促进表示的连通性；$\mathcal{L}_{\text{flare}} = -[d_3 - b_3]_{\mathcal{E}_{\boldsymbol{E}}^{-1}]-\infty,0.75}$ 则优化一个远离嵌入均值、包含（至少）三个簇的“喇叭形”结构。

我们观察到，普通 UMAP 嵌入更加分散（图 18a），而拓扑正则化嵌入则更趋于受限于连通的分叉形状（图 18c）。作为比较，我们针对初始化后的 UMAP 嵌入，在不使用 UMAP 嵌入损失的情况下，对损失$\mathcal{L}_{\text{top}}$ 进行了拓扑优化。结果得到的嵌入现在更加碎片化（图 18b）。因此，我们发现拓扑优化也可能受益于嵌入损失。两种损失缺一不可：拓扑损失用于施加拓扑先验知识，嵌入损失则确保学习到的表示忠实于学习到的表示。

空手道 空手道网络由两个不同的社群组成，分别以其关键人物（John A. 和 Mr. Hi）为代表，如图 13d 所示。为了对图进行嵌入，我们使用了一个改编自 Dagar 等人（2020年）的 DeepWalk 变体。普通 DeepWalk 嵌入（图 19a）虽然很好地遵循了点按社群划分的排列顺序，



(a) 普通 DeepWalk 嵌入。 (b)拓扑优化嵌入。(c) 拓扑正则化嵌入。

图 19：空手道网络及其各种嵌入。

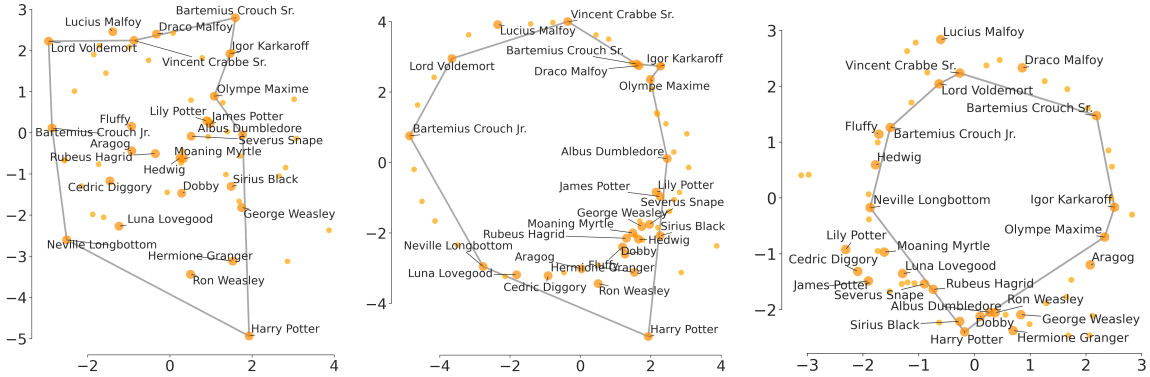
但这些社群仍然彼此靠近。因此，我们使用式（8）定义的带采样的拓扑损失，对该嵌入进行正则化，使其至少形成两个簇，其中$\mathcal{L}_{\text{top}}(\mathcal{D}_0) = -(d_2 - b_2)$衡量第二显著的 0 维孔洞的持久性，而 $f_{\mathcal{S}} = 0.25$, $n_{\mathcal{S}} = 10$。

得到的嵌入（图 19c）现在几乎完美地分开了图中存在的两个真实社群。使用相同的拓扑损失、但不包含 DeepWalk 损失，进一步优化普通的 DeepWalk 嵌入后，两个社群之间的分离程度更大（图 19b）。与正则化嵌入相比，两个社群内部可见的结构有所减弱，这再次表明同时纳入嵌入损失和拓扑损失的重要性。

哈利·波特 为了对《哈利·波特》图进行嵌入，我们使用了一个简单的图嵌入模型，其中嵌入节点之间内积的 Sigmoid函数值用于量化边出现的（伯努利）概率（Rendle 等人，2020年）。因此，嵌入中彼此接近的节点之间出现边的概率较高，而距离较远的节点之间则较低。随后，我们优化这些概率，使其与二元边指示向量相匹配。图 20a 展示了该嵌入的结果，以及 Vandaele 等人（2020b）提出的圆形模型。为清楚起见，角色标签仅标注在部分节点上（与 Vandaele 等人（2020b）标注的节点相同）。

为了更好地表示在“善良”与“邪恶”人物之间转换的圆形模型，我们使用一种拓扑损失函数 $\mathcal{L}_{\text{top}}(\mathcal{D}_1) = -(d_1 - b_1)$对嵌入进行正则化，该函数用于衡量最显著的一维孔洞的持久性。所得嵌入如图20c所示。有趣的是，经过拓扑正则化的嵌入现在能够更好地捕捉 Vandaele 等人（2020b）所识别模型的圆形性，并且更侧重于沿着该模型分布人物。请注意，尽管此前识别出的模型包含在可视化图中，但它既未用于推导嵌入，也不是由嵌入推导而来。作为比较，图20b展示了对普通图嵌入（用作初始化）进行优化以适应相同拓扑损失、但不包含图嵌入损失的结果。

我们得出结论，拓扑正则化确实成功地在嵌入中施加了指定的结构。



(a) 普通图嵌入。(b) 拓扑优化嵌入。（以普通图嵌入初始化）(c) 拓扑正则化嵌入。

图 20：《哈利·波特》图及其中圆形模型的各种嵌入。

4.2.2 Quantitative Evaluation

本节将分析前述实验中的嵌入损失和拓扑损失，并通过量化不同嵌入的预测性能变化来探讨问题 Q2。

表 6 总结了我们得到的普通嵌入、拓扑优化嵌入（以普通嵌入初始化，但未使用嵌入损失）以及拓扑正则化嵌入的损失。正如预期，拓扑正则化使嵌入损失处于普通嵌入与拓扑优化嵌入的嵌入损失之间。我们还发现，与正则化前后的嵌入损失相比，得到的拓扑损失差异更为显著。这表明，相对于拓扑损失对应的区域而言，嵌入损失的最优区域可能较为平坦。因此，拓扑正则化对局部嵌入最优解的轻微偏移，可能会带来更好的拓扑嵌入模型。

