Deep learning and computational methods on single-cell and spatial data for precision medicine in oncology

Quentin Blampey

université PARIS-SACLAY

Deep learning and computational methods on single-cell and spatial data for precision medicine in oncology

Méthodes d'apprentissage profond et computationnelles pour la médecine de précision sur données spatiales et à résolution cellulaire

Thèse de doctorat de l'université Paris-Saclay

École doctorale n° 582, Cancérologie : biologie - médecine - santé (CBMS)
Spécialité de doctorat: Sciences du Cancer
Graduate School : Life Sciences and Health, Référent : Faculté de médecine

Thèse préparée dans les unités de recherche **MICS EA 4037 Mathématiques et Informatique pour la Complexité et les Systèmes** (Université Paris-Saclay, CentraleSupélec) et **Prédicteurs moléculaires et nouvelles cibles en oncologie** (Université Paris-Saclay, Institut Gustave Roussy, Inserm), sous la direction de **Paul-Henry Cournède**, Professeur, et la co-direction de **Fabrice André**, Professeur.

Thèse soutenue à Paris-Saclay, le 18 décembre 2024, par

Quentin BLAMPEY

Composition du jury

Membres du jury avec voix délibérative

Gilles Faÿ Professeur, Université Paris-Saclay	Président du jury
Oliver Stegle Professeur, DKFZ - EMBL Heidelberg	Rapporteur & Examinateur
Anaïs Baudot Professeur, CNRS - INSERM - Aix Marseille Université	Rapporteure & Examinatrice
Hannah Spitzer PhD, Helmholtz Center Munich	Examinatrice
Kevin Lebrigand PhD, IPMC - CNRS - Sophia-Antipolis	Examinateur

THESE DE DOCTORAT
NNT : 2024UPASL116

université PARIS-SACLAY | ÉCOLE DOCTORALE Cancérologie : biologie - médecine - santé (CBMS)

Titre: Méthodes d'apprentissage profond et computationnelles pour la médecine de précision sur données spatiales et à résolution cellulaire
Mots clés: Apprentissage profond, Modèles génératifs, Auto-supervision, Omique à résolution cellulaire, Omique spatiale, Oncologie de précision

Résumé: La médecine de précision en oncologie a pour but de personnaliser les traitements en fonction des profils génétiques et moléculaires uniques des tumeurs des patients, et ce, afin d'améliorer l'efficacité thérapeutique ou de minimiser les effets secondaires. À mesure que les avancées technologiques produisent des données de plus en plus précises sur le micro environnement tumoral, la complexité de ces données augmente également. Notamment, les données spatiales — un type récent et prometteur de données omiques — fournissent des informations moléculaires à la résolution de la cellule tout en conservant le contexte spatial des cellules au sein des tissus. Pour exploiter pleinement cette richesse et cette complexité, l'apprentissage profond émerge comme une approche capable de dépasser les limitations des approches traditionnelles. Ce manuscript détaille le développement de nouvelles méthodes de deep learning et computationnelles ayant pour but d'améliorer l'analyse des systèmes complexes des données single-cell et spatial. Trois outils sont décrits : (i) Scyan, pour l'annotation de types cellulaires en cytométrie, (ii) Sopa, une pipeline générale de preprocessing de données spatiales, et (iii) Novae, un modèle de fondation pour données spatiales. Ces méthodes sont appliquées à plusieurs projets de médecine de précision, approfondissant notre compréhension de la biologie du cancer et facilitant la découverte de nouveaux biomarqueurs et l'identification de cibles potentiellement actionnables pour la médecine de précision.

Title: Deep learning and computational methods on single-cell and spatial data for precision medicine in oncology
Keywords: Deep learning, Generative models, Self-supervision, Single-cell omics, Spatial omics, Precision oncology

Abstract: Precision medicine in oncology customizes treatments based on the unique genetic and molecular profiles of patients' tumors, which is crucial for enhancing therapeutic efficacy and minimizing adverse effects. As technological advancements yield increasingly precise data about the tumor microenvironment, the complexity of this data also grows. Notably, spatial data — a recent and promising type of omics data — provides molecular information at the single-cell level while maintaining the spatial context of cells within tissues. To fully exploit this rich and complex data, deep learning is emerging as a powerful approach that overcomes multiple limitations of traditional approaches. This manuscript details the development of new deep learning and computational methods to enhance our analysis of intricate systems like single-cell and spatial data. Three tools are introduced: (i) Scyan, for cell type annotation in cytometry, (ii) Sopa, a general pipeline for spatial omics, and (iii) Novae, a foundation model for spatial omics. These methods are applied to multiple precision medicine projects, exemplifying how they deepen our understanding of cancer biology, facilitating the discovery of new biomarkers and identifying potentially actionable targets for precision medicine.

Acknowledgments

To my PhD supervisors:

I am deeply grateful to Paul-Henry Cournède for your unwavering support and belief in my projects. This journey began after you contacted me to discuss the PhD student position. At that point, I wasn't sure about pursuing a PhD, but you convinced me, and now, after three years, I am truly grateful for this wonderful adventure. Thank you for giving me so much freedom, for your trust when I wanted to start new projects, and for your insightful comments and suggestions. You provided many opportunities for growth, learning, and achievement. I also express my sincere gratitude to Fabrice André for driving so many innovative research projects at Gustave Roussy, including those I contributed to, and for strengthening the collaboration with CentraleSupélec.

To my collegues at CentraleSupélec:

A PhD thesis wouldn't be possible without great colleagues with whom to discuss ideas, share a cup of tea, and enjoy some laughter. Hakim, the early bird of the lab, thank you for our discussions, your curiosity, and your energy (and also for this very nice PhD template)! It was a pleasure working with you on Novae - perhaps we will continue working together in the future. Thank you, Laura, for all the fun and relaxing tea breaks and for bringing a sports culture to the lab! Karmen, you have always been super dynamic and motivated. It was a pleasure supervising you during your internship and having you as a friend. Stergios, thank you for your advice on Novae and Sopa and your contributions to the histopathological capabilities of Sopa. Thanks also to all the other members of the lab, including Maria, Félicie, Aaron, Léo M., Léo F., Inès, Gurvan, Lily, Imane, Vesna, Othmane, Paul, Romain, and many others.

To my colleagues at Gustave Roussy:

First, thanks to Kevin, my biological counterpart, for patiently explaining the basics of biology whenever I needed help. This summer school, and mainly the party at Albufeira, are unforgettable (I still need the same picture for you, by the way)! Thanks to Nadège, who also helped me to understand the biology and the data, who helped me on Scyan/Novae, and who motivated people for the Triathlon des Roses (2023 and 2024). I would like to thank Charles-Antoine, who saw potential in me and gave me the opportunity to start working on spatial omics; you also hosted many really enjoyable barbecues and parties at your house. Thanks to Loïc, the AI colleague from the open space, with whom I shared many ideas despite our (apparently) different PhD topics. I also extend my gratitude to Florent, Margaux, and all the other colleagues who assisted me on this biology journey.

To external collaborators:

A special thank you goes to Luca Marconato, who gave me the opportunity to enjoy Heidelberg for a month and meet amazing people at both EMBL and DKFZ. Working and discussing with you was truly inspiring and a real pleasure. This collaboration opened me to the scverse community, with whom we met many times for various incredible events: The hackathon in Ghent, the first scverse conference in Munich, and also the Basel workshop. Expecially, thanks to the other amazing spatialdata developers/contributors, including notably Giovanni Palla, Benjamin Rombaut, and Wouter-Michiel Vierdag. I also thank Elsa Bernard, Fragkiskos Malliaros, and Lionel Larue for their valuable advice through the follow-up committee meetings.

À mes amis:

Un aspect indispensable de ma thèse a été de pouvoir me détendre et de me changer les idées en dehors du travail. Merci donc à tous mes potes de Centrale, qui sont toujours là pour partir à l'aventure les week-ends, grimper, ou simplement prendre une bière au bar. Je pense notamment à Nathan, avec qui j'ai passé tant de soirées à poncer les blocs de Climb Up (désolé pour la colocation éclair). Un énorme merci aussi à Didom, mon coloc à Meudon pendant un an, et qui nous fait des pizza meilleures qu'en Italie (bon, finalement, j'ai pas fait de Docker désolé!). Merci également à toute la team pour les moments inoubliables : Natacha, Stutz, Oriane, Hugo, Laure, Carla, Hugues, Claire, Adrian, Alix. Entre les assos, la coloc de Gif, et tous les souvenirs qu'on continue à créer ensemble, vous avez rendu ces années uniques. Merci aussi à Pasteau, avec qui on a surmonté la prépa puis intégré Centrale ensemble ; et merci aussi la team de la DTY, je pense notamment à Clément et Julien, avec qui on a réussi à se voir plusieurs fois autour d'une bière pour papoter. Enfin, merci à Benjamin, mon ami et voisin d'enfance, de nous rejoindre pour cette soutenance ; j'espère que cette thèse sera à la hauteur de notre TPE.

À ma famille:

Tout a commencé à Montbéliard, au 1 allée du Pupillin, où j'ai grandi entouré d'une super famille. Merci Papa et Maman pour votre amour, l'éducation que vous nous avez donnée, et plus récemment pour tout votre soutien durant cette thèse. Je sais que mon sujet n'a pas été facile à expliquer, mais je sens à quel point vous êtes fiers de moi. J'espère que ce manuscrit et cette soutenance seront suffisamment clairs pour que vous puissiez en saisir les enjeux ! Merci à ma soeur Marie pour une enfance incroyable à poncer tous les jeux et consoles Nintendo, et pour nos inoubliables vacances en Haute-Savoie. Ces dernières années, nos passions communes nous ont rapprochés : je t'ai d'abord initié à l'escalade, puis c'est ensuite toi qui m'as montré la grande voie ! Sans oublier nos journées de ski alpin ou de rando avec Justin, dans votre petit cocon Chamoniard. Et, oui, je me souviens que tu as soutenu ta thèse avant moi : ce resto, promis, j'ai pas oublié ! Un grand merci aussi à Jérôme et Alexis

pour toutes les vacances fabuleuses à Faverges et à Praz. À l'école, je comptais les semaines avant les vacances pour pouvoir vous retrouver. Merci à mes grands-parents, qui sont fiers de moi malgré leur absence le jour de la soutenance - on va prendre plein de photos pour que vous puissiez quand même en profiter. Merci à tout le reste de la famille : Cathy, Philippe, Babeth, Fred, Nicolas, Sophie, Marie-Hélène, et à ma marraine Claudine ainsi qu'à Éric, pour leur présence et leur soutien tout au long de mon parcours. Je remercie aussi toute la belle famille : Marie, Olivier, Eulalie, et Luc, pour votre accueil chaleureux et votre gentillesse. On a passé de bons moments ensemble aux Gobelins ou à Arêches, entre repas, sorties padel et badminton, ou escape games.

À Léa:

Toi, ma Léa chérie, t'es arrivée un peu plus tard dans ma vie : c'est après un an de thèse que notre aventure a commencé ! Peu à peu, tu as pris une place essentielle, devenant un véritable pilier non seulement de ma thèse, mais aussi de ma vie personnelle. T'as toujours été là pour m'écouter, même quand mes récits de fin de journée sur mes préoccupations de thèse s'éternisaient (souvent trop). Tu as su m'épauler dans les moments de stress - je pense notamment à notre escapade à Arêches, en plein rush de fin de thèse - et tu m'as soutenu dans mes moments de doute. Mais surtout, grâce à toi, j'ai pu mieux déconnecter du travail : en soirée, le week-end, et lors de nos belles vacances ensemble (La Réunion, Alpes, Portugal...). Je me sens incroyablement chanceux de t'avoir à mes côtés, et, surtout, je t'aime très fort. T'as embelli mes deux dernières années de thèse, et je sais que tu continueras à le faire bien au-delà. Merci ma chérie pour ton amour, ta patience et ton soutien !

Contents

Introduction

Contents

1.1 . Context

This manuscript presents the work conducted over three years during my PhD at the MICS laboratory at CentraleSupélec and the U981 and U1015 laboratories at the Gustave Roussy Institute, all located in the Greater Paris area. My research focused on the development of advanced methodologies in single-cell and spatial omics, initially motivated by specific applicative projects at Gustave Roussy that exposed several limitations in existing state-of-the-art methods. This led to the creation of new methodological tools designed to (i) complete the applicative projects I was involved in, but also (ii) bring new packages to the open-source community. Therefore, the consistent approach I had throughout this PhD was to understand a biological unmet need, and bring a solution that could be used by the community (both inside and outside our institute).

The collaboration between CentraleSupélec and Gustave Roussy was essential, as CentraleSupélec researchers provided expertise in the mathematical aspects of deep learning, while Gustave Roussy researchers provided biological knowledge and practical applications. This interdisciplinary partnership facilitated the development of tools that are both practically relevant and broadly applicable across various research projects all over the world. In particular, as a data scientist by training, many expert immunologists helped me better understand the complex systems I was working on.

Additionally, my doctoral research benefited from collaborations with numerous developers from the **scverse** community. These international interactions were enriching for diverse reasons. Notably, it (i) gave me a better overall understanding of research and the different ways to conduct it, (ii) showed great examples of good development practices, and (iii) showed many different ways to communicate research.

1.2 . Motivation

Note: For non-specialists, consider reading the background chapter (chapter 2) first.

Single-cell and spatial omics technologies have provided new opportunities to study the cellular and molecular characteristics of cancer at high resolution. However, the complexity and scale of these datasets present significant analytical challenges, making it difficult to extract meaningful biological insights using traditional methods. This work is motivated by the need for advanced computational approaches that can effectively handle the volume and complexity of these data.

1.2.1 . Challenges in cytometry

In the context of cytometry, manual cell type annotation—once a common practice—is no longer reliable due to the increasing complexity and dimensionality of the data. Manual annotations are often biased, time-consuming, and difficult to reproduce, especially as datasets grow larger. The presence of batch effects (i.e., when experimental variations introduce discrepancies between datasets generated at different times or under different conditions) further complicates this process. Batch effects in omics data can obscure true biological signals, making consistent, fast, and accurate cell type identification more difficult. This research is motivated by the need for computational approaches that can automate this process, ensuring reproducibility and scalability, while minimizing the influence of batch effects.

1.2.2 . Challenges in spatial omics

In spatial omics, the complexity is even greater. The data generated by various vendors often come in different file formats, use different coordinate systems, and have incompatible versioning, creating a fragmented landscape that makes standardization a pressing need. Without standardization, it is challenging to compare results across platforms or integrate datasets from different sources. Furthermore, spatial omics datasets are often very large, typically exceeding the capacity of standard memory, necessitating the use of sophisticated data handling methods like lazy loading (where only necessary portions of data are loaded into memory on-demand). This allows for the analysis of large datasets without overwhelming computational resources, but it also requires advanced computational tools that can handle these workloads efficiently.

The issue of batch effects is particularly severe in spatial omics, where variations occur almost at the slide level due to differences in technology, reagents, or even the gene panels used. These batch effects can obscure biological signals and make cross-slide comparisons difficult.

Additionally, the diversity of tissues and technologies used in spatial omics adds another layer of complexity, making it hard to operate across multiple datasets and requiring methods that can generalize across different tissues, technologies, and gene panels.

1.2.3 . Regarding the need for deep learning

As datasets grow larger and the technologies used to generate them become more complex, there is a critical need for advanced mathematical methods that can handle these challenges. Traditional approaches are no longer sufficient for the scale and complexity of single-cell and spatial omics data. In comparison, deep learning offers distinct advantages over other approaches due to its ability to automatically learn complex patterns and relationships from high-dimensional data without extensive manual feature

selection. Unlike conventional statistical models or simpler machine learning techniques, deep learning models can capture intricate interactions across multiple layers of biological information, which is crucial for understanding the heterogeneity of tumors and their microenvironments. This work aims to develop deep learning models specifically designed for these cytometry and spatial omics, with the goal of providing robust, scalable solutions that can fully exploit the richness of these datasets and improve our understanding of cancer biology.

1.3 . Outline

The first chapter of the manuscript introduces the two main fields of this PhD, i.e. immunology and deep learning. The two latter are explained to be understandable by a broad audience, as they are the foundation of the work presented in the following chapters. I also provide more context specific to my PhD topic, i.e. on single-cell and spatial omics. This allows us to understand the different challenges of this field, and the reasons why there is a need to develop new deep learning methods.

The second chapter presents the first methodological contribution of my PhD, i.e. **Scyan** [Blampey et al., 2023]. This project was initiated by the need to better annotate cell types in cytometry data in a large clinical study from Gustave Roussy, known as PopDurva (detailed in Chapter 4). This study aims to include up to 150 patients in a few years, meaning that the cohort suffers from significant batch effects. Existing annotation methods were very sensible to this batch effect, significantly decreasing the robustness and quality of the annotation. Afterward, Scyan was made more general, so that it can be applied to a broad range of studies, being published, and used by the community.

The third chapter concerns another topic: spatial omics data, a field I was involved-in when the first MERSCOPE machine was acquired at Gustave Roussy. In this chapter, I present the second methodological contribution of my PhD, i.e. **Sopa** [Blampey et al., 2024b], which was also initiated by a specific need, i.e. processing MERSCOPE data. In order to broaden the impact of Sopa, the pipeline was made technology-invariant, so that it can be applied to any image-based spatial omics data. Therefore, this project establishes strong foundations to standardize the analysis of spatial omics data, with a focus on time and memory efficiency to handle the large datasets produced by these machines.

The fourth chapter presents the last methodological contribution of my PhD, that is **Novae** [Blampey et al., 2024a]. This project naturally follows Sopa, as it consists in analyzing preprocessed spatial transcriptomics data. In other words, Sopa is used as preprocessing, and Novae is used as the downstream analysis. It is a foundation model that captures cell representations within their spatial environments, and is trained on a large

dataset of nearly 30 million cells across 18 different tissues. As spatial transcriptomics studies are increasing in size, it became possible to collect a large dataset to train Novae, and to provide a model that can be used without re-training on new datasets.

In the fifth chapter, I list various applicative projects I was involved in during my PhD. Especially, these projects cover a wide range of applications, and exemplifies how the methods I developed can be used.

Finally, the conclusion chapter summarizes the different contributions of my PhD, and discusses the potential impact of the methods I developed.

1.4 . Contributions

I summarize below the different contributions of my PhD, including the publications, packages, and communications I have been involved in.

1.4.1 . Articles

As mentioned above, I was involved in several applicative projects during my PhD, which led to several co-author publications. In parallel to this, motivated by the need to develop new methods, I also published three first-author papers. Note that the papers below are intricate since most of my applicative projects (i.e., the co-author papers) use the methods developed in my first-author papers.

First author papers

- *Nature Communications* - Sopa: a technology-invariant pipeline for analyses of image-based spatial omics [Blampey et al., 2024b]
- *Briefings in Bioinformatics* - A biology-driven deep generative model for cell-type annotation in cytometry [Blampey et al., 2023]
- *(Preprint)* - Novae: a graph-based foundation model for spatial transcriptomics data [Blampey et al., 2024a]

Co-author papers

- *Cancer Discovery* - Trem2-expressing multinucleated giant macrophages are a biomarker of good prognosis in head and neck squamous cell carcinoma [Gessain et al., 2024]
- *(Unpublished yet, title not definitive)* - 1510 Multiomic functional biomarkers for cancer prediction and early detection

- *(Unpublished yet, title not definitive)* - Short-term Pre-OPerative Durvalumab in early small triple-negative breast cancer patients (POP-Durva)
- *(Unpublished yet, title not definitive)* - Antecedent viral immunization and efficacy of immune checkpoint blockade: an extensive serum antibody profile to predict outcomes in Non-small Cell Lung Cancer
- *(Unpublished yet, title not definitive)* - Efficacy of INCA033989 and Ruxolitinib in chronic and advanced forms of CALRdel52 and CALRins5 myeloproliferative neoplasms (MPN) models
- *(Unpublished yet, title not definitive)* - Von Willebrand Factor-Positive Hematopoietic Stem Cells Are CALRdel52 Disease-Initiating Cells associated with signaling of the PERK/EiF2 Branch of Unfolded Protein Response
- *(Unpublished yet, title not definitive)* - Unravelling DC subsets and states across human normal adjacent and malignant tissues
- *(Unpublished yet, title not definitive)* - LegendScreen project
- *(Unpublished yet, title not definitive)* - TCE-CRS project

1.4.2 . Open-source packages

The development of the methods presented in this manuscript was done in the open-source community, and led to the creation of several packages. Since these packages are based on core packages from the **scverse** community (see section 2.3 for more details about this community), I also contributed to these core packages.

Open-source packages creator

- novae (Graph-based foundation model for spatial transcriptomics data)
- sopa (Spatial omics pipeline analysis)
- scyan (Single-cell cytometry annotation network)
- spatialdata_xenium_explorer (Conversion between the Xenium Explorer and *spatialdata*)

Open-source packages contributions

- spatialdata (scverse core package for spatial omics data)
- pytometry (scverse package for cytometry)
- spatialdata-io (readers for *spatialdata*)

1.4.3 . Communications

Developing new methods is essential, but it is also important to communicate about them. I have been involved in several communications, including oral and poster presentations.

- *Oral and poster presentation* - Scyan ([Blampey et al., 2023]) at the *Joint annual meeting of the SFI and AFC* (Nice, France; Nov. 2022)
- *Poster presentation* - Sopa ([Blampey et al., 2024b]) at the *Cancer Core Europe summer school* (Albufeira, Portugal; Oct. 2023)
- *Oral presentation* - Scyan ([Blampey et al., 2023]) at the *scverse community meetings* (Online; May. 2024)
- *Oral and poster presentation* - Sopa ([Blampey et al., 2024b]) at the *VIB spatial omics conference* (Ghent, Belgium; June. 2024)
- *Oral and poster presentation* - Novae ([Blampey et al., 2024a]) at the *scverse conference* (Munich, Germany; Sept. 2024)

1.4.4 . Teaching and supervision

Finally, I have been involved in several teaching and supervision activities at CentraleSupélec during my PhD.

- *Practical sessions* - Statistics and machine learning course at CentraleSupélec (2022)
- *Project supervision* - 5-student project at CentraleSupélec (2022)
- *Intern supervision* - Master 2 internship on Cytometry (2022-2023)

Background

Contents

Abstract

This background section provides a basic overview of immunology and oncology, covering the minimal required concepts to understand the PhD. Notably, it details what is a cell, the immune system, cancer, and precision medicine. It then introduces omics data, explaining the different modalities, resolutions (bulk, single-cell, spatial), key vendors, and the associated tasks and challenges. The importance of open-source communities in advancing single-cell analysis is emphasized, with a focus on scverse, the Python ecosystem. A simple explanation of deep learning follows, highlighting its utility in single-cell and spatial analysis with examples such as Cellpose and scVI.

2.1 . Introduction to immunology and oncology

In this section, we provide a fundamental understanding of key biological concepts to help deep learning experts with no background in biology appreciate the relevance and application of computational methods in precision medicine, particularly in oncology.

2.1.1 . Cell: the basic unit of life

A cell is the basic building block of all living organisms. It is a small, self-contained unit that carries out the essential functions necessary for life. Cells can be broadly categorized into two types: prokaryotic cells, like bacteria, which lack a nucleus, and eukaryotic cells, like human cells, which have a nucleus and specialized structures called organelles. In this manuscript, we will focus on eukaryotic cells only. The cell nucleus holds the cell's DNA and controls its functions. The cytoplasm is a jelly-like substance that contains organelles such as mitochondria, which produce energy, and the endoplasmic reticulum, which helps make proteins and lipids. The cell membrane is a thin layer that surrounds the cell, providing structure and regulating what goes in and out. Within a cell, three key components play crucial roles: DNA, RNA, and proteins.

DNA (Deoxyribonucleic Acid)

DNA is the cell's genetic material. It contains the instructions needed for building and maintaining the organism. These instructions are coded in the form of sequences of nucleotides. DNA is organized into structures called chromosomes, which are found in the cell's nucleus.

RNA (Ribonucleic Acid)

RNA is a molecule that helps translate the instructions in DNA into proteins (see below). The process begins with transcription, where a specific segment of DNA is copied into RNA. This RNA copy, known as messenger RNA (mRNA), carries the genetic information from the nucleus to the ribosomes, which are the cell's protein-making machinery.

Proteins

Proteins are complex molecules that perform most of the functions within a cell. They are made up of amino acids and are synthesized by the ribosomes based on the instructions carried by mRNA. Proteins play a wide variety of roles, including: (i) structural roles (building and maintaining cell structure), (ii) enzymatic roles (catalyzing biochemical reactions), or (iii) regulatory roles (controlling the expression of genes and the activity of other proteins).

In simple words, DNA provides the blueprint for life, RNA serves as the messenger that conveys these instructions, and proteins are the workers that carry out the tasks specified by the genetic code. Understanding these components and their interactions is fundamental to studying biology and developing new medical treatments.

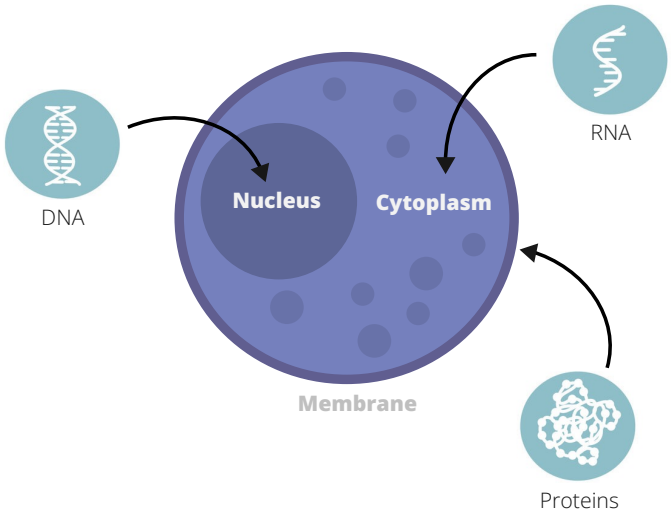

Figure 2.1: ***Illustration of an eukaryotic cell.*** *The main components of a eukaryotic cell include the nucleus, cytoplasm, and cell membrane. The DNA is located in the nucleus and contains the genetic instructions for the cell. RNA molecules help translate these instructions into proteins, which carry out various functions within the cell (proteins can be located at different places in the cell, including the membrane).*

Depending on the location in the human body (e.g., bones, muscles, nerves, or organs), cells will have different functions, and will specialize to carry out specific tasks. We denote by "cell type", a group of cells with similar functions. The diversity of cell types allows the body to function efficiently and respond to a wide range of needs, as each type is adapted to perform a unique role. For example, muscle cells are specialized for movement, contracting and relaxing to enable physical activity. Nerve cells, or neurons, are specialized for communication, transmitting signals throughout the body to coordinate actions and responses. Blood cells, such as red blood cells, transport oxygen and

nutrients to tissues, while white blood cells are involved in defending the body against infections.

2.1.2 . The immune system

The immune system is the body's defense mechanism against infections and diseases. It is a complex network of cells, tissues, and organs that work together to protect the body from harmful invaders like bacteria, viruses, or cancer. The immune system can be divided into two main parts: the innate immune system and the adaptive immune system. The innate immune system is the body's first line of defense. It responds quickly to invaders in a general way, without needing to recognize specific pathogens. The adaptive immune system provides a targeted and more specific response to invaders. It takes longer to respond but has a memory component that allows it to respond more rapidly and effectively to pathogens it has encountered before. The immune system can be described by two main lineages of cells: the myeloid and lymphoid lineages. Below, we briefly describe the main types of immune cells and their roles, and their relation with the innate and adaptive immune systems:

T Cells

T cells are a type of lymphocyte that plays a central role in the immune response, notably in identifying and killing infected cells. Helper T Cells (also known as CD4+ T Cells) are crucial for coordinating the immune response. They help activate other immune cells by releasing signaling molecules called cytokines. CD4+ T cells enhance the ability of B cells to produce antibodies and stimulate macrophages to destroy ingested microbes. Cytotoxic T Cells (also known as CD8+ T Cells) directly kill infected cells, cancer cells, and cells that are damaged in other ways. They recognize and bind to antigens presented on the surface of infected cells, leading to their destruction.

B Cells

B cells, another type of lymphocyte, produce antibodies (i.e. are proteins that specifically target and neutralize pathogens like bacteria and viruses). B cells can also present antigens to T cells, facilitating a more robust immune response. When B cells encounter their specific antigen, they can differentiate into plasma cells that secrete large amounts of antibodies or become memory B cells that provide long-term immunity.

Natural Killer (NK) Cells

Natural killer cells are a type of lymphocyte that plays a key role in the innate immune response. Unlike T cells and B cells, NK cells do not require antigen presentation to recognize and kill their targets. They can detect stressed cells in the absence of antibodies and MHC (Major Histocompatibility Complex), making them effective against tumor cells and virally infected cells.

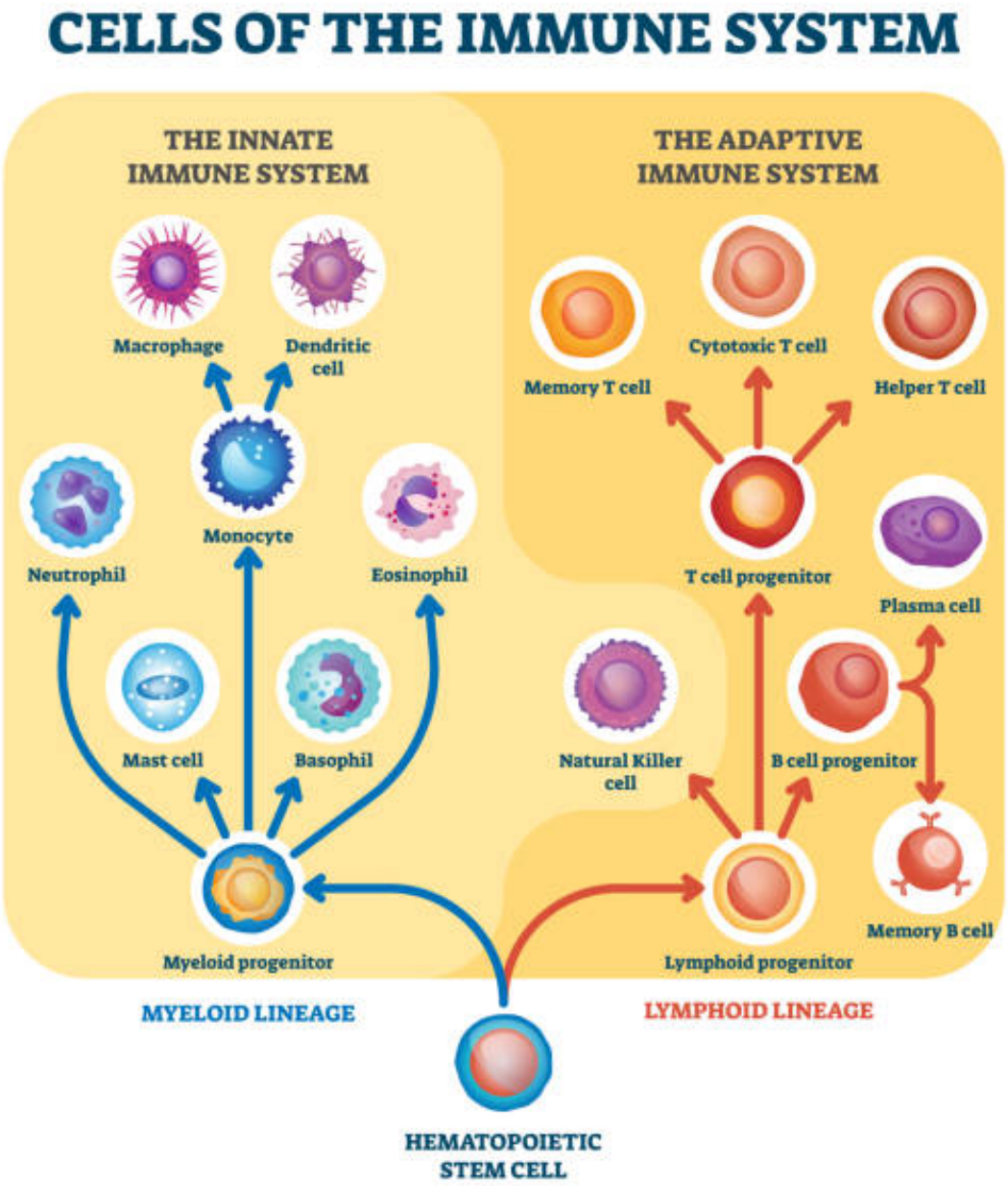

Figure 2.2: ***Illustration of the main immune cell types.*** *The two main lineages are the myeloid and lymphoid lineages. The myeloid lineage gives rise to monocytes, macrophages, dendritic cells (DC), neutrophils, eosinophils, mast cells and basophils. The lymphoid lineage gives rise to B cells, T cells, and natural killer (NK) cells.*

Macrophages

Macrophages are large phagocytic cells that can be found in almost all tissues and are particularly abundant in areas of infection or inflammation. Macrophages digest pathogens, dead cells, and cellular debris. They also present antigens to T cells and release cytokines that recruit and activate other immune cells.

Dendritic Cells (DCs)

Dendritic cells are antigen-presenting cells that act as messengers between the innate and adaptive immune systems. They capture antigens from pathogens and present them on their surface to T cells, thereby initiating and regulating the adaptive immune response. Dendritic cells are found in tissues that are in contact with the external environment, such as the skin and mucous membranes.

These various immune cells work together in a highly coordinated manner to detect, respond to, and eliminate pathogens and other threats, maintaining the body's health and defending against disease. Understanding the functions and interactions of these immune cells is crucial for developing effective treatments and therapies for various diseases, including infections, autoimmune disorders, and cancers.

2.1.3 . An overview of cancer

Cancer is characterized by the uncontrolled growth and spread of abnormal cells. It occurs when the normal regulatory mechanisms that control cell growth and division fail, leading to the formation of tumors and potential spread to other parts of the body. Normal cells in the body grow, divide, and die in an orderly fashion, but when cancer develops, this process is disrupted. The genetic material (DNA) within cells becomes damaged or altered, leading to mutations that affect normal cell functions. These mutations can cause cells to grow uncontrollably, avoid programmed cell death (apoptosis), and invade other tissues. Cancer cells can also spread to other parts of the body through the bloodstream or lymphatic system, a process known as metastasis. Cancer can arise in virtually any part of the body, and the main types include carcinomas, sarcomas, leukemias, lymphomas, and central nervous system cancers. Carcinomas originate in the epithelial cells (a cell type that lines the inside and outside surfaces of the body, such as the skin, lungs, and breasts). Sarcomas develop in the bone, cartilage, fat, muscle, or other connective tissues. Leukemias are cancers of the blood and bone marrow, characterized by the overproduction of abnormal white blood cells. Lymphomas originate in the lymphatic system, which is part of the immune system, and central nervous system cancers begin in the tissues of the brain and spinal cord. Cancer treatment depends on the type, stage, and location of the cancer, as well as the patient's overall health. Common treatments include surgery, radiation therapy, chemotherapy, targeted therapy, hormone therapy, and immunotherapy.

The tumor itself is not only composed of tumor cells, it is a complex and dynamic ecosystem, comprising various cell types, signaling molecules, and extracellular matrix components. We call it the tumor microenvironment (TME). It includes immune cells, fibroblasts, blood vessels, and other elements that interact with cancer cells, influencing tumor growth, progression, and response to therapy. Therefore, this complex system

requires sophisticated machines to measure as subtle information as possible. Notably, global information about the TME ("bulk resolution") is a good indicator of the patient's tumor. But, instead, information at the cell resolution ("single-cell resolution"), is more informative and can be better used for precision medicine, for instance to predict the treatment response.

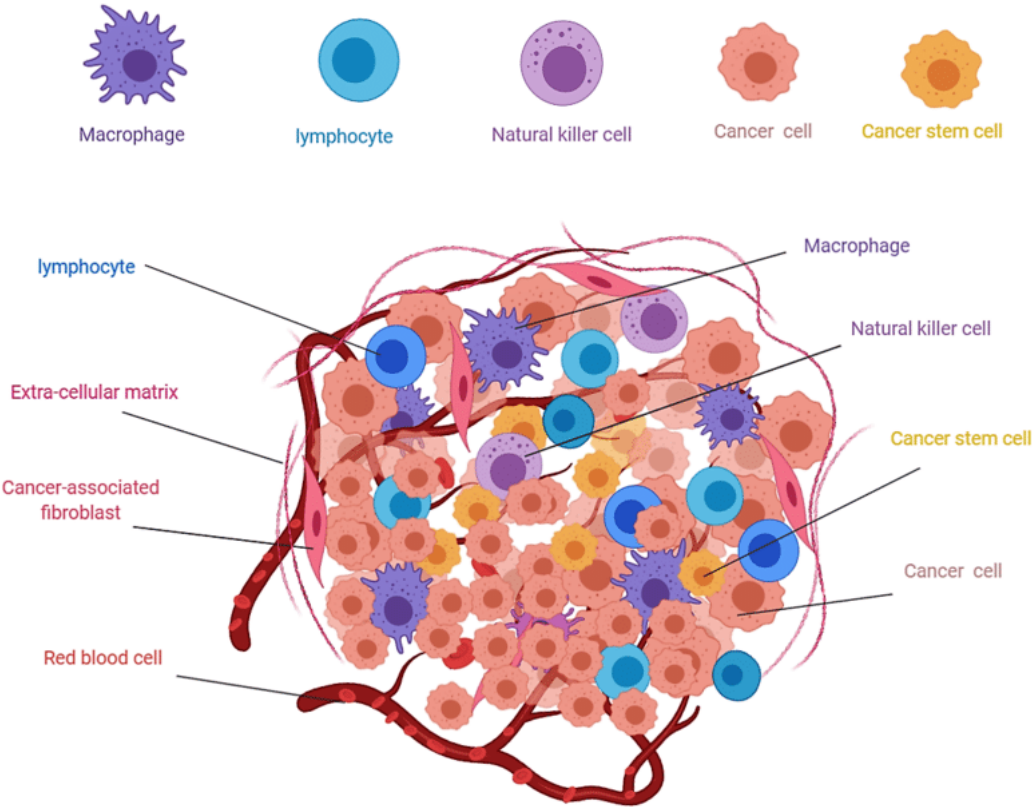

Figure 2.3: ***Illustration of the tumor microenvironment.*** *The tumor microenvironment is a complex ecosystem of a large range of cell types. Illustration from [Hassan and Seno, 2020].*

2.1.4 . Precision medicine in oncology

Precision medicine [Tsimberidou et al., 2020, Waldman et al., 2020] is an innovative approach to patient care that tailors treatment to the individual characteristics of each patient and their disease. In oncology, precision medicine involves using detailed information about a patient's genetic profile, the specific characteristics of their cancer, and other relevant information to develop personalized treatment plans [Tsimberidou et al., 2020, Alturki, 2023]. This approach aims to improve the effectiveness of therapies, minimize side effects, and enhance overall patient outcomes. Therefore, it is key to develop computational methods that can analyze and interpret large-scale biological data to identify patterns and relationships to guide treatment decisions. Such a measurable pattern

is called a biomarker, i.e. a biological indicator that can be used to diagnose diseases or personalize therapy plans. Biomarkers play a crucial role in precision medicine by helping to identify which patients are most likely to benefit from specific treatments. For example, the presence of certain genetic mutations or the expression of specific proteins on cancer cells can indicate whether a patient will respond to a particular therapy. Biomarker testing can guide the selection of the most appropriate and effective treatment for each patient, increasing the likelihood of a successful outcome.