我们通过评估第4.2.1节中展示的嵌入可视化在预测真实数据标签方面的信息量，来评估其质量。对于合成周期数据，这些标签是无噪声数据在 $\mathbb{R}^2$单位圆上的二维坐标，我们使用了多输出支持向量回归器模型。对于细胞轨迹数据和空手道网络，我们分别使用真实的细胞分组和社区划分，并采用支持向量机模型。随后，我们将二维嵌入中的点划分为训练集（90%）和测试集（10%）；合成周期数据集除外，其划分比例为80/20。接着，我们在训练数据上进行5折交叉验证，以调整正则化超参数 $C \in \{1e-2, 1e-1, 1, 1e1, 1e2\}$。其他设置均采用 scikit-learn的默认值。随后，我们在测试数据上评估最终调优并训练完成的模型的性能：回归问题使用 r^2 决定系数，所有分类问题则使用准确率。

	嵌入损失			拓扑损失		
Data	ord.	拓扑优化	拓扑正则化	ord.	拓扑优化	拓扑正则化
合成周期	**0.0632**	0.0639	0.0634	−0.42	**−1.63**	−1.38
细胞周期	**6.70**	7.00	6.78	−10.18	**−68.34**	−63.52
细胞分叉	**8496**	9865	8896	116.03	**24.03**	63.90
空手道	**2006**	3239	2112	−0.59	**−4.58**	−1.82
哈利·波特	**0.20**	4.25	0.23	−0.82	**−5.84**	−3.05

表6：最终嵌入的嵌入/重构损失和拓扑损失。最低值以粗体标出。如果拓扑损失函数是在数据的随机子集上计算的，则报告其在完整数据集上的损失。

Data	度量指标	普通嵌入	拓扑优化	拓扑正则化
合成周期	r^2	0.62 ± 0.15	0.60 ± 0.14	$\mathbf{0.67 \pm 0.20}$
细胞周期	准确率	$\mathbf{0.79 \pm 0.07}$	$\mathbf{0.79 \pm 0.07}$	0.78 ± 0.07
细胞分叉	准确率	0.78 ± 0.08	$\mathbf{0.85 \pm 0.07}$	0.82 ± 0.08
空手道	准确率	$\mathbf{0.97 \pm 0.08}$	$\mathbf{0.97 \pm 0.08}$	$\mathbf{0.97 \pm 0.08}$

表 7：用于标签预测的嵌入性能评估。最高值以粗体显示。

表 7 汇总了对 100 次随机训练-测试划分的测试性能指标取平均后得到的结果及其标准差。我们观察到，普通嵌入、优化后嵌入和正则化嵌入之间的定量差异很小。然而，使嵌入与预期拓扑保持一致似乎并不会降低预测性能。

4.2.3 Scalability

为了展示拓扑正则化的实际运行时间，我们使用圆损失（不使用嵌入损失），对一个大小为 n 的随机二维数据集进行了 100 次迭代（学习率为 0.01）。在二维嵌入中，圆是计算上最难优化的拓扑孔洞。实验在一台 Dell Latitude 5511 笔记本电脑上进行，该电脑配备 Intel(R) Core(TM) i7-10850H 2.70GHz 处理器和 16.0 GB 内存。图21以双对数图的形式展示了不进行采样时的运行时间。使用采样计算拓扑损失，采用 $f_{\mathcal{S}} = 0.1$ 和 $n_{\mathcal{S}} = 10$ 时，分别耗时 1.9,17,191, 和 2284 秒。这与不进行采样时 100 次迭代的运行时间相当，但要达到类似结果，我们可能需要更少的迭代次数。这是因为一维孔洞的拓扑损失只有在 4 个点上才具有非零梯度（见第2节 2.5.1）。通过聚合 $n_{\mathcal{S}}$ 个子集的损失，拓扑损失的梯度最多可影响 $n_{\mathcal{S}} \cdot 4$ 个点。

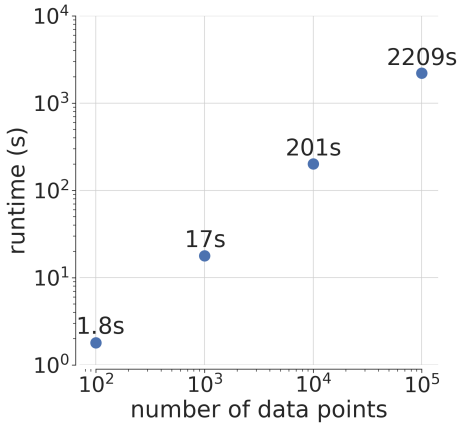


图21：展示数据集大小与在不进行采样的情况下，针对一维孔洞进行100次迭代优化所需运行时间之间关系的双对数图。

4.3 拓扑正则化的鲁棒性

在本节中，我们将进一步直观地探讨拓扑正则化的局限性，并尝试回答以下问题：

Q4. 拓扑损失的具体形式会如何影响最终嵌入？

Q5. 较低的信噪比会如何影响优化？

Q6. 使用错误的先验信息进行正则化会产生什么影响？

本节所介绍的用于正则化嵌入的拓扑损失函数概览见表8。

4.3.1 Varying the Parameters of Topological Priors

在第3节中，我们介绍了带采样的拓扑损失函数（见公式（8））以及用于耀斑形状的拓扑损失函数（见公式（9））。这里我们考察拓扑正则化对这些损失函数中所用超参数的不同取值有何反应。我们改变公式（8）中的采样比例 $f_{\mathcal{S}}$ 和重复次数 $n_{\mathcal{S}}$ 、公式（9）中的函数阈值 τ ，以及用于评估持久图中点的函数。请注意，尽管更改这些参数可能会改变所指定的几何 特征，但相对于所选的拓扑损失函数，拓扑 特征基本保持不变。

变化 $f_{\mathcal{S}}$ 和 $n_{\mathcal{S}}$ 图22展示了在不同取值 $f_{\mathcal{S}}$ 和 $n_{\mathcal{S}}$下，合成周期的拓扑正则化主成分分析嵌入。我们再次针对一维孔洞进行优化（见表 8），并采用表4中所述的相同参数设置。我们选择重复次数 $n_{\mathcal{S}}$ ，使计算嵌入所需的运行时间大致相同。我们观察到，增大 $f_{\mathcal{S}}$后，孔洞往往会变得更加明显，且边界内没有点。更具体地说，人们可以

Data	Fig.	拓扑损失函数	Dim	$f_{\mathcal{S}}, n_{\mathcal{S}}$
合成周期	22	$-(d_1 - b_1)$	1	0.25, 4 0.5, 2 1, 1
随机合成周期	27, 25	$-(d_1 - b_1)$	1	0.25, 4
合成周期	26b	$-(d_2 - b_2)$	1	NA
合成周期	26c	$\sum_{k=2}^{\infty} d_k$	0	NA
细胞分叉	23	$\sum_{k=2}^{\infty}(d_k - b_k)^{p_{\mathrm{CC}}} - [(d_3 - b_3)^{p_{\mathrm{Flare}}}]_{\mathcal{E}_{\boldsymbol{E}}^{-1}]-\infty,0.75]}$	0 - 0	NA
细胞分叉	24	$\sum_{k=2}^{\infty}(d_k - b_k) - [d_3 - b_3]_{\mathcal{E}_{\boldsymbol{E}}^{-1}]-\infty,\tau]}$	0 - 0	NA
细胞分叉	28	$-(d_1 - b_1)$	1	0.25, 10

表 8：第4.3节中用于正则化嵌入的拓扑损失汇总。请注意，对于零维同调图 $d_1 = \infty$。

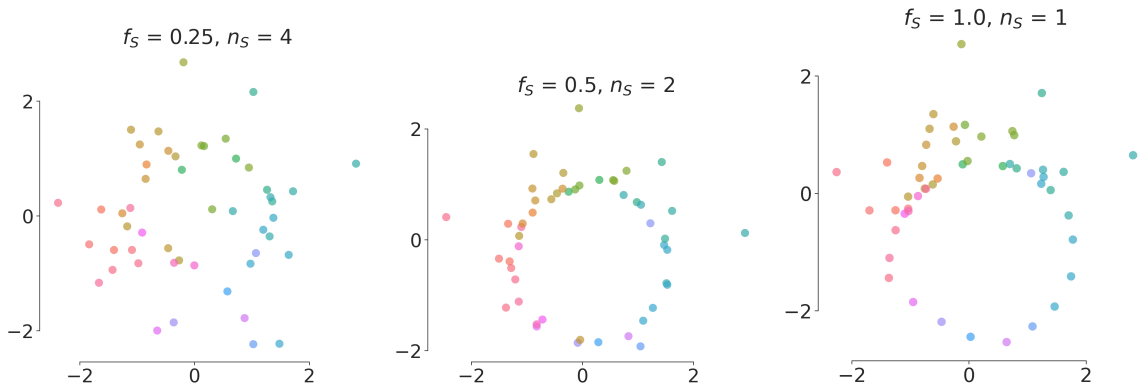


图22：合成周期数据集在不同采样比例 $f_{\mathcal{S}}$ 和重复次数 $n_{\mathcal{S}}$下的拓扑正则化PCA嵌入。

更直观地推断出数据的一个子集，该子集在欧几里得平面上构成并位于一个近乎完美的圆上。尽管如此，我们也注意到，在不进行采样的情况下（$f_{\mathcal{S}} = 1$），嵌入后的圆并未与真实值完美对齐，且只有部分数据位于其上。我们得出结论，采样比例应足够小，以避免优化伪洞，同时又要足够大，以确保感兴趣的 d维特征存在且稳定。在此示例中，使用 $f_{\mathcal{S}} = 0.25$ 进行正则化并不稳定，因为嵌入中的最大洞会因中心的两个点是否被采入样本而发生变化。

相同拓扑先验的不同损失函数（变化 g）对于某一特定类型的拓扑信息先验，可以通过改变损失函数 (4) 中使用的、在持久图中的点上计算的实值函数 g，来设计不同的拓扑损失函数。具体而言，我们的拓扑损失函数 (7)（其中令 $g \coloneqq b_k - d_k$）受到拓扑损失函数的启发

$$\mathcal{L}_{\mathrm{top}}(\boldsymbol{E}) \coloneqq \mathcal{L}_{\mathrm{top}}(\mathcal{D}) = \mu \sum_{k=i, d_k<\infty}^{|\mathcal{D}^{\mathrm{reg}}|} (d_k - b_k)^p \left(\frac{d_k + b_k}{2}\right)^q, \qquad \text{where } d_1 - b_1 \geq d_2 - b_2 \geq \ldots, \tag{12}$$

该损失函数由 Br¨uel-Gabrielsson 等人（2020年）提出，并由 Adcock 等人（2013）在代数框架中进行了更正式的研究。此处，p 和 q 是控制惩罚强度的超参数，而 i 以及持久图（或同调维数）的选择则用于设定感兴趣的拓扑信息。$\mu \in \{1, -1\}$ 的选择决定了人们希望增加（$\mu = -1$）还是减少（$\mu = 1$）为数据中的拓扑损失函数所设计的拓扑信息。术语 $(d_k - b_k)^p$ 可用于控制最显著拓扑孔洞的显著程度，即其持久性；而术语 $\left(\frac{d_k+b_k}{2}\right)^q$ 可用于控制显著拓扑特征是在过滤的早期还是较晚阶段持续存在。

我们的拓扑损失函数 (7) 与 (12) 相同，其中 $p = 1$ 和 $q = 0$，另外，我们对超参数 j 求和，而不是对 $|\mathcal{D}^{\mathrm{reg}}|$求和，这使得先验拓扑信息更加灵活。请注意，这是提出拓扑损失函数的一种最简单、最直观且灵活的方式。事实上，$(d_k - b_k)^{p=1}$直接衡量了所选感兴趣维度中最显著的第 k 个拓扑孔洞的持久性。$q \neq 0$ 的作用仍有待在拓扑正则化及先验拓扑信息的构建背景下进一步研究。

或许已经很清楚，可以通过对 $p > 0$作出不同选择来提出相同的拓扑信息。为探究这一点，我们针对真实的分叉单细胞数据，考察了拓扑正则化 UMAP 嵌入，并采用表 8 中所述的拓扑损失函数，同时改变参数 p，即

$$\mathcal{L}_{\mathrm{top}}(\boldsymbol{E}) = \sum_{k=2}^{\infty} (d_k - b_k)^{p_{\mathrm{CC}}} - \left[(d_3 - b_3)^{p_{\mathrm{Flare}}}\right]_{\mathcal{E}_{\boldsymbol{E}}^{-1}]-\infty, 0.75]}. \tag{13}$$

我们只改变其中一个指数，即 p_{CC} 或 p_{Flare}，以分别研究它们的影响，并在图 23 中展示嵌入。

我们观察到，较小的 p_{CC} 值会导致许多点相互重叠，因为即使是最小的间隙也会对损失产生显著影响。对于 $p_{CC} = 2$，不再有较大的间隙，所有点与其最近邻的距离也大致相同。随着 p_{Flare} 值增大，距离中心足够远的点会被拉向所表示的拓扑模型的叶端。这会提高三个簇的持久性。不过，分叉中心有少数点会使所有点保持连通，这正是拓扑损失第一部分正则化的结果。尽管如此，虽然不同的 p 值似乎会影响点之间的整体间距，但我们发现，嵌入后的拓扑形状整体上仍清晰可辨且保持一致。