One of the most promising areas of precision medicine in oncology is immunotherapy [Waldman et al., 2020]. Immunotherapy harnesses the body's own immune system to fight cancer. Unlike traditional treatments such as chemotherapy and radiation, which directly target cancer cells, immunotherapy works by stimulating or enhancing the immune system's natural ability to recognize and destroy cancer cells. It has the potential for higher success rates and better responses compared to traditional therapies, as it can be tailored for each patient. However, it remains difficult to predict which patient will respond or which patient will suffer from severe toxicities because of the treatment. There are several types of immunotherapy used in precision medicine, we describe some of them below:

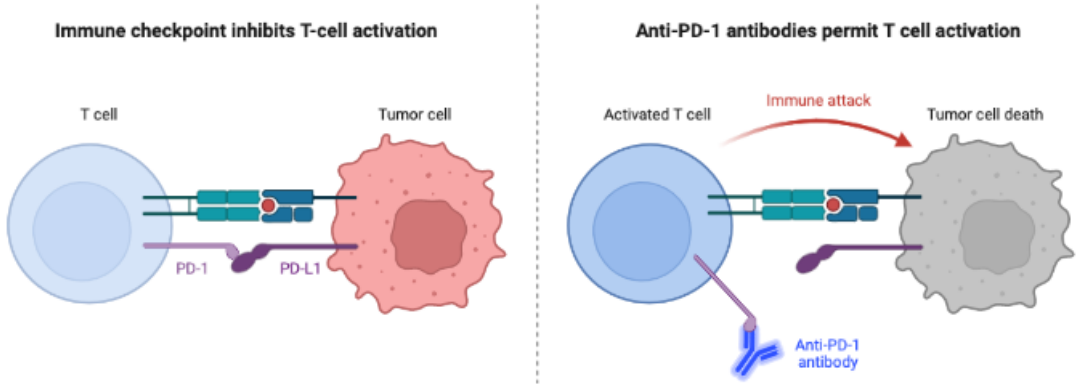

Figure 2.4: ***Illustration of checkpoint inhibitors.*** *Checkpoint inhibitors are a type of immunotherapy that block proteins that prevent the immune system from attacking cancer cells. By inhibiting these proteins, the immune response against cancer cells is enhanced. In this figure, the mechanism of an anti-PD-1 checkpoint inhibitor is shown.*

Checkpoint Inhibitors

These drugs block proteins that prevent the immune system from attacking cancer cells [Alturki, 2023]. For example, drugs that inhibit the proteins PD-1 or PD-L1 can boost the immune response against cancer cells, leading to their destruction. Checkpoint inhibitors have shown remarkable success in treating certain types of cancers, such as melanoma and lung cancer.

CAR-T Cell Therapy

This innovative treatment involves modifying a patient's own T cells to better recognize and attack cancer cells [Sterner and Sterner, 2021]. T cells are collected from the patient, genetically engineered to express a receptor specific to the cancer cells (chimeric antigen receptor or CAR), and then infused back into the patient. CAR-T cell therapy has been particularly effective in treating certain blood cancers, like acute lymphoblastic leukemia and some types of lymphoma.

Cancer Vaccines

These vaccines are designed to stimulate the immune system to attack cancer cells [Fan et al., 2023]. Unlike traditional vaccines that prevent infections, cancer vaccines are therapeutic and aim to treat existing cancers by enhancing the immune response against specific cancer antigens.

Overall, precision medicine and immunotherapy are revolutionizing cancer treatment by providing more targeted, effective, and personalized therapies. By integrating computational methods with biological data, researchers and clinicians can identify biomarkers, predict treatment responses, and optimize patient outcomes. The development of deep learning models that can analyze complex biological data and extract meaningful insights is a critical step towards advancing precision medicine in oncology.

2.2 . Exploring biological systems through omics data

Omics data refers to large-scale biological data generated from high-throughput technologies that capture various aspects of biological systems. These technologies allow researchers to study the molecular components of cells, providing a detailed view of biological processes.

2.2.1 . Omics data modalities

Omics data can be generated from different modalities, such as genomics, transcriptomics, proteomics, metabolomics, and epigenomics, as detailed below. Each omics modality captures a specific aspect of the biological system, providing information on different biological layers. In the context of precision medicine, omics data plays a crucial role in identifying biomarkers, predicting treatment responses, and guiding personalized therapeutic strategies.

Genomics

Genomics is the study of the complete set of DNA (the genome) in an organism. It involves sequencing and analyzing the entire genetic material to understand the structure, function, and evolution of genes. Genomic data helps identify genetic variations and mutations that may contribute to diseases, providing insights into molecular alter-

ations, inherited conditions, and potential therapeutic targets.

Transcriptomics

Transcriptomics is the study of the complete set of RNA transcripts produced by the genome at any given time. This includes messenger RNA (mRNA), which is translated into proteins, as well as other types of RNA that have regulatory and structural functions. Transcriptomic data reveals which genes are actively being expressed and to what extent, offering a snapshot of gene activity under various conditions.

Proteomics

Proteomics is the study of the complete set of proteins (the proteome) produced by a cell, tissue, or organism. Proteins are the functional molecules that perform a vast array of tasks within the cell. Proteomic data helps identify the types, quantities, and modifications of proteins present, providing insights into cellular processes and signaling pathways.

Metabolomics

Metabolomics is the study of the complete set of small molecules, or metabolites, within a cell, tissue, or organism. Metabolites are the intermediates and products of metabolic reactions. Metabolomic data helps understand the biochemical activities within cells, revealing how cellular metabolism changes in response to various factors such as disease, drug treatment, or environmental conditions.

Epigenomics

Epigenomics is the study of the complete set of epigenetic modifications on the genetic material of a cell. These modifications, such as DNA methylation and histone modification, regulate gene expression without altering the underlying DNA sequence. Epigenomic data provides insights into how gene activity is controlled and how it can be influenced by environmental factors and lifestyle.

In this manuscript, we will focus mainly on the analysis of transcriptomics and proteomics data, which provide crucial information about cellular functions and disease mechanisms. By studying both gene expression (transcriptomics) and protein levels (proteomics), we get a comprehensive view of how cells operate and how they change in disease conditions.

2.2.2 . Different technologies with different resolutions

In modern biological research, various technologies are employed to measure and analyze biological data at different resolutions, each providing unique insights into the

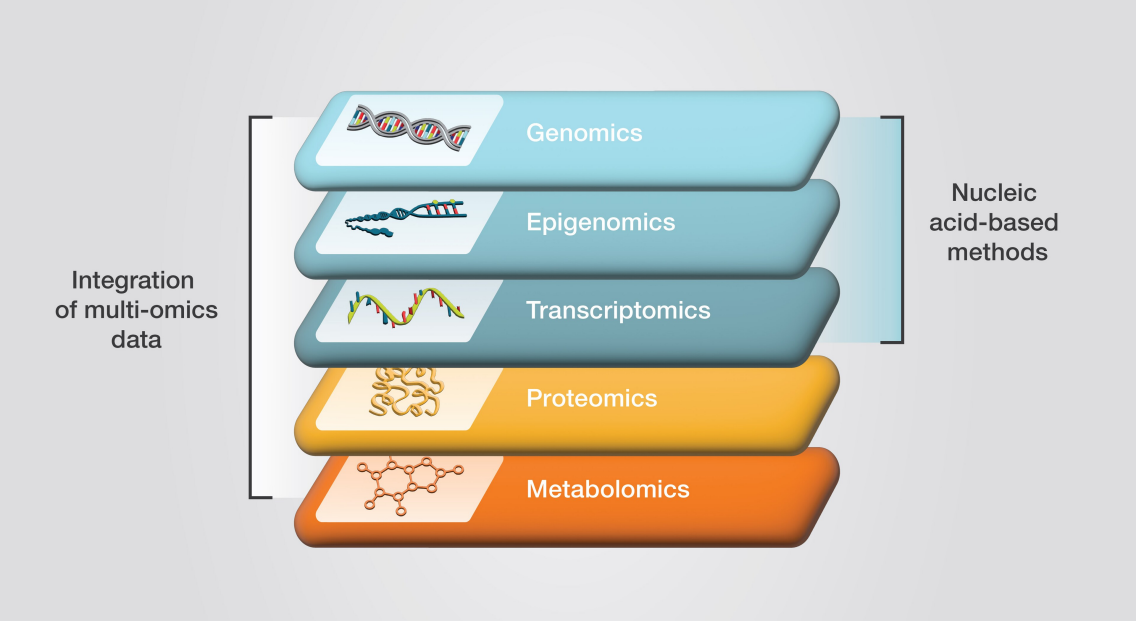

Figure 2.5: ***Illustration of omics data modalities.*** *Omics data captures various aspects of biological systems, including genomics, transcriptomics, proteomics, metabolomics, and epigenomics. If multiple modalities are used at the same time, this is called multi-omics.*

complex mechanisms of life. These technologies can be broadly categorized into bulk data, single-cell data, spatial data with spot resolution, and spatial data with single-cell resolution.

Bulk data

Bulk data refers to measurements taken from a large population of cells, providing an average signal across the entire sample. This approach is useful for identifying general trends and overall gene or protein expression levels in tissues or large cell populations. However, it masks the variability and unique characteristics of individual cells, potentially overlooking rare but important cellular behaviors (especially in heterogeneous populations of cells, like the tumor and its microenvironment).

Single-cell resolution data

Single-cell data addresses the limitations of bulk data by measuring gene or protein expression in individual cells. This high-resolution approach reveals the heterogeneity within a cell population, allowing researchers to identify distinct cell types, states, and transitions. Single-cell technologies provide a detailed understanding of cellular diversity and function, which is crucial for studying complex biological systems and diseases like cancer.

Spatial data with spot resolution

Spatial data with spot resolution combines spatial information with gene or protein expression profiles. Technologies in this category capture data from defined regions or spots within a tissue sample. Each spot typically contains multiple cells, providing localized but not single-cell resolution. This approach helps map the spatial organization of different cell types and their interactions within the tissue, offering insights into the tissue architecture and microenvironment.

Spatial data with single-cell resolution

Spatial data with single-cell resolution (or subcellular resolution) takes the analysis a step further by providing detailed information at the level of individual cells and even subcellular structures. These advanced technologies allow researchers to see the precise localization of molecules within cells and how they interact with their immediate surroundings. This high resolution is critical for understanding the intricate details of cellular function and pathology.

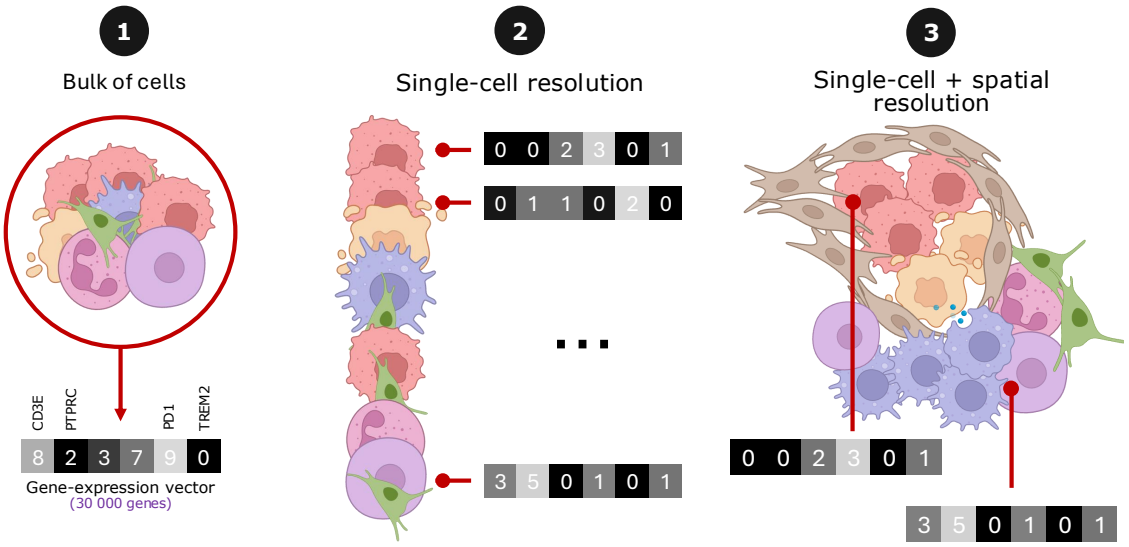

Figure 2.6: ***Illustration of omics data resolutions.*** *Omics data can be measured at different resolutions, including bulk data, single-cell data, and spatial data with single-cell resolution. For instance, bulk-RNA-seq measures a vector of gene expression (more than 30,000 genes) for the whole sample. Also, scRNAseq data measures the same information for each single cell, while spatial transcriptomics data also maintain the spatial location of cells within the tumor.*

In this manuscript, we focus on single-cell resolution data as well as spatial data with single-cell resolution, for both transcriptomics and proteomics. These high-resolution technologies, while expensive, provide the most detailed and informative data about cellular processes and interactions.

2.2.3 . A myriad of vendors and machines

The field of omics research is rapidly evolving, with numerous vendors and machines offering a wide range of technologies for single-cell and spatial-resolution data. All machines are different, and each has its own strengths and limitations, making it essential to choose the right technology for the specific research question. The key vendors include *10x Genomics*, *Vizgen*, *Nanostring*, *Akoya Biosciences*, *Beckman Coulter*, *Fluidigm*, and many others. Below, we provide a brief overview of some of the technologies offered by these vendors, according to their modality and resolution:

Machine	**Vendor**	**Omics**	**Resolution**	**Spatial**	**Imaging-based**
Chromium	10X Genomics	Transcriptomics	Single-cell	No	No
Visium	10X Genomics	Transcriptomics	Spots	Yes	No
Visium HD	10X Genomics	Transcriptomics	Subcellular	Yes	No
Xenium	10X Genomics	Transcriptomics	Subcellular	Yes	Yes
MERSCOPE	Vizgen	Transcriptomics	Subcellular	Yes	Yes
CosMX	Nanostring	Transcriptomics	Subcellular	Yes	Yes
GeoMX	Nanostring	Transcriptomics	Spots	Yes	No
CytoFLEX	Beckman Coulter	Proteomics	Single-cell	No	No
Cytek Aurora	Cytek Biosciences	Proteomics	Single-cell	No	No
Phenocycler	Akoya	Proteomics	Single-cell	Yes	Yes
MACSIma	Miltenyi	Proteomics	Single-cell	Yes	Yes
Hyperion	Fluidigm	Proteomics	Single-cell	Yes	Yes

Table 2.1: Non-exhaustive list of omics machines and vendors, and their characteristics.

All these machines come with different file formats, a different number of genes or proteins measured, different coordinate systems, different sensitivities, and different resolutions. Therefore, it is crucial to develop computational methods that can handle these differences and extract meaningful insights from the data. We will further provide more details on the advantages and drawbacks of each of these machines.

2.2.4 . Tasks and challenges in omics data analysis

Analyzing omics data involves a range of tasks, each presenting unique challenges. Some of the main tasks include (i) cell type annotation, (ii) handling batch effects, (iii) integrating multi-omics data, or (iv) dealing with the computational complexities of the data. Below, we explore these tasks and the associated challenges in more detail.

Cell type annotation

Cell type annotation is one of the primary tasks in single-cell omics analysis, involving the identification and classification of the various cell types present in a dataset. This can for instance be done by comparing the gene expression profiles of individual cells to known markers or reference datasets [Biancalani et al., 2021]. While essential for un-

derstanding cellular diversity and function, this task can be challenging due to the high dimensionality of the data, the biological variation within and between cell types, and the lack of reference data for some rare or poorly characterized cell types.

Batch effects

Batch effects are non-biological variations in the data, that arise from technical variability between different batches of experiments (or samples), which can obscure true biological differences. These effects can be introduced during sample preparation, sequencing, or data processing. Handling batch effects is crucial for ensuring the validity of the analysis. It can be addressed by normalization methods that adjust for technical variability, or also Deep Learning models that can learn and correct for batch effects [Lopez et al., 2018, Korsunsky et al., 2019].

Multi-omics integration

Integrating multi-omics data provides a comprehensive view of biological systems but introduces several challenges [Picard et al., 2021]. Different omics data types have distinct characteristics and scales, making integration complex. Combining large datasets requires significant computational resources and sophisticated algorithms to handle the increased complexity and ensure meaningful integration.

Computational complexity

Omics data can represent from gigabytes up to terabytes of data for each patient, mostly when dealing with spatial omics data. Indeed, spatial omics datasets can be extremely large, with multi-channel images reaching up to 1TB and individual slides containing billions of transcripts. Managing and processing such vast amounts of data require robust computational infrastructure and efficient algorithms. High storage and memory requirements necessitate efficient data management strategies, such as data compression and distributed computing. Computational performance can be affected by high-resolution spatial data, which can slow down analysis pipelines. Optimizing algorithms for speed and efficiency and utilizing high-performance computing (HPC) clusters can help mitigate this issue.

Advances in computational methods and hardware infrastructure are continually improving our ability to tackle these challenges. Open-source communities and collaborative efforts play a crucial role in developing and disseminating tools that address these issues, enabling researchers to push the boundaries of omics data analysis [Virshup et al., 2023]. With the increasing size and complexity of the data, deep learning methods are becoming increasingly efficient and popular for analyzing omics data.

2.3 . Open-source communities and packages for omics data analysis

The analysis of omics data, particularly single-cell and spatial data, requires sophisticated computational tools to clean, process, and analyze, the vast and complex datasets generated by modern technologies. Open-source communities play a crucial role in advancing these tools by fostering collaboration, innovation, and accessibility. *Seurat* (in R) and ***scverse*** [Virshup et al., 2023] (in Python) are the two leading communities in single-cell data analysis. The contributions of open-source communities like *Seurat* and *scverse* have transformed omics data analysis, providing powerful, flexible, and accessible tools that drive scientific discovery. As a contributor to *scverse*, I will highlight and provide more details about this community. *scverse* is a Python ecosystem of interoperable packages tailored for single-cell and spatial omics data analysis. The *scverse* community is dedicated to developing tools that are flexible, efficient, and easy to integrate into various analytical workflows. Here are the main packages of *scverse*:

anndata [Virshup et al., 2021]

This package provides a scalable way to store annotated data matrices. It is designed to handle large single-cell datasets efficiently, ensuring that metadata and experimental data are stored together in a structured format. anndata serves as the foundational data structure for many scverse tools.

scanpy [Wolf et al., 2018]

Built on top of anndata, scanpy is a powerful toolkit for analyzing single-cell gene expression data. It includes modules for data preprocessing, clustering, trajectory inference, and visualization.

squidpy [Palla et al., 2022]

This package extends scanpy's capabilities to spatial transcriptomics data. squidpy provides tools for analyzing spatial patterns, interactions, and neighborhood relationships within tissues. It integrates seamlessly with anndata and scanpy, allowing users to perform comprehensive spatial analysis alongside their single-cell analyses.

spatialdata [Marconato et al., 2024]

spatialdata is an universal framework for processing spatial omics data. It provides a common interface for reading, writing, and manipulating any spatial dataset, enabling interoperability between different spatial technologies and analysis tools.

Open-source communities like *scverse* are essential for several reasons. First, it brings

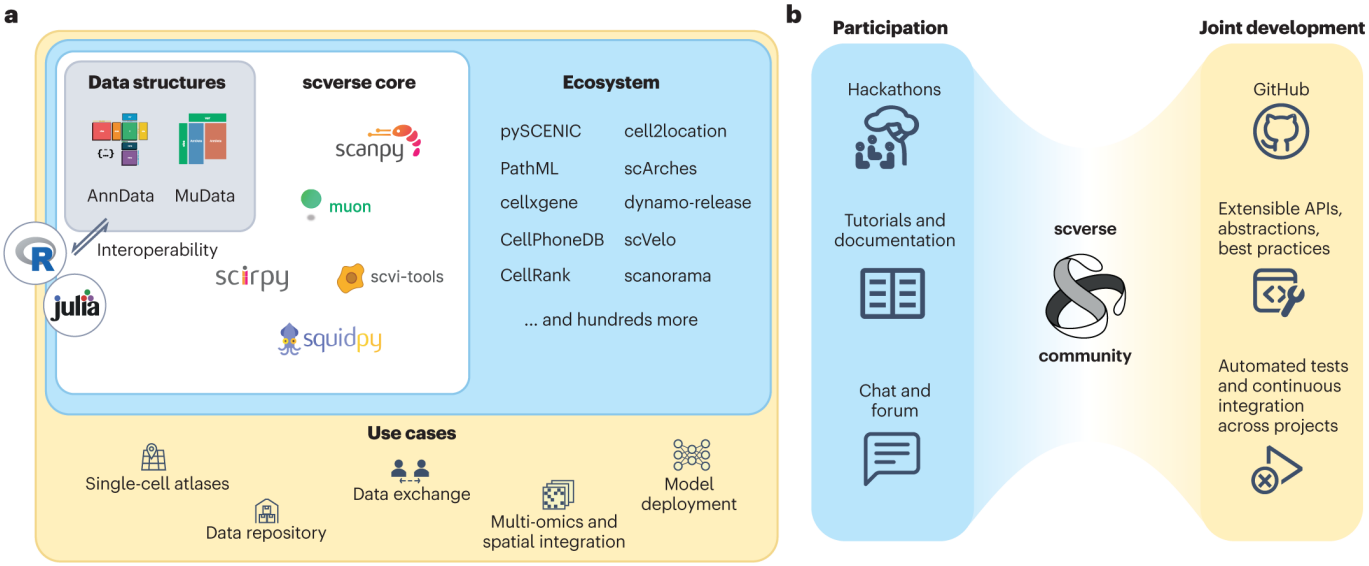

Figure 2.7: ***Overview of the scverse ecosystem. a,*** *Core packages and ecosystem. AnnData, MuData and SpatialData are data structures used by core packages such as Scanpy or Squidpy. On top of that, contributors can add other packages to the ecosystem.* ***b,*** *The scverse community is a large community with many ways to contribute and exchange. [Image credits: [Virshup et al., 2023]]*

together researchers, developers, and data scientists from around the world to collaborate on improving and expanding analytical tools. This collective effort accelerates the development of innovative methods and best practices. Secondly, these open-source software are freely available for anyone to use, modify, and distribute. This transparency ensures that scientific analyses can be reproduced and validated by independent researchers, which is essential for the credibility and reliability of scientific findings. Finally, by providing free tools and resources, open-source communities democratize access to advanced analytical methods. This accessibility allows researchers from institutions with limited resources to perform cutting-edge analyses and contribute to scientific progress.

2.4 . Introduction to Deep Learning

2.4.1 . Basics concepts

Deep learning [LeCun et al., 2015] is a subset of machine learning, which is itself a subset of Artificial Intelligence (AI). Deep learning uses mathematical models known as neural networks to analyze, perform tasks, and interpret complex data. These neural networks are trained to recognize patterns and make predictions by optimizing a set of parameters through iterative learning processes. In simple words, a deep learning model is a parametrized function with specific inputs and outputs, and the training process consists of finding the right parameters that make this function "good" (according to a certain measure of performance). The function itself, the neural network, consists of multiple layers of interconnected nodes (or neurons). Each neuron performs a simple mathematical

operation, typically involving a weighted sum of its inputs followed by a non-linear activation function. The connections between neurons are represented by weights, which are adjusted during training to minimize the error between the network's predictions and the actual (or expected) outcome. The fundamental building block of a neural network, i.e. the neuron, is mathematically represented by the following equation:

$$y = f(\sum_{i=1}^{n} w_i x_i + b), \tag{2.1}$$

where $(x_i)_i$ are the input features, $(w_i)_i$ are the corresponding trainable weights, b is the trainable bias term, and f is a chosen activation function (a non-linear function). Common activation functions include the Rectified Linear Unit (ReLU), sigmoid, and tanh functions. The output y is the result of the neuron's computation. By "trainable", we mean a parameter that is optimized during the training process.

As mentioned above, before training a model, a "loss function" (denoted $\mathcal{L}$), must be defined to quantify the performances of the model given a specific set of parameters (the weights and biases). Usually, the complete set of parameters is denoted as $\theta := (w_1, \dots, w_n, b, \dots)$. Note that the parameters $(w_1, \dots, w_n, b)$ are only the parameters of one neuron, therefore the complete model, composed of possibly millions of neurons, may have a very large set of parameters θ. Then, training a neural network involves finding the optimal set of weights and biases that maximizes the performances of the model (i.e., finding θ that minimizes the loss function $\mathcal{L}(\theta)$). This process is done using a technique called backpropagation, which calculates the gradient of the loss function with respect to each weight by applying the chain rule of calculus. The weights are then updated using gradient descent or one of its variants, such as stochastic gradient descent (SGD), Adam, or RMSprop.

Deep learning has proven to be exceptionally powerful in handling large-scale and high-dimensional data, making it ideal for applications in image and speech recognition, natural language processing, and more recently omics data. Its ability to automatically learn representations of data eliminates the need for manual feature engineering, allowing the model to uncover intricate patterns and relationships within the data. By leveraging the power of deep learning, researchers can analyze complex biological datasets, such as single-cell and spatial omics data, to gain new insights into cellular processes, disease mechanisms, and potential therapeutic targets.

2.4.2 . Main deep learning architectures

Deep learning encompasses various neural network architectures, each designed to address specific types of problems and data structures. Below, we describe some of the most prominent architectures: Multi-layer perceptrons, convolutional neural networks,

recurrent neural networks, transformer networks, and graph neural networks. Each of the architectures below has unique strengths and is suited to specific types of problems.

Multi layer perceptrons (MLPs)

Multi-layer perceptrons are the simplest type of artificial neural network, where the information moves in one direction only: from the input layer, through the hidden layers, to the output layer. They are mainly used to process one-dimensional vector data (e.g., patient clinical data or a single-cell gene expression vector). Each layer consists of neurons that are fully connected to the neurons in the subsequent layer. The architecture of a MLP can be mathematically expressed as:

$$\mathbf{h}^{(l)} = f(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}),$$

where $\mathbf{h}^{(l)}$ is the output of the l-th layer, $\mathbf{W}^{(l)}$ are the weights, $\mathbf{b}^{(l)}$ are the biases, and f is the activation function. The size and number of hidden layers can vary. The input $\mathbf{x}$ is provided as $\mathbf{h}^{(0)} := \mathbf{x}$, and the final output layer provides the prediction, i.e. $y = \mathbf{h}^{(L)}$, where L is the number of layers.

Convolutional Neural Networks (CNNs)

Convolutional neural networks [Yamashita et al., 2018] are specifically designed for processing structured grid data, such as images. Therefore, they can be used for tasks such as cell segmentation. CNNs use convolutional layers that apply a set of filters to the input data to capture spatial hierarchies and patterns. These layers are followed by pooling layers that reduce the dimensionality of the data, making the network more computationally efficient. The operation of a convolutional layer can be described as:

$$\mathbf{h}_{i,j}^{(l)} = f\left(\sum_{m,n} \mathbf{W}_{m,n}^{(l)} \mathbf{h}_{i+m,j+n}^{(l-1)} + \mathbf{b}^{(l)}\right)$$

where $\mathbf{h}_{i,j}^{(l)}$ is the output of the l-th layer at position (i, j), $\mathbf{W}_{m,n}^{(l)}$ are the convolutional filters, and f is the activation function.

Transformer Networks

Transformer networks [Vaswani et al., 2023] have revolutionized the field of natural language processing by using self-attention mechanisms to process sequential data. Transformers allow for parallelization and can handle long-range dependencies. The self-attention mechanism is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where Q (queries), K (keys), and V (values) are linear transformations of the input, and d_k is the dimension of the key vectors. The transformer architecture typically consists of an encoder-decoder structure, where both the encoder and decoder are composed of multiple layers of self-attention and MLPs. More recently, transformers were adapted for vision tasks (Vision Transformers [Dosovitskiy et al., 2021]), showing high performances for tasks such as image classification and segmentation.

Graph Neural Networks (GNNs)

Graph neural networks are designed to process data represented as graphs, which consist of nodes and edges. GNNs are particularly useful for tasks involving relational data, such as networks of cells (e.g., in spatial omics). They leverage the structure of the graph to learn representations for nodes, edges, or the entire graph. The operation of a GNN layer can be described as:

$$\mathbf{h}_i^{(l+1)} = f\left(\mathbf{W}^{(l)}\mathbf{h}_i^{(l)} + \sum_{j \in \mathcal{N}(i)} \mathbf{W}_e^{(l)}\mathbf{h}_j^{(l)}\right)$$

where $\mathbf{h}_i^{(l+1)}$ is the updated representation of node i at layer $l+1$, $\mathbf{W}^{(l)}$ and $\mathbf{W}_e^{(l)}$ are weight matrices, and $\mathcal{N}(i)$ represents the neighbors of node i. GNNs can capture complex dependencies and interactions within the graph, making them powerful for a variety of applications. There are many types of GNN, and some involve attention mechanisms (known as Graph Attention Networks, or GAT [Brody et al., 2022]).

2.4.3 . Deep learning for omics data analysis

Deep learning has shown great promise in analyzing omics data, particularly single-cell and spatial omics data. Notably, deep learning led to significant advancements in various tasks, including segmentation, clustering, and cell-type annotation. Below are detailed examples of applications in omics data analysis.

Cell segmentation

In imaging data, CNNs are employed to accurately segment individual cells and sub-cellular structures. Architectures like U-Net [Ronneberger et al., 2015] and its variants are widely used due to their ability to capture fine details and spatial patterns, enabling precise delineation of cells in complex tissue environments. Examples of deep learning tools used for segmentation include Cellpose [Stringer et al., 2021], an open-source framework for cell segmentation that works across a wide variety of imaging modalities using a U-Net-based approach, and StarDist [Schmidt et al., 2018], a deep learning method that represents objects by star-convex polygons, which is particularly effective for segmenting objects with varying shapes and sizes, such as nuclei in microscopy images.

Representation learning

Representation learning aims to reduce the dimensionality of omics data while preserving its essential features, facilitating tasks such as clustering, classification, and visualization. Deep learning methods like autoencoders and in particular variational autoencoders (VAEs [Kingma and Welling, 2019]) are particularly effective for this task. VAEs, in addition to providing low-dimensional representations, are generative models that learn the underlying distribution of the data. This is invaluable for tasks such as data imputation (very useful in transcriptomics, where the data is very sparse). Another technique, variational inference, can be used to approximate complex probability distributions. It allows for scalable and efficient inference in models with high-dimensional latent variables. A successful application of variational inference in omics data analysis is the scvi-tools ecosystem, which provides a comprehensive suite of tools for single-cell analysis. In particular, the scVI [Lopez et al., 2018] model within this suite uses variational inference to learn a latent representation of single-cell RNA sequencing data, enabling robust normalization, clustering, differential expression analysis, and integration of multiple datasets.

In brief, deep learning has transformed omics data analysis by providing sophisticated methods for segmentation, representation learning, generative modeling, and variational inference. Tools like Cellpose and StarDist enhance segmentation accuracy, while VAEs and frameworks like scVI enables robust data integration and analysis. These techniques enable researchers to handle the complexity and scale of omics data, uncovering valuable insights into cellular functions, disease mechanisms, and potential therapeutic targets. The integration of deep learning into omics research is driving significant advancements in precision medicine, disease research, and our fundamental understanding of biology. In spatial omics data analysis, a relatively recent field, deep learning methods are now predominant, which can be explained by the high complexity of the data and the large size of datasets, which are very suitable conditions for applying deep learning.

Biology-driven deep generative modelling for cytometry data analysis

Contents

Abstract

Cytometry enables precise single-cell phenotyping within heterogeneous populations. These cell types are traditionally annotated via manual gating, but this method lacks reproducibility and sensitivity to batch effect. Also, the most recent cytometers — spectral flow or mass cytometers — create rich and high-dimensional data whose analysis via manual gating becomes challenging and time-consuming. To tackle these limitations, we introduce Scyan (`https://github.com/MICS-Lab/scyan`), a Single-cell Cytometry Annotation Network that automatically annotates cell types using only prior expert knowledge about the cytometry panel. For this, it uses a normalizing flow — a type of deep generative model — that maps protein expressions into a biologically relevant latent space. We demonstrate that Scyan significantly outperforms the related state-of-the-art models on multiple public datasets while being faster and interpretable. In addition, Scyan overcomes several complementary tasks, such as batch-effect correction, debarcoding, and population discovery. Overall, this model accelerates and eases cell population characterization, quantification, and discovery in cytometry.

3.1 . Context and motivation

The simultaneous detection of several cellular proteins by spectral and mass cytometry opens up an unprecedented way to detect, quantify, and monitor the function of highly specific cell populations from complex biological samples [Behbehani, 2019]. These rich analyses are made possible using large panels of markers, typically more than 30 or 40 markers, which considerably increases the information in the data [Spitzer and Nolan, 2016]. They provide key insights to better understand specific diseases, immune cell functions, or monitor the response to therapies [McKinnon, 2018].

When studying cytometry, population annotation must be performed to provide each cell with a biologically meaningful cell type. Yet, due to the data's high dimensionality and complexity, manual annotations become challenging and labor intensive [Newell and Cheng, 2016]. This process, called gating [Staats et al., 2019], corresponds to the visual inspection of a series of two-dimensional scatterplots. At each step, a subset of cells, either positive or negative for the two visualized markers, is selected and further stratified in the subsequent iterations until populations of interest are captured (see Figure 3.2 for more details). Manual gating is very time-consuming and subjective, with a bias towards well-known cell types [Newell and Cheng, 2016]. These drawbacks are amplified as the number of cytometry samples increases, reinforcing the need to develop and use automatic tools in population annotation and data analysis [Newell and Cheng, 2016, Aghaeepour et al., 2013]. Usually, a few scatterplots (up to 20) are used during manual gating, which is far

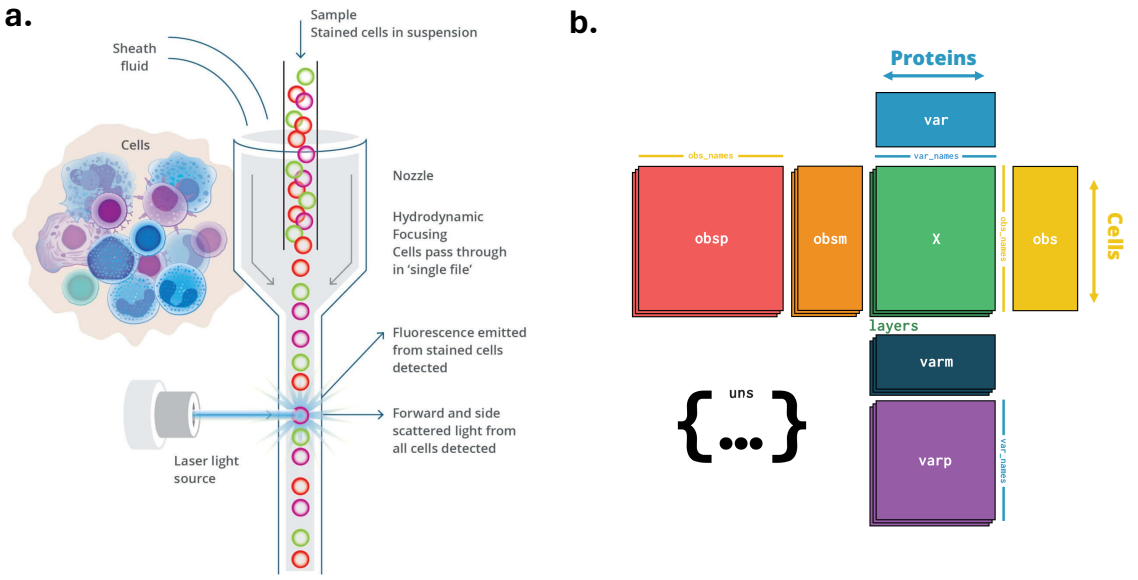

Figure 3.1: ***Overview of cytometry data.*** ***a.*** *Illustration of the process to acquire cytometry data. Cells are stained, and then, using a laser, the fluorescence emitted by each single-cell is measured. This allows us to measure the expression of the proteins used when staining the cells.* ***b.*** *After the experiment, the data can be transformed into a cell-by-protein table, usually using an AnnData [Virshup et al., 2021] object. The number of cells can reach millions, or hundreds of millions, while the number of proteins typically goes up to 50.*

less than the total number of 2D-scatterplots. Indeed, when the panel consists of 40 different markers, it is possible to generate up to 820 different 2D-scatterplots.

Many clustering tools [Levine et al., 2015, Traag et al., 2019, Qiu et al., 2011] have been developed for automatic data exploration and population discovery. However, a manual analysis of marker expressions is still required to name each cluster with a meaningful cell type. Indeed, clusters do not necessarily correspond to one specific cell type, and it is up to the investigator to decide to which population each cluster corresponds. Clustering tools also do not scale well and are sensitive to batch effects, making this approach less suited for large datasets with a large inter-sample variation. An alternative approach to clustering is to use automatic annotation models. The first category of annotation models are supervised or semi-supervised models [Li et al., 2017, Abdelaal et al., 2019, Kaushik et al., 2021, Liu et al., 2020], but they rely on prior manual gating to train the models. Moreover, these models can only annotate populations with predefined types of cells, and they cannot be used to discover new ones. The second category, to which our model belongs, corresponds to unsupervised annotation models that leverage prior biological knowledge about the panel of markers. Although some models have been developed [Lee et al., 2017, Ji et al., 2018, Liu et al., 2020], they either (i) lack interpretability, (ii) cannot discover new populations, (iii) require the usage of batch-effect correction models before being applied, or (iv) scale poorly to large datasets. Surprisingly, deep learning has been un-

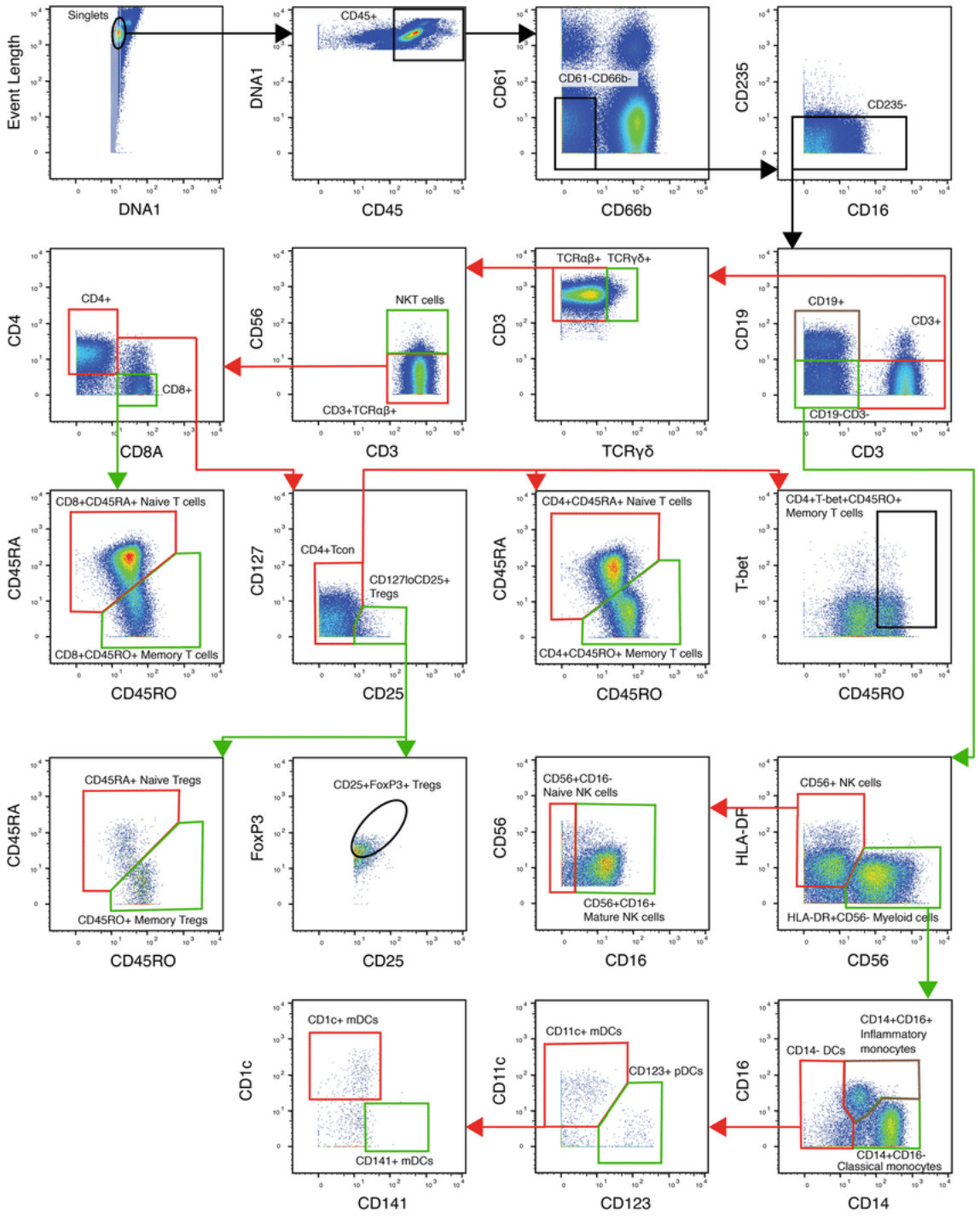

Figure 3.2: ***Illustration of population annotation by manual gating.*** *The manual gating process consists of sequentially selecting cell populations based on scatterplots of the expression of one or several markers (usually, two markers, as shown in this figure). The selections are performed manually, usually by defining a rectangle or a polygon around the dots of interest on the 2D scatterplot.*

derused for cytometry annotations, while proving efficient and flexible for many related applications of single-cell biology [Lopez et al., 2018, Zhang et al., 2019, Amodio et al., 2019].