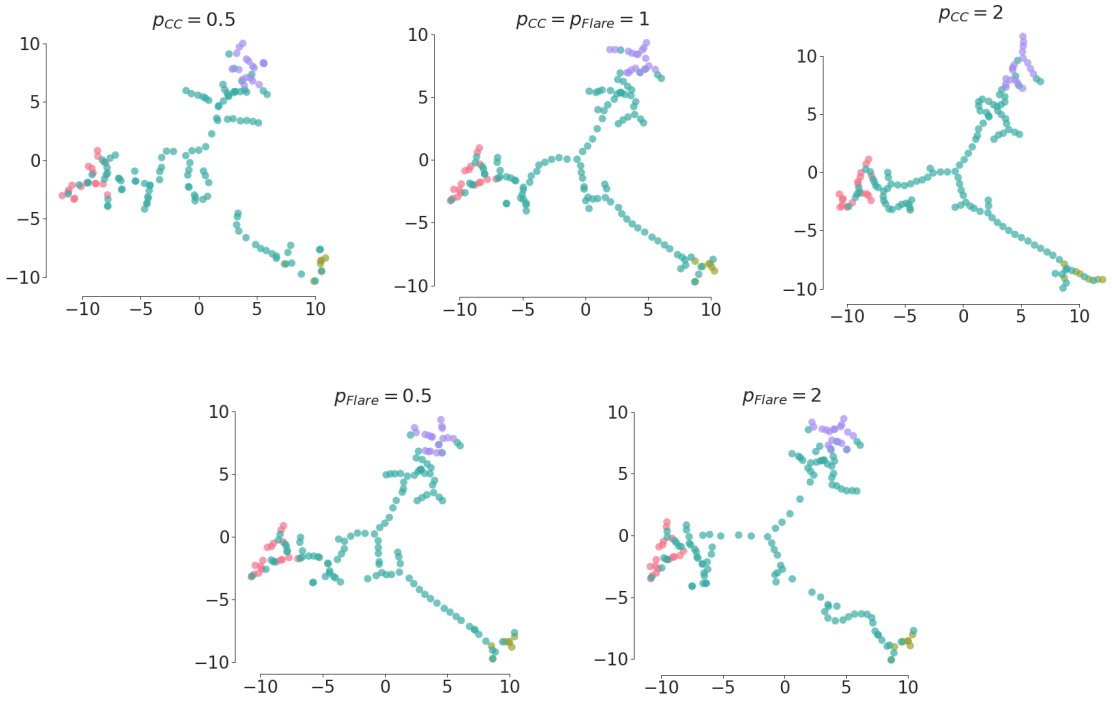


图 23：根据（13）得到的分叉型真实单细胞数据的拓扑正则化 UMAP 嵌入，适用于不同的 p 值。

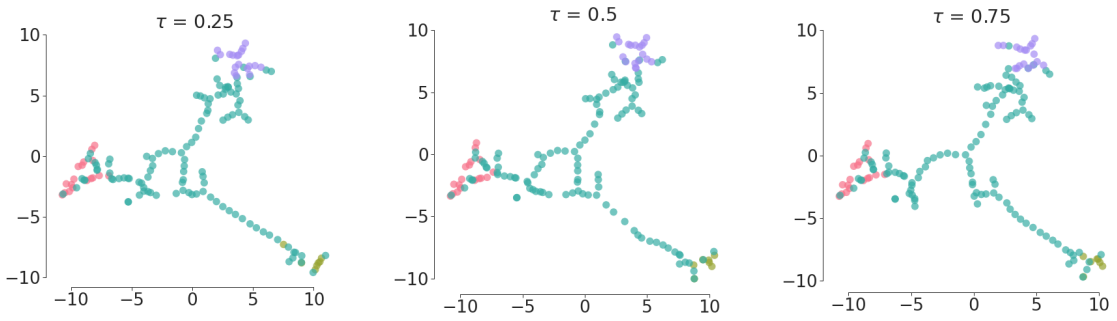


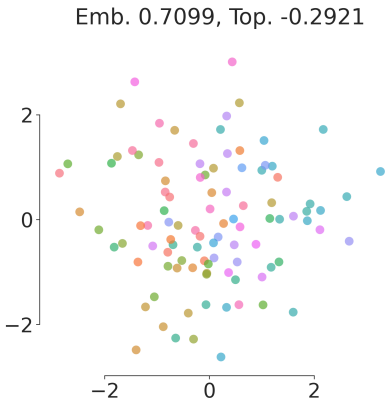
图 24：针对不同函数阈值 τ，分叉型真实单细胞数据的拓扑正则化 UMAP 嵌入。

变化 τ 图 24 展示了遵循分叉型细胞分化模型的真实单细胞数据的拓扑正则化 UMAP 嵌入，以及函数阈值 τ 取不同值时的情况。回顾一下，$0 \leq \tau \leq 1$ 是嵌入点到其均值的归一化距离阈值，定义见式 (10)。直观而言，τ 指定了考察范围与嵌入均值的接近程度；τ 越大，在优化远离中心的三个簇时，纳入的点就越多（因而也越接近嵌入均值）。$\tau = 0.25$ 和 $\tau = 0.5$ 之间几乎没有差异，但对于 $\tau = 0.75$，嵌入模型中靠近分叉中心的点会更明显地向端点延展。这与以下事实有关：τ 越大，在优化三个分离簇时，纳入的靠近中心的点就越多。

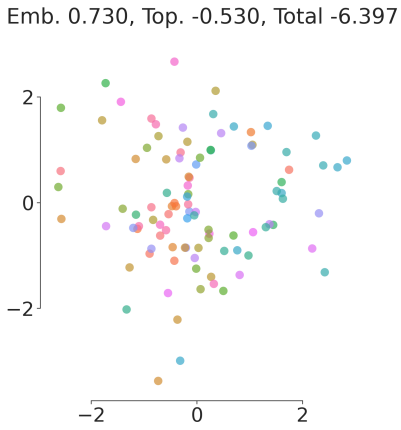
4.3.2 Regularization with Weak Signal to Noise Ratio

在整个实验中使用的合成数据里，按照构造方式，环位于方差最高的维度中。因此，采用主成分分析进行嵌入初始化时，已经在一定程度上与底层的圆形结构对齐。即使采用不同的初始化，主成分分析（均方误差）损失也会促使结果达到这一配置。我们研究了一种包含 10 维合成数据集（n=100）的设置，其中环仍位于前两个维度，但将所有维度标准化为具有相同的方差。