In this chapter, we introduce a single-cell cytometry annotation network called Scyan that annotates cell types and corrects batch effects concurrently without any label or gating needed. Scyan is a Bayesian probabilistic model composed of a deep invertible neural network called a normalizing flow [Rezende and Mohamed, 2015, Papamakarios et al., 2021, Izmailov et al., 2020]. This flow transforms cell data into a latent space that is used for annotation, does not contain batch effect, and is key for population discovery.

We demonstrate Scyan efficiency, scalability, and interpretability on three public mass cytometry datasets for which manually annotated cell populations are used to evaluate models. We compare Scyan classification performance to two knowledge-based approaches [Lee et al., 2017, Ji et al., 2018], one clustering method [Levine et al., 2015, Lee et al., 2017], and two supervised models [Abdelaal et al., 2019, Kaushik et al., 2021]. Additionally, we compare Scyan batch-effect correction to four state-of-the-art batch correction methods [Korsunsky et al., 2019, Lun et al., 2017, Johnson et al., 2007, Amodio et al., 2019]. We also show that our model can be used for population discovery, as well as for the general task of debarcoding. Overall, these properties make Scyan an end-to-end analysis framework for mass/spectral/flow cytometry.

3.2 . Methods

Let $\mathbf{x_1}, \dots, \mathbf{x_N} \in \mathbb{R}^M$ represent the vectors of M marker expressions for N cells. We assume these expression levels have already been transformed using the $asinh$ or logicle [Parks et al., 2006] transformation and standardized (see supplementary subsection A.1.11). Our objective is to associate each cell to one of the P predefined cell types using a marker-population table $\rho \in \mathbb{R}^{P \times M}$, with $\rho_{z,m}$ summarising the knowledge about the expression of marker m for population z. If it is known that population z expresses m then $\rho_{z,m} = 1$; if we know that it does not express m then $\rho_{z,m} = -1$. Otherwise, if we have no knowledge or if the expression can vary among the population, then $\rho_{z,m} =$ NA. Note that it is also possible to choose values in $\mathbb{R}$; for instance, for mid or low expressions, we can choose 0 and 0.5, respectively (see supplementary subsection A.1.6). In addition, we can add covariates $\mathbf{c_1}, \dots, \mathbf{c_N} \in \mathbb{R}^{M_c}$ associated with each cell, e.g., information about the batch or which antibody has been used by the cytometer. M_c denotes the number of covariates; it can be zero if no covariate is provided.

3.2.1 . Broad overview of the Scyan methodology

This short section details broadly the method behind Scyan. It is described in more details in the following sections.

Scyan is composed of two core components: (i) f_ϕ, a neural network called normalizing flow, and (ii) a latent space on which a target distribution $\mathbf{U}$ is defined (Figure 3.3b). This target distribution is a mixture of distributions — one per population — built using prior biological knowledge about the cell types. This knowledge is provided as a table: for all populations, each expected marker expression is given or left unknown (more details in supplementary subsection A.1.6). This table is then used to mathematically define the target distribution $\mathbf{U}$. Also, the latent space (on which $\mathbf{U}$ is defined) has the same dimension as the original space; therefore, each marker has its corresponding latent expression.

The purpose of the normalizing flow is to learn an invertible mapping between the actual marker expression distribution and the target $\mathbf{U}$. By mapping marker expressions to a biologically-defined latent space, we force the transformation to provide latent expressions on a scale that is shared for every marker, going from negative (-1) to positive (+1). These latent marker expressions are meant to be free of batch effect or any non-biological factor. By the design of f_ϕ and of the objective function, the normalizing flow is not allowed to make huge space distortions, which helps preserve the biology. After learning the model parameters ϕ, annotations are performed on the latent space. We annotate a cell by choosing the population distribution whose likelihood is the highest for the cell latent representation. If a cell latent representation does not correspond to any component of the mixture, then the cell remains unlabelled, but population discovery can be run afterward to annotate it eventually (see Figure 3.6).

3.2.2 . Generative process

In this section, we detail the generative process of Scyan (illustrated in Figure 3.3b/c). Let $\mathbf{X}$ be the random vector of size M representing one cell by its standardized marker expressions; in other words, $\mathbf{X}$ is the random variable from which $\mathbf{x_1}, \dots, \mathbf{x_N}$ are sampled. We model $\mathbf{X}$ by the following deep generative process:

$$
\begin{aligned}
Z &\sim Categorical(\pi) \\
\mathbf{E} \mid Z = (e_m)_{1\leq m\leq M}, \text{ where } &\begin{cases} e_m = \rho_{Z,m} & \text{if } \rho_{Z,m} \neq \text{NA} \\ e_m \sim \mathcal{U}([-1,1]) & \text{otherwise,} \end{cases} \\
\mathbf{H} &\sim \mathcal{N}(\mathbf{0}, \sigma \mathbf{I_M}) \\
\mathbf{U} &= \mathbf{E} + \mathbf{H} \\
\mathbf{X} &= f_\phi^{-1}(\mathbf{U}).
\end{aligned}
\tag{3.1}
$$

In the above equations, $\pi = (\pi_z)_{1\leq z\leq P}$ represents the weights of each population, with the constraints $\pi_z \geq 0$ and $\sum_z \pi_z = 1$. Z is the random variable corresponding to a cell type among the P possible ones. $\mathbf{E}$ is a population-specific variable whose terms are either known according to the expert knowledge table ρ or drawn from a uniform distribution between negative expressions (represented by -1) and positive expressions (represented by +1). $\mathbf{H}$ contains cell-specific terms, such as autofluorescence. Finally,

$\mathbf{U}$ is the cell's latent expressions, summing a population-specific component and a cell-specific one. Note that $\mathbf{E}$, $\mathbf{H}$ and $\mathbf{U}$ have a dimension of M. Also, $\mathbf{U}$ can be transformed into a measured cell marker expressions vector $\mathbf{X}$ by the inverse of a deep invertible network f_ϕ detailed below. The normalizing flow aims to learn an invertible mapping between the actual marker expression distribution and the target $\mathbf{U}$. By mapping marker expressions to a biologically-defined latent space, we force the transformation to provide latent expressions on a scale that is shared for every marker, going from negative (-1) to positive (+1). These latent marker expressions are meant to be free of batch effect or any non-biological factor. By the design of f_ϕ and of the objective function, the normalizing flow is not allowed to make huge space distortions, which helps preserve the biology. Also, an ablation study shows that both f_ϕ and the uniform term are key for good performances (see supplementary Table A.2).

3.2.3 . Invertible transformation network

The core network, f_ϕ (illustrated in Figure 3.3b), is a normalizing flow [Rezende and Mohamed, 2015, Papamakarios et al., 2021, Izmailov et al., 2020]. It transforms the target distribution p_X into the known base distribution p_U, which was described in the previous section. Using a change of variables, we can compute the exact likelihood of a sample $\mathbf{x}$ by:

$$p_X(\mathbf{x};\theta) = p_U(f_\phi(\mathbf{x});\pi) \cdot \left| det \frac{\partial f_\phi(\mathbf{x})}{\partial \mathbf{x}^T} \right|. \tag{3.2}$$

To be able to compute this expression, we need to choose an invertible network with a tractable Jacobian determinant. We have chosen a set of transformations called Real Non-Volume-Preserving (Real NVP [Dinh et al., 2017], we justify this choice in supplementary subsection A.1.1) transformations, which are compositions of functions named coupling layers $f_\phi := f^{(L)} \circ f^{(L-1)} \circ \cdots \circ f^{(1)}$ with L the number of coupling layers. Each coupling layer $f^{(i)} : (\mathbf{x}, \mathbf{c}) \mapsto \mathbf{y}$ splits both $\mathbf{x}$ and $\mathbf{y}$ into two components $(\mathbf{x^{(1)}}, \mathbf{x^{(2)}}), (\mathbf{y^{(1)}}, \mathbf{y^{(2)}})$ on which distinct transformations are applied. We propose below an extension of the traditional coupling layer [Dinh et al., 2017] to integrate covariates $\mathbf{c}$ (illustrated in Figure 3.3c):

$$\begin{cases} \mathbf{y^{(1)}} = \mathbf{x^{(1)}} \\ \mathbf{y^{(2)}} = \mathbf{x^{(2)}} \odot exp\Big(s([\mathbf{x^{(1)}}; \mathbf{c}])\Big) + t([\mathbf{x^{(1)}}; \mathbf{c}]). \end{cases} \tag{3.3}$$

In the equations above, $\odot$ stands for the element-wise product, $[.;.]$ is the concatenation operator, and (s, t) are functions from $\mathbb{R}^{d+M_c}$ to $\mathbb{R}^{M-d}$ where d is the size of $\mathbf{x^{(1)}}$. These functions can be arbitrarily complex, in our case, multi-layer-perceptrons. Note that the indices used by the coupling layer to split $\mathbf{x}$ into $(\mathbf{x^{(1)}}, \mathbf{x^{(2)}})$ are set before training and are different for every coupling layer. This way, we ensure that the flow transforms all the markers. Each coupling layer has an easy-to-compute log Jacobian determinant, which is $\sum_i s([\mathbf{x^{(1)}}; \mathbf{c}])_i$, and is easily invertible as shown in the following equations:

$$\begin{cases} \mathbf{x^{(1)}} = \mathbf{y^{(1)}} \\ \mathbf{x^{(2)}} = (\mathbf{y^{(2)}} - t([\mathbf{y^{(1)}}; \mathbf{c}])) \odot exp\Big(- s([\mathbf{y^{(1)}}; \mathbf{c}])\Big). \end{cases} \tag{3.4}$$

As f_ϕ is a stack of coupling layers, it is also invertible, and its log Jacobian determinant is obtained by summing each coupling layer log Jacobian determinant. Stacking many coupling layers is essential to learning a rich target distribution and complex variables interdependencies. Overall, the normalizing flow has some interesting properties: (i) the coupling layers preserve order relation for two different expression values, and (ii) penalize huge space distortion (the log determinant term). The two properties are useful to preserve biological variability as much as possible.

3.2.4 . Learning process

The model parameters are $\theta = (\pi, \phi)$. For computational stability during training, instead of learning π itself we actually learn logits $(l_z)_{1 \leq z \leq P}$ from which we obtain $\pi_z = \frac{e^{l_z}}{\sum_k e^{l_k}}$. By doing this, we ensure the positivity of each weight and guarantee they sum to 1. To train the model, we minimize the Kullback-Leibler (KL) divergence between the cell's empirical marker-expression distribution p_{X^*} and our model distribution p_X. It is equivalent to minimizing the negative log-likelihood of the observed cell expressions (mathematically defined as $-\mathbb{E}_{\mathbf{x} \sim p_{X^*}}\Big[log\ p_X(\mathbf{x}; \theta)\Big]$) over θ. Using Equation 3.2 and adapting it to integrate covariates leads to minimizing the following quantity:

$$\mathcal{L}_{KL}(\theta) = -\sum_{1 \leq i \leq N} \left[log\left(p_U(f_\phi(\mathbf{x_i}, \mathbf{c_i}); \pi)\right) + log\left|det \frac{\partial f_\phi(\mathbf{x_i}, \mathbf{c_i})}{\partial \mathbf{x}^T}\right|\right]. \tag{3.5}$$

In the above equation, the term $p_U(f_\phi(\mathbf{x_i}, \mathbf{c_i}); \pi) = \sum_{z=1}^{P} \pi_z \cdot p_{U|Z=z}(f_\phi(\mathbf{x_i}, \mathbf{c_i}))$, which is not computationally tractable because the presence of NA in ρ leads to the summation of a uniform and a normal random variable. We approximate the density of the sum of the two random variables by a piecewise density function that is constant on $[-1+\sigma, 1-\sigma]$ with Gaussian queues outside of this interval. In practice, we choose a normal law with a low standard deviation, which leads to a good piecewise approximation (see supplemental subsection A.1.2). If we consider the KL-divergence as described above, some modes may collapse; that is, one small population may not be predicted. Indeed, a small population z that has a small weight π_z leads to smaller gradients towards this population. To solve this issue, we favor small populations once every two epochs. For that, for all z, we replace π_z by $\pi_z^{(-T)} = \frac{e^{-l_z/T}}{\sum_k e^{-l_k/T}}$ where T is called temperature [Ackley et al., 1985, Ficler and Goldberg, 2017] as it increases the entropy of $\pi^{(-T)}$. Note that here we added the minus signs to reverse the weights of the populations so that it favors small ones. A temperature close to 0 leads to high weights for small populations, while an infinite temperature leads to equal population weights, i.e., the maximum entropy. Alternating between π and $\pi^{(-T)}$ allows for a better balance of population sizes at the end of the training.

We optimize the loss on mini-batches of cells using the Adam optimizer [Kingma and Ba, 2015]. Once finished training, the annotation process $\mathcal{A}_\theta$ consists in choosing the most likely population according to the data using Bayes's rule. So, for a cell $\mathbf{x}$ with covariates $\mathbf{c}$, we have:

$$\mathcal{A}_\theta(\mathbf{x}, \mathbf{c}) = argmax_{1 \leq z \leq P} \ \ \pi_z \cdot p_{U|Z=z}(f_\phi(\mathbf{x}, \mathbf{c})). \tag{3.6}$$

We also define a log threshold t_{min} to decide whether or not to label a cell (see supplementary subsection A.1.4 to determine its value). That is, we don't label a cell if:

$$max_{1 \leq z \leq P} \ \ p_{U|Z=z}(f_\phi(\mathbf{x}, \mathbf{c})) \leq e^{t_{min}}.$$

3.2.5 . Batch-effect correction

Batch information is one-hot-encoded and added to the covariates (see covariates usage in Equation 3.4). Minimizing $\mathcal{L}_{KL}(\theta)$ (as in Equation 3.5) will naturally reduce inter-batch variability in the latent space. Note that we don't need to add any additional loss terms, and Scyan will naturally use the batch information inside the covariates to better align the distribution of the different batches on the target distribution $\mathbf{U}$. Since the normalizing flow is invertible, we can conserve this batch alignment when mapping back in the original space by using a simple trick: we map all cells back with the same reference covariates $\mathbf{c_{ref}}$. Note that $\mathbf{c_{ref}}$ is simply one of the existing covariates vectors; usually, we choose the covariates of the patient with the most cells. More formally, to correct the batch effect of a sample $\mathbf{x}$ with covariates $\mathbf{c} \neq \mathbf{c_{ref}}$, we first transform $\mathbf{x}$ into its latent expressions via f_ϕ. Since the latent space is batch-effect free, latent expressions can then be transformed back into the original space using the covariates of the reference batch and f_ϕ^{-1}. Formally, we denote by $\tilde{\mathbf{x}}$ the batch-effect corrected cell associated to $\mathbf{x}$; that is, $\tilde{\mathbf{x}} = f_\phi^{-1}\Big(f_\phi(\mathbf{x}, \mathbf{c}), \mathbf{c_{ref}}\Big)$. In this manner, we get expressions $\tilde{\mathbf{x}}$ as if $\mathbf{x}$ were cell expressions from the reference batch. After training, we can also infer the missing values ("NA") from the table by using the mean latent expressions for all populations: it allows to refine batch correction for all markers.

3.2.6 . Interpretability and population discovery

Understanding Scyan predictions

One important thing to notice is that $U_1 \mid (Z = z), \dots, U_M \mid (Z = z)$ are independent for every population z. It means that we can decompose $log\ p_{U|Z=z}(\mathbf{u})$ as follow: $log\ p_{U|Z=z}(\mathbf{u}) = \sum_m log\ p_{U_m|Z=z}(u_m)$, and we can gather all these terms into a matrix of scores $\Big(log\ p_{U_m|Z=z}(u_m)\Big)_{z,m}$. The term $log\ p_{U_m|Z=z}(u_m)$ can be interpreted as the impact of marker m towards the prediction of the population z for the latent cell expression $\mathbf{u}$. Based on that, we can interpret Scyan predictions for a group of cells $(\mathbf{x_i}, \mathbf{c_i})_i$ by transforming the cells into their latent expressions and then averaging the score matrices. The resulting matrix is typically displayed on a heatmap (Figure 3.6d), and populations are

sorted by their score (sum over a score matrix row). Note that, in the figure, each population score is scaled to make it easier to read.

Latent expressions

Considering a cell $\mathbf{x}$ and its covariates $\mathbf{c}$, its latent representation is $\mathbf{u} = f_\phi(\mathbf{x}, \mathbf{c})$. The information of which marker is positive or negative is contained in $\mathbf{u}$. Indeed, $u_m \approx 1$ corresponds to a positive expression, while $u_m \approx -1$ represents a negative expression, whatever the marker m (i.e., expression levels for all markers are unified). Similarly, $u_m \approx 0$ is a mid-expression, and so on. We average the latent cell expressions over one population to obtain a latent expression at the population level. These population-level latent expressions can be displayed for one population (Figure 3.6c) or for all of them at once (Figure 3.6b).

3.2.7 . Benchmark-related methods

Datasets used

We compare Scyan to the related works on three public mass cytometry datasets. One is from patients with acute myeloid leukemia [Levine et al., 2015] (AML, N = 104 184 cells, mass cytometry), one from bone marrow mononuclear cells [Bendall et al., 2011] (BMMC, N = 61 725 cells, mass cytometry), and the last one from peripheral blood mononuclear cells (PBMCs) samples of peanut-allergic individuals [Chinthrajah et al., 2019] (POISED, N = 4 178 320 cells, mass cytometry). The latter contains 30 samples, divided among 7 batches, and under two different conditions (peanut stimulated or unstimulated). Finally, one flow cytometry dataset [Zunder et al., 2015] has been used for debarcoding (N=100 000 cells). Manual gating has been performed in previous studies [Levine et al., 2015, Bendall et al., 2011, Chinthrajah et al., 2019], providing ground truth labels to evaluate annotation models. Note that the unsupervised models listed below do not use these labels during training.

Compared models

We compared Scyan to six other annotation models: three knowledge-based models (ACDC [Lee et al., 2017], a baseline model defined by the authors of ACDC, and MP [Ji et al., 2018]), one clustering method (Phenograph [Levine et al., 2015]), and two supervised models (LDA [Abdelaal et al., 2019] and CyAnno [Kaushik et al., 2021]). We also benchmarked our model ability to correct batch effect to four models: Cydar [Lun et al., 2017], Combat [Johnson et al., 2007], SAUCIE [Amodio et al., 2019], and Harmony [Korsunsky et al., 2019]. We used the POISED dataset on which we had 7 biological batches, and we amplified the batch effect to complex the batch correction (see subsection 3.2.7).

Evaluation

We evaluated the models for the classification task using accuracy, macro averaged F1-score, and balanced accuracy [Jiao and Du, 2016]. The results are detailed in Figure 3.4a. For the debarcoding task, the Silhouette score and the Calinski Harabasz Score were used. All the above metrics were implemented in Scikit-learn [Pedregosa et al., 2011]. Concerning the batch-effect-correction task, we provide two metrics: the cell-type LISI (cLISI), which measures if the biological variability is kept, while the integration LISI (iLISI) measures how well the batches overlap, i.e., if the batch-effect was corrected. We used the implementation from Harmony [Korsunsky et al., 2019]. For more details, see supplementary Figure A.6.

Batch effect amplification

On the POISED dataset, we amplified the batch effect so that the benchmark becomes more complex. Let $\sigma_{BE} > 0$ a scale factor, and $b_1, \ldots, b_N \in [1 \ldots 7]$ the batch number associated to each of the N cells. Then, we sample 7 matrices $\mathbf{S}_1, \ldots, \mathbf{S}_7 \in \mathbb{R}^{M \times M}$, whose elements are drawn from $\mathcal{N}(0, \sigma_{BE})$. For a cell i of expression $\mathbf{x}_i$, the batch-effect-amplified expression $\mathbf{x}_i^{'}$ is multiplied by some batch-relative term: $\mathbf{x}_i^{'} = (\mathbf{I}_M + \mathbf{S}_{b_i})\mathbf{x}_i$. In this equation, $\mathbf{I}_M$ is the identity matrix of size $M \times M$, and the multiplication operation is matrix multiplication. In practice, we use $\sigma_{BE} = 0.01$. Note that the UMAPs were computed on the cell-type related markers and did not use cell-state markers.

Implementation details and hyperoptimization

We implemented our model using Python and the Deep Learning framework Pytorch [Paszke et al., 2019]. We used between 6 and 8 coupling layers whose multi layer perceptrons (s, t) have each between 6 and 8 hidden layers depending on hyperparameter optimization. The hidden layer size can vary between 16 and 32. Model hyperoptimization can be performed using an unsupervised heuristic (see supplementary subsection A.1.9), but Scyan is robust to small changes of the main hyperparameter σ (see supplementary subsection A.1.5).

3.3 . Results

3.3.1 . Scyan provides a better and faster annotation than unsupervised methods

Classification metrics comparison

We evaluated Scyan to four unsupervised or semi-supervised models for the classification task (ACDC and its baseline [Lee et al., 2017], MP [Ji et al., 2018], and Phenograph [Levine et al., 2015]), on three public datasets, and over 3 different metrics (see subsec-

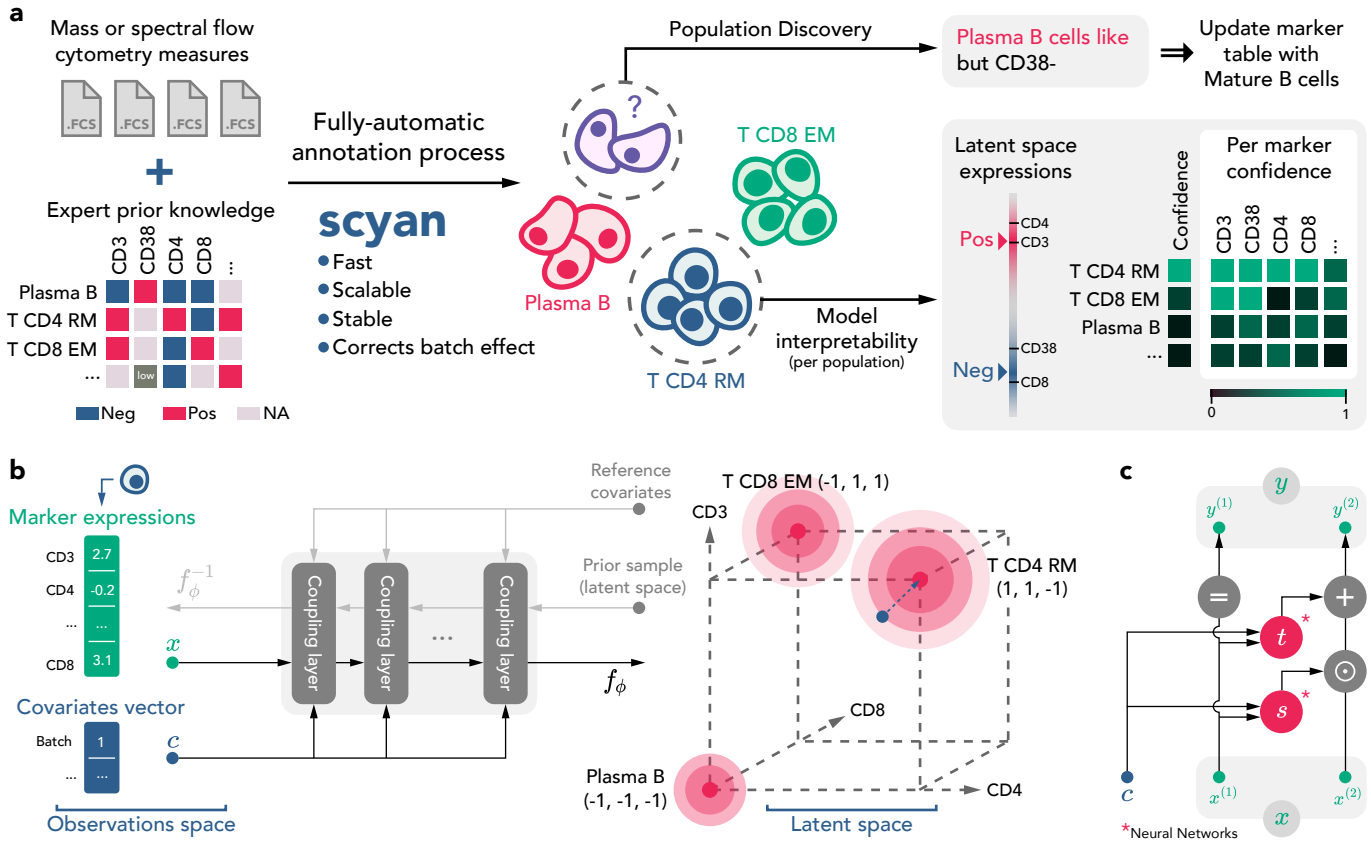

Figure 3.3: ***Overview of Scyan usage and architecture.*** ***a****, Illustration of Scyan typical use case. It requires (i) one or multiple cytometry acquisitions and (ii) a knowledge table that details which population is expected to express which markers: Then, Scyan annotates cells in a fast and unsupervised (or fully-automatic) manner while removing batch effect (if any). After training, we provide interpretability tools to understand Scyan annotations and discover new populations that can eventually be added to the table afterward.* ***b****, Illustration of Scyan architecture: it is composed of two core components: (i)* f_ϕ*, a neural network called normalizing flow, and (ii) a latent space on which a target distribution* $\mathbf{U}$ *is defined (cube on the right). The table from (a) is used to define this target distribution* $\mathbf{U}$ *mathematically. Also, the latent space (on which* $\mathbf{U}$ *is defined) has the same dimension as the original space; therefore, each marker has its corresponding latent expression. Finally, one cell is represented by its marker expressions vector and eventual covariates. Once a cell is mapped into the latent space, annotation can be made by choosing the highest probable population, whose distribution is Gaussian-like and on a hypercube vertex.* ***c****, One coupling layer, the elementary unit that composes the transformation* f_ϕ*, contains two multi-layer perceptrons (*s *and* t*) and uses cell covariates such as the batch information.*

tion 3.2.7 for more details). The tests show that Scyan outperforms the other models. In particular, Scyan is about 20 points higher than the other models on POISED and BMMC for the F1-score and the balanced accuracy, which is explained by the capacity of Scyan to detect better small populations (Figure 3.4a). On these datasets, multiple populations represent less than one percent of the total number of cells, making these populations more difficult to detect and label. Yet, small population annotations can still be essential, and thus so is Scyan's capacity to detect them. Also, the gap between Scyan and the other models is more stringent for POISED, showing our model's ability to better annotate large, complex datasets with batch effect.

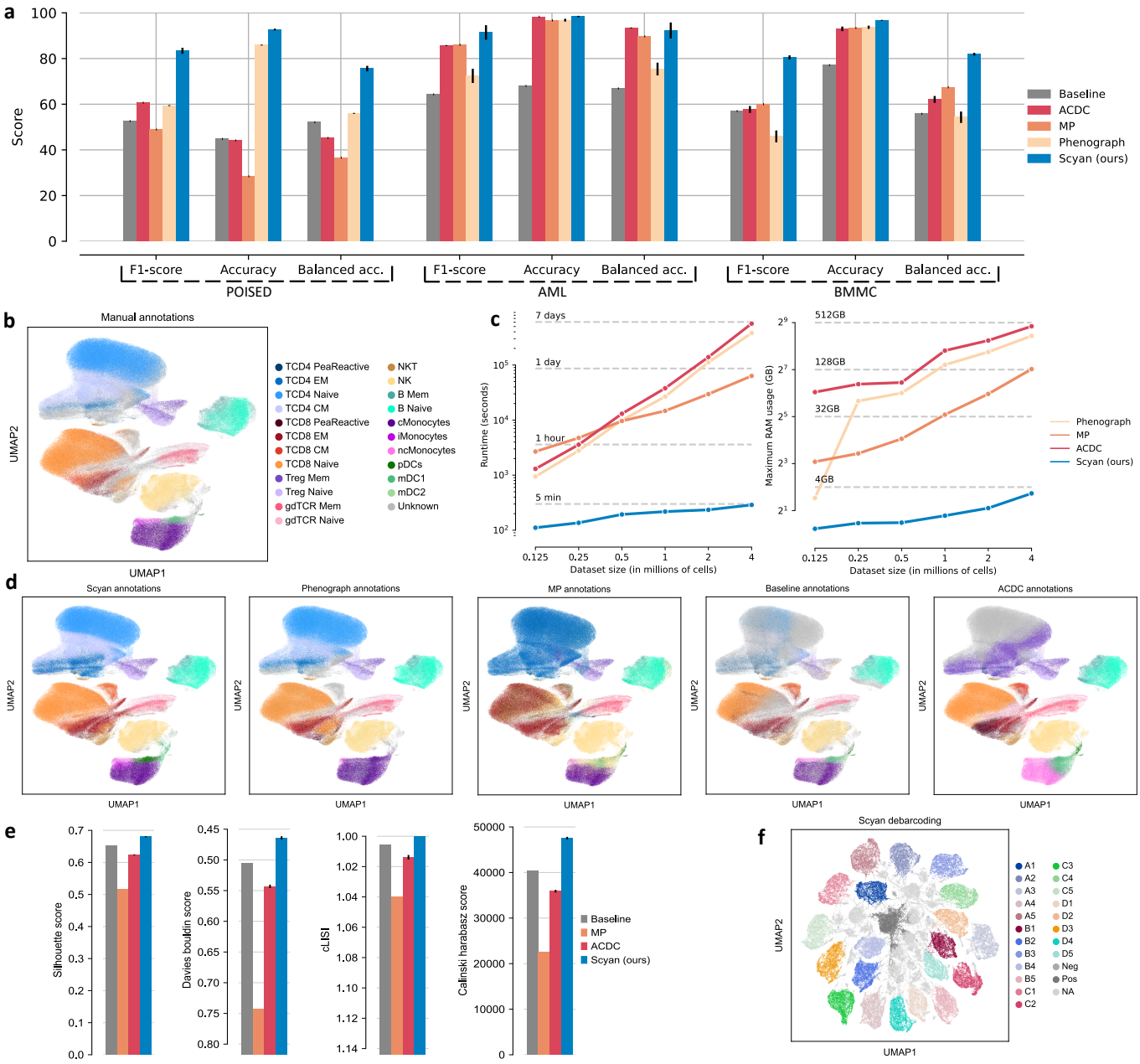

Figure 3.4: ***Comparison to state-of-the-art unsupervised methods.*** ***a****, Performance comparison of Scyan and four other unsupervised methods on three datasets (POISED, AML, BMMC) using three metrics for each.* ***b****, UMAP representing the manually annotated populations on the POISED dataset.* ***c****, Models runtime comparison (left) and RAM usage comparison (right) over multiple dataset sizes.* ***d****, UMAPs representations of the annotations of all five models on the POISED dataset.* ***e****, Unsupervised metrics for the debarcoding task.* ***f****, UMAP representing Scyan debarcoding. Cells that did not correspond to any desired barcode were left unclassified (NA).*

Computational speed, scalability, and memory usage

To demonstrate the scalability of Scyan on large datasets, we compare the execution times and the random access memory (RAM) usage of the different algorithms over multiple dataset sizes (Figure 3.4c). The different sizes were obtained by sub-sampling the POISED dataset, for various sample sizes from 125,000 to 4 million cells. All experiments

were run using the same hardware; in our case, CPUs only (i.e. no GPU acceleration, even though Scyan can use GPUs, supplementary subsection A.1.12). On $N = 4$ million cells, Scyan runs in five minutes, while ACDC/MP/Phenograph need between one day and seven days. Scyan scales well to large datasets, as shown by the low slope on Figure 3.4c. Concerning RAM consumption, Scyan uses less than 4GB of RAM, which means it can be run on any standard laptop. In comparison, ACDC, MP, and Phenograph all require between 128GB and 512GB of RAM, which is only available on large computer clusters. See supplementary subsection A.1.8 for comments about supervised models runtime.

Comparison for barcoding deconvolution

Barcoding is a method that reduces the batch effect and data variability by allowing the processing of multiple cell samples together, each cell sample being labeled — or barcoded — with a unique combination of antibodies. This protocol requires (i) the dedication of a few markers to make barcodes and (ii) the identification of each cell sample based on its barcode. The latter task, called debarcoding [Zunder et al., 2015], can also be expressed as a knowledge-based annotation task. In this situation, we annotate samples instead of populations, and the expert knowledge required for this task simply corresponds to the known barcodes. Figure 3.4e shows that Scyan outperforms ACDC, MP, and the baseline on a public dataset with 20 barcodes and 6 markers [Zunder et al., 2015].The UMAP on Figure 3.4f shows a clear separation of the different barcodes, with some small residual clusters (not to be considered) corresponding to non-existing barcodes. The UMAPs corresponding to the debarcoding of the other methods can be found in supplementary Figure A.7.

3.3.2 . Scyan corrects batch effect

A batch effect is a phenomenon that induces data variability due to non-biological factors such as the use of a different antibody or slightly different cytometer settings. In practice, these factors may introduce variability that interferes with the analysis and can lead to confusion, over-interpretation, and difficulties in annotating populations. To tackle this issue, Scyan can align the distribution of different batches (see methods subsection 3.2.5). Classically, batch effect correction is performed before annotation, but our method allows for correcting it at the same time as the annotation. Taking into account the batch helps Scyan to annotate the populations better. Figure 3.5a shows the batch effect we want to correct (from the amplified POISED dataset, see subsection 3.2.7). The next figures, i.e., Figure 3.5b/c, show that Cydar's and Combat's corrections are very limited, even though they keep the biological variability. SAUCIE provides the best iLISI, so it mixes well the different batches, but it also removes most of the biological variability (high cLISI). On the opposite, Scyan and Harmony successfully remove the batch effect while preserving the biological variability. Another benchmark on POISED without amplification can be found in supplementary Figure A.6.

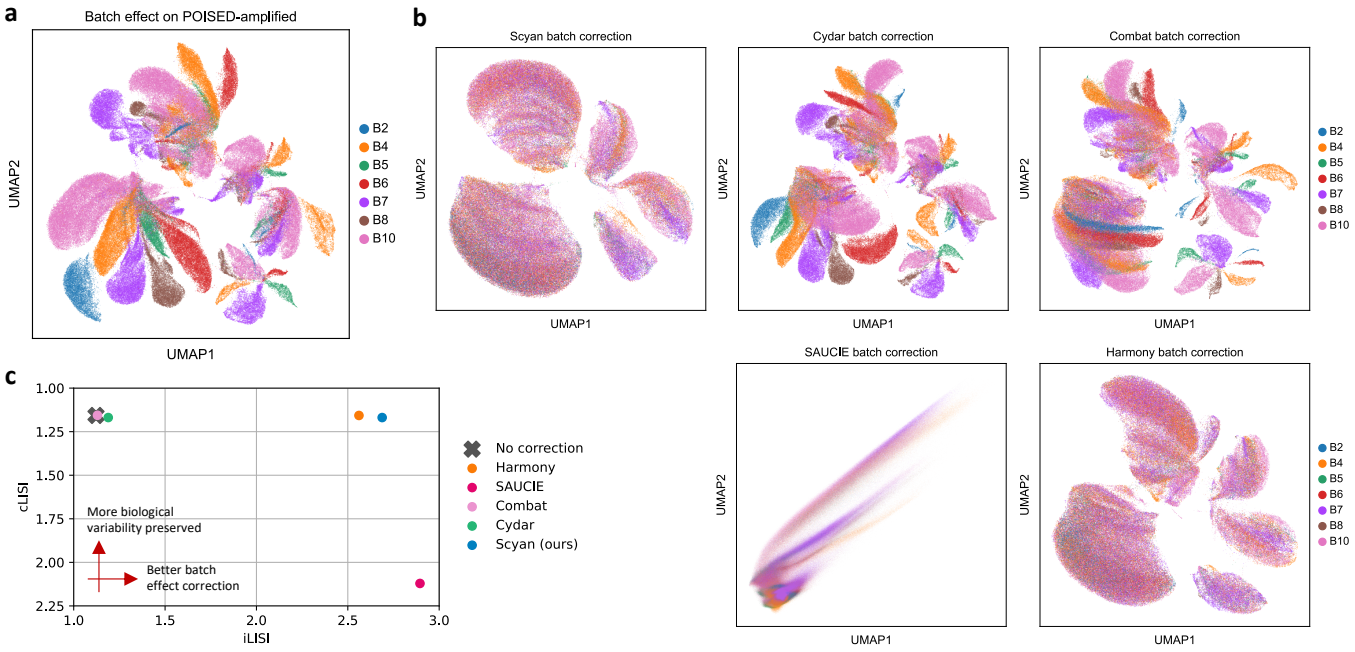

Figure 3.5: ***Batch-effect correction on the POISED dataset with batch-effect amplification.*** ***a****, UMAP showing the 7 different batches (before batch effect correction). The batch effect is visible since different batches form separated clusters.* ***b****, Batch-effect correction of Scyan, Cydar, Combat, SAUCIE, and Harmony. A superposition of all batch distributions can show a good batch effect correction.* ***c*** *Batch-effect correction metrics. A low cLISI (at the top of the figure) denotes good cell-type variability preservation, while a high iLISI (on the right of the figure) denotes better batch mixing.*

3.3.3 . Scyan latent space provides interpretability and helps population discovery

Scyan's latent space (see subsection 3.2.6) is key for interpretability. Specifically, it enables the understanding of the Scyan annotation process, and also helps to quickly characterize new populations of cells to improve the annotation. We illustrate population discovery on the POISED dataset. For this purpose, we show that we could annotate six populations that were missed during manual gating, such as differentiated effector T cells [Sallusto et al., 1999] (TCD8 TEM) and $\gamma\delta$TCR CD16+ cells (Figure 3.6a shows the populations we had before running population discovery). To demonstrate two different ways of discovering new populations, we show that we can (i) annotate more precise populations among known ones using Leiden [Traag et al., 2019] sub-clustering and Scyan latent space (see subsection 3.2.6 and Figure 3.6b), or (ii) discover a population that was missing from the table (see Figure 3.6c/d). For the latter case, we show that cells corresponding to a population that is not in the table will be annotated as 'unknown' by Scyan, and its interpretability (subsection 3.2.6) will help characterize this missing population.One advantage of Scyan is its table flexibility: the new populations, once characterized, can be added to the knowledge table, and Scyan will then be able to annotate them. This is shown by Figure 3.6e that summarizes all the populations we annotated.