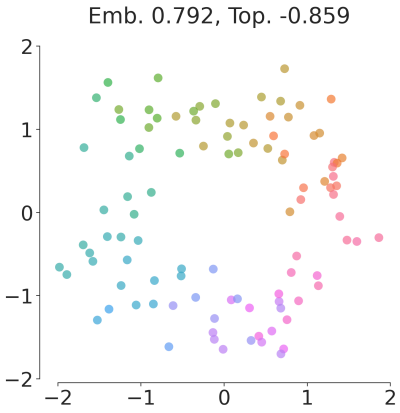
该数据的 PCA投影（图 25a）不一定与前两个维度对齐，也无法显示出圆。以此 PCA投影初始化的拓扑正则化会产生一个带有伪孔洞的嵌入（图 25b）。前两个维度上的投影及其拓扑正则化优化（图 25c 和图 25d）由于拓扑损失较低，目标值都更优。看来，25b 中使用 PCA 初始化得到的拓扑正则化嵌入是损失的一个局部最优，而当前的局部优化技术无法跳出这一局部最优。由于仅使用前两个维度的投影的总损失显著更低，而以真实值初始化的正则化嵌入的总损失更低，因此我们推测，如果采用不同的优化技术或更智能的初始化方式，或许能够



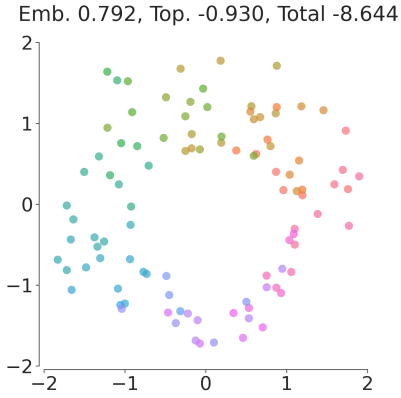
(a) PCA投影。



(b) 以25a初始化的拓扑正则化嵌入。



(c) 前两个
数据维度的可视化。



(d) 拓扑正则化嵌入——
在图 25c 中初始化的嵌入。

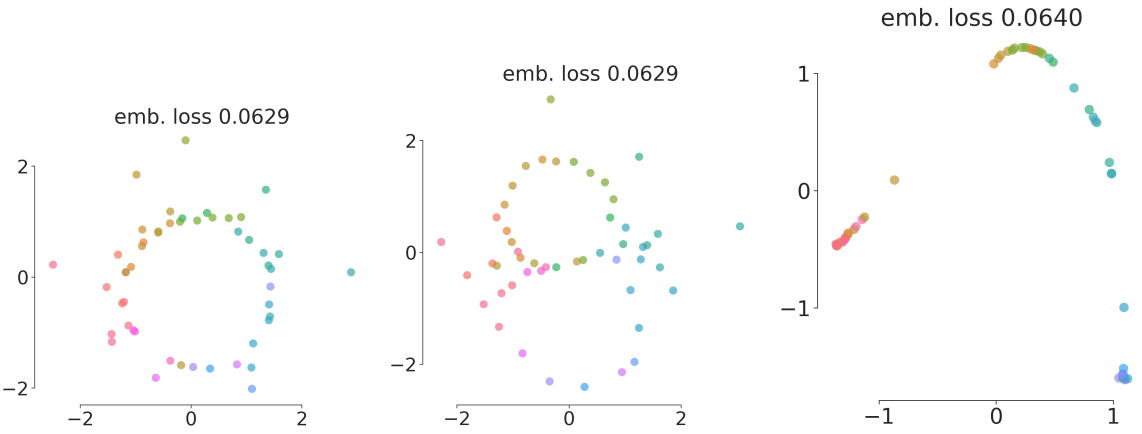
图 25：对 (b) 和 (d) 中的 (n=100, d=10) z分数标准化数据进行正则化嵌入，分别以 PCA 投影 (a) 和对应真实圆的前两个维度 (c) 作为初始化。总目标函数计算为 $\mathcal{L}_{\text{emb}} + 10 \cdot \mathcal{L}_{\text{top}}$。

在这一具有挑战性的情形下仍然识别出正确的环。然而，这也凸显出，在更困难的情形下，要很好地优化该损失并不容易。

4.3.3 Optimizing for Different Topological Priors

由于用户可能并不总是拥有可用的强先验专家拓扑知识，我们通过研究拓扑正则化对不同拓扑损失函数的响应来探究 Q6，这些损失函数并未对数据的真实拓扑结构进行建模。

合成周期的先验错误我们探讨两种不同的拓扑损失函数对合成周期嵌入的正则化效果。对于这个数据集，我们知道



(a) 一个圆　(b) 两个大小相同的圆　(c) 连通分支

图 26：真实模型为圆的 500 维合成数据的拓扑正则化嵌入。为便于直观比较，我们在 (a) 中加入了 14c 中已展示的嵌入。

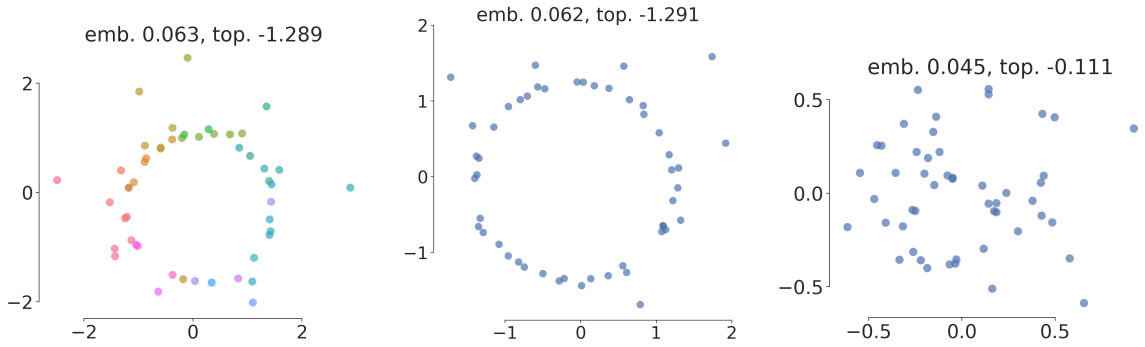
恰好有一个拓扑模型（即圆）能够生成这些数据。我们使用与之前相同的参数计算正则化主成分分析嵌入（见表4），并在表 8 中汇总损失函数。