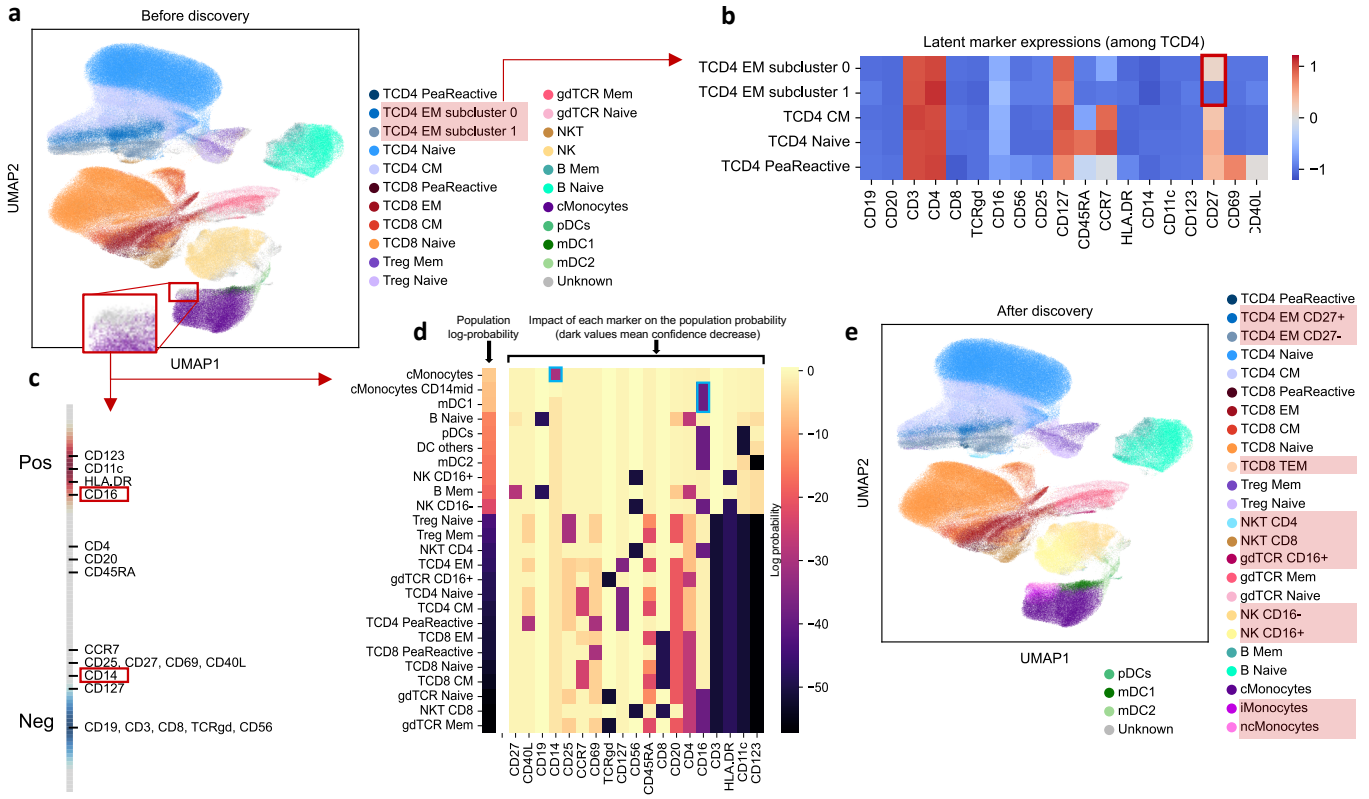

Figure 3.6: **Interpretability and population discovery with Scyan.** **a**, UMAP on POISED before population discovery. Two subclusters of TCD4 EM cells have been defined and characterized in (b). Also, intermediate and non-classical monocytes were removed from the knowledge table to show that we can retrieve existing populations that are missing from the table: as shown by the red magnifying window, Scyan annotated these cells as unknown, and (c/d) helped to characterize them as intermediate and non-classical monocytes. **b**, Latent space expressions for subsets of TCD4 cells, displayed on a heatmap. We can easily see the difference between the two clusters: one is CD27+, the other CD27-. The subclusters were obtained by running Leiden [Traag et al., 2019] clustering on the TCD4 EM populations. **c**, Scyan helps characterize the unknown cells defined in (a) by showing its latent marker expressions, displayed on a shared scale going from Negative to Positive expressions. We can see, for instance, that these "Unknown" cells are CD16 positive and CD14 negative. **d**, Scyan provides soft predictions for all populations (first column), i.e., a log probability is associated with each population. Then, each population probability is decomposed into a sum of marker impact on the probability (one row). Dark colors indicate that the corresponding marker expression decreased the population probability of the corresponding row. According to the first column, we see that the first guesses of Scyan are classical monocytes (both CD14 high and mid) and mDC1. Then, we look at the three corresponding rows: they correspond to the confidence of Scyan for these populations decomposed by markers. For instance, we see that the expression of CD14 (which is negative, according to (c)) decreased Scyan's confidence toward the prediction of classical monocytes. Thus, based on the first row, we can conclude that the 'Unknown' cells are similar to classical monocytes but are CD14- instead of CD14+, and that these cells are non-classical or intermediate monocytes. Similarly, the third row shows that they look like mDC1 cells but with a CD16+ expression instead of negative (again, (c) was needed to see the expression of CD16). Once more, we indeed conclude that these cells are non-classical/intermediate monocytes, and they can be added back to the table for the annotation. **e**, UMAP of Scyan annotations after population discovery. The red boxes denote new populations compared to (a): such populations were found using subclustering and analyzing Scyan's latent space.

Understanding Scyan annotation process

Scyan annotation process can be interpreted on one cell or a group of similar cells (see methods subsection 3.2.6). Typically, we can select one population and interpret Scyan's annotation process on this group of cells. First, we can display all the latent marker expressions corresponding to these cells (Figure 3.6c). It opens up a new simple way to understand which marker is positive or negative at a glance. Indeed, the latent space has a shared scale for all markers, and a simple scale indicates expression levels between Negative (-1) and Positive (+1). But mostly, we can provide a confidence measure (or log-probability) to belong to each cell type (Figure 3.6d, left column). Then, for each population, we can decompose the population probability as a sum of marker impact (expressed

as log-probabilities). It allows explaining which marker contributed the most to the probability of each of the populations. This interpretability property can be used to discover new populations (see subsubsection 3.3.3).

Annotating unknown populations

Sometimes, users may forget some populations in the table given to Scyan, and the corresponding cells will be left unclassified. Since every population from the POISED dataset was already described, we decided to remove two populations from the table provided to Scyan (non-classical and intermediate monocytes) to see if we could retrieve them. As shown in Figure 3.6a on the red magnifying glass, cells corresponding to these populations were annotated as being "Unknown" (light grey color). We can further investigate these "Unknown" cells to retrieve their corresponding population, see Figure 3.6c/d. To summarise, the process is the following: (i) we choose a group of cells that were unclassified by Scyan, (ii) we quickly characterize these cells using Scyan latent space, and (iii) we update the table to annotate them. Combining Figure 3.6c and Figure 3.6d provides a description of the Scyan annotation process that is understandable by humans, through decomposition into confidence measures by marker and by population.

3.3.4 . Comparison to supervised models

Performances on POISED

Two supervised models (LDA [Abdelaal et al., 2019] and CyAnno [Kaushik et al., 2021]) were compared to Scyan on the POISED dataset. Figure 3.7a shows the performances of the three models on POISED. Even though the benchmark is in favor of supervised models (since they use labels), Scyan still has a higher performance. Also, since we are comparing an unsupervised method to supervised methods, comparing results to manual gating is biased. For this reason, we additionally compare the models' agreement (in a pairwise manner) using Cohen's Kappa score (Figure 3.7b). We see that LDA and Scyan are the two models whose agreement is the highest, even higher than LDA and CyAnno (while they are two supervised methods trained for the same task). Concerning the disagreement between Scyan and the manual gating, we show in the supplementary Figure A.9a/b that most disagreement is partly due to the subjective delimitation boundaries between non-classical/intermediate/classical monocytes. As shown by supplementary Figure A.9a/b, although Scyan is also properly annotating these populations, it has slightly different decision boundaries than manual gating, which still significantly decreases F1-score or balanced accuracy. It emphasizes again the importance of comparing the agreement between all models instead of only comparing to manual gating.

Annotations of the ungated cells

One key aspect of annotation models is whether they annotate more cells than traditional manual gating. This can enhance the biomarker discovery and provides higher statistical significance during post-annotation analyses. We compared the number of

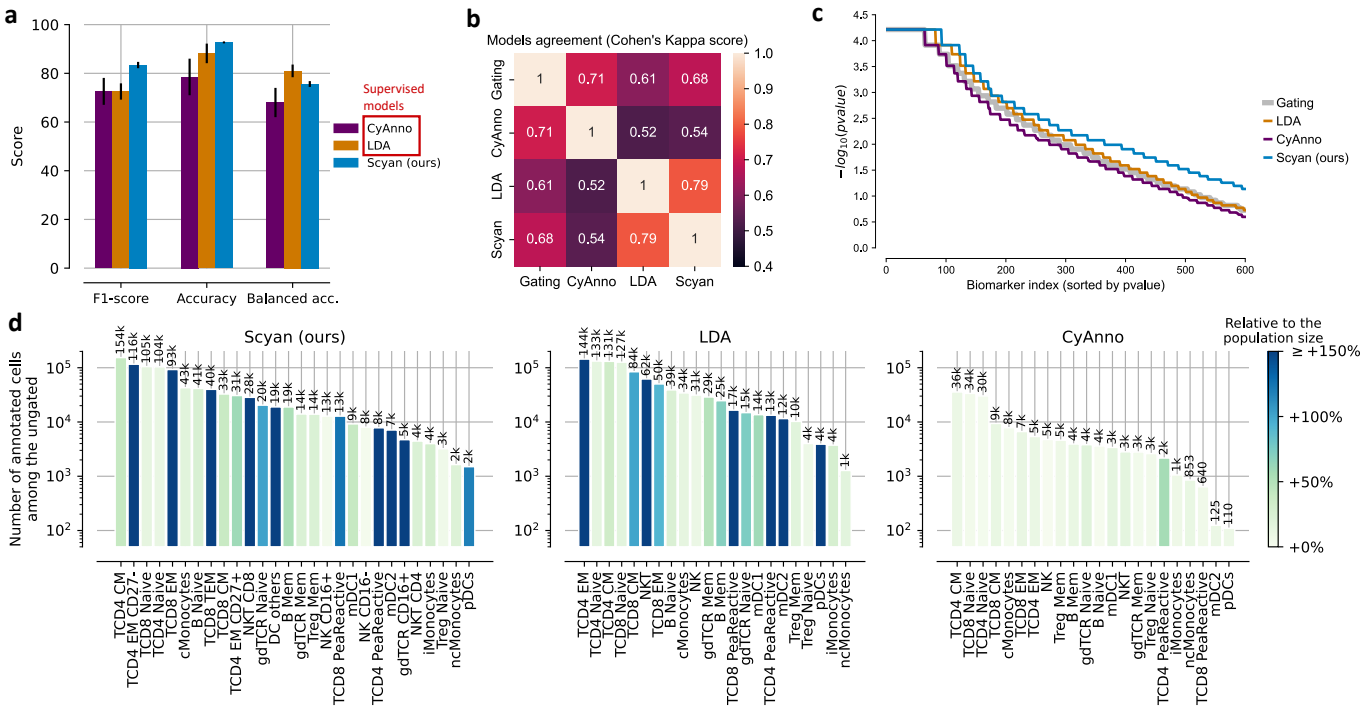

Figure 3.7: ***Comparison to supervised models.*** *The last two figures were done after Scyan's population discovery.* ***a****, Metrics on POISED. Note that among the three methods tested, CyAnno and LDA are supervised methods (i.e., using training labels from manual gating).* ***b*** *Heatmap representing pairwise models agreement using Cohen's Kappa score. A high value indicates a better agreement (the highest value is 1).* ***c****, After annotation, one can extract biomarkers and run differential expression relative to a clinical condition. Here, we show the significance of the biomarkers for all methods (higher is more significant).* ***d****, Number and percentage of cell types that were annotated by the models among the ungated ones.*

cells annotated by Scyan, LDA [Abdelaal et al., 2019], and CyAnno [Kaushik et al., 2021] on POISED, and demonstrated that Scyan annotates more cells than CyAnno, and a similar amount of cells to LDA (Figure 3.7d). Indeed, CyAnno annotated 15% of the ungated cells, Scyan 97%, and LDA was set up to annotate all cells. Moreover, Scyan annotated 6 more populations compared to CyAnno and LDA. Most importantly, we show by back gating that the annotated cells were properly classified (see supplementary Figure A.9). Indeed, one limitation of supervised models such as LDA or CyAnno is that they can not annotate new populations, i.e., they are limited to manually gated populations. Although knowledge-based annotation models (like ours) are limited to populations from the provided table, the table can be easily extended. This property is, therefore, crucial for population discovery with Scyan.

Usage for biomarker discovery

The POISED dataset is decomposed into two conditions: peanut-stimulated samples, and unstimulated ones. We try to find biomarkers that are differentially expressed on peanut-stimulated samples. For that, for all models, biomarkers were extracted, and we ran Wilcoxon signed-rank tests [Wilcoxon, 1945] between the two conditions (we assume

patients are independent). On Figure 3.7c, we sorted the biomarkers by p-value for all models, and we display the $-log_{10}(p - value)$ of the first 400 biomarkers. We show that Scyan extracts more biomarkers of higher significance. Note that a similar process could be run for other clinical conditions, such as the patient response to treatment. Having more significant biomarkers means it will be easier to predict such an outcome.

3.4 . Discussion

3.4.1 . Summary of Scyan

We have introduced Scyan, a multi-purpose neural network for cytometry annotation, batch-effect removal, debarcoding, and population discovery. It provides a robust and broad pipeline to analyze cytometry cell populations, monitor their dynamics over time, and compare the populations' proportions among patients. Such analyses can help discover biomarkers or specific populations characteristic of response to treatment, for example. Scyan can perform fast and automatic annotations for these large datasets and correct potential batch effects. Some studies use barcoding to reduce the batch effect, hence requiring a debarcoding step that Scyan can also perform. Thus, Scyan is suitable for various types of cytometry projects and does not rely on any extra cytometry analysis library.

Scyan annotates populations without needing labels and, therefore, can fully replace manual gating. It uses a marker-population table containing expert knowledge. The literature offers many resources and existing knowledge to construct such tables, but some marker expressions remain unexplored. For this reason, we offer the possibility to handle "not applicable" values inside the table and, to improve flexibility, intermediate expressions such as "mid" or "low". In the case where the panel remains not well known enough to build the input table, Scyan can help discover new populations: analyses start by annotating large populations and then gradually target smaller and smaller cell types. Also, with the increasing usage of cytometry, we expect the marker knowledge to improve over time, reinforcing Scyan performance and ease of use.

In terms of model architecture, normalizing flows is a recent and promising field of research in generative models. They benefit from interesting mathematical properties such as (i) exact likelihood computation and (ii) invertibility. We show that normalizing flows can be used to leverage marker knowledge in a biologically natural way, providing interpretability. Indeed, the network invertibility allows switching between the measured marker expressions and their latent expressions. In this space, all latent markers have unified expression ranges, which is convenient for human analysis, especially for population discovery. It also makes the model reliable and transparent to biologists, which can

help build trust toward the model annotations and validate them. Moreover, normalizing flows are smooth transformations that control how the space is deformed, ensuring that we do not alter the biological meaning behind marker expressions. At the same time, it benefits from the expressiveness and flexibility of deep neural networks. In fact, the usage of neural networks allows adding additional terms in the loss function to handle the batch effect, which is naturally corrected with the network invertibility. Eventually, we can further push the usage of these convenient mathematical properties for other tasks in single-cell analysis, for instance, single-cell RNA sequencing data or imaging mass cytometry data [Chang et al., 2017].

Overall, Scyan promises to be powerful in several ways. Scyan is robust for identifying unique cell populations, like dendritic cells or stromal cells, which are underrepresented in complex biological samples, although they play essential roles in shaping disease resolution or progression. The ability of Scyan to analyze large datasets in a fast and accurate manner will be essential for this task and open up the possibility of unraveling the heterogeneity of such rare populations of cells. These analyses can be done on all types of cytometers, regardless of the presence or absence of batch effect.

3.4.2 . Position of Scyan in an open-source context

Scyan is positioned as a post-processing tool. That is, it is intended to be used after basic processing steps such as reading FCS files, normalization, data cleaning, and quality control, which can be handled by tools like Pytometry, a scverse package for cytometry data preprocessing. At first, preprocessing was expected to be done inside Scyan, and I provided methods for this. Later, when Pytometry was being developed, I contributed to it in order to build a common preprocessing package for the Python community. As a contributor to Pytometry, I have ensured that Scyan complements this preprocessing toolbox. Scyan utilizes AnnData, making it fully compatible with the scverse ecosystem. This allows it to integrate seamlessly with other tools in the community, providing a streamlined workflow from data preprocessing to advanced post-processing tasks like cell-type annotation and batch effect correction.

3.4.3 . Advantages and limitations of cytometry

Cytometry, particularly flow cytometry, is a powerful technique widely used in oncology for analyzing the physical and chemical characteristics of cells or particles. One of the primary advantages of cytometry is its cost-effectiveness and relative ease of use. Compared to more complex and expensive spatial data analysis techniques, cytometry requires less sophisticated equipment and can be performed more quickly, making it accessible to a broader range of laboratories. Furthermore, cytometry provides high-throughput analysis, allowing for the examination of thousands of cells in a short period, which is invaluable in clinical settings where rapid and accurate diagnostics are crucial.

However, a significant disadvantage of cytometry is the loss of spatial context. Unlike spatial data analysis, which allows researchers to visualize the precise location and organization of cells within a tissue, cytometry provides a more generalized view, lacking information about cell positioning and microenvironmental interactions. This limitation can obscure critical insights into the spatial heterogeneity of tumors, which is increasingly recognized as a key factor in cancer progression and treatment resistance. Consequently, while cytometry remains an indispensable tool in oncological research, its application must be carefully considered, particularly when spatial information is essential for understanding the biological processes under study. In the following chapters, we will dive into spatial omics technologies, offering more precise information about the tumor microenvironment, but whose complexity demands specific computational tools.

Technology-invariant preprocessing and analysis of spatial omics data

Contents

Abstract

Spatial omics data allow in-depth analysis of tissue architectures, opening new opportunities for biological discovery. In particular, imaging techniques offer single-cell resolutions, providing essential insights into cellular organizations and dynamics. Yet, the complexity of such data presents analytical challenges and demands substantial computing resources. Moreover, the proliferation of diverse spatial omics technologies, such as Xenium, MERSCOPE, CosMX in spatial-transcriptomics, and MACSima and PhenoCycler in multiplex imaging, hinders the generality of existing tools. We introduce Sopa (`https://github.com/gustaveroussy/sopa`), a technology-invariant, memory-efficient pipeline with a unified visualizer for all image-based spatial omics. Built upon the universal SpatialData framework, Sopa optimizes tasks like segmentation, transcript/channel aggregation, annotation, and geometric/spatial analysis. Its output includes user-friendly web reports and visualizer files, as well as comprehensive data files for in-depth analysis. Overall, Sopa represents a significant step toward unifying spatial data analysis, enabling a more comprehensive understanding of cellular interactions and tissue organization in biological systems.

4.1 . Context and motivation

4.1.1 . Overview of existing single-cell resolution technologies

Spatial omics data offer opportunities to improve our understanding of cellular interactions within their micro-environment and the intricacies of tissue organization [Bressan et al., 2023, Rao et al., 2021]. Recent advancements in imaging technologies have expanded these capabilities, enabling the measurement of 1000+ genes through Spatial Transcriptomics [Moses and Pachter, 2022] and/or the analysis of 50+ proteins via Multiplex Imaging [Lewis et al., 2021]. These include MERFISH [Chen et al., 2015], ISH [Jin and Lloyd, 1997], ISS [He et al., 2022], MICS [Kinkhabwala et al., 2022], PhenoCycler [Jhaveri et al., 2023] and IMC [Chang et al., 2017], all of which provide single-cell resolution that could not be achieved by previous spot-based techniques like 10X Visium or Nanostring GeoMX [Merritt et al., 2020].

Therefore, image-based technologies provide a higher resolution — up to the subcellular level — which is needed for a detailed exploration of individual cells and their gene expression profiles within their spatial context. This level of precision has been essential for unraveling tissue architecture and understanding cellular interactions; it marks the beginning of a significant leap forward in our comprehension of biological systems [Kumar et al., 2023, Jhaveri et al., 2023, Chu et al., 2023].

Recently, 10X Genomics released the Visium HD, an NGS-based machine whose se-

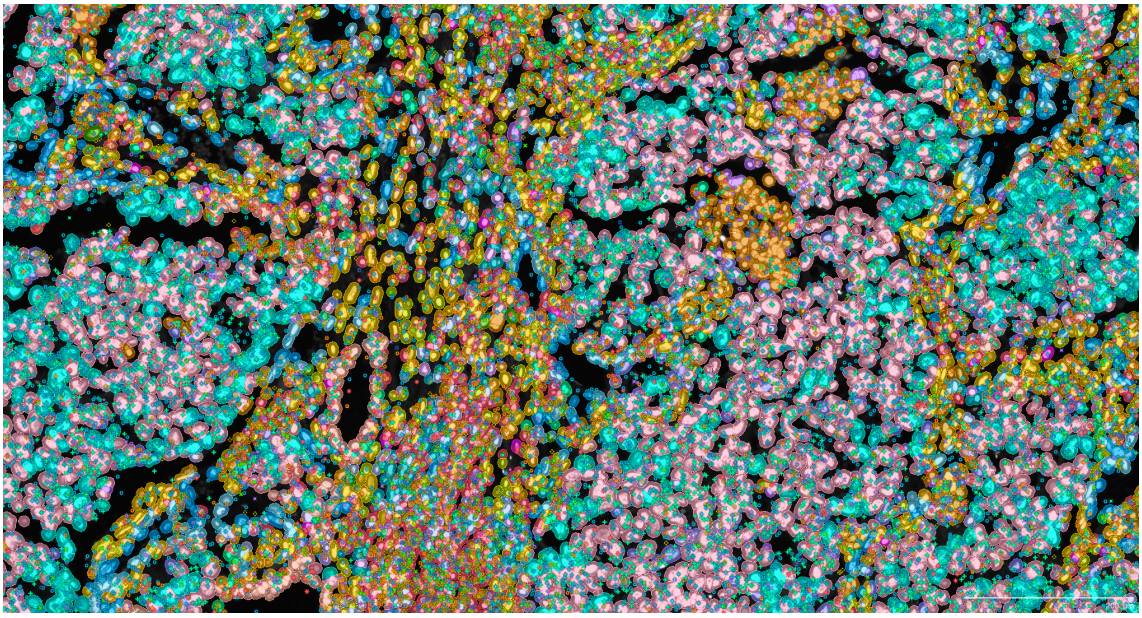

Figure 4.1: ***Example of Xenium data (lung tissue).*** *Cells are contoured by their cell type. Each point represents one transcript, colored by the corresponding gene name.*

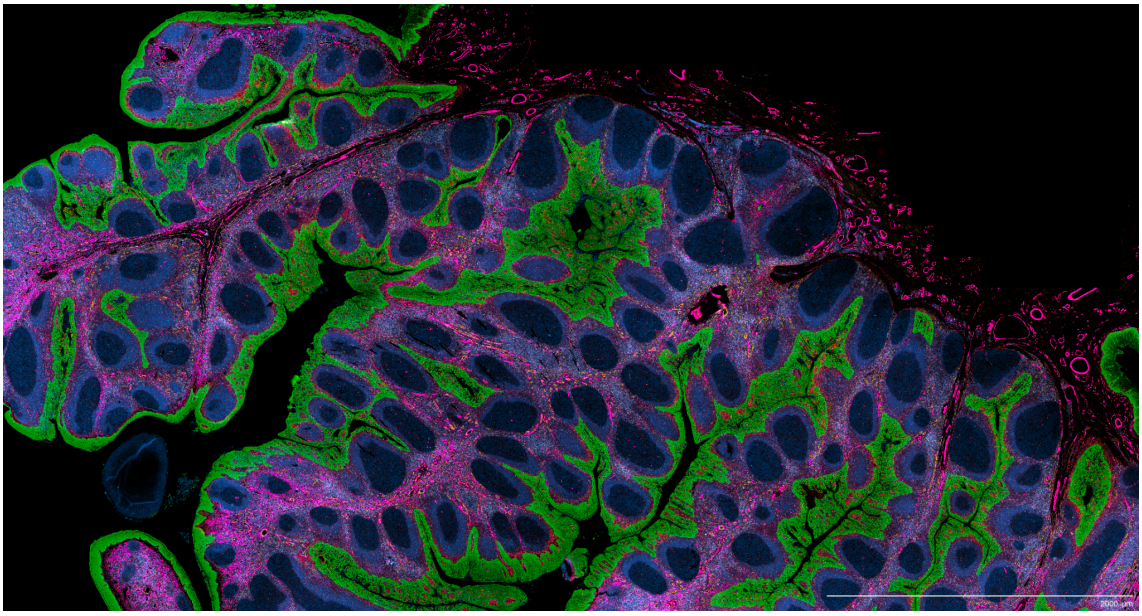

Figure 4.2: ***Example of PhenoCycler data (tonsil tissue).*** *Four protein stainings are shown.*

quencing is performed inside 2-microns-wide bins (while the normal Visium has spots of 55 microns). With such a resolution, the Visium HD is, therefore, also a single-cell resolution technology. An important thing to note is that, since the Visium HD is NGS-based, it is, therefore, sequencing the whole genome, contrary to imaging-based methods, which are limited to thousands of genes.

In addition, a few technologies are multi-omics, for instance measuring transcripts

and proteins, but they still have limitations. For instance, the MERSCOPE is limited to less than 10 proteins, while the CosMX can't perform both proteomics and transcriptomics on the same slide.

		Xenium	Visium HD	MERSCOPE	CosMX	PhenoCycler	MACSima	Hyperion
General information	Vendor	10X Genomics	10X Genomics	Vizgen	Nanostring	Akoya Biosciences	Miletnyi	Standard BioTools
	Type	Imaging	NGS	Imaging	Imaging	Imaging	Imaging	Imaging
	Resolution	Subcellular	2-microns bins	Subcellular	Subcellular	Single-cell	Single-cell	Single-cell
Data properties	Direct data access	Yes	Yes	Yes	No	Yes	Yes	Yes
	Processed output	Yes	No single-cell aggregation	Yes	Yes	No	No	No
	Stitched image	Yes	Yes	Yes	No	Yes	Yes	Yes
Omics	Proteins	< 10	0	< 10	< 100	< 100	< 100	< 100
	Transcripts	> 1000	> 30k	> 1000	> 1000	0	0	0
Alignment	H&E and omics aligned	Yes	Yes	No	No	No	No	No
	Proteins and transcripts aligned	Yes	NA	Yes	No	NA	NA	NA
Post-processing	Visualizer	Free but closed	Free but closed	Free but closed	Paying	Paying	Paying	Free but closed
	Analysis platform	Yes	Yes	No	Yes	Yes	Yes	No

Figure 4.3: ***Comparison of some of the existing single-cell resolution technologies.*** *We compare the resolution, the type of omics data measured, and the data output, among other properties. Each machine has its own advantages and limitations. This table is not an exhaustive list of all the existing technologies and does not mention all the important criteria for choosing a machine.*

4.1.2 . A universal data structure for spatial omics: *SpatialData*

SpatialData is a versatile data structure within the scverse ecosystem, designed to manage spatial omics data with both a disk format (using zarr) and an in-memory format. It standardizes data handling by providing a specific reader for each technology, transforming raw data into a standardized SpatialData object. This structure supports multiple coordinate systems and allows for transformations between them, ensuring flexibility in spatial data analysis. The library capabilities is summarized in Figure 4.4 below.

As a SpatialData contributor, I added multiple functionalities. Notably, I contributed to the readers (MERSCOPE reader, CosMX reader, Phenocycler reader). I also added a few core geometric operations, such as shapes rasterization into labels, labels vectorization into shapes, bins rasterization, and bins aggregation.

Internally, SpatialData creates an abstraction for various spatial objects. That is, each object will fall inside of the five types of spatial elements (points, shapes, labels, images, and tables), allowing for a standardized representation of the object (both on disk and

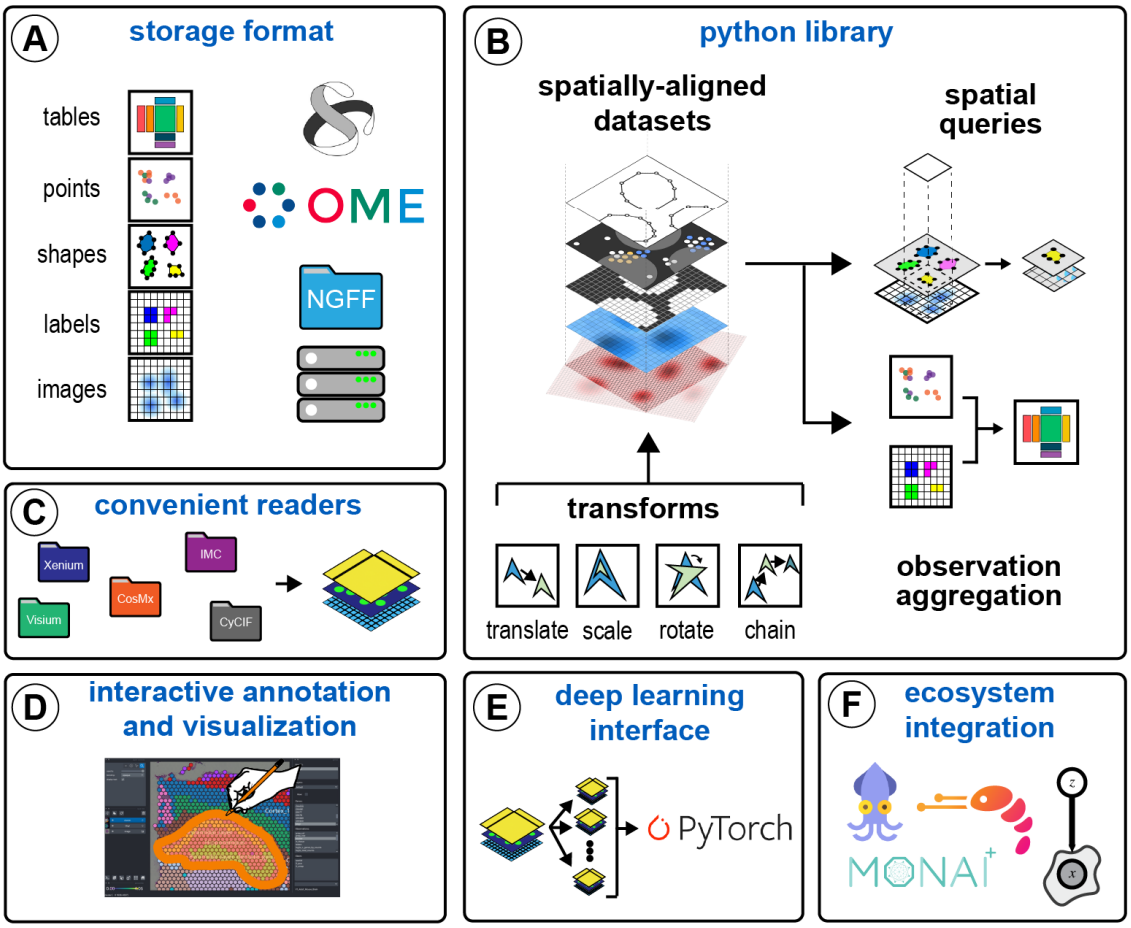

Figure 4.4: ***Overview of the SpatialData data structure*** *[Marconato et al., 2024].*

in memory). We summarise below these different elements, which are also illustrated in Figure 4.5:

- **Points:** DataFrame of 2D or 3D points locations. For instance, this can be used for the names and locations of transcripts for imaging-based technologies (MERSCOPE, Xenium, CosMX). Internally, it uses Dask for efficient memory management.
- **Shapes:** DataFrame of geometries (for instance, circles, lines, polygons). This can be used for cell or tissue boundaries. Internally, it uses GeoPandas for efficient geometry operations.
- **Labels:** Rasterized images, typically used for rasterized segmentation. Internally, it uses Xarray [Hoyer and Hamman, 2017].
- **Images:** Image data, typically used for DAPI or protein staining. Internally, it uses Xarray [Hoyer and Hamman, 2017].
- **Tables:** AnnData objects, typically used for transcriptomics data to store a cell-by-gene matrix of counts. Internally, it uses AnnData [Virshup et al., 2021].

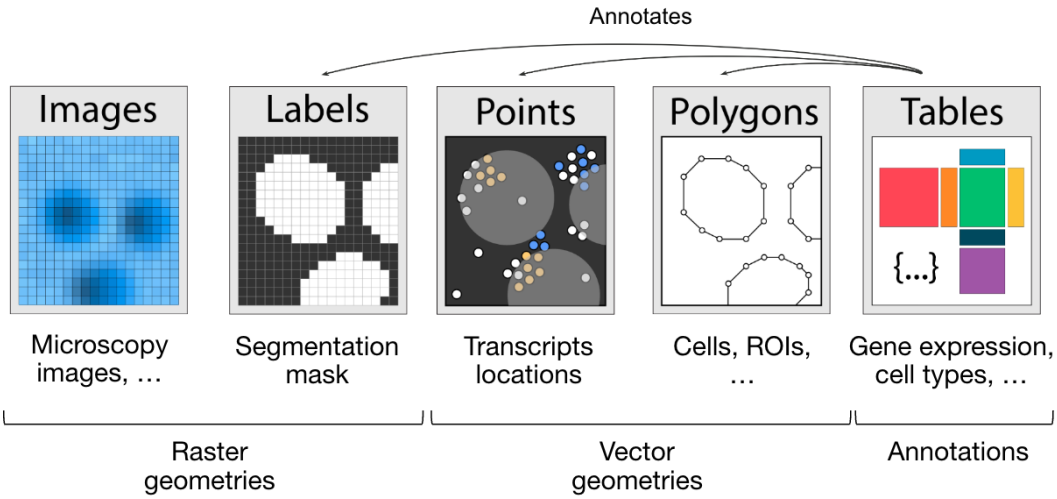

Figure 4.5: ***Overview of the SpatialData abstraction of spatial elements.***

Importantly, SpatialData is not an analysis library but rather a robust framework for organizing and managing spatial data. Therefore, the analysis tool introduced in this chapter is built on top of SpatialData. That is, it uses SpatialData as a data structure to process and analyze spatial omics data.

4.1.3 . Challenges and objectives

Despite this data structure standardization, the analysis of image-based spatial omics has encountered significant computational challenges and limitations [Atta and Fan, 2021, Moses and Pachter, 2022, Zeng et al., 2022, Vandereyken et al., 2023, Dries et al., 2021]. Most existing methods [Stringer et al., 2021, Petukhov et al., 2022, Biancalani et al., 2021] are not designed to handle large images with millions of cells. Their usage typically demands high-performance computational clusters with substantial memory resources, which limits its accessibility to spatial omics due to cost and hardware constraints. As a result, most companies have developed proprietary tools for their own data types, primarily focusing only on segmentation and visualization. Yet, these proprietary tools have certain constraints, such as (i) a limit on specific functionalities, (ii) no incorporation of the latest state-of-the-art methods, and (iii) a lack of versatility, as they cannot be applied to other technologies. This tool diversity has other limitations in that each suite has a learning and adaptation process and that the tools' specificities lead to variations in the analysis of comparable data types. Similarly, current open-source analysis libraries often rely on (i) already-segmented data [Hao et al., 2023, Palla et al., 2022], (ii) specific data types [Axelrod et al., 2021, Cisar et al., 2023], or (iii) a subset of analysis tasks [Axelrod et al., 2021, Cisar et al., 2023], resulting in fragmented approaches and difficulty in adapting one approach to a different type of technology. The absence of a unified data representation and modular programming interface further complicates the integration of various analysis steps.

To address these gaps, our work introduces Spatial Omics Pipeline and Analysis, or Sopa, a computational framework that enhances the accessibility, efficiency, and interpretability of image-based spatial omics data. Sopa is a memory-efficient pipeline that works across all image-based spatial omics technologies and that can display results in a common visualizer. This includes the most recent Spatial Transcriptomics technologies (Xenium, MERSCOPE, CosMX) and also the multiplex imaging techniques (e.g., MACSima, PhenoCycler, Hyperion). Sopa's capabilities include segmentation and multilevel annotation, both based on transcripts and/or stainings, as well as spatial statistics and niche geometry analysis. We demonstrate Sopa's performance on four public datasets: two spatial-transcriptomics (MERSCOPE, Xenium) and two multiplex imaging technologies (PhenoCycler, MACSima), and provide a memory and time benchmark over multiple dataset sizes. Additionally, we demonstrate Sopa's capabilities for geometric and spatial analysis on the MERSCOPE dataset by analyzing cell colocalization with regard to cell types and niches, showing promise for biological discoveries. All these functionalities are accessible via our open-source code, which includes a Command Line Interface (CLI), an Application Programming Interface (API), and a flexible Snakemake [Köster and Rahmann, 2018] workflow, enabling users with various levels of expertise to process spatial omics data seamlessly, from no-code simplicity to full flexibility. The pipeline's generic nature ensures effortless transitions to other types of spatial omics data, making it a versatile and powerful tool for the scientific community.

4.2 . Broad overview of the pipeline and its key properties

To establish versatile tools, a common strategy involves adopting a shared data structure that seamlessly integrates across diverse technologies. SpatialData [Marconato et al., 2024] serves as one such comprehensive framework, including readers tailored for the most widely used spatial omics technologies. Building upon this, Sopa converts any data into a SpatialData object, on which all of the six following tasks are performed. First, if needed, users can interactively select a region of interest, facilitating the exclusion of less relevant or lower-quality areas. Next, we generate overlapping patches of images and/or transcripts. Segmentation can then be performed for each individual patch, and we currently support Cellpose [Stringer et al., 2021] (image-based segmentation) and Baysor [Petukhov et al., 2022] (transcripts-based segmentation). Afterwards, the cell segmentation masks are converted into polygons and merged over all patches to remove potential artefacts. Following these first four steps, we average the staining intensities and count the transcripts inside each cell (see subsection 4.3.2 and subsection 4.3.2), allowing further tasks such as annotation. For example, Sopa currently supports Tangram [Biancalani et al., 2021] for transcript-based annotation, and a simple Z-score method for staining-based annotation (subsection 4.3.4). Finally, we implemented spatial and geometric anal-

ysis tools to fully exploit the spatial nature of the data (subsection 4.3.5). For convenience, all image-based technologies can be visualized in a shared explorer (see section 4.4), and an HTML report is provided for pipeline quality checks. The full process described above is summarised in Figure 4.6.