图 26b 展示了通过拓扑损失正则化嵌入的结果，该损失旨在最大化第二显著环的持久性。我们认为，这种损失指定了一种部分错误的先验信息，因为它也要求嵌入中最显著环的持久性较高。结果是两个环具有相近的持久性。图 26c 所示的嵌入则通过拓扑损失进行正则化，以最小化所有有限的（0 维）死亡时间，从而缩短邻近点之间的距离。为便于比较，我们还展示了使用正确的圆形先验正则化的嵌入（26a）。

这些实验表明，即使采用错误的拓扑先验，也可以优化嵌入。在这种情况下，具有一个圆和两个圆的嵌入，其重构误差非常相似，无法据此判断哪一种是正确的。因此，必须谨慎解读这些结果：通常不能将所施加拓扑信息在视觉上的呈现视为数据中确实存在这种拓扑结构的证据。

随机高维数据中的拓扑模型 在前一个实验中，合成数据集相对于其维数而言非常小（n=50、d=500），这使投影拥有很大的灵活性：它可以改变某个点的低维嵌入，而不影响其他点。因此，该实验的结果并不令人意外，而一个更有意思的问题是在什么情况下拓扑正则化会产生呈现伪拓扑的低维嵌入。

我们并不试图穷尽所有情况，而是通过对大小和维数相同（n=50, d=500）的随机高维数据集进行拓扑正则化，来探究这个问题；



(a) 500 维合成包含一个环的数据。

(b) 500 维点没有圆形底层结构的点云真实值。

(c) 不具有圆形真实结构的10维点云。

图27：包含环的500维数据集的拓扑正则化嵌入，与采用相同拓扑损失进行正则化的500维和10维点云进行比较。为了在10维数据的投影中嵌入圆形结构，我们提高了拓扑损失的权重（$\lambda_{\text{top}} = 10$）。

该数据集的噪声量与合成周期相同，但不包含圆形信号。正则化所用的拓扑损失函数也与表 8 中所述相同。我们观察到，或许并不令人意外，对随机数据的嵌入进行正则化（图 27b）后，得到的圆形在视觉上与合成周期的嵌入相似。然而，当将数据的维数降至 d=10 时，环的形态就不那么明显了（图 27c），即使采用更高的拓扑损失权重（λ_{top} =10）也是如此。

这些实验表明，当数据的维度数相对于数据点数量较高时，无论数据中是否确实存在所指定的拓扑结构，结果都可能呈现该结构。即使拓扑损失的权重较小，也可能发生这种情况。然而，当数据点数量相对于维度数足够大时，拓扑正则化受此影响的程度较低。尽管这些结果对大多数读者来说可能并不令人意外，但我们认为有必要强调，拓扑正则化嵌入不能也不应被用作检验关于数据的某种先验是否正确的方法。

分叉单细胞数据上的圆形先验在前面的实验中，我们展示了在不同设置下，使用错误的拓扑先验对正交投影进行正则化所产生的影响。这里，我们考察这种正则化对 UMAP 非线性嵌入的影响。我们认为，由于点坐标是在嵌入空间中直接优化的，因此这些嵌入可能更容易对错误的先验拓扑信息进行建模。无论输入数据的维数如何，这都提供了很大的灵活性。为探究这一点，我们通过拓扑损失函数–($d_1 - b_1$)，使用 $f_{\mathcal{S}} = 0.25$ 和 $n_{\mathcal{S}} = 10$，在第一维持久图上进行评估，将真实的分叉单细胞数据嵌入，以采用圆形先验。

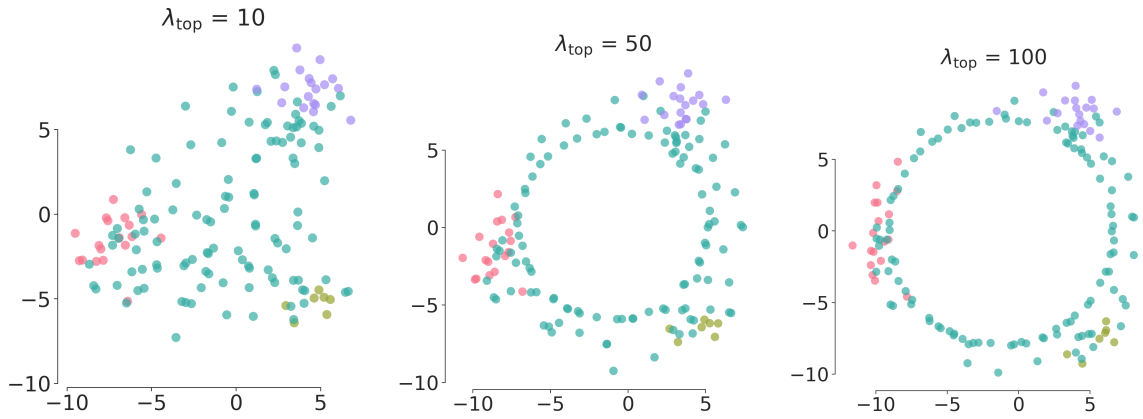


图 28：使用可能错误的先验拓扑（圆形）信息，对分叉细胞数据进行拓扑正则化 UMAP 嵌入。我们展示了采用不同拓扑正则化强度 λ_{top}、经过 250 个训练轮次优化得到的嵌入。

图 28 展示了针对不同拓扑正则化强度 λ_{top}、经过 250 个训练轮次优化的圆形先验拓扑正则化 UMAP 嵌入。显然，拓扑正则化强度 λ_{top} 越高，对数据嵌入中拓扑表示的影响就越显著。我们观察到，使用 $\lambda_{\text{top}} = 10$ 时，UMAP 损失会阻止嵌入呈现孔洞。正则化强度为 $\lambda_{\text{top}} = 50$ 时，我们观察到正则化嵌入中出现了圆形。使用 $\lambda_{\text{top}} = 100$ 时，圆的半径增大，但分叉端点处的点大多位于圆外。

回到研究问题Q6，我们得出结论：拓扑正则化不能也不应被用于检验数据集是否包含某种特定的拓扑结构。我们证明，尤其是对于样本较少的高维数据，以及 UMAP 这类灵活的嵌入方法，可以优化出任意结构。此外，不应根据拓扑损失函数需要多大的λ_{top} 才能对嵌入产生可见影响，来评估其适用性。该数值主要取决于嵌入损失与拓扑损失之间的相对差异。因此，在缺乏可靠的拓扑先验信息时，应保持谨慎。

5. 结论

我们提出了一种名为拓扑正则化的新型表征学习方法，该方法建立在近期开发的拓扑优化微分框架之上。为了让拓扑正则化尽可能易于理解和使用，我们全面介绍了持久同调所需的最低限度理论基础。我们的方法为依据专家拓扑先验知识嵌入数据提供了一种灵活而有效的途径，这些先验知识通过我们新提出的拓扑损失函数直接表述。所提出的拓扑损失函数能够对广泛的全局和局部拓扑结构进行建模。此外，通过将采样方法与这些损失函数结合，我们克服了一项重要的鲁棒性