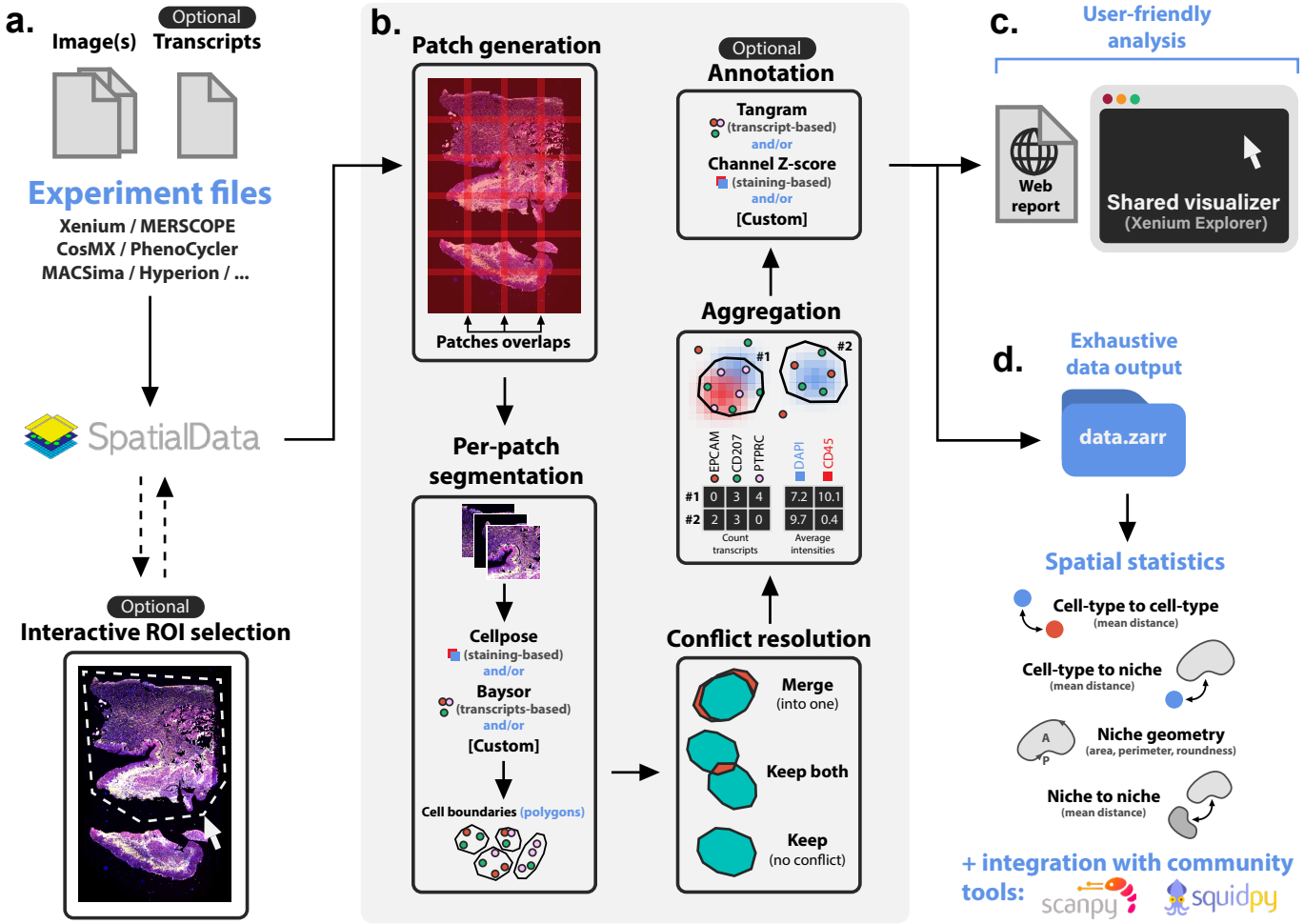

Figure 4.6: ***Overview of Sopa.*** ***a.*** *The pipeline input consists of experimental files of any image-based spatial omics. It is transformed into a SpatialData object, on which we can optionally select a region of interest (ROI) interactively.* ***b.*** *Afterwards, the data is split into overlapping patches, and segmentation is run on each patch (for instance, Cellpose, Baysor, or a custom segmentation tool). Since patches are overlapping, some cells can be segmented multiple times on different patches. Therefore, these conflicts have to be resolved: two boundaries with a significant overlap are merged into one cell, while two cells barely touching are kept separate. The next step is aggregation, i.e., counting transcripts and averaging each channel intensity inside each cell. This allows annotation, either based on transcripts (using Tangram) or on channel intensities.* ***c.*** *Afterwards, Sopa outputs a user-friendly report and files to be opened in the Xenium Explorer (whatever the input technology).* ***d.*** *All data files are kept for further analysis in Sopa, such as spatial statistics, or integration with community tools.*

4.3 . Methods

4.3.1 . Segmentation on patches

For computational efficiency, segmentation is performed on patches, i.e., small image regions. These patches have a certain overlap, which is typically chosen to be at least twice as big as the average diameter of cells (e.g., 20 microns). This way, each cell should be complete in at least one patch, which avoids artefacts due to cutting cells at the border of the patches. Subsequently, any segmentation algorithm compatible with images and/or transcripts can be applied. While Cellpose [Stringer et al., 2021] and/or Baysor [Petukhov et al., 2022] are commonly used, Sopa does allow the integration of other segmentation algorithms. Following segmentation on individual tiles, the cell boundaries are transformed into polygons using Shapely. Since patches overlap, some cells may be segmented across different patches, leading to segmentation conflicts where multiple polygons correspond to a single cell. To resolve this, we adopt a method similar to the one used in Vizgen's preprocessing tool (VPT). Specifically, we merge pairs of cells when the intersection area exceeds half the area of the smaller cell, ensuring a substantial overlap. If the intersection area is too small, indicating distinct cells, both polygons are retained. When the overlap area divided by the smallest cell area is close to 1, this corresponds to two almost identical cells, while a score close to 0 corresponds to two cells barely touching. On Figure 4.7e, we studied the distribution of this score, showing that most of the conflicts are associated with a score that is either very close to 0 or very close to 1, indicating a good conflict resolution. Indeed, statistical considerations indicate that scores above 0.8 or below 0.07 are good resolutions (see supplementary subsection A.2.1). Additionally, note that, before segmentation, the user can decide to select a region of interest: this can be done interactively with matplotlib [Barrett et al., 2005] on a low-resolution image.

4.3.2 . Channel and transcript aggregation inside cells

Channel averaging

When dealing with image-based technologies, a crucial step involves averaging the intensity of each channel within each cell. While this task can be achieved using cell masks, it proves highly inefficient in terms of both time and memory consumption. To address this challenge, we adopt a chunk-level approach: (i) For each chunk, we identify cell boundaries (i.e., polygons) that intersect with the chunk coordinates, then (ii) we determine the bounding box for each of these cells, then (iii) we extract the image values for each of these bounding boxes, and finally (iv) we rasterize the cell polygons to average the staining intensity over the local bounding box. In this way, we only load small arrays corresponding to each cell, instead of loading large cell masks. This process is repeated over all chunks, and we make sure that the channel intensity for cells located on multiple chunks is computed correctly.

Counting transcripts

GeoPandas is a Python library that enhances Pandas [Wes McKinney, 2010] Dataframes by incorporating support for Shapely geometries. It facilitates scaling operations on ge-

ometries, making it particularly suitable for transcript counting, where transcripts can be represented as Shapely points and cells as Shapely polygons. However, without Sopa, the memory requirements for such operations can be substantial, especially for spatial transcriptomics datasets that may contain up to one billion transcripts. To optimize this process, we leverage Dask and execute the GeoPandas "join" operation at the partition level to assign each point (i.e., a transcript) to a polygon (i.e., a cell). Thus, each operation is carried out on smaller data frames, each less than 100MB in size. Dask efficiently assigns each partition to different workers in parallel, mitigating memory concerns. This approach proves highly effective on both laptops and high-performance clusters, as Dask is designed to seamlessly scale these processes without necessitating any code modifications.

4.3.3 . Conversion to the Xenium Explorer

Converting a spatial omics object into the Xenium Explorer requires the creation of six files: (i) the image, (ii) a JSON metadata file, (iii) the cell boundaries, (iv) the cell categories (e.g., cell type or clustering), (v) the gene counts table, and (vi) the transcripts (if they exist). The conversion is done automatically by Sopa, but it can also be done manually via our CLI: `sopa explorer write <sdata_path> <output_path>`.

For image creation, a Python function is recommended in the Xenium Explorer documentation (https://www.10xgenomics.com/support/software/xenium-explorer/tutorials/xe-image-file-conversion) but is not optimized for large images. We updated it to support Dask arrays, i.e. (the image type used by Sopa). Pyramids of resolutions are generated via the SpatialData library [Marconato et al., 2024]. To decrease memory usage, each (1024x1024) image tile is generated using an iterator that only computes the minimally required data from the Dask array at each tile generation. For higher pyramidal levels, where the image size decreases, we allow loading an image into memory if it fits, accelerating conversion.

As transcripts typically cannot be loaded entirely into memory, the Xenium Explorer avoids loading all transcripts. On low-resolution levels, only a subset of transcripts is displayed (subsampled), while zooming in reveals all transcripts from the current field of view. This pyramidal transcript view ensures low memory usage during visualization. The highest-resolution tiles are 250-micron-wide squares. For each pyramid level, the tile width doubles, and only one-fourth of the transcripts from the previous level are retained. The process stops when there is only one remaining tile that is larger than the original slide. Transcript coordinates are stored as separate chunks for each tile and resolution, saved as a Zarr file. This allows loading only the transcripts corresponding to the displayed tiles when zooming in.

Cell boundaries are padded to have the same number of vertices (13). Polygon simplification is applied to polygons with more than 13 vertices using the Shapely library, reducing the number of vertices while preserving shape geometry. A fixed number of vertices enables lighter cell-boundary storage and faster visualization.

Transcript counts (cell-by-gene table) use sparse array storage. One 1D array stores all non-zero transcript counts, another array stores the cell index for each count, and a third array is a pointer indicating the gene index for these counts. Cell categories are similarly saved using indices and corresponding pointers. Once again, the file format employed is a Zarr file.

4.3.4 . Cell-type annotation

Transcript-based annotation.

Tangram [Biancalani et al., 2021] is used for cell-type annotation based on an annotated scRNAseq reference. To make Tangram [Biancalani et al., 2021] scalable for large datasets, we adopt a strategy of splitting the data into "bags of cells", with the size determined by the user. This approach ensures that each Tangram iteration operates within manageable memory limits, and we subsequently merge the results to obtain the annotation for the entire dataset. Following this, Leiden [Traag et al., 2019] clustering can be applied to refine the annotation, associating each Leiden cluster with its most prevalent Tangram cell type. Additionally, we have implemented a multi-level annotation feature based on Tangram to enhance the annotation of minor cell types if needed. The process involves initially annotating global cell populations, followed by running Tangram on specific cell groups (e.g., Myeloid cells) for a more detailed annotation (e.g., pDCs, *TREM2* macrophages, etc.). All that is required is to provide multiple cell-type annotation columns in the reference scRNAseq data, and Sopa will seamlessly execute the multi-level annotation.

Staining-based annotation.

For non-transcriptomics data, we also provide a fluorescence-based annotation. As each channel intensity is averaged inside each cell, we obtain a matrix $\mathbf{X}$ of shape (N, P), where N is the number of cells, and P the number of stainings/channels. Then, these intensities are preprocessed as in a recent article [Wu et al., 2022b]:

$$\mathbf{X}' = (\mathbf{X'}_j)_{1 \leq j \leq P}, \text{ with } \mathbf{X'_j} = \text{arcsinh}(\frac{\mathbf{X_j}}{5Q(0.2, \mathbf{X_j})}), \tag{4.1}$$

where $\mathbf{X}'$ is the preprocessed matrix, arcsinh is the inverse hyperbolic sinus function, and $Q(0.2, \mathbf{X_j})$ is the 20th percentile of $\mathbf{X_j}$. Afterwards, we use a list of stainings corresponding to a population (defined by a biologist), and each cell is annotated according to the channel whose preprocessed intensity is the highest. If desired, Leiden clustering [Traag

et al., 2019] can be run to have a deeper annotation. Each cluster can be annotated via differential analysis or by showing a heatmap of staining expression per cluster.

4.3.5 . Spatial statistics

All spatial statistics are performed after computing a Delaunay graph based on the spatial location of cells. This is done with Squidpy [Palla et al., 2022], which is itself based on Scipy [Virtanen et al., 2020]. We also prune long edges that cannot correspond to a physical cell-cell interaction (typically, edges longer than 40 microns). In the paragraphs below, N denotes the number of cells.

Cell category to cell-category statistics.

One relevant spatial statistic is the computation of the mean or minimum distance between two cell categories. This includes the pairwise distance between cell types (e.g., the mean distance between CD8 T cells and tumour cells), as well as the distance between cell types and niches (e.g., the distance between tumour cells and tertiary lymphoid structures). Let $(C_1, \dots, C_N)$ represent categories assigned to the N cells (e.g., cell types), and $(C'_1, \dots, C'_N)$ represent other categories (such as the niche to which the cell belongs). For instance, if cell "i" is a T cell inside the stroma, then C_i = "T cell" and C'_i = "stroma". The sets of unique categories are denoted G and G', respectively; for instance, G can be the set of unique cell types, and G' can be the set of unique niches. Then, $\forall g \in G$ and $\forall g' \in G'$, we define the mean distance between the category g and g' as follow:

$$D(g, g') = \frac{1}{\mathrm{Card}(\{i \,|\, C_i = g\})} \sum_{i \,|\, C_i = g} \min_{j \,|\, C'_j = g'} d_{ij}, \tag{4.2}$$

where Card represents the cardinal, and d_{ij} is the hop-distance between cell i and cell j. Note that $\min_{j \,|\, C'_j = g'} d_{ij}$ is the distance between cell i and the closest cell of category g', that is how many hops are needed for cell i to "find" the category of interest. In practice, we compute $D(g, g')$ by multi-node graph traversal, starting from all nodes whose category is g'. In this way, for each $g' \in G'$, we compute $(\min_{j \,|\, C'_j = g'} d_{ij})_{1 \leq i \leq N}$ in a single graph traversal. All the resulting distances can be stored in a matrix $((D(g, g')))_{g \in G, g' \in G'}$ and shown as a heatmap. Note that this heatmap is asymmetric because of the "minimum" usage in the above distance definition. To prevent confusion while reading the asymmetrical heatmaps, we precise that one row corresponds to the distances from the cell type of the row index to all other cell types. Additionally, we combine the four matrices of distances (cell-type to cell-type, cell-type to niches, niches to cell type, and niches to niches) into an adjacency matrix whose weights are the inverse of the distance. Then, the corresponding network can be plotted using the netgraph [Brodersen, 2023] library, as in Figure 4.9g, providing an interpretable visualization of the tumour microenvironment's structure.

Niche geometry statistics.

When niches (or spatial domains) are performed with an algorithm such as STAGATE [Dong and Zhang, 2022], users can decide to extract these niches as geometries to compute some relevant statistics, such as their area, perimeter, or roundness. From now on, for each cell i, $1 \leq i \leq N$, C_i denotes the niche to which the cell belongs, and G is the corresponding set of unique niches (i.e., for all cell i, $C_i \in G$). First, we prune all the edges (i, j) that are in between niches from the Delaunay graph, i.e., if $C_i \neq C_j$. Then, we extract the connected components of the graph. Because of the way we pruned the edges, each component corresponds to one niche, but one niche can be composed of multiple components (or occurrences). For each component, we search simplices (i.e., triangles from the Delaunay graph) at the component's border, that is, the simplices that have one or two simplex neighbours. From all the border simplices, we extract the corresponding border edges; these edges are then linked to make one or multiple rings (i.e. cyclic lines). If we have only one ring, it is transformed into a polygon, which corresponds to a "full" component. If there are multiple rings, the largest ring is the outer polygon, and the others correspond to "holes" inside the main polygon: this can happen when some components are completely surrounded by another niche. Repeating this process for all components allows the transformation of each niche $g \in G$ into multiple polygons. We can then count how many occurrences (or polygons) each niche is made of, and we can also compute the mean area A_g, perimeter L_g, and roundness R_g of each niche using Shapely. Note that $R_g = \frac{4\pi A_g}{L_g^2} \in [0, 1]$, where higher values correspond to a "circle-like" shape. The density of cells inside a niche is computed as the total number of cells in this niche divided by the total area of the niche. Also, for each niche, we filter out components whose areas are less than 5% of the area of the same niche's largest component, as they usually correspond to low-quality artefacts from the clustering of niches.

4.3.6 . Datasets, metrics, and computational details

Four public datasets were used to demonstrate Sopa's abilities. First, we used a MERSCOPE dataset (from Vizgen) of the human liver hepatocellular carcinoma (HCC), called FFPE Human Immuno-oncology Data Set May 2022. It is composed of a 500-gene panel, and has DAPI staining and PolyT staining. It contains about 500,000 cells, depending on the segmentation. Secondly, we used a Xenium dataset (from 10X Genomics) of pancreatic cancer (adenocarcinoma, Grade I-II) with the Xenium Human Multi-Tissue and Cancer Panel, in parallel with corresponding H&E image, and a protein-staining image with DAPI/CD20/PPY/TROP2. Note that the two latter images has to be aligned on the default main DAPI image. It contains about 180,000 cells, depending on the segmentation. Thirdly, we used a PhenoCycler dataset (from Akoya Biosciences) of the human tonsil (FFPE) with 31 protein stainings. It contains about 2,500,000 cells, depending on the segmentation. Finally, we used a MACSima dataset (from Miltenyi) of head and neck squamous cell carcinoma (HNSCC) with 61 protein stainings. It contains about 40,000 cells, depending on

the segmentation.

The Calinski-Harabasz-Score is defined as the ratio of the sum of between-cluster dispersion and of within-cluster dispersion. To compute this score, we used the implementation in scikit-learn [Pedregosa et al., 2011]. The mean cluster distance is the average distance between all pairwise combinations of cells between two different clusters; thus, a higher distance indicates a better cluster separation. For the differential expression analysis, we ran the scanpy [Wolf et al., 2018] *rank_genes_groups* function, and we averaged the score of the 20 most significant genes for each cell type. Since we could not run Baysor on the full datasets in Figure 4.7, we run it on 16,000-pixels-wide crops of the MERSCOPE and Xenium datasets, and we computed the ratios between the run with the patches and without patches. We then averaged these ratios across these two datasets, with two runs on each dataset, for each of the above metrics and used the resulting ratios to extrapolate the Baysor score on the full datasets. The time and memory benchmarks were performed on a Slurm cluster on the same CPU nodes. The benchmark related to Cellpose was performed on crops of the MERSCOPE dataset, while the other time and memory benchmarks were performed on a synthetic dataset (see supplementary subsection A.2.1). Figure 4.7e was generated based on the corresponding 16,000-pixels-wide datasets; this involves 25 Cellpose patches and 4 Baysor patches. The percentage of conflicts for Cellpose (compared to all pairs of cells) was 0.007%, while this percentage was 0.001% for Baysor. The UMAPs of Figure 4.8 were generated with scanpy [Wolf et al., 2018], using the default parameters. The MERSCOPE and Xenium datasets have been segmented with Baysor, while the PhenoCycler and MACSima datasets have been segmented with Cellpose. Both the MERSCOPE and Xenium datasets have been annotated using Tangram (see supplementary subsection A.2.1 for more details). Concerning the H&E niches, they were obtained by running a ResNet [He et al., 2016] model pre-trained on ImageNet and applied on patches of size 250x250 pixels.

4.4 . Results

Visualisation of spatial omics in a cross-technology interactive visualizer

In spatial omics analysis, effective visualization is crucial but has presented challenges due to the size of the datasets. While open-source initiatives like Napari [Perkel, 2021] are emerging, they currently face limitations in handling large amounts of transcripts. Also, most companies provide technology-specific visualizers, offering limited user possibilities (see supplementary subsection A.2.1). Yet, 10X Genomics has introduced the Xenium Explorer, an optimized visualizer whose file format is open, i.e., formats that can be generated for various SpatialData types. In Sopa, we have incorporated a converter that transforms the pipeline output into the input files compatible with the Xenium Explorer (see

subsection 4.3.3 and Figure 4.6c). This integration ensures access to an efficient and robust visualizer, extending its functionalities to any technology whose data is readable by Sopa. Importantly, this adaptation applies to both spatial transcriptomics and multiplex imaging data, with the "Transcripts" panel selectively available for transcriptomics data. Figure 4.8b/e shows views using this Explorer, while supplementary Figure A.12 and Figure A.13 provide full-window examples. In addition to visualisation, the Xenium Explorer contains an interactive tool to align images from which we can export a transformation matrix and use it to align images on the SpatialData object to benefit from all the functionalities in Sopa (see supplementary subsection A.2.1).

4.4.1 . Memory and time efficient analysis of spatial omics

Managing large datasets is a critical challenge in spatial omics, particularly when dealing with images that can reach hundreds of gigabytes and contain hundreds of millions of transcripts in spatial transcriptomics data. This necessitates implementing memory optimization techniques to ensure the scalability of the analysis. Notably, segmentation algorithms like Cellpose [Stringer et al., 2021] and Baysor [Petukhov et al., 2022] encounter scalability issues with large images, as illustrated in Figure 4.7a/b. To tackle this, these segmentation models are applied to smaller regions called patches, drastically decreasing random-access-memory (RAM) usage and time. While this patching process generates possible segmentation conflicts, we show in Figure 4.7d/e and in supplementary Figure A.11 that this does not impact segmentation quality, since most conflicting cell boundaries have an intersection-over-min-area (IOMA) lower than 0.07 or higher than 0.8 (see supplementary subsection A.2.1 for more details). Indeed, for cells on overlapping regions, most of the boundary conflicts correspond to either (i) the same cell segmented twice on the two patches (at least one cell is complete, as shown in Figure 4.7c, with one boundary being included in the other), or (ii) different cells slightly overlapping (as shown in the right of Figure 4.7c). Additionally, the conventional storage of cell boundaries as raster masks demands significant memory for storage and processing (see Figure 4.7f). To address this, we adopt a more efficient approach by storing cell boundaries as polygons using Shapely, which proves highly effective for both on-disk and in-memory storage. This also facilitates geometry-related operations, such as cell expansion, area/perimeter computations, and cell-cell intersections. Combined with the image lazy loading feature from SpatialData [Marconato et al., 2024] and Xarray [Hoyer and Hamman, 2017], we implement a fast channel averaging on cell boundaries by combining geometry operations and image chunk lazy loading (see Figure 4.7f), i.e., deferring memory loading until needed for processing. Additionally, using memory-efficient tools like Dask, we extend geometric operations of GeoPandas on chunks of transcripts, ensuring parallel processing of as many chunks as possible without exceeding memory limits (see Figure 4.7g). For image conversion to a pyramidal '.tif', we significantly lower the memory usage compared to what is recommended by 10X Genomics (see subsection 4.3.3) by writing tiles in a lazy man-

ner, which avoids loading the full image in memory (see Figure 4.7h). To highlight Sopa's memory efficiency, we compared its RAM usage against standard methods for all tasks mentioned above across various dataset sizes, summarized in Figure 4.7. Overall, the latter figure shows significant improvements in terms of RAM and time: depending on the tasks, Sopa can require between 10 and 100 times less memory than normal techniques and can be up to 100 times faster. Even on the largest image, Sopa can be run with a simple laptop with 16GB of RAM.

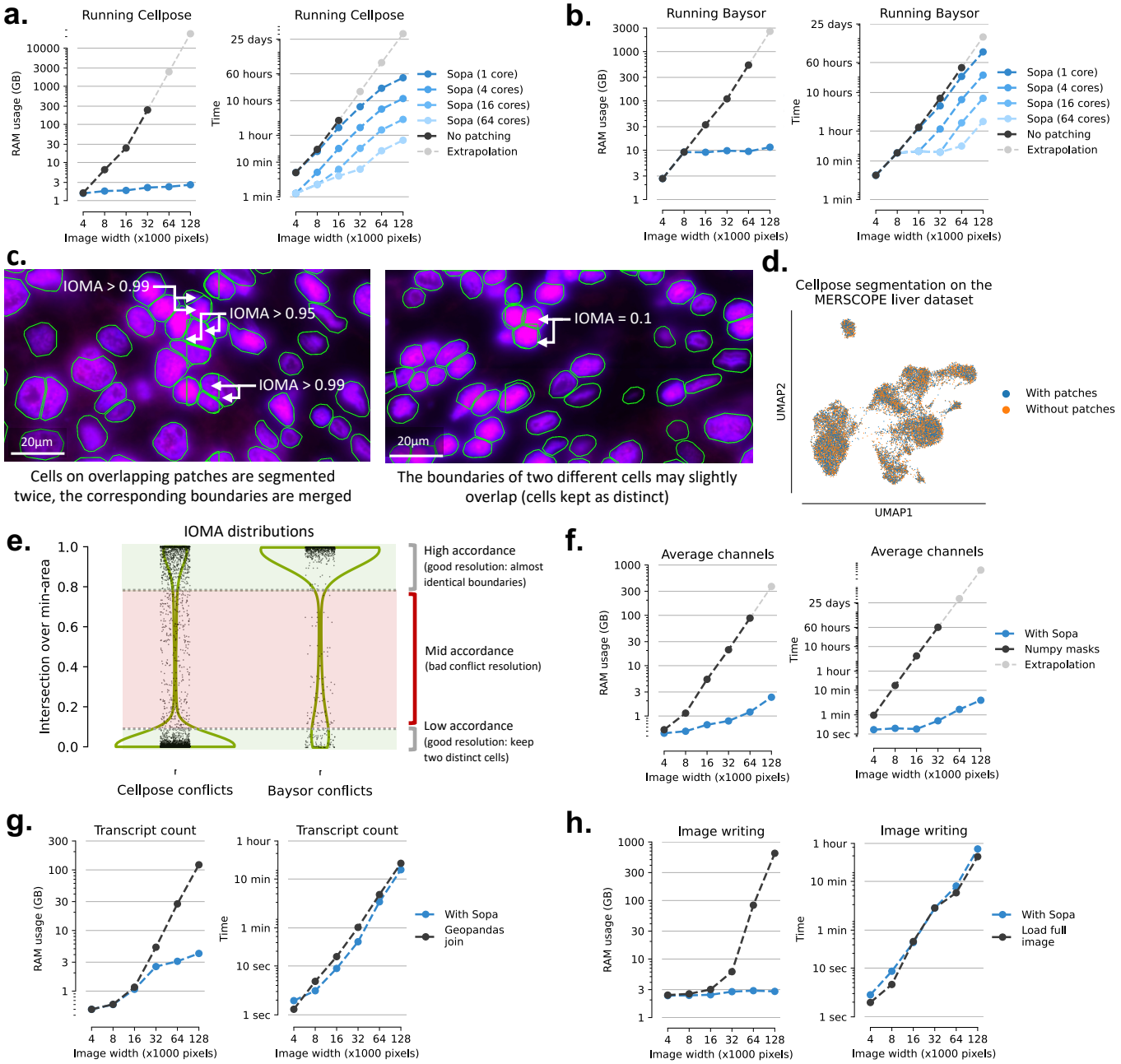

Figure 4.7: **Computational efficiency of Sopa in terms of RAM and time on different dataset sizes. a.** Cellpose segmentation comparison: with and without patching. The RAM usage is given per core. **b.** Baysor segmentation comparison: with and without patching. The RAM usage is given per core. **c.** Examples of cell boundaries before resolving the conflicts over overlapping patches when running Cellpose segmentation on DAPI staining (MERSCOPE human liver hepatocellular carcinoma dataset). On overlapping regions, cells are segmented twice (middle and right). For each conflict, their IOMA determines whether or not to merge the two cell boundaries. **d.** UMAP showing the difference between the resolution with and without the patching process. **e.** Violin plots showing the intersection-over-min-area density of segmentation conflicts when using patches (for both Cellpose and Baysor). When resolving a conflict, the two good cases are either (i) a high concordance between the two cells (which will be merged), or (ii) a low concordance between them (the two cells are kept). IOMA below 0.07 or above 0.8 correspond to good conflict resolution cases. **f.** Channels averaging for each cell: Sopa and standard average inside numpy masks. **g.** Counting each gene inside each cell: with Sopa compared to GeoPandas join operation on the whole DataFrame. **h.** Writing image as a tiff file for the Xenium Explorer: with Sopa compared to what is recommended by 10X Genomics, i.e. loading the whole image in memory. Source data are provided as a Source Data file.

4.4.2 . A wide range of use cases for different levels of expertise

Sopa offers three distinct options, each tailored to different use cases: (i) a Snakemake [Köster and Rahmann, 2018] pipeline that enables a quick start within minutes, (ii) a CLI that facilitates rapid prototyping of a personalized pipeline, and (iii) an API that allows direct usage of Sopa as a Python package (`https://github.com/gustaveroussy/sopa`), providing full flexibility and customization. The Snakemake pipeline remains consistent across various technologies, with only its configuration differing. Users can leverage existing configuration files, selecting one that aligns with their technology, which then enables them to execute the pipeline without any code updates. Another advantage of Sopa's generality and scalability is that more advanced users seeking customisable pipelines can use the CLI. Notably, Sopa's general design allows for an easy integration of any state-of-the-art or custom segmentation methods such as ComSeg [Defard et al., 2024], rendering them memory-efficient and accessible for all image-based spatial omics applications. Additionally, the Python API is available for users interested in incorporating specific parts of Sopa into their personal libraries. This API also facilitates integration with other tools of the scverse [Virshup et al., 2023] ecosystem, such as Scanpy [Wolf et al., 2018] or Squidpy [Palla et al., 2022] (see supplementary subsection A.2.1). In particular, the integration with Squidpy enables the use of post-processing tools for cell-cell interaction and spatially variable gene analysis.

4.4.3 . High resolution of the tumour microenvironment

Segmentation plays a crucial role in image-based spatial omics analysis. Sopa focuses significantly on improving this step (see subsection 4.3.1) by enabling the usage of state-of-the-art segmentation models like Baysor [Petukhov et al., 2022] on large datasets. Indeed, as shown on Figure 4.7a/b, these high-quality segmentation tools use a lot of memory, which hinders their usage on large spatial datasets. To evaluate the resolution provided by Sopa after segmentation, we annotated major cell types and conducted tests on four datasets: two spatial-transcriptomics datasets (MERSCOPE and Xenium) and two multiplex-imaging datasets (PhenoCycler and MACSima), see subsection 4.3.6 and supplementary subsection A.2.1 for more details. For the MERSCOPE and Xenium datasets, proprietary segmentations were provided by Vizgen and 10X Genomics, respectively. In comparison to these segmentations, Sopa shows an improved cell-type distinction on UMAP [McInnes et al., 2018] plots (see Figure 4.8a/d) by leveraging Baysor. To support these visual observations, we used multiple metrics (see subsection 4.3.6), indicating that Sopa can generate more significant population-specific genes, greater intra-cluster distance, and improved cluster separation (see Figure 4.8c/f). The increased resolution in spatial omics data allows for a more in-depth exploration compared to previous segmentations.

Sopa also facilitates the concurrent analysis of both RNA and proteins. To demon-

strate this, we used the Xenium dataset, which includes transcriptomic expression and protein stainings (CD20, PPY and TROP2). CD20 is a common marker for B cells, PPY is expressed by endocrine cells, and TROP2 is overexpressed in tumour cells. 10X Genomics currently does not produce files with protein expression per cell, while Sopa does support the analysis of proteins. To demonstrate this feature, we aligned the Xenium staining image to the original coordinate system (see supplementary subsection A.2.1), and Sopa computed the CD20/PPY/TROP2 intensity within all cell boundaries. Combined with transcriptomic expression, CD20 staining greatly facilitates the annotation of B cells, as shown by their clear delimitation on Figure 4.8d and supplementary Figure A.14. In the future, we expect technologies to be able to run more protein stainings in parallel with transcriptomics data, making this kind of analysis even more valuable.

Regarding multiplex imaging, Sopa shows efficiency in (i) managing large protein staining panels and (ii) segmenting millions of cells (using Cellpose). The former is exemplified by the MACSima dataset with 61 stained proteins. Again, we computed staining intensity per cell, and Figure 4.8g demonstrates Sopa's capacity to annotate high-resolution cell types. Secondly, the PhenoCycler dataset underscores Sopa's ability to handle datasets of substantial size, with an area of 3cm², containing approximately 2,500,000 cells. The corresponding cell resolution is shown in Figure 4.8h/i.

In summary, these studies demonstrate that Sopa can (i) be applied across diverse technologies, (ii) efficiently handle millions of cells, and (iii) seamlessly operate on both transcriptomics and protein stainings.

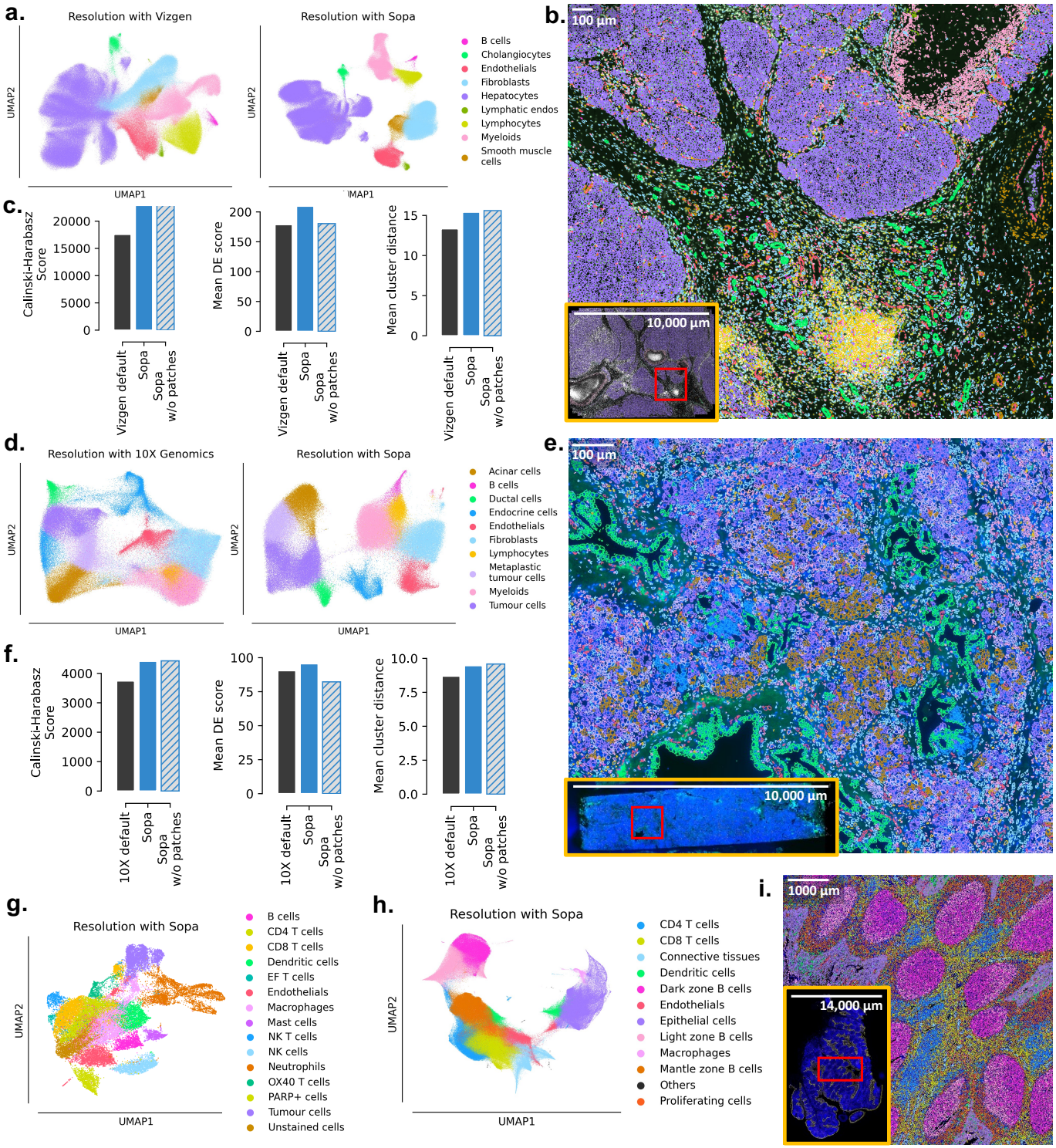

Figure 4.8: **Data resolution after Sopa segmentation over two spatial-transcriptomics technologies (MERSCOPE (a-c) and Xenium (d-f)) and over two multiplex-imaging technologies (g-i). a.** UMAPs after Vizgen proprietary segmentation on the MERSCOPE human liver hepatocellular carcinoma dataset (left) and after Sopa segmentation on the same dataset (right). **b.** Visualization of cell types on the MERSCOPE dataset after annotation with Sopa. Colours correspond to the legend of (a). **c.** Three cluster separation metrics compare the quality of these two segmentations on the MERSCOPE dataset. The grey hatched boxes extrapolate the score Sopa would have without running on patches. **d.** UMAPs of cells after 10X Genomics proprietary segmentation on the Xenium human pancreatic cancer dataset (left) and after Sopa segmentation on the same dataset (right). **e.** Visualization of cell types on the Xenium dataset after annotation with Sopa. Colours correspond to the legend of (d). **f.** Three cluster separation metrics compare the quality of these two segmentations on the Xenium dataset. The grey hatched boxes extrapolate the score Sopa would have without running on patches. **g.** UMAP of cell types on the MACSima dataset (head and neck squamous cell carcinoma), based on 61 protein stainings. **h.** UMAP of cell types on the PhenoCycler dataset (human tonsil), based on 31 protein stainings. **i.** Cells of the PhenoCycler dataset visualized. The colours correspond to the legend of (h). Source data for (c, f) are provided as a Source Data file.

4.4.4 . Demonstration of geometric and spatial analyses capabilities

Spatial omics naturally unlocks multiple biological questions related to spatial organization. While some are addressed in libraries such as Squidpy [Palla et al., 2022], metrics related to the distance between cell-types/niches and the geometric characteristics of those niches are not provided. These metrics could help in the understanding of the morphology of the tumour micro-environment and its location with regard to different cell types. Such statistics have been shown to be relevant for predicting disease progression or response to treatment [Jass, 2007, Sharma et al., 2005]. For instance, it is known that tertiary lymphoid structures (TLS) have a good prognosis [Sautès-Fridman et al., 2019], but their geometry has not been studied. TLS may come in different sizes, shapes, occurrences, or locations with regard to other niches. Such statistics are generalized in subsection 4.3.5 for all cell categories (usually, cell types or niches). Leveraging this spatial analysis, we demonstrate a better understanding of the dynamics among different cell types and their relation to different spatial niches on the MERSCOPE liver dataset (Figure 4.9). To use Sopa geometric analysis, we run STAGATE [Dong and Zhang, 2022] to identify eight distinct niches (or "spatial domains") across various tumour regions (Figure 4.9a). First, we show in Figure 4.9b four geometric properties related to these niches: for each niche compartment, we counted their occurrence on the same slide, as well as their mean area, perimeter, and roundness (see subsection 4.3.5 for more details). For instance, our geometric analysis shows a high occurrence of vascular niches, that are small in area and perimeter, but have a high roundness. Conversely, the stroma has only one occurrence and is highly "unround", and Figure 4.9c shows that this shape enables a "proximity" to every other niche. Figure 4.9c also highlights how far the vascular niche is from the necrosis. While such observations are not novel, our geometric computation allows for statistical comparisons over multiple patients, which could lead to the discovery of significant geometric biomarkers in large-scale studies. Finally, Squidpy [Palla et al., 2022] already incorporates functionalities on neighbourhood enrichments, which is a local metric and, therefore, not suited to capture niche-level information. In comparison, the distance metric used in Sopa can capture asymmetrical observations and global organizations (see supplementary Figure A.16 for more details).

We also utilised Sopa to assess the intricacies of the tumour complexity. We annotated the immune populations of the MERSCOPE dataset in higher definition (see supplementary Figure A.15) and, in parallel, performed a differential analysis on each niche to better understand niche complexity. This revealed a distinct necrotic niche correlated with *TREM2* macrophages (expressing *TREM2*, *C1QC* and *CSF1R*), a population of macrophages reported across cancer types and often associated with bad prognosis [Molgora et al., 2020, Binnewies et al., 2021] (see Figure 4.9d and supplementary Figure A.15). To deepen this understanding of tissue intricacies, we investigated whether these *TREM2* macrophages

were in close distance with any other cell type (see Figure 4.9e). Strikingly, this figure highlighted that three macrophage populations (*LRP1*, *CEBP*, and *TREM2*-macrophages) exclusively interacted with themselves. Correlating their location with the niche revealed that their co-occurrence is specific to the necrotic niche (see Figure 4.9f). When combining all (cell-cell/cell-niche/niche-niche) interactions, this affirms again the association of *LRP1*/*CEBP*/*TREM2*-macrophages in the necrotic niche, yet it also highlights the heterogeneity of all macrophage populations and their relation to the niche in the whole tissue environment (see Figure 4.9g). These combined interactions also showed that, inversely, the conventional dendritic cells (DCs) are not associated with any niche environment, accentuating how some populations can also be niche-independent. This observed spatial location underscores a potential reprogramming feature of macrophages based on their specific niche. While it is known that the accumulation of *TREM2* macrophages has been associated with enrichment in the tumour regions [Mulder et al., 2021, Sharma et al., 2020, Zhou et al., 2022], Sopa can provide insights for a refined understanding of a macrophage-specific tumour-associated phenotype. These examples illustrate that this geometric and spatial analysis — computed with Sopa — helps better understand the tumour's architecture and its relationship with cell type phenotypes.

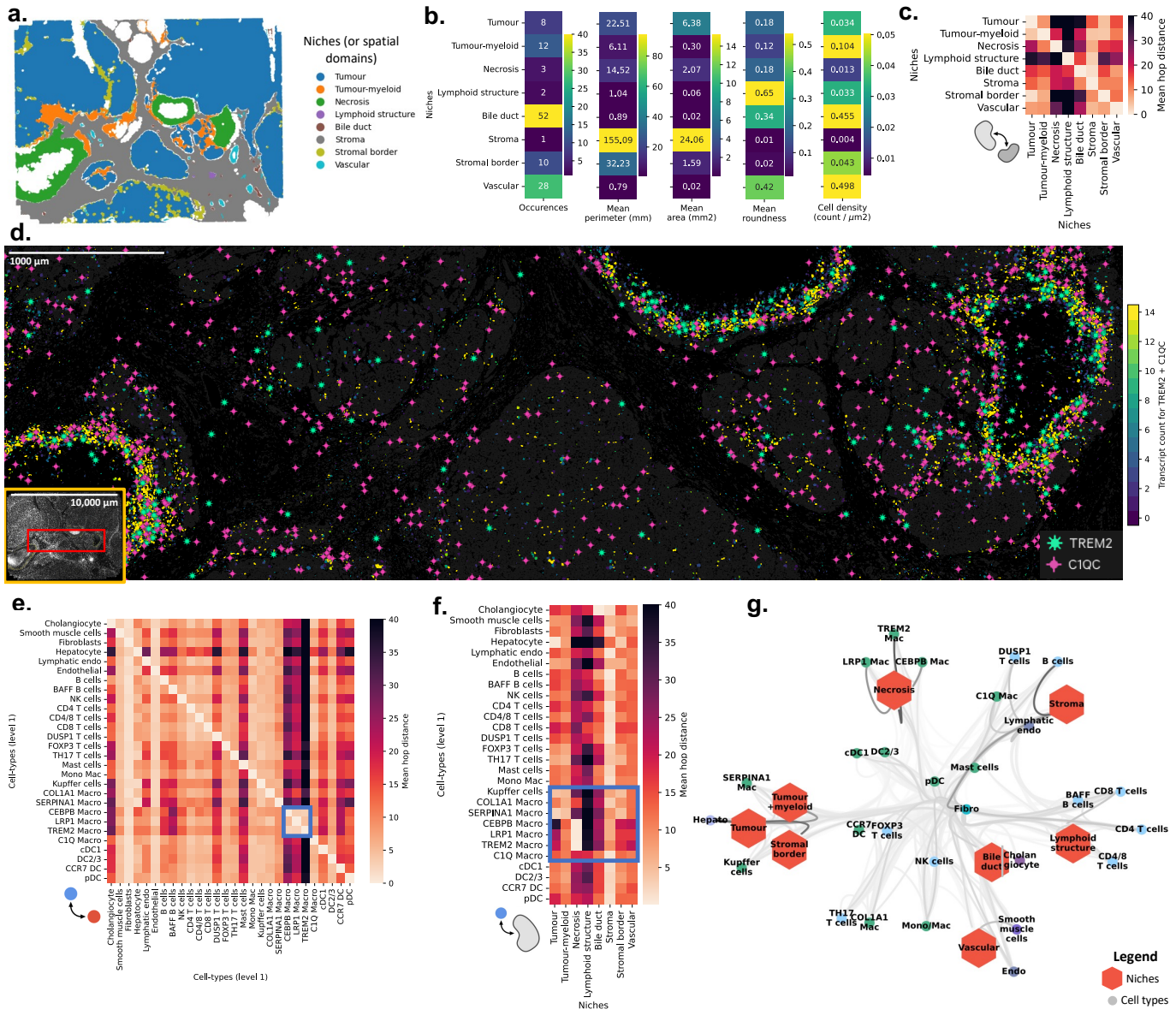

Figure 4.9: **Geometric analyses and spatial statistics on the MERSCOPE human liver hepatocellular carcinoma dataset. a.** Niches (or spatial domains) after geometric conversion to shapely polygons. **b.** Geometric statistics of the niches: their occurrence, perimeter, area, roundness, and inner cell density. **c.** Heatmap of average hop distance between niches and niches. **d.** Localisation of *TREM2* macrophages shown in the visualizer. The *TREM2* and *C1QC* genes are shown, and cells are coloured by their gene counts for the two selected genes. **e.** Heatmap of average hop distance between cell types and all other cell types. *LRP1*, *CEBP*, and *TREM2* macrophages show a high proximity. **f.** Heatmap of average hop distance between cell types and niches. The macrophage subpopulations show heterogeneous localisation with respect to the niches. *LRP1*, *CEBP*, and *TREM2* macrophages are enriched in the necrosis niche. **g.** Network plot summarising the distance metrics of (c)/(e)/(f). Each node of the network corresponds either to a niche (hexagon) or a cell type (circle). The lower the mean distance between the two nodes, the higher the weight of the edge between these two nodes. A high node-node proximity is shown by a dark edge. Overall, it provides an overview of the colocalisation of cell types and niches in the tumour environment.

4.4.5 . Incorporation of H&E into the multi-omics spatial analysis

Some technologies, such as the Xenium, have been developed to get Hematoxylin and Eosin (H&E) staining and protein staining on the same slide used for Spatial Transcriptomics. By aligning the modalities (with the Xenium Explorer, as detailed in the supplementary subsection A.2.1), Sopa enables analyses that can interplay with all three modalities. Especially, the H&E modality, via the colour and texture, captures extra information that the two other modalities do not contain. For instance, H&E may be stronger in regions with low RNA information, such as high collagen regions (see supplementary Figure A.17). In Figure 4.10, we perform analyses that couple the three layers to provide interpretability to the H&E niches (see subsection 4.3.6 for more details on the niches computation). Figure 4.10c shows that H&E-based niches are highly heterogeneous in terms of cell types, with some niches being highly enriched in some particular populations. Notably, niche 3 is highly specific to Acinar cells, niche 5 is specific to Ductal cells, while niche 4 is enriched in B cells and Myeloid cells. Also, Figure 4.10d shows differentially expressed genes inside each niche, providing complementary insights to Figure 4.10c, such as *TM4SF4* and *APCDD1* being highly specific to niche 5. Finally, Figure 4.10e exemplifies the analysis of the distribution of protein stainings inside these H&E niches, with TROP2 being more expressed in niche 0 and 3, which correspond to the tumour-specific niches identified in Figure 4.10c. Overall, these examples show the capability of Sopa to use one spatial modality to bring insights into another spatial modality.

4.5 . Discussion

4.5.1 . Summary of Sopa

Advances in technology development for spatial omics hold great promise for biological discoveries. Yet, to build strong and unified foundations for spatial omics data analysis, more tools are required. With this purpose in mind, we designed and built Sopa to address several crucial aspects of spatial omics analysis: versatility, reproducibility, and scalability. It offers a suite of tools — or building blocks — designed for spatial omics, which are assembled to build a pipeline for any image-based spatial omics technology. At the end of the pipeline, it produces standardized outputs, which ease exploration and visualization. While each company's technology comes with its own suite of tools — which differ in terms of capabilities and functionalities — Sopa does not require learning from multiple data types and software. In addition, Sopa is scalable from simple laptops to high-performance clusters, offering further versatility for users.

Moreover, Sopa can easily integrate recent methods and tools: as future segmentation or annotation methods are developed, they can be added to Sopa once published and validated. This incorporation into Sopa enables scalability and availability to any fu-

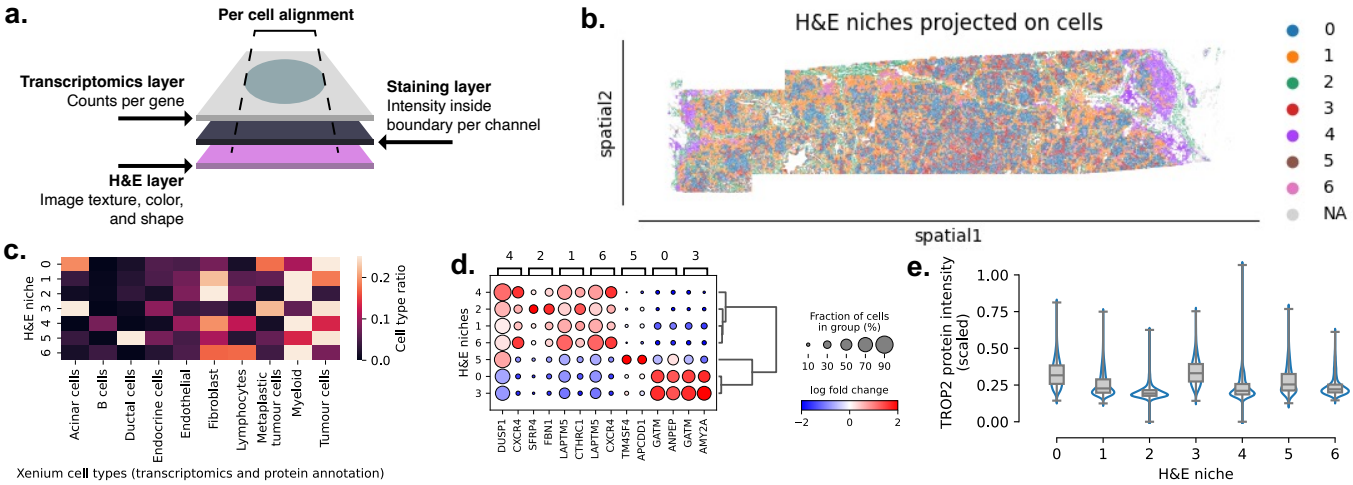

Figure 4.10: ***Spatial multi-omics analyses on the Xenium pancreatic cancer dataset.*** ***a.*** *Overview of spatial multi-omics alignment. For each single cell, the information of (i) transcriptomics, (ii) stainings, and (iii) H&E is combined after the alignment of all the different layers.* ***b.*** *H&E clusters of patch-level embeddings based on a pre-trained computer vision model (denoted as H&E niches). The figure shows the cells obtained from spatial transcriptomics data and coloured by the H&E patch cluster inside which they are included.* ***c.*** *Proportion of cell types inside each H&E niche. The cell types are the cell types annotated using both spatial transcriptomics and protein information as in Figure 4.8.* ***d.*** *Differential gene expression performed on the H&E niches using single-cell resolution. The size of the dots indicates the percentage of cells in the given niche that are have a positive count of transcript for the given gene on the x-axis.* ***e.*** *Distribution of TROP2 intensities per cell (N=175,022) inside each H&E niche, showcasing the usage of the staining layer coupled with the H&E information. Source data for (e) are provided as a Source Data file.*

ture technology with only minor configuration changes. As datasets become increasingly bigger, Sopa's scalability is crucial. For instance, Sopa enabled the possibility of running Baysor on data produced by the MERSCOPE, which was previously impossible due to RAM usage and time. Assessing the effect of patch-based segmentation showed no significant difference in segmentation quality. We also demonstrated that Baysor significantly increases data quality compared to the default Vizgen and 10X Genomics segmentation tools, which aligns with Hartman et al. [Hartman and Satija, 2024].

As shown on the MERSCOPE liver dataset, we were able to annotate spatial-specific macrophages, particularly *TREM2* macrophages, in the necrotic niche. Additionally, *TREM2* has been shown to increase with Hepatocellular Carcinoma, suggesting a potential immunosuppressive role of *TREM2* [Zhou et al., 2022, Mulder et al., 2021], while necrosis has been associated with worse prognosis [Wei et al., 2021, Bijelic and Rubio, 2021]. With the help of Sopa, the exploration of this relationship between tissue architecture and cell phenotypes can advance biological knowledge.

Besides higher data resolution, Sopa can also incorporate protein and H&E information into spatial analysis. Without this protein layer, extracting the B cell population in the

Xenium data would not have been possible. Concerning the H&E layer, we can benefit from the transcriptomics layer to bring interpretability to the H&E tissue characterization or also build upon Sopa to develop tools that predict refined spatial-transcriptomics cell types based on H&E images. While current spatial technologies involve either a high number of proteins or transcripts, future developments could add extra layers of information, contributing to a better understanding of biological systems. This paper has demonstrated through various techniques that Sopa is ready to handle large multi-modal spatial technologies.

4.5.2 . Position of Sopa in an open-source context

Sopa is an analysis and pre-processing tool within the scverse ecosystem, specifically designed to work with SpatialData as its underlying data structure. Its primary function is to extract high-quality information from raw spatial data, serving as a replacement for proprietary pre-processing methods. Once processed by Sopa, the data is ready for further analysis, whether using other tools within the scverse ecosystem or leveraging deep learning capabilities through Novae, which is described in the next chapter. Sopa bridges the gap between raw data acquisition and advanced spatial data analysis, integrating seamlessly with the broader scverse workflow.

A graph-based foundation model for spatial transcriptomics data

Contents

Abstract

Spatial transcriptomics is advancing molecular biology by providing high-resolution insights into gene expression within the spatial context of tissues. This context is essential for identifying spatial domains, enabling the understanding of micro environment organizations and their implications for tissue function and disease progression. To improve current model limitations on multiple slides, we have designed Novae (`https://github.com/MICS-Lab/novae`), a graph-based foundation model that extracts representations of cells within their spatial contexts. Our model was trained on a large dataset of nearly 30 million cells across 18 tissues, allowing Novae to perform zero-shot domain inference across multiple gene panels, tissues, and technologies. Unlike other models, it also natively corrects batch effects and constructs a nested hierarchy of spatial domains. Furthermore, Novae supports various downstream tasks, including spatially variable gene or pathway analysis and spatial domain trajectory analysis. Overall, Novae provides a robust and versatile tool for advancing spatial transcriptomics and its applications in biomedical research.

5.1 . Introduction

Spatial transcriptomics [Atta and Fan, 2021, Bressan et al., 2023] data provide invaluable insights into cellular interactions within their micro-environment and the complexities of tissue organization. A key advantage over current single-cell RNA sequencing (scRNAseq) [Du et al., 2023] is that spatial transcriptomics maintains the spatial context of cells, enabling a deeper understanding of how cells interact within their native environments. Technologies in spatial transcriptomics can be broadly categorized into two types: (i) Next-Generation Sequencing [McCombie et al., 2019] (NGS)-based methods that offer whole-genome sequencing, and (ii) imaging-based techniques like the Xenium [Janesick et al., 2023], MERSCOPE [Chen et al., 2015], or CosMX [He et al., 2022], that provide subcellular resolution. The former allows for comprehensive gene analysis but lacks fine spatial detail, while the latter offers detailed spatial resolution but with a limited gene panel size. As imaging-based technologies continue to evolve, they expand their gene panel capabilities, enabling the inclusion of larger panels or the replacement of low-quality genes during studies. However, this flexibility often results in experiments conducted across different machines or using varying panels, which introduces new challenges. In more general cases, when performing analysis over multiple spatial transcriptomics slides (for both NGS or imaging-based techniques), it is common to observe a strong batch effect, such that it can be challenging to identify common spatial patterns across multiple slides without careful consideration of the batch effect.

A key focus in spatial transcriptomics is the identification and categorization of spatial

micro-environments, often referred to as spatial domains or niches. Various methods, such as STAGATE [Dong and Zhang, 2022], GraphST [Long et al., 2023], SpaceFlow [Ren et al., 2022], and SEDR [Xu et al., 2024], have been developed for this purpose. While these methods show promising results (especially with NGS-technologies with spot resolution like Visium), they are limited by (i) their reliance on predefined gene panels, (ii) sensitivity to batch effects, and (iii) dependence on external tools like Harmony [Korsunsky et al., 2019] for batch effect correction and Leiden [Traag et al., 2019] or Mclust [Scrucca et al., 2016] for clustering. These dependencies can slow down processing and reduce robustness, since external tools need to be re-run for each new analysis or when adjusting spatial domain resolutions (i.e., choosing different number of spatial domains). Additionally, due to their reliance on specific gene sets, these methods often necessitate training on the intersection of gene sets, which can significantly reduce the number of available genes and, consequently, impact performance. Most importantly, even when applied to slides with a shared panel, these models tend to identify primarily slide-specific domains, which limits the comparison of domains across a broader study and reduces the potential for discovering new spatial biomarkers.

To address these limitations, we introduce Novae, a self-supervised [Jaiswal et al., 2021] graph attention network [Brody et al., 2022] that encodes local environments into spatial representations. Unlike existing methods, the same Novae model can operate with multiple gene panels, allowing for the application across diverse technologies and tissues. It includes native batch effect correction methods, directly correcting for variations and enhancing robustness and scalability. Therefore, Novae's design allows it to seamlessly integrate data from different platforms and gene panels without compromising performance. We trained Novae on a large dataset comprising 78 slides, representing nearly 30 million cells across 18 tissues and three different subcellular resolution technologies (Xenium [Janesick et al., 2023], MERSCOPE [Chen et al., 2015], CosMX [He et al., 2022]). This broad training allows Novae to compute relevant representations via zero-shot [Xian et al., 2019] or fine-tuning on any new slide from any tissue. These representations can be directly used for spatial domain identification, eliminating the need for external clustering tools. Beyond spatial domain identification, these representations can be applied to various downstream tasks, including (i) spatial domain trajectory analysis, (ii) spatially variable gene analysis, and (iii) spatial pathways analysis. Novae's versatility, robustness and ease of use make it a powerful tool for advancing spatial transcriptomics research within the scientific community.

5.2 . Methods

5.2.1 . Broad overview of the Novae methodology

This short section details broadly the method behind Novae. It is described in more details in the following sections.

Novae is a graph-based model that we trained on a large single-cell spatial transcriptomics dataset composed of about 30 millions cells. This model, aware of 18 different tissues, is hosted on a public hub and can be reused by the user for multiple downstream tasks on any technology or gene panel, as illustrated in Figure 5.1a. The main application of Novae is to learn spatial domains, which can be performed with the shared model without any re-training (called zero-shot inference [Xian et al., 2019]). If desired, the user can also re-train Novae to obtain refined results (fine-tuning). Two distinguishing properties of Novae are that (i) it provides a nested organization of spatial domains for different resolutions and (ii) it natively corrects batch effect across slides. This consistent domain assignment across slides enables comparison analyses of a study containing multiple slides. Additionally, Novae can perform multiple downstream analysis tasks, such as (i) spatial pathway analysis, (ii) spatial domain organization/architecture analysis, or (iii) spatial variable genes analysis. These tasks and properties are summarized in Figure 5.1b.

In terms of technical details, Novae is a graph-based neural network that is trained in a self-supervised manner based on the SwAV [Caron et al., 2020] framework. It learns representations of the local microenvironment at the single-cell (or spot) resolution. More specifically, for each cell or spot, a representation of its neighborhood is provided by a Graph Attention Network [Brody et al., 2022], after embedding each cell into a panel-invariant representation. In order to effectively assign domains, we learn embeddings in the latent space, called prototypes, which represent elementary spatial domains. Depending on the desired level of resolution, these elementary domains are regrouped to form the domains at the desired resolution. We project the spatial-domain representations into these prototypes to get probabilities of assignments. These probabilities are corrected via an algorithm of Optimal Transport [Peyré and Cuturi, 2020] to ensure a smooth representation of the different prototypes. Afterwards, we use cross-entropy loss to make the representation of two cells in the same domain closer together. This methodological approach is illustrated in Figure 5.1c.

5.2.2 . Model input

In Novae, we consider a total of G genes, such as those in the human or mouse genome. A spatial transcriptomics slide captures the expression profile of a subset of these genes, denoted by $\mathcal{P} \subseteq \{1, \dots, G\}$, representing the indices of the genes in this specific panel. Throughout this article, we use the term "cell" to refer to either a spot or a cell, although Novae is applicable to both spot-resolution and single-cell resolution technologies. The expression data for a single slide is represented by a matrix $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_N) \in \mathbb{R}^{N \times P}$, where N denotes the number of cells and $P = \text{Card}(\mathcal{P})$ represents the number of

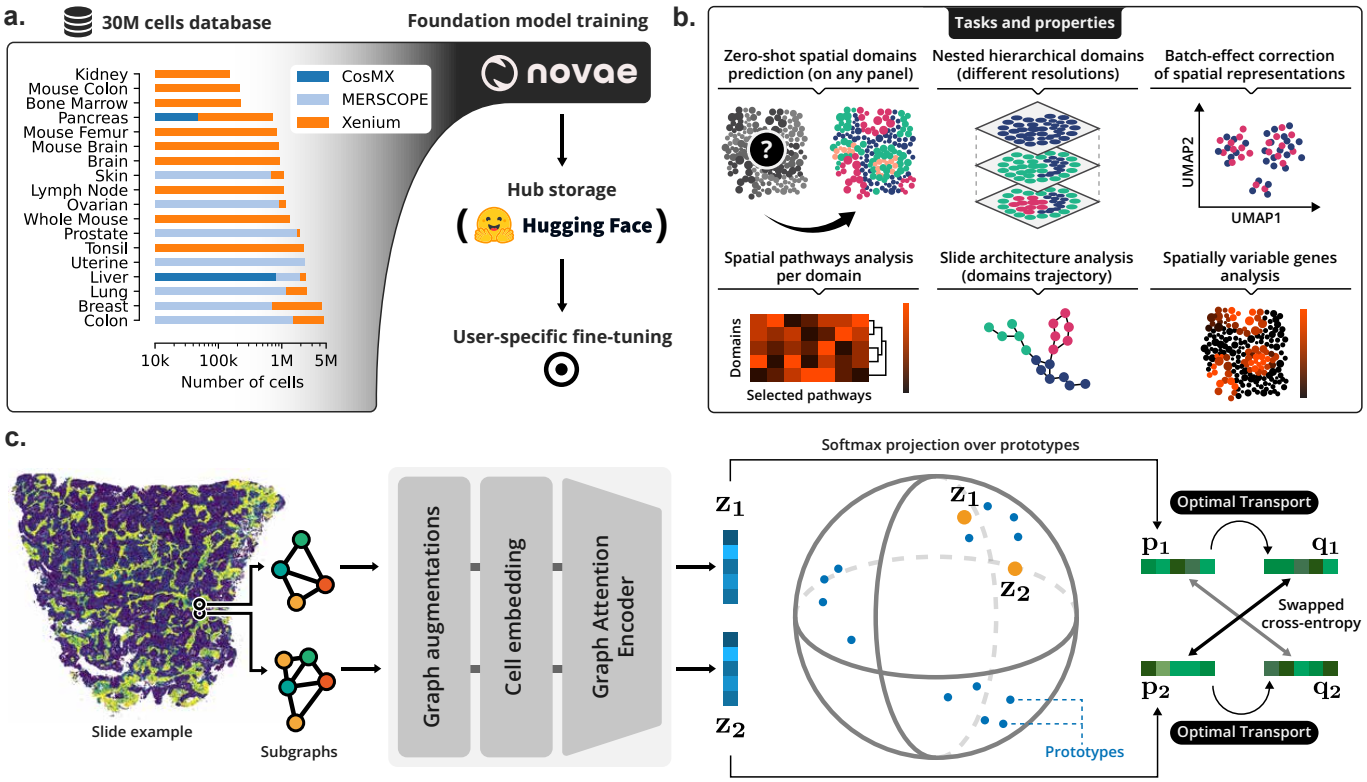

Figure 5.1: ***Overview of Novae. a.*** *The dataset is composed of 29 million cells across 18 different tissues for three different single-cell spatial transcriptomics technologies. This dataset was used to train Novae, which was shared online on Hugging Face Hub. Users can easily load the pre-trained model for reuse or fine-tuning.* ***b.*** *A visual summary of the key tasks and capabilities of Novae. Notably, Novae can infer spatial domains in a zero-shot manner, organizing them hierarchically. Additionally, based on learned representations, Novae can align representations across different batches. Other downstream tasks include spatial pathway analysis, slide architecture analysis, and spatially variable gene analysis.* ***c.*** *An overview of Novae's underlying method. The process involves extracting two nearby subgraphs of cells (or spots), which are then augmented and embedded. These graph inputs are processed through a graph attention encoder, generating representations for both subgraphs. These representations are projected onto prototypes—learnable vectors from the latent space. Finally, an optimal transport task is applied to the assignment probabilities over these prototypes, and a swapped cross-entropy loss is computed for backpropagation.*

genes in the panel $\mathcal{P}$. This matrix $\mathbf{X}$ contains normalized and logarithmized gene expression values. To incorporate spatial information, we utilize the 2D localization of cells to construct a Delaunay graph. In this graph, nodes correspond to cells, and edges indicate neighboring cells. The adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ encodes the connectivity between cells, with edge weights representing the distances between neighboring cells in the Delaunay graph. Also, an entire spatial transcriptomics dataset consists of multiple slides, each described by a tuple $(\mathbf{X}, \mathcal{P}, \mathbf{A})$, where the size of each gene panel Card$(\mathcal{P})$ can vary across different slides. Additionally, for a given slide and a specific cell i, we define a subgraph $\mathcal{G}_i$ consisting of cells within a distance of n_{local} edges from cell i. The features of a node (or cell) in this subgraph are the gene expression vectors of the corresponding cells. For instance, the feature vector for cell i is $\mathbf{x}_i \in \mathbb{R}^P$. In the following sections, $\mathcal{G}_i$ is used

to obtain a spatial domain representation for the subgraph origin cell i.

5.2.3 . Augmentation

Augmentation is a technique used to introduce artificial variations into the data, enhancing the dataset's diversity. This approach is commonly employed to improve the robustness and generalization abilities of models. In the context of spatial transcriptomics, we apply two biologically relevant augmentations to strengthen the robustness of Novae.

Firstly, we introduce a "pseudo batch effect" noise to reduce the model's sensitivity to batch effects. For a given subgraph input, we sample two vectors: $\mathbf{a} \sim \text{exponential}(\lambda)^P$ and $\mathbf{s} \sim \text{Normal}(0, \sigma^2 \mathbf{I}_P)$, where λ and σ are hyperparameters. The gene expression vector for each cell i is then updated as $\mathbf{x}_i^{(noise)} = \mathbf{a} + \mathbf{x}_i \odot (1 + \mathbf{s})$.

Secondly, we randomly subset the gene panel according to a ratio $\gamma \in]0, 1[$. For a given panel $\mathcal{P}$, we select $\lfloor \gamma P \rfloor$ genes, resulting in a new panel $\mathcal{P}' \subset \mathcal{P}$. This augmentation simulates the effect of different panels of genes, as if multiple machines generated the data, or as if the panel was updated during a study. After applying these augmentations, each cell is represented by the expression vector $\mathbf{x}_i^{(augmented)} = (x_{ij}^{(noise)})_{j \in \mathcal{P}'}$. This augmented representation helps the model better generalize across varying conditions and datasets.

5.2.4 . Cell embedding

After augmentation, cells are transformed into panel-invariant embeddings, which serve as the node features for the graph encoder described in the subsequent section. Let $\mathbf{v_1}, \dots, \mathbf{v_G} \in \mathbb{R}^E$ denote a list of trainable gene embeddings, where E represents the embedding size. For a given gene panel $\mathcal{P} \subseteq \{1, \dots, G\}$, we consider only the embeddings corresponding to the genes in this panel. These embeddings are L2-normalized and then multiplied by the cell's gene expression vector $\mathbf{x} \in \mathbb{R}^P$. Specifically, the embedding for a cell i is calculated as follows:

$$\text{embed}(\mathbf{x}_i, \mathcal{P}) = \frac{\sum_{j=1}^{P} x_{ij} \mathbf{v}_{\mathcal{P}[\mathbf{j}]}}{\sqrt{\sum_{j=1}^{P} \mathbf{v}_{\mathcal{P}[\mathbf{j}]}^2}} \in \mathbb{R}^E,$$

where the square root, square, and division operations are performed element-wise. The purpose of the L2 normalization is to ensure that the embedding weights are comparable across different gene panel sizes. This can be compared to a principal component analysis (PCA) reduction, where the components are trainable gene embeddings, and the L2 normalization ensures that each panel has consistent weights across different "gene programs". Specifically, for each $e \leq E$, $\mathbf{v}_{ge}$ represents the weight of gene $\mathbf{v}_g$ in the gene program e. Additionally, rather than training all gene embeddings from scratch, we can initialize with pre-trained gene embeddings from scGPT [Cui et al., 2024]. To prevent domain shift for genes not present in the Novae dataset, these pre-trained embeddings are frozen, and we introduce a trainable linear layer afterward. Thus, the updated cell em-

bedding formula becomes

$$\text{embed}(\mathbf{x}_i, \mathcal{P}) = \frac{\sum_{j=1}^{P} x_{ij}(\mathbf{W}\mathbf{v}_{\mathcal{P}[\mathbf{j}]} + \mathbf{b})}{\sqrt{\sum_{j=1}^{P}(\mathbf{W}\mathbf{v}_{\mathcal{P}[\mathbf{j}]} + \mathbf{b})^2}} \in \mathbb{R}^E,$$

where $\mathbf{W} \in \mathbb{R}^{E\times E}$ and $\mathbf{b} \in \mathbb{R}^E$ are trainable parameters. During training, the function $\text{embed}(\cdot,\cdot)$ is applied to the augmented gene expression vector $\mathbf{x}_i^{(augmented)}$, whereas during prediction or inference, it is applied to the original gene expression vector $\mathbf{x}_i$.

5.2.5 . Graph encoder

The graph encoder utilized in Novae is a Graph Attention Network [Brody et al., 2022] (GAT), which employs attention mechanisms to aggregate information from neighboring cells. The GAT is composed of multiple layers, with each layer potentially having multiple attention heads. The input to the GAT is a subgraph $\mathcal{G}$, where node features are embedded cell features, as described in subsection 5.2.4. For each cell i, the initial node feature is $\mathbf{h}_i^{(0)} = \text{embed}(\mathbf{x}_i^{(augmented)}, \mathcal{P}')$ during training, and $\mathbf{h}_i^{(0)} = \text{embed}(\mathbf{x}_i, \mathcal{P})$ during inference. For each layer l, the node features for the next layer are calculated as

$$\mathbf{h}_i^{(l+1)} = \sum_{j\in\mathcal{N}(i)\cup\{i\}} \alpha_{i,j}^{(l)} \mathbf{\Theta}_t^{(l)} \mathbf{h}_j^{(l)},$$

where the attention coefficients $\alpha_{i,j}^{(l)}$ are defined as

$$\alpha_{i,j}^{(l)} = \frac{\exp\left(\mathbf{a}^{(l)\top}\text{LeakyReLU}\left(\mathbf{\Theta}_s^{(l)}\mathbf{h}_i^{(l)} + \mathbf{\Theta}_t^{(l)}\mathbf{h}_j^{(l)} + \mathbf{\Theta}_e^{(l)}A_{ij}\right)\right)}{\sum_{k\in\mathcal{N}(i)\cup\{i\}}\exp\left(\mathbf{a}^{(l)\top}\text{LeakyReLU}\left(\mathbf{\Theta}_s^{(l)}\mathbf{h}_i^{(l)} + \mathbf{\Theta}_t^{(l)}\mathbf{h}_k^{(l)} + \mathbf{\Theta}_e^{(l)}A_{ik}\right)\right)}.$$

In these equations, $(\mathbf{a}^{(l)}, \mathbf{\Theta}_s^{(l)}, \mathbf{\Theta}_t^{(l)}, \mathbf{\Theta}_e^{(l)})_{1\le l\le L}$ denotes parameters of the model (biases or matrices), and $\mathcal{N}$ represents the set of neighbors of a cell in the subgraph $\mathcal{G}$. After L sequential layers, each node i has a feature vector $\mathbf{h}_i^{(L)}$. Then, to obtain a unified representation for the entire subgraph $\mathcal{G}$, we employ an attention aggregation layer, resulting in a graph-level representation $\mathbf{z} := \text{attention-aggregation}((\mathbf{h}_i^{(L)})_i)$. The subgraph representation $\mathbf{z}$ is learned through the self-supervised task described in subsection 5.2.6. Note that, for simplification, the above equations detail the graph encoder with only one head. With multiple attention heads, the GAT computes separate attention coefficients and node features for each head. The outputs from all heads are then concatenated (or averaged) to form the final node representation for each layer, allowing the model to capture diverse aspects of the node's neighborhood.

5.2.6 . Prototypes and swapped assignment task

Since the dataset lacks ground truth, we need to train Novae using an unsupervised approach. Specifically, as we aim to pre-train a foundation model, we will leverage self-

supervised learning [Jaiswal et al., 2021, Rani et al., 2023], which is well-suited for capturing meaningful data representations. Among the different self-supervision frameworks, SwAV [Caron et al., 2020] is a self-supervised learning algorithm that integrates contrastive learning and clustering. The main concept is to learn representations by predicting cluster assignments derived from different augmentations (views) of the same image. In our context, instead of using two views of the same image, we utilize two closely related subgraphs of cells. Specifically, we select a pair (i, j) of cells separated by n_{view} edges, from which we derive the corresponding subgraphs $(\mathcal{G}_i, \mathcal{G}_j)$. When n_{view} is small enough (typically 2 or 3), we expect that the subgraph representations $(\mathbf{z}_i, \mathbf{z}_j)$ will belong to the same spatial domain. To actually assign these representations to spatial domains, we train a set of K vectors $\mathbf{c}_1, \ldots, \mathbf{c}_K \in \mathbb{R}^O$ in the unit sphere (i.e., $\forall k, ||\mathbf{c}_i||_{L^2} = 1$, where $|| \cdot ||_{L^2}$ is the L2-norm). These trainable embeddings are referred to as *prototypes* in the original SwAV [Caron et al., 2020] paper. In our application, they represent high-resolution spatial domain (i.e., in practice, K is large). The representation $\mathbf{z}_i$ of each cell is projected onto the prototypes, over which a softmax function is applied:

$$\mathbf{p}_i = \left(\frac{\exp\left(\frac{1}{\tau} \frac{\mathbf{z}_i^T}{||\mathbf{z}_i||_{L^2}} \mathbf{c}_k\right)}{\sum_{1 \leq k' \leq K} \exp\left(\frac{1}{\tau} \frac{\mathbf{z}_i^T}{||\mathbf{z}_i||_{L^2}} \mathbf{c}_{k'}\right)} \right)_{1 \leq k \leq K} \in \mathbb{R}_+^K,$$

where τ is a temperature parameter that controls the sharpness of the softmax distribution. Intuitively, p_{ik} represents the probability that cell i belongs to prototype k. To prevent the representations from collapsing into a single prototype, we define a "corrected" assignment $\mathbf{q}_i$ that aligns with the distribution $\mathbf{p}_i$ while considering global mini-batch statistics. Essentially, we aim for each mini-batch to represent all prototypes as evenly as possible. This assignment $\mathbf{q}_i$ is derived from the result of an optimal transport (OT) problem over a mini-batch of size B (with each mini-batch being dedicated to one slide). Specifically, given a mini-batch of B representations $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_B) \in \mathbb{R}^{B \times O}$ and the matrix of all prototypes $\mathbf{C} = (\mathbf{c}_1, \ldots, \mathbf{c}_K) \in \mathbb{R}^{K \times O}$, the OT problem is defined as:

$$\mathbf{Q}^* := \text{argmin}_{\mathbf{Q} \in \mathcal{Q}} \Big(\text{Tr}(\mathbf{Q}\mathbf{C}\mathbf{Z}^\top) - \epsilon H(\mathbf{Q}) \Big),$$

where H is the Shannon entropy, ϵ is a regularization hyperparameter, and $\mathcal{Q}$ is the transportation polytope defined by $\mathcal{Q} = \left\{ \mathbf{Q} \in \mathbb{R}_+^{B \times K} \mid \mathbf{Q}\mathbf{1}_K = \frac{1}{B}\mathbf{1}_B, \mathbf{Q}^\top \mathbf{1}_B = \frac{1}{K}\mathbf{1}_K \right\}$. This OT problem ensures a smooth distribution of assignments across the different prototypes, preventing mode collapse. It also helps avoid learning "artificial" prototypes that are specific to certain slides or batch effects. The Sinkhorn-Knopp algorithm is employed to approximate $\mathbf{Q}^*$, as detailed in the supplementary notes. Subsequently, we obtain $\mathbf{q}_i = \mathbf{Q}_i^* \in \mathbb{R}^K$, which represents the "corrected" assignments for cell i. The loss function used in Novae is a cross-entropy applied to the "swapped" prediction problem. Specifically, $\mathbf{q}_i$ serves as the "self-supervised ground truth" for $\mathbf{p}_j$, while $\mathbf{q}_j$ is used for $\mathbf{p}_i$. We

formally define the loss as:

$$\mathcal{L}(\theta, i, j) = -\sum_{k=1}^{K} \big(q_{ik} \log p_{jk} + q_{jk} \log p_{ik}\big),$$

where θ represents all the model parameters, and (i, j) denotes a pair of cells separated by n_{view} edges. This loss function encourages the soft spatial domains of cell i to resemble the soft "pseudo-ground-truth" spatial domains of cell j, and vice versa. The model is trained via backpropagation of this loss using the Adam optimizer. Note that the gradients are detached during the optimal transport part, meaning that the loss is backpropagating via $(\mathbf{p}_i, \mathbf{p}_j)$.

5.2.7 . Pan-tissue prototypes

The loss defined in the previous section is limited when training on multiple tissues, as the optimal transport problem enforces similar distribution assignments to the prototypes across all slides. However, many tissues possess unique, non-overlapping spatial domains, making it biologically unrealistic to train shared prototypes for all tissues. To address this, we introduce a slide-by-prototype matrix of weights, allowing the model to not map a slide to all domains. Specifically, we initialize a queue $\mathbf{W}^{(queue)} \in \mathbb{R}^{S \times size \times K}$, where S is the number of different slides, $size$ is the queue size, and K is the number of prototypes (filled with $1/K$ everywhere). For each slide and mini-batch of size B (the mini-batch contains cells of the same slide), we calculate the maximum assignment probability for each prototype, $(\max_{i=1}^{B} \ p_{ik})_{1 \leq k \leq K} \in \mathbb{R}^K$, and store this in the queue at the corresponding slide's position. Technically, this involves rolling the queue weights for a specific slide s and updating the entry as $\mathbf{W}_{0s}^{(queue)} \leftarrow (\max_{i=1}^{B} \ p_{ik})_{1 \leq k \leq K}$. Afterward, we compute the maximum probabilities over the queue size, i.e. the second dimension, which we denote $\mathbf{W}^* \in \mathbb{R}^{S \times K}$. This slide-by-prototype matrix $\mathbf{W}^*$ is used for each mini-batch to subset the prototypes on which the optimal transport is performed to define the corrected assignments $\mathbf{q}$. For a given slide $s \in \{1, \ldots, S\}$, the codes are computed using the prototypes k for which $\mathbf{W}_{sk}^* > 0.9 * \max_{j=1}^{S}(\mathbf{W}_{jk}^*)$, meaning that the probability of having this prototype is high enough (compared to the slide which most likely have this domain). Yet, in order to not only learn slide-specific prototypes, we ensure that each slide is mapped over at least ρK prototypes, where $\rho \in]0, 1[$ is a hyperparameter. If less than ρK prototypes match the above condition, we choose the $\lfloor \rho K \rfloor$ prototypes with the highest $\mathbf{W}_{sk}^*$ score. Typically, we choose $\rho \gg \frac{1}{S}$ to ensure that prototypes will be shared by multiple slides. Indeed, we expect slides from the same tissue to have shared prototypes (or "tissue-specific" prototypes), while some prototypes should be shared across multiple tissues.

5.2.8 . Assignment to spatial domains

Prototypes can be considered as centroids of elementary spatial domains. Since the desired number of spatial domains may vary, we employ hierarchical clustering on the prototypes. In this setup, the prototypes serve as the leaves of a hierarchical tree, with each level of the tree representing increasingly coarse-grained spatial domains. For a given subgraph representation $\mathbf{z}_i$, we assign it to the closest prototype by selecting the one with the highest dot product, defined as $C_i^{(\text{leaf})} := \text{argmax}_{k=1}^{K} \mathbf{z}_i^T \mathbf{c}_k$. This assignment $C_i^{(\text{leaf})}$ indicates to which tree-leaf the cell i is associated. At a particular level l of the tree, the spatial domain assignment for cell i is determined by $C_i^{(l)} = \text{Map}(l, C_i^{(\text{leaf})})$, where $\text{Map}(l, \cdot)$ is a mapping function that associates the leaf prototype $C_i^{(\text{leaf})}$ with a cluster at level l (the level l is chosen according to the number of desired clusters). This hierarchical approach allows each cell representation to be assigned to spatial domains of varying resolutions efficiently (constant time). In other words, the prototypes serve as elementary spatial domains, used to define spatial domains at the desired resolution.

5.2.9 . Zero-shot and fine-tuning

We train Novae on a large dataset composed of 18 different tissues, as detailed in subsubsection 5.2.11. Therefore, we can save this model on a hub (in particular, Hugging Face Hub), and then anyone can download this model and re-use it on their own dataset without re-training. For that, for a given new dataset, we compute the spatial domain representation $\mathbf{z}$ of all cells in all slides. Afterward, we run a KMeans algorithm from sklearn [Pedregosa et al., 2011] on the representations to define K centroids, which will act as the new prototypes. These prototypes are used for spatial domain assignment as detailed in the previous section. We denote this as zero-shot since Novae is not re-trained. For fine-tuning, we apply the same approach but then re-train the model for a few epochs, potentially one only epoch.

5.2.10 . Batch effect correction

The optimal transport approach ensures the retrieval of relatively similar spatial domains across slides of the same tissue. Therefore, the spatial domain assignments C_i^l are corrected for the different slides, meaning that correcting the actual representations $\mathbf{z}$ is not always necessary. If correcting the representation is actually desired, for each spatial domain we compute the centroid representation of the slide that has the most cells in this spatial domain, which generates a list of centroids $\mathbf{z}_0^{(centroid)}, \dots, \mathbf{z}_L^{(centroid)} \in \mathbb{R}^O$. Then, for each slide and for each spatial domain l, we compute the mean representation $\bar{\mathbf{z}}_l$, and the representation of a cell i in spatial domain l becomes $\mathbf{z}_i^{(corrected)} = \mathbf{z}_i - \bar{\mathbf{z}}_l + \mathbf{z}_l^{(centroid)}$. Essentially, we translate the representations to align the centroids to the reference centroid.