挑战。实验表明，除最困难的情形（低信噪比）外，拓扑正则化在增强所施加拓扑结构的显著性方面非常有效。我们还表明，纳入拓扑先验知识为改进后续学习任务（即使是非拓扑学习任务）提供了一种有前景的方法（另见第4节 4.2.1）。

6. 未来工作

我们的实验表明，未来工作有多个方向。首先，拓扑正则化的一个明显局限在于，拓扑先验知识并非总是可用。因此，如何从一系列先验中选出最佳先验，仍有待进一步研究（另见第4.3.3节）。一个有趣的问题是，能否通过将先验知识与高维数据推断出的（拓扑）特征相结合，或部分从这些特征中推导出先验知识，来克服这一局限。

其次，为嵌入与拓扑损失之间的权衡参数设定一个合理范围，或许可以缓解这样的问题：当数据集中的数据点数量相较于维度数量较少，或使用 UMAP 等灵活的（非线性）嵌入方法时，几乎任何拓扑先验都能在嵌入中得到优化。研究如何为这一参数设定最佳值，或许能让拓扑正则化也可用于检验数据集是否包含指定的拓扑结构。

第三，对于信噪比较低的问题，可能需要更稳健的（非局部）非线性优化方法或更有针对性的初始化方案。正如我们在第4.3.2节所见，尽管目标值更低，我们对拓扑正则化主成分分析的概念验证实现仍未达到预期的配置。

第四，拓扑损失函数的设计仍可说相当复杂，因此研究如何让非专业用户更容易进行设计会很有价值。从理论基础的角度来看，我们的工作为研究如何将已发展出的拓扑优化理论（Carriere 等，2021）扩展到新引入的损失函数，并将其整合进数据嵌入，提供了新的机会。

最后，基于 α-复形以外的组合结构进行拓扑优化，可能具有理论和实践价值。例如，基于图近似（如最小生成树）进行点云优化（Vandaele 等，2021a），或者在损失函数（9）中改变函数阈值 τ，同时调整过滤时间（Chazal 等，2009），都可能产生超参数更少的自然拓扑损失函数。

致谢

产生这些成果的研究获得了欧洲研究委员会在欧盟第七框架计划（FP7/2007-2013）项下的资助（ERC 资助协议编号：615517），以及欧盟地平线2020研究与创新计划项下的资助（ERC 资助协议编号：963924）；同时还获得了根特大学特别研究基金（BOF）的资助（BOF20/IBF/117）、弗兰德斯政府“Onderzoeksprogramma Artifici¨ele I ntelligentie (AI) Vlaanderen”计划的资助，以及 FWO 的资助（项目编号：G0F9816N、3G042220、11J2322N）。

参考文献

Aaron Adcock、Erik Carlsson 和 Gunnar Carlsson。持久条形码上的代数函数环。*arXiv*预印本 *arXiv:1304.0530*，2013年。Charu C Aggarwal、Alexander Hinneburg 和 Daniel A Keim。高维空间中距离度量的令人惊讶的行为。载于数据库理论国际会议，第420–434页。Springer，2001年。Nina Amenta、Dominique Attali 和 Olivier Devillers。低维多面体上点的德劳内三角剖分的复杂性。载于*ACM-SIAM* 离散算法研讨会，第1106–1113页，2007年。Shuanghua Bai、Huo-Duo Qi 和 Naihua Xiu。球面上受约束的最佳欧几里得距离嵌入：一种矩阵优化方法。*SIAM* 优化学报，25 (1)：439–467，2015年。Jean-Daniel Boissonnat、Fr´ed´eric Chazal 和 Mariette Yvinec。几何与拓扑推断，第57卷。剑桥大学出版社，2018年。Rickard Br¨uel-Gabrielsson、Bradley J Nelson、Anjan Dwaraknath、Primoz Skraba、Leonidas J Guibas 和 Gunnar Carlsson。用于机器学习的拓扑层。载于人工智能与统计国际会议，第1553–1563页。PMLR，2020年。Florian Buettner、Kedar N Natarajan、F Paolo Casale、Valentina Proserpio、Antonio Scial- done、Fabian J Theis、Sarah A Teichmann、John C Marioni 和 Oliver Stegle。对单细胞 RNA 测序数据中细胞间异质性的计算分析揭示了隐藏的细胞亚群。《自然·生物技术》，33(2)：155–160，2015年。Robrecht Cannoodt、Wouter Saelens 和 Yvan Saeys。单细胞转录组学轨迹推断的计算方法。《欧洲免疫学杂志》，46(11)：2496–2506，2016年11月。ISSN 00142980。Robrecht Cannoodt、Wouter Saelens、Helena Todorov 和 Yvan Saeys。包含轨迹的单细胞组学数据集，2018年10月。URL https://doi.org/10.5281/ zenodo.1443566。Gunnar Carlsson。面向点云数据的拓扑模式识别。《数值学报》，23：289–368，2014年。Mathieu Carriere、Fr´ed´eric Chazal、Marc Glisse、Yuichi Ike、Hariprasad Kannan 和 Yuhei Umeda。优化基于持久同调的函数。载于国际机器学习会议，第1294–1303页。PMLR，2021年。Fr´ed´eric Chazal、David Cohen-Steiner、Leonidas J Guibas、Facundo M´emoli 和 Steve Y Oudot。使用持久性表示形状的 Gromov-Hausdorff 稳定签名。载于《计算机图形学论坛》，第28卷，第1393–1403页。Wiley 在线图书馆，2009年。