5.2.11 . Implementation details

Downstream tasks examples

For downstream tasks, we either use the spatial representations of Novae (i.e., vectors), or directly the spatial domain assignments per cells (i.e., categories). For instance, the spatially variable genes (SVG) are defined by the `scanpy.tl.rank_genes_groups` function from Scanpy [Wolf et al., 2018] over the categorical spatial domains of Novae, and increasing the resolution of the Novae domains results in more local SVG. Regarding pathways, we use the `scanpy.tl.score_genes` function from Scanpy applied to gene sets from the Gene Set Enrichment Analysis (GSEA) database [Subramanian et al., 2005, Mootha et al., 2003]. This gives a score for every cell and every pathway. Afterward, we average these scores per spatial domain, resulting in a domain-by-pathway heatmap. We plot this heatmap using the `seaborn.clustermap` function from Seaborn [Waskom, 2021], which groups domains with similar patterns of scores across pathways. More specifically, in Figure 5.5, we used the LEE_AGING_CEREBELLUM_UP [Lee et al., 2000] pathway (https://www.gsea-msigdb.org/gsea/msigdb/mouse/geneset/LEE_AGING_CEREBELLUM_UP.html?ex=1). Regarding trajectory inference, we can run any method on the representations of Novae, grouped by categorical spatial domain. In practice, in this article, we used PAGA [Wolf et al., 2019] and more specifically the `scanpy.tl.paga` implementation from Scanpy.

Metrics for model comparison

The F1-score of inter-domain edges (denoted FIDE score) is used to quantify the spatial domain continuity. More specifically, let $\mathbf{C} = (C_i)_{1 \leq i \leq N}$ be the categorical spatial domain predictions for N cells of a slide. The FIDE score is defined as $\text{FIDE}(\mathbf{C}, \mathbf{A}) = \text{F1-score}((C_i, C_j)_{i,j \text{ s.t. } A_{ij}>0})$, where A_{ij} is positive when the cells i and j are graph neighbors. Intuitively, for an edge $i \leftrightarrow j$, the edge is an inter-domain edge if $C_i \neq C_j$. A high number of inter-domain edges shows a low domain continuity, therefore quantified by a low FIDE score. On multiple slides, we average the per-slide FIDE scores. The Jensen-Shannon divergence (JSD) is used to quantify the homogeneity of domains across multiple slides. For each slide $s \in [\![1, S]\!]$, we compute $\pi_s \in [0, 1]^D$, the proportion of each domain among the slide, where D is the total number of spatial domains. Then, the JSD metric is defined as $\text{JSD}(\pi_{\mathbf{1}}, \dots, \pi_{\mathbf{S}}) = H(\frac{1}{S}\sum_{s=1}^{S} \pi_s) - \frac{1}{N}\sum_{s=1}^{S} H(\pi_s)$, where H is the Shannon entropy. The Adjusted Rand Index [Hubert and Arabie, 1985] (ARI) is defined as the percentage of concordance between two clusterings. Mathematically, $ARI = \frac{TP+TN}{TP+FP+FN+TN}$, where TP is the number of true positives, TN the number of true negatives, FP the number of false positives, and FN the number of false negatives.

Datasets used

In total, 78 public spatial transcriptomics slides were collected from single-cell resolution technologies (Xenium, MERSCOPE, CosMX). The datasets are available on the vendor websites, i.e., respectively 10X Genomics, Vizgen and Nanostring. This dataset covers 19 tissues (human bone marrow, human brain, human breast, human colon, human kid-

ney, human liver, human lung, human lymph node, human ovarian, human pancreas, human prostate, human skin, human tonsil, human uterine, mouse femur, mouse brain, mouse colon, and whole mouse) and a total of 28,909,516 cells. It is comprised of 853,622 cells from the CosMX technology, 10,196,835 cells from the MERSCOPE technology, and 17,859,059 cells from the Xenium technology. The CosMX covers 2 different tissues, 8 for the MERSCOPE, and 17 for the Xenium technology. In total, 8 tissues contain slides from multiple technologies (breast, colon, liver, lung, ovarian, pancreas, prostate, and skin). For the benchmark, we selected five colon slides from the Xenium technology, divided into three distinct gene panels. The first panel includes 325 genes and comprises 270,984 cells, the second panel contains 422 genes and 924,597 cells, and the third panel consists of 480 genes, with a total of 388,175 cells. The slides can be found in the Novae dataset (see data availability statement), and their name is listed in Supplementary Table 1. We also selected two breast slides from different platforms: Xenium and Merscope, containing a total of 1,290,084 cells. Again, their name is listed in Supplementary Table 1. Similarly, we listed in Supplementary Table 1 the name of the lymph node and mouse brain slides used in Figure 5.5. Regarding the mouse brain slides, note that the "Alzheimer-like pathology" concerns the CRND8 APP-overexpressing (TgCRND8) transgenic mouses, as detailed in https://www.10xgenomics.com/datasets/xenium-in-situ-analysis-of-alzheimers-disease-mouse-model-brain-coronal-sections-from-one-hemisphere-over-a-time-course-1-standard. Additionally, we created a synthetic dataset in order to compare the different models to a ground-truth. More specifically, 7 domains were generated in a circular manner, for 5 different slides with 100 genes. For each domain of each slide, the gene expression was sampled from an exponential law whose parameter is the sum of a slide-specific parameter and a domain-specific marker. For more details, see the supplementals.

Spatial-domains assignment benchmark

In our benchmark study, we evaluated the performance of our method, Novae, against four state-of-the-art methods: SpaceFlow [Ren et al., 2022], GraphST [Long et al., 2023], STAGATE [Dong and Zhang, 2022], and SEDR [Xu et al., 2024], using three spatial transcriptomics datasets: the colon dataset, the breast dataset, and a synthetic dataset, as described in subsubsection 5.2.11. Regarding the colon dataset, each method was applied independently to the different gene panels (except Novae, which works across different gene panels). For the breast dataset, we focused on the intersection of the two slides, which included 185 common genes, ensuring that all methods were applied consistently to this shared gene set. Again, this excludes Novae, which can work on both gene panels at the same time. After training the different models on each dataset, we applied Harmony [Korsunsky et al., 2019] to address potential batch effects introduced by the use of different slides or experimental conditions, aligning the spatial transcriptomic data across slides. Following Harmony, we used the mclust [Scrucca et al., 2016] algorithm for cluster-

ing the representation, each cluster being considered as a spatial domain. Note that clustering was performed on the concatenated data to get consistent clusters across slides. Again, the usage of Harmony and mclust does not concern Novae, which includes both operations natively, as described in subsection 5.2.10 and subsection 5.2.8.

Other implementation and training details

The input of Novae is composed of one or many AnnData [Virshup et al., 2021] objects from the anndata [Virshup et al., 2021] data structure of the scverse [Virshup et al., 2023] community. They contain cell-by-gene tables, which can be typically obtained via (i) reading the proprietary data, e.g., using the readers from SpatialData [Marconato et al., 2024], or (ii) performing a new segmentation, e.g., using CellPose [Stringer et al., 2021] or Baysor [Petukhov et al., 2022]. When constructing the Delaunay graph, edges longer than 100 microns were dropped, which ensures long-distance cells are not considered neighbors. We implemented Novae using Python and the deep learning framework Pytorch [Paszke et al., 2019] as well as Pytorch Geometric [Fey and Lenssen, 2019]. Each mini-batch contains B subgraphs such that all these subgraphs come from the same slide. For RAM efficiency, the subgraphs are lazy-loaded when sampling a mini-batch. The slide queue $\mathbf{W}^{(queue)}$ is not used during the first epochs to ensure it is filled, and the prototypes are also frozen during these epochs. The model training was monitored with Weight & Biases. The hyperparameters were fined-tuned with the sweep option from Weight & Biases on a heuristic, the product of the FIDE score and the Shannon entropy of the spatial domains distribution. Novae was trained on a Nvidia HGX A100 GPU for 24 hours.

5.3 . Results

5.3.1 . Pan-tissue spatial domains

As outlined in the previous section, Novae can analyze multiple panels and tissues, with the ability to identify both shared and tissue-specific spatial domains. While we anticipate some overlap in domains across tissues, we also expect certain domains to be specific to particular conditions or diseases. In Figure 5.2a/c, we present the spatial domains identified across various slides for both human (Figure 5.2a) and mouse (Figure 5.2c) tissues. Notably, the lymph node and tonsil exhibited similar spatial domain distributions, with some domains also observed in other tissues such as breast and lung. Also, the heatmap dendrogram regroups slides with similar patterns of domains, and is not a perfect tissue match. Indeed, the spatial domains include also context related to diseases or certain mechanisms, which can be retreived for multiple different tissues. For instance, a lung tumor and a breast tumor may share some cancer-related domains that cannot be found in a healthy breast tissue [André et al., 2024]. The UMAP [McInnes et al., 2018] representation of spatial context is shown in Figure 5.2b/d, where each dot corresponds to a cell's spatial context, clustered into distinct spatial domains. Regarding

the mouse study, we considered brain slides, as well as colon, femur, and also a whole mouse sample, as detailed in subsubsection 5.2.11. More specifically, the mouse brain slides showed strong inter-slide similarity, as many domains were consistently identified across all slides (see Figure 5.2c). Additionally, we observed that many domains, including "brain-specific" and "colon-specific" domains, were retrieved in the whole mouse sample. Examples of spatial domains for the whole mouse sample are shown in Figure 5.2e. For instance, we find the areas characterizing the bones and costal cartilage (D997), the lung lobes (D983), the liver (D973), and the intestine (D975) in the abdominal cavity [Fonseca et al., 2009], as highlighted in Figure 5.2f. In summary, while expecting results were shown in Figure 5.2, it shows the capability of Novae to train across multiple tissues, while avoiding over-correction (i.e., the identification of all domains in all slides) or under-correction (i.e., all spatial domains being specific to one slide).

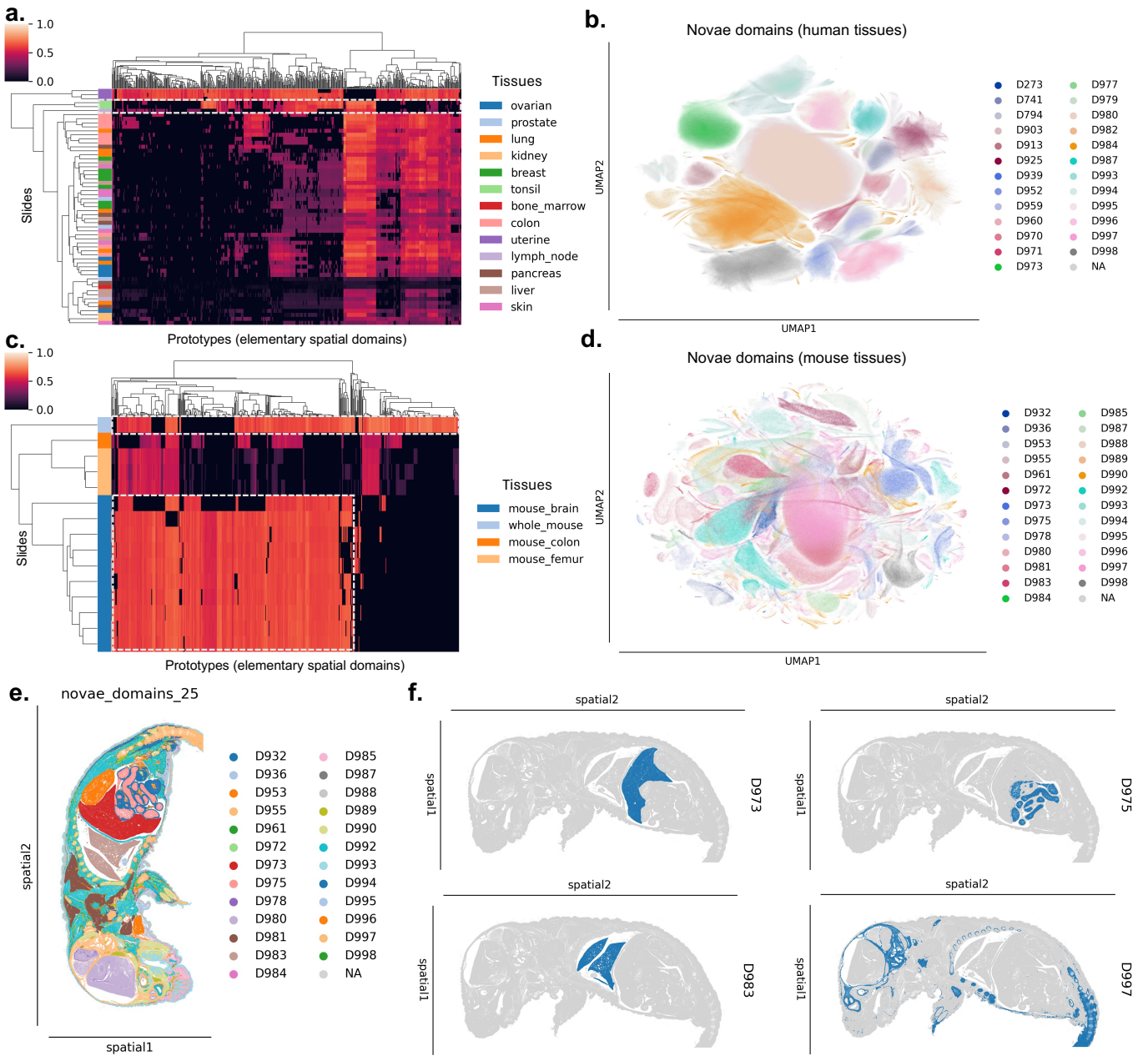

Figure 5.2: **Spatial domains across tissues and species. a.** Heatmap showing the spatial domain weights for each slide, organized by tissue type. A weight of 0 (black) indicates the absence of the domain in that slide, while higher weights represent increased confidence in the presence of the domain. **b.** UMAP visualization of spatial domain representations in human tissue slides. Each dot corresponds to the spatial context representation of a cell. The panels **c.** and **d.** show similar figures to (a) and (b), but for the mouse samples. **e.** Spatial domains identified for the whole mouse sample. **f.** Zoom-in on four domains in the whole mouse: liver (D973), intestine (D975) within the abdominal cavity, lung lobes (D983), bones and costal cartilage (D997).

5.3.2 . High integration and continuity of the spatial domains

In this study, we compared Novae in both zero-shot and fine-tuning modes to four state-of-the-art methods: SpaceFlow [Ren et al., 2022], GraphST [Long et al., 2023], SEDR [Xu et al., 2024], and STAGATE [Dong and Zhang, 2022]. We evaluated the performance of these methods across three distinct test cases. In the first test case, we used the breast dataset, composed of two slides with different gene panels. As mentioned earlier, Novae can be trained across multiple gene panels; hence, a single Novae model was trained for both panels. In contrast, the other methods were trained on the intersection of 185 common genes across the two panels (as illustrated in Figure 5.3a), followed by batch effect correction using Harmony [Korsunsky et al., 2019] and clustering with mclust [Scrucca et al., 2016] (using 7, 10, and 15 clusters). To evaluate model performance, we used the FIDE score and Jensen-Shannon Divergence (JSD) score to assess spatial domain continuity and cross-slide homogeneity, respectively (see subsubsection 5.2.11 for more details). Figure 5.3b presents the results of this benchmark, highlighting a significant improvement in performance by Novae, even in the zero-shot case (i.e., using a pre-trained model directly).

The second test case involved the colon dataset, where Novae was again trained across all slides. For the other methods, due to the limited intersection of genes between panels, we employed a different approach: a separate model was trained for each slide (as illustrated in Figure 5.3c). The results from each model were then concatenated, followed by batch-effect correction and clustering. Figure 5.3d presents the result metrics for all methods on the colon dataset, again showing a superior performance by Novae in both zero-shot and fine-tuning modes.

For visual comparison, Figure 5.4 shows the spatial domain assignments for the breast samples (see supplementary materials for colon samples). Notably, Figure 5.4 shows that Novae (i) better identifies cross-slide domains (see the third figure column) and (ii) better aligns the batches/slides in the latent space (see the fourth figure column). Regarding the biological interpretation of the results from Novae, domains D498 and D499 were identified as stromal regions characterized by the expression of *FN1* and *COL1A1* (see supplementals), which are present in both samples. Domain D504 was composed of glandular cells, marked by the expression of *TAPBP*, *PGR*, and *CDH1*. In the Xenium sample, an expansion of domain D503 was observed, potentially resulting from clonal expansion of cancer cells. This expanded D503 became embedded within the stromal areas of D498 and D499, while excluding the immune cell-rich region D485, which is known for high levels of *PTPRC*, *CD52*, and *CD3E*. This exclusion pattern, as depicted in the PAGA graphs in the supplementals, is characteristic of tumors with an immune-excluded phenotype.

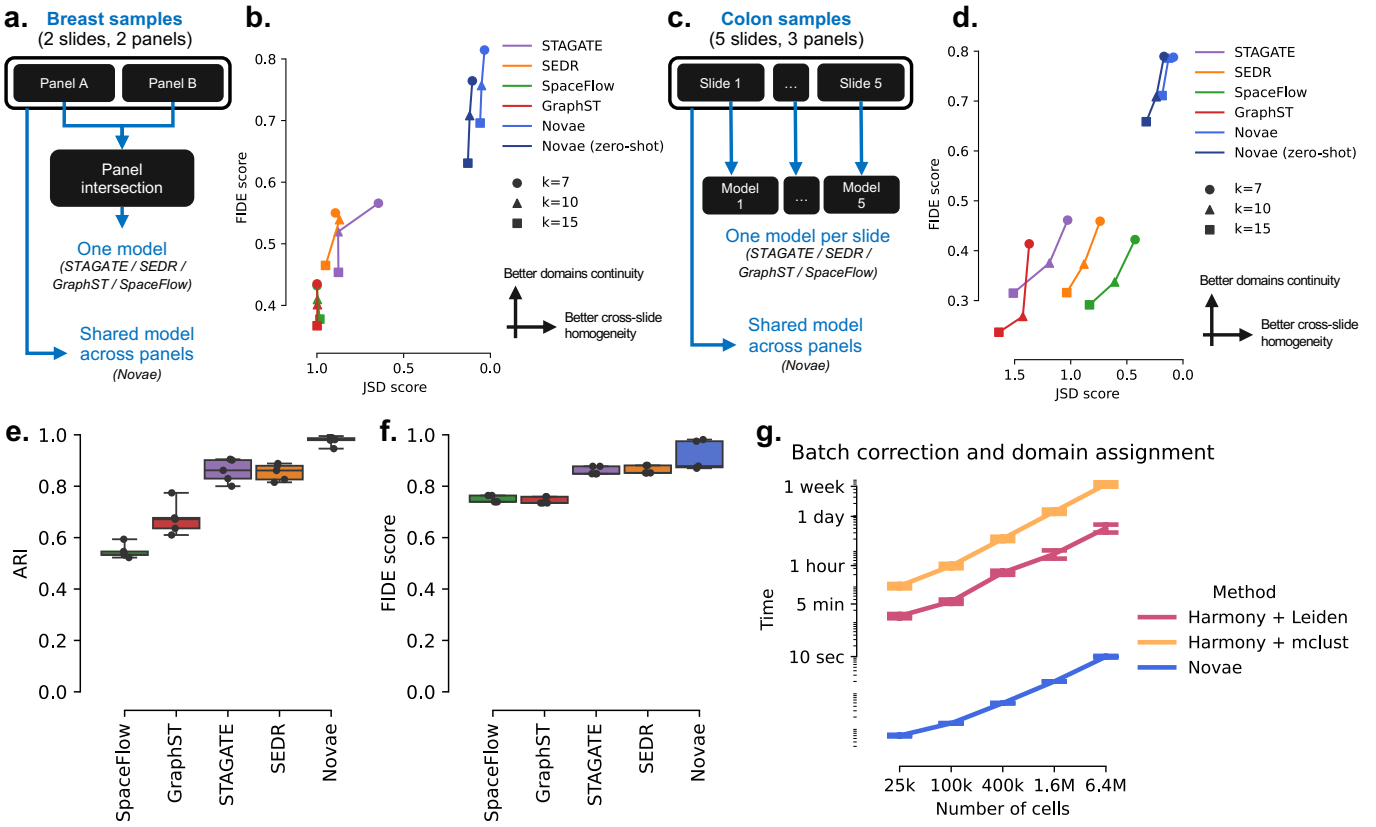

Figure 5.3: ***Quantitative benchmark of the spatial domain assignment.*** ***a.*** *Schematic representation of the benchmarking approach for the breast dataset. The same Novae model can be trained on both panels, while other models were run using genes shared between the panels.* ***b.*** *FIDE and JSD scores of different methods for three domain counts (7, 10, and 15) on breast samples. The FIDE score evaluates domain continuity, while the JSD score assesses cross-slide homogeneity.* ***c.*** *Schematic representation of the benchmarking approach for the colon dataset. The same Novae model can be trained on all panels, whereas other models were run in a one-per-slide manner.* ***d.*** *FIDE and JSD scores of different methods for three domain counts (7, 10, and 15) on colon samples.* ***e.*** *ARI comparison on the synthetic dataset.* ***f.*** *FIDE score comparison on the synthetic dataset.* ***g.*** *Runtime comparison of post-inference tasks (batch-effect correction and domain assignment) across different dataset sizes.*

The third comparison was conducted on a synthetic dataset consisting of 5 slides and 7 spatial domains (see subsubsection 5.2.11 for more details). Unlike the previous cases, this dataset used the same gene panel across all slides, allowing the other methods to run without requiring gene panel intersection. In this case, we did not use Novae in zero-shot mode, as the gene expression was synthetic, meaning the generated cell types may not correspond to real data. The models were evaluated using the Adjusted Rand Index (ARI) for clustering accuracy against the known ground truth, as well as the FIDE score. Figure 5.3e shows the ARI of the different methods across 5 seeds, while Figure 5.3f presents the FIDE score across 5 seeds. Again, Novae outperformed the other methods, demonstrating higher ARI and FIDE scores with notably low standard deviation in ARI (Figure 5.3e).

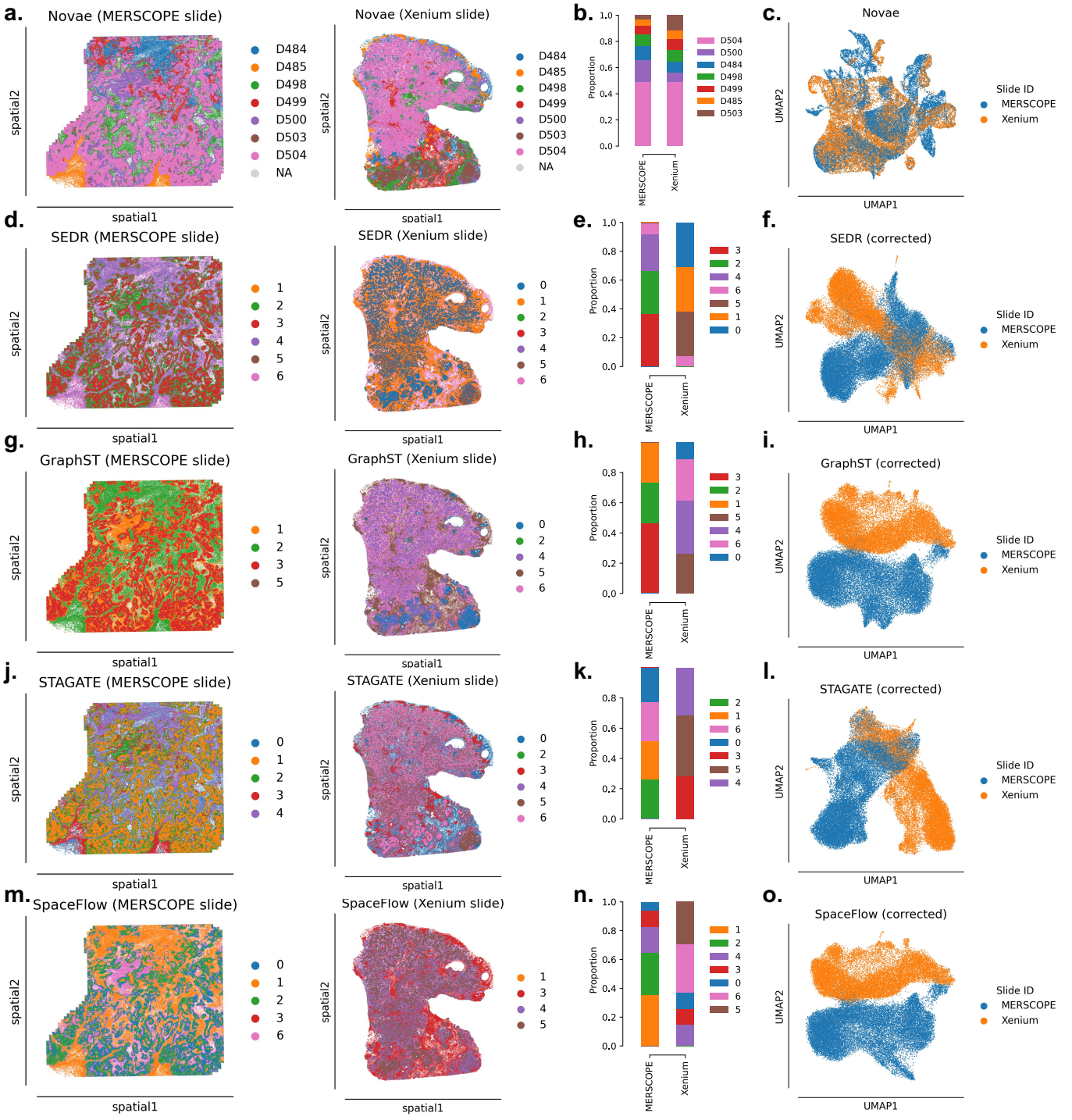

Figure 5.4: **Visualization of the spatial domains for the breast dataset.** Spatial domains results across two slides from the breast dataset (Merscope and Xenium slides, see more details in subsubsection 5.2.11) using five different methods: Novae, SEDR, GraphST, STAGATE, and SpaceFlow. **a.** Spatial domains assigned by Novae on the two breast slides. **b.** Proportions of each Novae domain for each slide. **c.** UMAP of spatial representations, colored by slide ID. Each dot is the UMAP spatial representation of one cell. The next four rows, that is **d.** to **o.**, show the same figures as the first row but for the four other methods (SEDR, GraphST, STAGATE, and SpaceFlow, respectively).

5.3.3 . Time and memory efficiency

After running inference, i.e., computing the cell's spatial representations, Novae can perform the attribution of the niches and correct batch effect in a short time. Indeed, the attribution of spatial domains is a mapping between the prototypes and the desired resolution (see subsection 5.2.8 for more details), which is performed in a low (constant) time. Regarding batch-effect correction, since the categorical domain assignments are already corrected through Novae, we can use the assignments to align the spatial representations (as detailed in subsection 5.2.10). This vectorial operation is also fast and in linear time. Contrastively, for the two latter operations, the other state-of-the-art models depend on external tools: usually (i) Harmony [Korsunsky et al., 2019] for batch-effect correction and (ii) Leiden [Traag et al., 2019] or mclust [Scrucca et al., 2016] to assign spatial domains to the representations. As shown by Figure 5.3g, this can be slow, especially on large datasets of millions of cells. Indeed, this can take up to several days on 6 millions cells, while Novae can perform these two operations in several seconds. Furthermore, during experimentation, it is common to try multiple resolutions of spatial domains, hence requiring clustering to be run multiple times. Novae can perform this very rapidly, thus easing the analysis of different resolutions, while the other methods require running a time-consuming clustering again. Also, regarding random-access-memory (RAM) usage, Novae supports lazy loading. That is, instead of storing the full graph dataset in memory, each subgraph is created on the fly before running through the model. This prevents a significant amount of RAM from being dedicated to loading the dataset. This allowed us to train Novae on a dataset composed of nearly 30 million cells using a GPU with 40GB of RAM (see subsubsection 5.2.11 for more details). In addition, Novae runs on local subgraphs instead of running on a full slide, which is original for two reasons. First, the subgraphs are smaller and therefore require less memory. Secondly, it is possible to use a larger number of layers in the neural network without aggregating information from further environments. For instance, having a graph neural network of 16 layers operating on the full slide would lead to mixed information between two cells at a distance of 300 microns, which is usually not intended. Instead, using local subgraphs ensures only aggregating information from local microenvironments.

5.3.4 . A multitude of downstream tasks

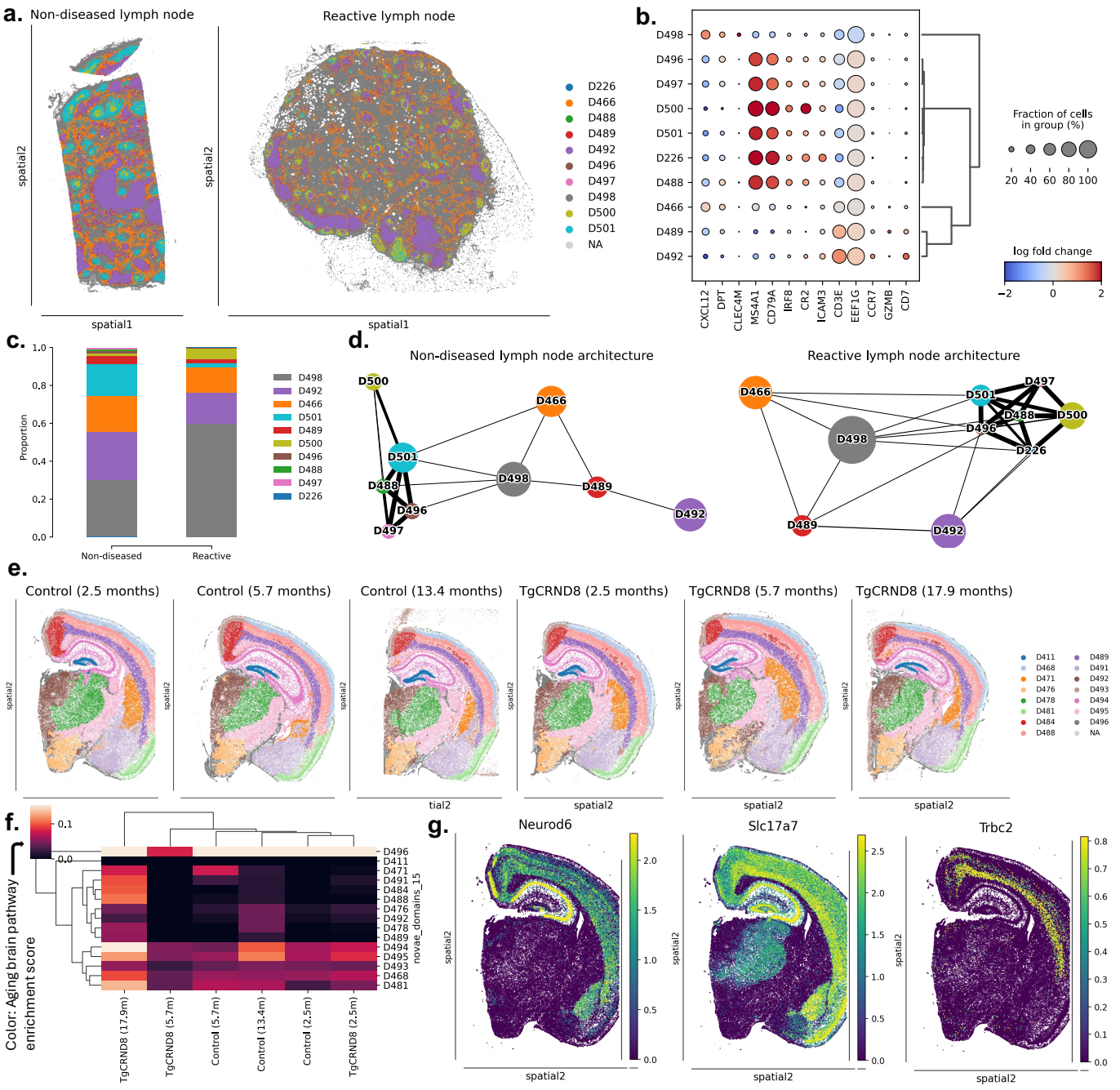

Figure 5.5: **Downstream tasks examples on human lymph node and mouse brain slides. a.** Spatial domains of the non-diseased (left) and reactive (right) lymph nodes. **b.** DEGs between the spatial domains of the reactive lymph node slide. **c.** Proportions of spatial domains for each lymph node slide. **d.** PAGA graphs comparing the domain organization of non-diseased (left) and reactive (right) lymph nodes. **e.** Spatial domains of mouse brain across different time points, comparing control conditions and Alzheimer-like pathology (*TgCRND8* samples). **f.** Heatmap of brain aging pathway activation across different domains and time points in the mouse brain slides. **f.** Expression patterns of three spatially variable genes on the 2.5-month control mouse brain slide.

Novae's spatial representations enable the performance of multiple downstream tasks. To illustrate this, we applied three tasks across two datasets. The first dataset consists of a non-diseased lymph node and a reactive lymph node (shown in panels Figure 5.5a-d), while the second dataset comprises six mouse brain slides collected at different time points, with half exhibiting Alzheimer's-like pathology (further details in subsubsection 5.2.11). For the lymph node dataset, spatial domains are displayed in Figure 5.5a, with Figure 5.5b/c providing details of gene expression and domain proportions to help the characterisation of these spatial domains. To compare the spatial organization (or "slide architecture"), users can apply trajectory inference methods to the spatial representations generated by Novae (more details in subsubsection 5.2.11). As shown in Figure 5.5d, we observed changes in the spatial organization of these domains. Notably, the D500 domain in the germinal center, enriched with (*CD79+ CR2+*) mature B cells, was previously connected only to the D501 domain before clonal expansion. Additionally, Figure 5.5c reveals an inversion in the proportions of the D500 and D501 domains, also visible in Figure 5.5a.

Regarding the mouse brain dataset, the spatial domains identified by Novae are presented in Figure 5.5e. Although no significant changes in brain architecture were observed (see supplementals for details), we do observe differences in brain aging pathway [Lee et al., 2000] enrichment that we computed for each spatial domain. As shown in Figure 5.5f, the 17.9-month-old *TgCRND8* mouse, which exhibits Alzheimer's-like pathology, shows higher brain aging, particularly in specific spatial domains such as D494 and D481. For example, the D494 domain has a high expression of the *Neurod6* gene, a key gene associated with brain aging [Lee et al., 2000]. This analysis highlights how identifying spatial domains enhances the understanding of pathway activation, demonstrating the utility of spatial domain identification in pathway analysis. In addition, differential gene expression analysis can be performed on cells grouped by spatial domains, allowing for the identification of spatially variable genes (SVGs). SVGs are genes whose expression varies significantly across spatial domains. In Figure 5.5g, we demonstrate this for the 2.5-month control mouse, showing the three most spatially variable genes identified by Novae. Therefore, this helps better identify genes that have different activations depending on the spatial domain.

5.4 . Discussion

The field of spatial transcriptomics is advancing rapidly, leading to the generation of increasingly larger datasets [Velten and Stegle, 2023]. While this growth holds the promise of uncovering new insights that were difficult to achieve with scRNAseq [Wu et al., 2022a], it also presents significant challenges in performing comparative spatial analyses across multiple slides. Current algorithms [Dong and Zhang, 2022, Long et al., 2023, Xu et al., 2024, Ren et al., 2022] are often designed for single or consecutive slide analyses and typi-

cally rely on external batch-effect correction methods, which may not be sufficient for the complexities of multi-slide spatial datasets. Novae addresses this gap by offering a more flexible solution that can operate across various tissues, conditions, technologies, and gene panels. Compared to related models, we have shown that Novae can better identify cross-slide domains while not over-correcting the spatial domain assignments. Indeed, the flexibility introduced in the prototypes allows for certain domains to be tissue-specific or disease-specific. In addition, Novae is less dependent on external tools, as it performs domain assignment and batch-effect correction inside the same model. This has many advantages, such as (i) a better batch-effect correction due to the spatial awareness of the model and (ii) improved speed performances. Also, as a shared open-source model, Novae can be easily loaded and used for user-specific tasks without the need for hyper-parameter optimization or large re-training.

Our study demonstrates Novae's potential, particularly when applied to large cohorts, to identify spatial domains enriched or specific to certain conditions or diseases. Using spatial trajectory analysis, we demonstrated how tissue organization and structure can be characterized, as shown by the comparison of a reactive lymph node with a non-diseased one. Lymph nodes are crucial for immune surveillance across different organs, with their structure enabling a targeted immune response during infections. In our analysis, we observed spatial reorganization in the reactive lymph node, notably the expansion of the D498 domain, which is enriched in the chemoattractant CXCL12, a key factor in recruiting inflammatory cells. Concerning the second use case example, we identified an enriched aging-related pathway in specific mouse brain regions linked to Alzheimer's-like pathology at defined time points, as well as brain-specific spatially variable genes. Overall, this study has showcased Novae's versatility in handling multiple downstream tasks, such as spatial domain identification, slide architecture, and spatially variable gene analysis, all within a single framework.

Methodologically, Novae is built upon the SwAV [Caron et al., 2020] framework, a self-supervision technique initially developed for computer vision. In this study, SwAV was adapted for graph learning, and many components were updated to better suit our biological problem, e.g., using biologically relevant augmentations. Furthermore, compared to other self-supervised methods, SwAV has two major properties that make it suitable for our purpose. First, SwAV's inclusion of optimal transport [Peyré and Cuturi, 2020] (OT) within the model is particularly noteworthy, as OT has proven effective in addressing batch effects natively, offering a significant advantage over traditional methods that require external corrections. Secondly, SwAV's approach to learning embeddings in the latent space (that is, prototypes) allows for efficient assignment of spatial domains, with prototypes serving as centroids for elementary classes. Since we typically have less than

a thousand prototypes, hierarchical clustering on prototypes is computationally efficient and facilitates the definition of nested spatial domains, providing an advantage over more conventional clustering techniques on the latent space.

Integrating additional modalities into Novae, for instance, protein data, could lead to the development of a multi-omics spatial model. This could be achieved by incorporating protein embeddings into the model, allowing it to analyze both transcriptomic and proteomic data concurrently. Another area for improvement lies in the segmentation process of our training dataset, which currently relies on proprietary methods that may benefit from refinement. Given the critical role of segmentation in determining the quality of spatial data [Blampey et al., 2024b], the application of newer segmentation tools to the raw data could enhance overall dataset quality. Additionally, using multiple segmentation methods could serve as a form of data augmentation, potentially improving the model's robustness. Addressing these areas will further enhance Novae's utility and effectiveness in spatial transcriptomics research.

Applications for research and clinical projects in oncology

Contents

Abstract

Although the developed tools are designed for broad usage, they were originally created to address specific needs in the field of oncology. My involvement in several projects at the *Gustave Roussy Cancer Center* provided a rich and diverse context for their application. Each project had distinct objectives and utilized various types of data, ranging from clinical data to spatial data. This diversity underscores the utility and versatility of the developed methods, demonstrating their applicability across a wide range of scenarios in precision medicine, also helping me to have a better overview of precision medicine in oncology. Therefore, in this chapter, I provide various applicative projects. Due to the results being published in papers from which I'm not a first author, I remain short on the details of the results, but focus on the methods instead.

6.1 . Cytometry related projects

In this section, I present projects from Gustave Roussy for which I used Scyan to analyze the corresponding Cytometry datasets. Apart from the projects listed below, I also helped people using Scyan for their own project at Gustave Roussy, so that they become more autonomous in their analysis.

6.1.1 . Pre-operational Durvalumab in triple-negative breast cancer

The Pop-Durva study explores the use of Durvalumab, an immune checkpoint inhibitor (ICI), in treating early-stage triple-negative breast cancer (TNBC). The trial aims to assess the pathological complete response (pCR) after two doses of Durvalumab monotherapy. The study will involve 195 patients and seeks to understand the effectiveness of this treatment and identify biomarkers that predict immune response. Secondary objectives include evaluating the overall response rate and safety, while exploratory goals focus on immune dynamics, genetic factors, and the role of the microbiome in treatment response. The first type of data collected was spectral cytometry data. At the beginning of the project, cell types were manually gated, but this process was time-consuming and error-prone. Scyan [Blampey et al., 2023] was developed to overcome the limitations of gating for this specific study, and was then made more general to be applied to other projects and published.