Chao Chen、Xiuyan Ni、Qinxun Bai 和 Yusu Wang。通过持久同调为分类器构建拓扑正则项。载于人工智能与统计国际会议，第 2573–2582 页。PMLR，2019年。Paolo Cignoni、Claudio Montani 和 Roberto Scopigno。Dewall：一种快速的分治德劳内三角剖分算法。载于计算机辅助设计，30(5):333–341，1998年。Frank H Clarke。优化与非光滑分析。SIAM，1990年。Ronald R Coifman、Stephane Lafon、Ann B Lee、Mauro Maggioni、Boaz Nadler、Frederick Warner 和 Steven W Zucker。几何扩散作为数据调和分析与结构定义的工具：扩散映射。美国国家科学院，102(21): 7426–7431，2005年。Etienne Cˆome、Marie Cottrell、Michel Verleysen 和 J´erˆome Lacaille。用于健康监测的自组织星（sos）。载于欧洲人工神经网络会议，，第 99–104 页。D Facto，布鲁塞尔，2010年。Michel Coste。*o*-极小几何导论。比萨国际出版与印刷学院，2000年。A. Dagar、A. Pant、S. Gupta 和 S. Chandel。图网络，2020年。网址：https://github. com/dsgiitr/graph_nets。Boris Delaunay 等人。空球问题。苏联科学院通报，数学与自然科学部，7(793-800):1–2，1934年。Olivier Devillers 和 Charles Dum´enil。光滑曲面的泊松采样是良好的采样。载于欧洲计算几何研讨会（*EuroCG*），2019年。Rex A Dwyer。线性期望时间内的高维沃罗诺伊图。离散与计算几何，6(3):343–367，1991年。Allen Hatcher。代数拓扑。剑桥大学出版社，2002。ISBN 0521795400。Ian T Jolliffe。回归分析中的主成分。载于主成分分析，第 129–155 页。Springer，1986年。Giampaolo Luiz Libralon、Andr´e Carlos Ponce de Leon Ferreira、Ana Carolina Lorena 等人。用于基因表达分类数据噪声检测的预处理。巴西计算机学会杂志，15(1):3–11，2009年。Zehua Liu、Huazhe Lou、Kaikun Xie、Hao Wang、Ning Chen、Oscar M Aparicio、Michael Q Zhang、Rui Jiang 和 Ting Chen。通过单细胞转录组数据重建细胞周期伪时间序列。《自然·通讯》，8(1):1–9，2017年。Cl´ement Maria。计算拓扑中的算法与数据结构。博士论文，Universit´e Nice Sophia Antipolis，2014。Leland McInnes、John Healy 和 James Melville。UMAP：用于降维的均匀流形逼近与投影。*arXiv*预印本 *arXiv:1802.03426*，2018年。

Geoffrey J McLachlan。判别分析与统计模式识别。John Wiley & Sons，2005年。Michael Moor、Max Horn、Bastian Rieck 和 Karsten Borgwardt。拓扑自编码器。载于国际机器学习会议，第 7045–7054 页。PMLR，2020年。Bradley J Nelson 和 Yuan Luo。通过交错优化实现保拓扑降维。*arXiv*预印本 *arXiv:2201.13012*，2022。Nina Otter、Mason A. Porter、Ulrike Tillmann、Peter Grindrod 和 Heather A. Harrington。持久同调计算路线图。*EPJ Data Science*，6(1):17，2017年8月。ISSN 2193-1127。Steve Y. Oudot。持久性理论：从箭图表示到数据分析。数学综述与专著第 209 卷。美国数学会，2015年。Bryan Perozzi、Rami Al-Rfou 和 Steven Skiena。DeepWalk：社会表征的在线学习。载于*ACM SIGKDD*知识发现与数据挖掘国际会议，第 701–710 页，2014。Chi Seng Pun、Kelin Xia 和 Si Xian Lee。基于持久同调的机器学习及其应用——综述。*arXiv*预印本 *arXiv:1811.00252*，2018年。Steffen Rendle、Walid Krichene、Li Zhang 和 John Anderson。重新审视神经协同过滤与矩阵分解。载于*ACM*推荐系统大会，第 240–248 页，2020年。Wouter Saelens、Robrecht Cannoodt、Helena Todorov 和 Yvan Saeys。单细胞轨迹推断方法比较。《自然·生物技术》，37:1，2019年4月。Yitzchak Solomon、Alexander Wagner 和 Paul Bendich。一种快速、稳健的全局拓扑函数优化方法。载于人工智能与统计国际会议，第 109–117 页。PMLR，2021年。Eashwar V Somasundaram、Shael E Brown、Adam Litzler、Jacob G Scott 和 Raoul R Wadhwa。R软件包持久同调计算基准测试。《*R*期刊》，2021年。SageMath 开发者。SageMath，2022年1月。网址：https://github.com/sagemath/sage。Csaba D Toth、Joseph O'Rourke 和 Jacob E Goodman。离散与计算几何手册。CRC出版社，2017年。James Townsend、Niklas Koep 和 Sebastian Weichwald。Pymanopt：使用自动微分进行流形优化的 Python工具箱。《机器学习研究期刊》，17(137):1–5，2016年。网址：http://jmlr.org/papers/v17/16-177.html。Lou van den Dries。驯性拓扑与 *o*-极小结构，第 248 卷。剑桥大学出版社，1998年。

Laurens Van der Maaten 和 Geoffrey Hinton。使用 t-SNE 可视化数据。《机器学习研究期刊》，9(11)，2008年。Robin Vandaele、Guillaume Adrien Nervo 和 Olivier Gevaert。用于目标检测和皮肤病变拓扑图像处理的拓扑图像修改。《科学报告》，10(1):21061，2020a。Robin Vandaele、Yvan Saeys 和 Tijl De Bie。通过森林表示挖掘图中的拓扑结构。《机器学习研究期刊》，21(215):1–68，2020b。Robin Vandaele、Bastian Rieck、Yvan Saeys 和 Tijl De Bie。通过图近似获取度量树的稳定拓扑签名。《模式识别快报》，147:85–92，2021a。Robin Vandaele、Yvan Saeys 和 Tijl De Bie。度量图上测地线的图近似。发表于国际模式识别会议 *(ICPR)*，第 7328–7334 页。IEEE，2021b。Robin Vandaele、Bo Kang、Tijl De Bie 和 Yvan Saeys。重新审视维数灾难：在含噪高维数据中，距离何时能为真实值提供信息？发表于人工智能与统计国际会议，第 2158–2172 页。PMLR，2022a。Robin Vandaele、Bo Kang、Jefrey Lijffijt、Tijl De Bie 和 Yvan Saeys。拓扑正则化数据嵌入。发表于国际学习表征会议，2022b。Alexander Wagner、Elchanan Solomon 和 Paul Bendich。利用分布式拓扑改进度量降维。*arXiv*预印本 *arXiv:2106.07613*，2021年。Junyuan Xie、Ross Girshick 和 Ali Farhadi。用于聚类分析的无监督深度嵌入。发表于国际机器学习会议，第 478–487 页。PMLR，2016年。W.W. Zachary。小群体中冲突与分裂的信息流模型。《人类学研究杂志》，33:452–473，1977年。Ruoyu Zhang、Gurinder S Atwal 和 Wei Keat Lim。噪声正则化消除了单细胞 RNA 测序数据预处理中的相关性伪影。《*Patterns*》，2(3):100211，2021年。