6.1.2 . Biomarkers associated with the diagnosis of tobacco-associated cancers

PREVALUNG is a prospective translational research trial, involving 508 smokers with cardiovascular disease (CVD), to identify biomarkers linked to the diagnosis of tobacco-associated cancers. Using omics technologies, analytes related to inflammation, immu-

nity, metabolism, microbiota, plasma proteins, and clonal hematopoiesis were measured. Concerning cytometry, cell-type annotations were performed using Scyan. It identified 26 populations based on 20 markers among the whole panel. Afterwards, for each patient, we extracted (i) the cell count ratio of all immune cell populations and (ii) the mean fluorescence intensity (MFI) of all markers for each population. It resulted in a total of 1,546 biomarkers per patient that we considered for further statistical analysis.

6.1.3 . Biomarker screening approach for NSCLC patients treated with anti-PD1

With the advancements in immunotherapy, it is crucial to identify biomarker(s) that can indicate the potential response to treatment. The Legendscreen project aims to develop a biomarker screening approach for non-small cell lung cancer (NSCLC) patients treated with anti-PD1 (pembrolizumab). Blood samples were collected from these patients before treatment, and a protein screen assay (Legendscreen) was performed. Using tools like Scyan and InfinityFlow, we can analyse up to 30 patients simultaneously and screen 396 proteins. For that, patients cells are barcoded, pooled, stained and processed together. Then, the data is debarcoded with Scyan, allowing to recover the 30 patients with minimal batch effect. Scyan was also used for the cell-type annotation.

6.2 . Spatial-omics related projects

This section introduces projects from the *Gustave Roussy Cancer Center* for which I used Sopa to analyze spatial-omics datasets.

6.2.1 . A standardized spatial-omics infrastructure

Many machines are installed at the *Gustave Roussy Cancer Center*, including two MERSCOPE, a Xenium, a MACSima, and a Hyperion. Therefore, it is important to have a standardized infrastructure and pipeline to analyze the data produced by these machines. In this context, Sopa is particularly useful, as it can run on all these machines. We connected the machines to a cold storage from the *Gustave Roussy Cancer Center*. Then, I developed a command prompt so that everyone could easily run Sopa on their data. More particularly, it displays a list of all the slides available in the cold storage, and the user can select the slide(s) he wants to analyze. Then, the user can choose the type of analysis he wants to perform (among existing configuration files). It will copy the data from the cold storage to the hot storage, execute the pipeline, and move the results back to the cold storage. Overall, this command prompt is a user-friendly interface that allows everyone to run Sopa on their data without any coding knowledge, while also ensuring standardized data analysis across different technologies.

6.2.2 . Biomarkers of residual cancer burden in triple-negative breast cancer

When a patient completes treatment, it is not a guarantee that they are free from the cancer they were treated for. Despite successful treatment outcomes, approximately half of the patients with triple-negative breast cancer (TNBC) experience a relapse. The risk of relapse can be associated with the residual cancer burden (RCB). Specifically, RCB I is associated with an 89% survival rate, while RCB II is associated with a 62% survival rate. Currently, the reasons why certain patients relapse remain unclear. By employing spatial transcriptomics, we aim to better understand the intricacies between the immune system and the tumour environment prior to residual surgery after successful immunotherapy combined with chemotherapy. A dataset of 50 MERSCOPE slides was collected and analysed with Sopa.

6.2.3 . Multinucleated giant cells microenvironment in HNSCC [Gessain et al., 2024]

Head and neck squamous cell carcinoma (HNSCC) patients often face poor outcomes due to ineffective risk management and treatment approaches, and integrating new prognostic biomarkers into clinical settings remains difficult. In this study, multinucleated giant cells (MGCs), a type of macrophage, are identified in HNSCC tumors, and their presence is correlating with better prognosis in both treatment-naive and preoperative-chemotherapy patients. In parallel to other analyses, Sopa was run on 8 slides of CosMX data (4 slides with transcriptomics information, and 4 slides with proteomics information). Then, the MGC(s) were manually selected by a pathologist via the lasso tool from the Xenium Explorer. Afterwards, the resulting polygons were used to update the shapes layer of the SpatialData object, and update the cell-by-gene or cell-by-protein tables. Finally, post processing analysis was performed to locate these MGCs in the tissue, and analyze their microenvironment.

6.3 . Clinical-oriented applications

This section details projects that are more clinically oriented, and that do not use one of the methods developed during this PhD. The diversity of these projects helped me to understand other crucial aspects of medicine. For instance, one project related to survival analysis, and another one concerns the prediction of the risk of toxicity, an important topic related to immunotherapy.

6.3.1 . Predictive risk factors of cytokine-released syndromes induced by T-Cell engagers

T-cell engagers (TCE) represent a promising new approach to cancer treatment, especially in cases where chemotherapy has been unsuccessful. However, a significant drawback of these therapies is the potential for cytokine release syndrome (CRS), a serious and potentially fatal side effect that can limit their application. The factors that contribute

to the development of CRS are still not well understood. The hypothesis of this study is that there may be identifiable risk factors that predispose patients to severe CRS when treated with T-cell engagers. In total, 37 patients with solid tumors, and 16 patients with lymphoma, received infusions of TCE. For these patients, 17 clinical variables were collected before treatment, as well as the grade of the CRS. The goal of the study was to identify risk factors that could predict the CRS grade (superior or inferior to grade 3). Due to the small amount of samples, we chose to combine multiple simple models and aggregate their predictions, i.e. the combined model prediction is the most represented prediction among the sub-models. In practice, the model was composed of a Random Forest, a k-Neirest-Neighbors, a Gradient Boosting, and a Logistic regression. The model input is composed of 5 features that were selected with recursive feature elimination. It consisted in removing the least important feature (in terms of AUC) to the model input recursively until the model performance decreases. Note that at each variable removal, we tested our model with Leave-One-Out Cross-Validation to ensure we evaluated our model on unseen samples at each time. To better interpret the model, we computed the SHAP [Lundberg and Lee, 2017] values of the main features of interest.

6.3.2 . Antecedent viral immunization and efficacy of immune checkpoint blockade

Immune checkpoint blockers (ICBs) have transformed the treatment of advanced non-small cell lung cancer (NSCLC), yet only a subset of patients respond, and predicting clinical benefit remains challenging. This study investigates whether antibody responses to past viral infections can help predict NSCLC patient responses to ICBs. For each patient in the dataset, peptide log fold changes are measured using Virscan technology (CDI Labs, US). Peptides whose log fold change is positive in less that 5% of the patients are removed, and a log1p transformation is applied on the other peptides log fold change. Leiden clustering [Traag et al., 2019] is run on peptides to create 90 clusters of highly-correlated peptides. For each of these clusters, the first principle component is extracted in order to get one feature per peptide cluster. These features were used to train a Cox model from sksurv (l1 ratio = 0.9) to predict the overall survival. Three different models were trained, based on three groups of patients depending on their treatment (chemotherapy, immunotherapy, or a combination of the two). The best alpha parameter of the Cox model is computed with 10-fold cross-validation.

6.3.3 . Vwf-positive hematopoietic stem cells in knock-in mice

This study explores two common mutations in the calreticulin (CALR) gene, a 52 base-pair deletion (del52) and a 5 base-pair insertion (ins5), both linked to essential thrombocythemia and myelofibrosis. In mouse models, CALRdel52 shows a more severe phenotype compared to CALRins5, including enhanced hematopoietic stem cell (HSC) expansion and progression to fibrosis. For this specific study, I analyzed three mouse samples with

scRNAseq data, using the scanpy [Wolf et al., 2018] package, as well as other packages from the scverse ecosystem, such as SCVI [Lopez et al., 2018]. In particular, I focused the analysis on the Vwf-positive hematopoietic stem cells. This project started early in my PhD, and was a great opportunity to learn about scRNAseq data analysis. At this time, I was not working on spatial transcriptomics data yet, so it was a good transition between my previous work on cytometry data and my future work on spatial data.

Conclusions & Perspectives

Contents

Abstract

In this chapter, we summarize the main contributions of this manuscript, both in terms of methods and applications. We then discuss the usage and impact of the developed methods, by analyzing how much and where the methods are used. Finally, we discuss the perspectives for future work, such as the development of foundation models.

7.1 . Synthesis

I focused on three main methodological projects, each advancing the analysis of single-cell resolution omics data. These projects — Scyan, Sopa, and Novae — tackle different types of omics data, with distinct objectives. Scyan addresses challenges in cytometry, a widely used clinical technique, while Sopa and Novae focus on spatial omics, a cutting-edge technology that offers deeper biological insights but comes at a higher cost. Consequently, Scyan is designed for clinical applications, whereas Sopa and Novae are more aligned with biological research and discovery. Also, Sopa and Novae seamlessly interact with each other. Indeed, Sopa enhances the preprocessing of spatial data to improve quality, serving as a foundation for Novae, which then extracts patterns and insights from this processed data. All three projects are open-source and built on top of the scverse core tools, a widely-used ecosystem for single-cell data analysis. By contributing to core data structures (like SpatialData) and adding new tools to the ecosystem, I have ensured that these tools are accessible to a broad audience and are built on reliable, widely used foundations.

The first project, Scyan, introduces a multi-purpose neural network designed for a range of tasks in cytometry, including annotation, batch-effect removal, debarcoding, and population discovery. Scyan's key advantage is its ability to perform rapid, automatic annotations, particularly for large datasets. By utilizing a marker-population table built from expert knowledge, Scyan can annotate cell populations without the need for manual gating. Researchers highly value this "biology-driven" approach, as it leverages expert knowledge while avoiding the time-consuming and error-prone nature of manual annotation.

The second project, Sopa, addresses the need for robust, general, and scalable tools in spatial omics. Designed as a versatile framework, Sopa improves the preprocessing of spatial omics data compared to vendor-provided defaults. It integrates and standardizes various spatial technologies, facilitating the exploration and visualization of diverse spatial datasets. A key challenge in spatial omics is handling large data volumes that exceed typical RAM capacities. Sopa addresses this by using lazy-loading techniques at every step of the pipeline, ensuring that the full dataset is never loaded into memory simultaneously.

Furthermore, Sopa supports multiple modalities, including protein and H&E data, a first step towards multi-omics spatial data.

Finally, Novae builds on Sopa's preprocessing and enables spatial analysis across large cohorts, based on multiple slides, tissues, and technologies. Novae is a foundational model that captures cell representations within their spatial environments. Trained on a large dataset of nearly 30 million cells across 18 different tissues, Novae offers zero-shot domain inference, meaning users can apply it without re-training. Additionally, Novae automatically corrects batch effects and generates a hierarchical structure of spatial domains, and facilitates various downstream analyses (such as identifying spatially variable genes and pathways, or exploring spatial domain trajectories).

To sum up, I developed three open-source tools — Scyan for automated cytometry analysis, Sopa for scalable spatial omics preprocessing, and Novae for spatial analysis across large cohorts — each addressing different challenges in single-cell resolution omics data and contributing to both clinical and biological research.

7.2 . Usage and impact of the developed methods

All the methods developed in this manuscript are open-source and hosted on Github. Therefore, it is possible to get access to a portion of the users of the packages via Github's API. Notably, we localize the users who starred the repositories, opened issues, or opened pull requests. This is not a direct measure of the impact of the methods, because some users may star a repository without using the package, and many users may be using the package while not starring the repository. Still, it gives a good overview of the number of institutions using the packages, and where they are located across the world. The maps in Figures 7.1 and 7.2 show the localization of the users of Scyan and Sopa, respectively.

The 7.1 shows that scyan is mostly used in France and Europe, with some usage in the US. On the other hand, 7.2 shows that sopa is much more international, with a significant usage in the US, in Asia, and in Europe. Novae is still a very young package (2 month old, at the time of writing, compared to 12 months for Sopa.), but we already see some usage in Europe and in the US.

7.3 . Limitations and proposed improvements

While the methods developed in this manuscript offer significant advancements in the analysis of single-cell resolution omics data, they are not without limitations. Recognizing and addressing these limitations is crucial for both understanding the current state of the

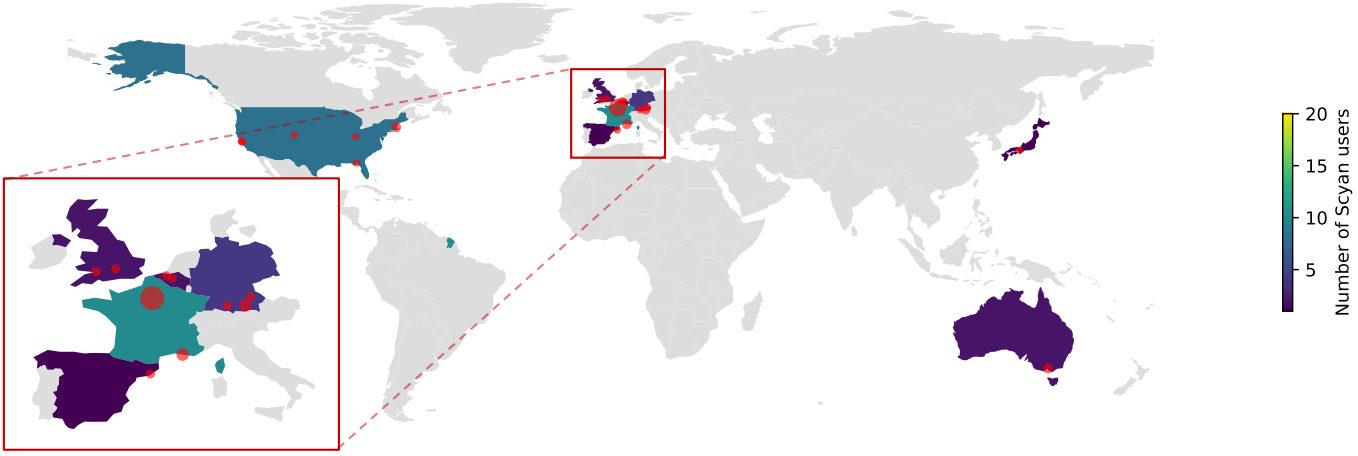

Figure 7.1: ***Overview of the usage of Scyan across the world.*** *The map shows the institutes or companies where Scyan is used. One count represents a user of the package that starred the repo, opened an issue, or opened a pull request. The dots represent the number of users in a specific city, while the color is used for the user count over the whole country. Data collected in November 2024 with Léa Boucher, that is 27 months after the first package release.*

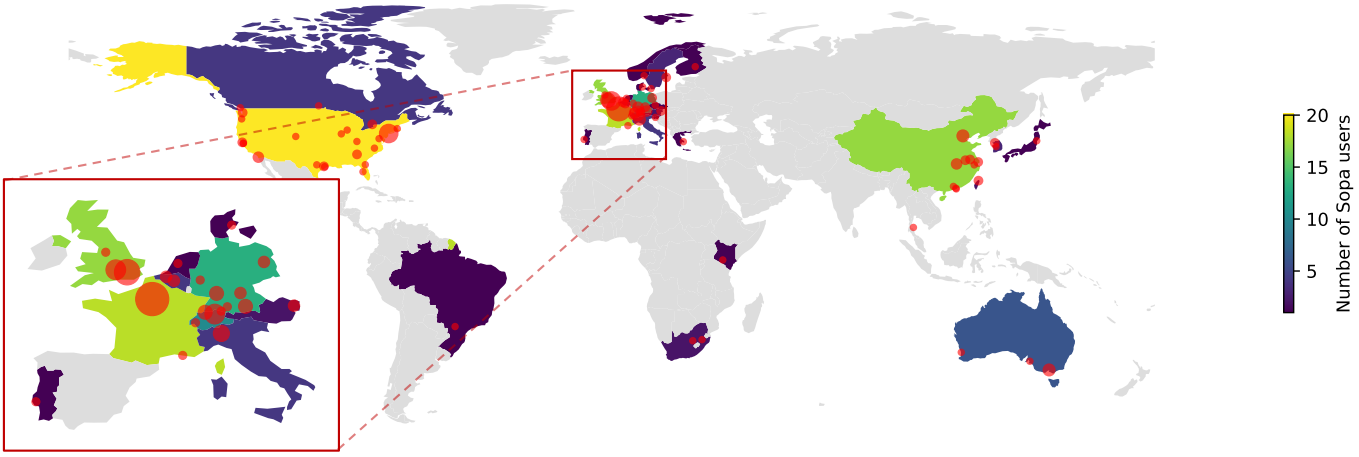

Figure 7.2: ***Overview of the usage of Sopa across the world.*** *The map shows the institutes or companies where Sopa is used. One count represents a user of the package that starred the repo, opened an issue, or opened a pull request. The dots represent the number of users in a specific city, while the color is used for the user count over the whole country. Data collected in November 2024 with Léa Boucher, that is 12 months after the first package release.*

methodologies and guiding future research directions.

Scyan next steps

First, Scyan, as a cytometry analysis tool, is restricted to specific panels in cytometry, which may limit its broad applicability across diverse datasets. This constraint arises because Scyan typically requires a predefined marker-population table for cell-type annotation. This requirement can become a bottleneck when dealing with large and heterogeneous cytometry datasets that contain multiple protein panels. Moreover, the reliance on

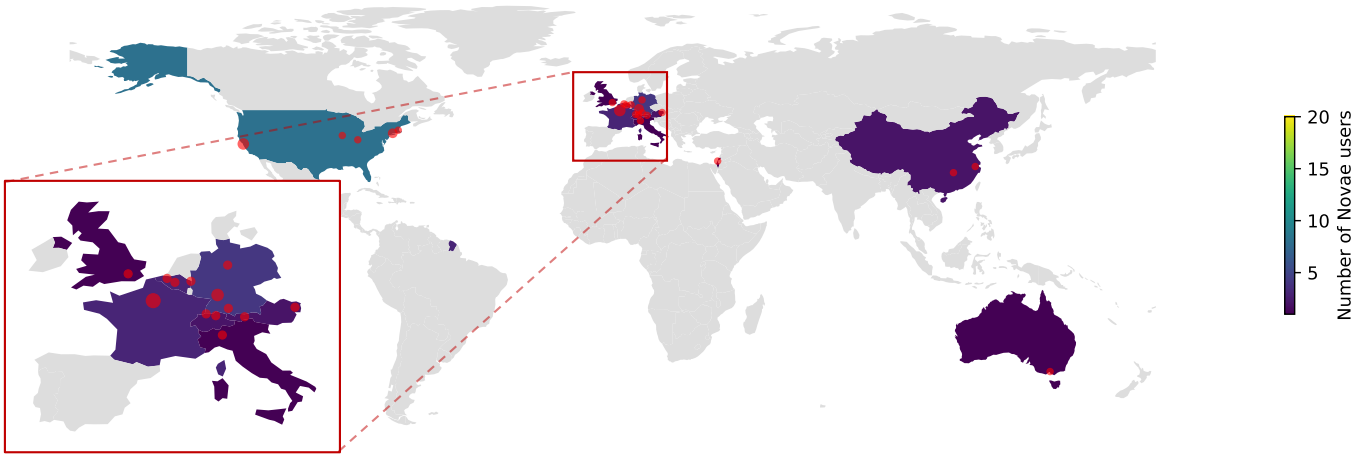

Figure 7.3: ***Overview of the usage of Novae across the world.*** *The map shows the institutes or companies where Novae is used. One count represents a user of the package that starred the repo, opened an issue, or opened a pull request. The dots represent the number of users in a specific city, while the color is used for the user count over the whole country. Data collected in November 2024 with Léa Boucher, that is 2 months after the first package release.*

a marker-population table implies that Scyan depends heavily on prior knowledge. This choice has several benefits, but it also has drawbacks. For instance, prior knowledge may not always be available or comprehensive, particularly for novel or under-explored populations. An opportunity for overcoming this limitation could involve the development of a more generalized approach, similar to Novae, that makes Scyan applicable to any protein panel. Another idea would be to automatically aggregate public tables of marker-population annotations from various sources (e.g., public curated databases), enabling Scyan to leverage a broader knowledge base, without requiring user inputs for each experiment.

Sopa next steps

The current scope of Sopa, another method developed in this work, is limited to 2D spatial data. While 2D spatial omics data provides valuable insights, it only captures a slice of the three-dimensional biological reality. Many biological processes, especially those involving tissue architecture, cellular interactions, and microenvironments, are inherently three-dimensional. Hence, the inability to process 3D spatial data represents a limitation. Extending Sopa to handle 3D spatial data would allow for more accurate and biologically relevant interpretations, particularly for tissues or organs where the spatial organization of cells is critical. In addition to the dimensionality constraint, Sopa's reliance on many Python dependencies introduces challenges in terms of maintenance and scalability. As its Python dependencies evolve and potentially become deprecated, ensuring that Sopa remains functional and up-to-date will require continual effort.

Novae next steps

Similar to the aforementioned limitations in Scyan and Sopa, Novae is constrained in its application, being currently designed for spatial transcriptomics data only. This restricts its utility for other emerging spatial omics modalities, such as spatial proteomics and spatial epigenomics, which are becoming increasingly important for understanding the full complexity of cellular regulation and interaction within tissues. The expansion of Novae to support additional spatial omics techniques would significantly enhance its applicability. This can be done by learning the representation of other modalities, e.g. learning protein embeddings, which requires collecting large spatial proteomics datasets. Such datasets may be available in the near future, as the field keeps evolving quickly.

7.4 . Perspectives

7.4.1 . Towards multi-omics foundation models

The methods developed in this manuscript are predominantly focused on single-omics data analysis, such as spatial transcriptomics. While Novae represents a significant step toward the creation of foundation models in omics, its current scope is limited to analyzing spatial transcriptomics data. However, there is an emerging need and a promising future direction in the development of multi-omics foundation models—models that integrate multiple layers of biological data, including genomics, transcriptomics, proteomics, and epigenomics. Indeed, multi-omics data integration is essential for providing a more comprehensive understanding of cellular and molecular processes [Picard et al., 2021, Benkirane et al., 2023]. Different omics layers capture distinct but complementary aspects of biology. For example, genomics provides information about the genetic blueprint of an organism, while transcriptomics reveals how genes are expressed in different conditions. Proteomics and epigenomics further contribute to understanding the functional outcomes of gene expression and how these processes are regulated. Analyzing these layers in isolation provides only a fragmented view of biological systems. By contrast, a multi-omics approach offers a more holistic view of cellular states and interactions.

One of the most promising directions for advancing multi-omics models is the application of deep learning techniques that have been successfully utilized in other domains. Models like transformer-based architectures, which have demonstrated remarkable success in natural language processing, are particularly promising for multi-omics integration. These architectures can handle complex, high-dimensional data and capture intricate relationships across different omics layers, making them ideal candidates for building foundation models in biology.

The integration of multi-omics data has the potential to improve our understanding

of complex diseases such as cancer, where multiple molecular layers interact in a non-linear and dynamic fashion. For instance, genetic mutations (genomics) may lead to aberrant gene expression (transcriptomics), which subsequently alters protein function (proteomics) and cellular signaling pathways (epigenomics). A multi-omics model would enable researchers to trace these cascading effects across biological layers, leading to more accurate disease models.

Moreover, such models hold immense promise for identifying novel biomarkers and therapeutic targets. By capturing the interplay between different omics layers, multi-omics models could reveal previously hidden relationships that are critical to disease progression or response to treatment. This could open up new avenues for precision medicine, where therapies are personalized to the specific molecular profile of an individual's disease.

7.4.2 . Towards drug discovery

Advanced spatial omics data, particularly at the single-cell level, has the potential to greatly accelerate drug discovery by providing deeper insights into the molecular architecture of diseases. Unlike bulk omics and scRNAseq data, spatial omics captures the precise localization and heterogeneity of cells within their native tissue environment, revealing context-specific molecular interactions that are critical in diseases such as cancer and neurodegenerative disorders.

By leveraging spatial omics, researchers can identify new drug targets that were previously hidden in bulk or scRNAseq data. For example, spatial transcriptomics can uncover signaling pathways active only in specific regions of a tumor, or pinpoint rare cell populations that are located in specific areas and drive disease progression. This level of resolution allows for the development of highly targeted therapies that are more precise and potentially more effective than conventional approaches.

Additionally, spatial omics can be instrumental in addressing one of the major challenges in drug discovery: drug resistance. By mapping how different cell populations within a tissue respond to treatment, spatial data can highlight resistant cell niches that evade therapy. Understanding these spatially distinct resistance mechanisms can inform the design of combination therapies that target both sensitive and resistant cells simultaneously, improving overall treatment efficacy.

In conclusion, spatial omics, with its capacity to unravel the spatial complexity of tissues and cellular interactions, will likely play a critical role in the next generation of drug discovery efforts, enabling more precise, context-aware therapeutic strategies.

7.5 . Looking ahead: AI and spatial omics in medicine and science

7.5.1 . Application to other domains

As we look to the future, the combination of spatial omics and deep learning (or AI, more broadly) is promising important contributions not just in oncology, but across many areas of medicine. The progress made in understanding cancer at the cellular level is just one example of how AI can help better understand complex biological insights. In fields like neurology, undergoing research using deep learning on spatial omics slides could for instance dive deeper into brain diseases, such as Alzheimer's, helping to develop earlier detection methods or more targeted treatments. Immunology may also benefit from AI models that analyze spatial data to better understand immune responses in autoimmune diseases or infections. While the tools developed in this manuscript were applied to oncology, they were developed not developed specifically for oncology: they can therefore be used for discovery in the above domains of medicine, among many others.

7.5.2 . Accessibility and affordability of spatial omics

However, it's important to acknowledge that spatial omics technologies are currently very expensive, limiting their widespread use. High costs for both the equipment and the computational resources needed to analyze the massive datasets make them accessible mainly to well-funded research institutions. Nevertheless, like many technologies in their early stages, costs are expected to decrease over time as the technology matures, just as we've seen with genomic sequencing. The development of more efficient data processing algorithms, miniaturization of equipment, and increased automation could lower prices, making spatial omics more accessible. Therefore, even in low-resource areas, while the high costs may seem like a barrier today, the future holds the potential to make spatial omics more affordable.

In addition to this, the discoveries that spatial omics enable could eventually be applied using more accessible and affordable methods. For example, if spatial omics help identify a new biomarker or disease pathway, this knowledge might later be used with less costly techniques like H&E staining or other standard diagnostics. This could allow the insights gained from spatial omics to be useful in clinical settings that rely on simpler, more widely available tools. Although this process is not guaranteed, it holds promise for making breakthroughs in biology more broadly applicable.

7.5.3 . Ethical considerations

As AI and spatial omics become more integrated into healthcare, several ethical challenges arise. One key concern is the reliability and interpretability of AI models. Many deep learning systems make decisions in ways that are difficult for clinicians to fully un-

derstand, which can hinder their acceptance in medical practice. For AI to be trusted, especially in critical areas like cancer treatment, these models need to be more transparent and explainable to human experts.

Another important issue is ensuring that AI models are equitable and generalizable. If AI systems are trained on limited or biased datasets, they may perform poorly when applied to diverse populations, potentially leading to unequal outcomes. Developing models based on diverse, representative data is essential to ensure fairness.

Data privacy and consent also come into focus, particularly when dealing with sensitive patient information. Strong safeguards must be in place to protect privacy while still enabling the data-sharing necessary for AI advancement.

Lastly, questions of accountability must be addressed. If an AI model leads to a misdiagnosis, who is responsible—developers, clinicians, or institutions? Clear guidelines are needed to ensure ethical AI use in healthcare while balancing AI's potential with human judgment to maintain patient trust.

Bibliography

[Abdelaal et al., 2019] Abdelaal, T. et al. (2019). Predicting cell populations in single cell mass cytometry data. *Cytometry Part A*, 95:769–781.

[Ackley et al., 1985] Ackley, D. H., Hinton, G. E., and Sejnowski, T. J. A. (1985). Learning algorithm for boltzmann machines. *Cognitive Science*, 9:147–169.

[Aghaeepour et al., 2013] Aghaeepour, N. et al. (2013). Critical assessment of automated flow cytometry data analysis techniques. *Nature Methods*, 10:228–238.

[Akiba et al., 2019] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2623–2631.

[Alturki, 2023] Alturki, N. A. (2023). Review of the Immune Checkpoint Inhibitors in the Context of Cancer Treatment. *Journal of Clinical Medicine*, 12:4301.

[Amodio et al., 2019] Amodio, M. et al. (2019). Exploring single-cell data with deep multitasking neural networks. *Nature Methods*, 16:1139–1145.

[André et al., 2024] André, F., Rassy, E., Marabelle, A., Michiels, S., and Besse, B. (2024). Forget lung, breast or prostate cancer: Why tumour naming needs to change. *Nature*, 626:26–29.

[Atta and Fan, 2021] Atta, L. and Fan, J. (2021). Computational challenges and opportunities in spatially resolved transcriptomic data analysis. *Nature Communications*, 12:5283.

[Axelrod et al., 2021] Axelrod, S. et al. (2021). Starfish: scalable pipelines for image-based transcriptomics. *Journal of Open Source Software*, 6:2440.

[Barrett et al., 2005] Barrett, P., Hunter, J., Miller, J. T., Hsu, J.-C., and Greenfield, P. (2005). matplotlib - a portable python plotting package. *Astronomical Data Analysis Software and Systems XIV ASP Conference*, 347.

[Behbehani, 2019] Behbehani, G. K. (2019). Immunophenotyping by mass cytometry. *Methods in Molecular Biology*, 2032:31–51.

[Bendall et al., 2011] Bendall, S. C. et al. (2011). Single-cell mass cytometry of differential immune and drug responses across a human hematopoietic continuum. *Science*, 332:687–696.

[Benkirane et al., 2023] Benkirane, H., Pradat, Y., Michiels, S., and Cournède, P.-H. (2023). CustOmics: A versatile deep-learning based strategy for multi-omics integration. *PLOS Computational Biology*, 19:e1010921.

[Bevins et al., 2023] Bevins, H., Handley, W., and Gessey-Jones, T. (2023). Piecewise normalizing flows. *arXiv*, 2305.02930.

[Biancalani et al., 2021] Biancalani, T. et al. (2021). Deep learning and alignment of spatially resolved single-cell transcriptomes with tangram. *Nature Methods*, 18:1352–1362.

[Bijelic and Rubio, 2021] Bijelic, L. and Rubio, E. R. (2021). Tumor necrosis in hepatocellular carcinoma–unfairly overlooked? *Annals of Surgical Oncology*, 28:600–601.

[Binnewies et al., 2021] Binnewies, M. et al. (2021). Targeting trem2 on tumor-associated macrophages enhances immunotherapy. *Cell Reports*, 37:109844.

[Blampey et al., 2024a] Blampey, Q., Benkirane, H., Bercovici, N., Andre, F., and Cournede, P.-H. (2024a). Novae: A graph-based foundation model for spatial transcriptomics data. *BioArxiv*, page 2024.09.09.612009.

[Blampey et al., 2023] Blampey, Q., Bercovici, N., Dutertre, C.-A., Pic, I., Ribeiro, J. M., André, F., and Cournède, P.-H. (2023). A biology-driven deep generative model for cell-type annotation in cytometry. *Briefings in Bioinformatics*, page bbad260.

[Blampey et al., 2024b] Blampey, Q., Mulder, K., Gardet, M., Christodoulidis, S., Dutertre, C.-A., André, F., Ginhoux, F., and Cournède, P.-H. (2024b). Sopa: a technology-invariant pipeline for analyses of image-based spatial omics. *Nature Communications*, 15:4981.

[Bressan et al., 2023] Bressan, D., Battistoni, G., and Hannon, G. J. (2023). The dawn of spatial omics. *Science*, 381:eabq4964.

[Brodersen, 2023] Brodersen, P. (2023). Netgraph: Publication-quality network visualisations in python. *Journal of Open Source Software*, 8:5372.

[Brody et al., 2022] Brody, S., Alon, U., and Yahav, E. (2022). How attentive are graph attention networks? *International Conference on Learning Representations*.

[Caron et al., 2020] Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., and Joulin, A. (2020). Unsupervised Learning of Visual Features by Contrasting Cluster Assignments. *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 831:9912 – 9924.

[Chang et al., 2017] Chang, Q. et al. (2017). Imaging mass cytometry. *Cytometry Part A*, 91:160–169.

[Chen et al., 2015] Chen, K. H., Boettiger, A. N., Moffitt, J. R., Wang, S., and Zhuang, X. (2015). Spatially resolved, highly multiplexed RNA profiling in single cells. *Science*, 348:aaa6090.

[Chinthrajah et al., 2019] Chinthrajah, R. S. et al. (2019). Sustained outcomes in oral immunotherapy for peanut allergy (poised study): a large, randomised, double-blind, placebo-controlled, phase 2 study. *The Lancet*, 394:1437–1449.

[Chu et al., 2023] Chu, Y. et al. (2023). Pan-cancer t cell atlas links a cellular stress response state to immunotherapy resistance. *Nature Medicine*, 29:1550–1562.

[Cisar et al., 2023] Cisar, C., Keener, N., Ruffalo, M., and Paten, B. A. (2023). Unified pipeline for fish spatial transcriptomics. *Cell Genomics*, 3.

[Cui et al., 2024] Cui, H., Wang, C., Maan, H., Pang, K., Luo, F., Duan, N., and Wang, B. (2024). scGPT: Toward building a foundation model for single-cell multi-omics using generative AI. *Nature Methods*, pages 1–11.

[Davies and Bouldin, 1979] Davies, D. and Bouldin, D. A. (1979). Cluster separation measure. pattern analysis and machine intelligence. *IEEE Transactions on*, PAMI-1:224–227.

[Defard et al., 2024] Defard, T., Laporte, H., Ayan, M., Soulier, J., Curras-Alonso, S., Weber, C., Massip, F., Londoño-Vallejo, J.-A., Fouillade, C., Mueller, F., and Walter, T. (2024). A point cloud segmentation framework for image-based spatial transcriptomics. *Communications Biology*, 7:1–13.

[Dinh et al., 2017] Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2017). Density estimation using real NVP. *International Conference on Learning Representations*.

[Dong and Zhang, 2022] Dong, K. and Zhang, S. (2022). Deciphering spatial domains from spatially resolved transcriptomics with an adaptive graph attention auto-encoder. *Nature Communications*, 13:1739.

[Dosovitskiy et al., 2021] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv*, 2010.11929.

[Dries et al., 2021] Dries, R. et al. (2021). Advances in spatial transcriptomic data analysis. *Genome Research*, 31:1706–1718.

[Du et al., 2023] Du, J., Yang, Y.-C., An, Z.-J., Zhang, M.-H., Fu, X.-H., Huang, Z.-F., Yuan, Y., and Hou, J. (2023). Advances in spatial transcriptomics and related data analysis strategies. *Journal of Translational Medicine*, 21:330.

[Fan et al., 2023] Fan, T., Zhang, M., Yang, J., Zhu, Z., Cao, W., and Dong, C. (2023). Therapeutic cancer vaccines: Advancements, challenges and prospects. *Signal Transduction and Targeted Therapy*, 8:1–23.

[Fey and Lenssen, 2019] Fey, M. and Lenssen, J. E. (2019). Fast graph representation learning with PyTorch Geometric. *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[Ficler and Goldberg, 2017] Ficler, J. and Goldberg, Y. (2017). Controlling Linguistic Style Aspects in Neural Language Generation. *Proceedings of the Workshop on Stylistic Variation*, pages 94–104.

[Fonseca et al., 2009] Fonseca, S., Abdelmassih, S., Lavagnolli, T., Serafim, R., Santos, E., Mendes, C., Pereira, V., Ambrosio, C., Miglino, M., Visintin, J., Abdelmassih, R., Kerkis, A., and Kerkis, I. (2009). Human immature dental pulp stem cells' contribution to developing mouse embryos: Production of human/mouse preterm chimaeras. *Cell proliferation*, 42:132–40.

[Gessain et al., 2024] Gessain, G. et al. (2024). Trem2-expressing multinucleated giant macrophages are a biomarker of good prognosis in head and neck squamous cell carcinoma. *Cancer Discovery*.

[Goodfellow et al., 2014] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. *arXiv*, 1406.2661.

[Hao et al., 2023] Hao, Y. et al. (2023). Dictionary learning for integrative, multimodal and scalable single-cell analysis. *Nature Biotechnology*, pages 1–12.

[Hartman and Satija, 2024] Hartman, A. and Satija, R. (2024). Comparative analysis of multiplexed in situ gene expression profiling technologies. *PubMed*.

[Hassan and Seno, 2020] Hassan, G. and Seno, M. (2020). Blood and Cancer: Cancer Stem Cells as Origin of Hematopoietic Cells in Solid Tumor Microenvironments. *Cells*, 9:1293.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

[He et al., 2022] He, S. et al. (2022). High-plex imaging of RNA and proteins at subcellular resolution in fixed tissue by spatial molecular imaging. *Nature Biotechnology*, pages 1–13.

[Hoyer and Hamman, 2017] Hoyer, S. and Hamman, J. (2017). xarray: N-d labeled arrays and datasets in python. *Journal of Open Research Software*, 5:10.

[Hubert and Arabie, 1985] Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of Classification*, 2:193–218.

[Izmailov et al., 2020] Izmailov, P., Kirichenko, P., Finzi, M., and Wilson, A. G. (2020). Semi-supervised learning with normalizing flows. *Proceedings of the 37th International Conference on Machine Learning*, 119:4615–4630.

[Jaiswal et al., 2021] Jaiswal, A., Babu, A. R., Zadeh, M. Z., Banerjee, D., and Makedon, F. (2021). A Survey on Contrastive Self-Supervised Learning. *Technologies*, 9:2.

[Janesick et al., 2023] Janesick, A. et al. (2023). High resolution mapping of the tumor microenvironment using integrated single-cell, spatial and in situ analysis. *Nature Communications*, 14:8353.

[Jass, 2007] Jass, J. R. (2007). Classification of colorectal cancer based on correlation of clinical, morphological and molecular features. *Histopathology*, 50:113–130.

[Jhaveri et al., 2023] Jhaveri, N. et al. (2023). Single-cell spatial metabolic and immune phenotyping of head and neck cancer tissues identifies tissue signatures of response and resistance to immunotherapy. *BioArxiv*.

[Ji et al., 2018] Ji, D., Nalisnick, E., Qian, Y., Scheuermann, R. H., and Smyth, P. (2018). Bayesian trees for automated cytometry data analysis. *Proceedings of the 3rd Machine Learning for Healthcare Conference (PMLR)*, pages 465–483.

[Jiao and Du, 2016] Jiao, Y. and Du, P. (2016). Performance measures in evaluating machine learning based bioinformatics predictors for classifications. *Quantitative Biology*, 4:320–330.

[Jin and Lloyd, 1997] Jin, L. and Lloyd, R. V. (1997). In situ hybridization: Methods and applications. *Journal of Clinical Laboratory Analysis*, 11:2–9.

[Johnson et al., 2007] Johnson, W. E., Li, C., and Rabinovic, A. (2007). Adjusting batch effects in microarray expression data using empirical Bayes methods. *Biostatistics*, 8:118–127.

[Kaushik et al., 2021] Kaushik, A. et al. (2021). Cyanno: a semi-automated approach for cell type annotation of mass cytometry datasets. *Bioinformatics*, 37:4164–4171.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, L. J. (2015). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*.

[Kingma et al., 2016] Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. (2016). Improved variational inference with inverse autoregressive flow. *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 4743–4751.

[Kingma and Welling, 2019] Kingma, D. P. and Welling, M. (2019). An introduction to variational autoencoders. *FNT in Machine Learning*, 12:307–392.

[Kinkhabwala et al., 2022] Kinkhabwala, A. et al. (2022). Macsima imaging cyclic staining (mics) technology reveals combinatorial target pairs for car t cell treatment of solid tumors. *Scientific Reports*, 12:1911.

[Korsunsky et al., 2019] Korsunsky, I. et al. (2019). Fast, sensitive and accurate integration of single-cell data with harmony. *Nature Methods*, 16:1289–1296.

[Köster and Rahmann, 2018] Köster, J. and Rahmann, S. (2018). Snakemake–a scalable bioinformatics workflow engine. *Bioinformatics*, 34:3600.

[Kumar et al., 2023] Kumar, T. et al. (2023). A spatially resolved single-cell genomic atlas of the adult human breast. *Nature*, 620:181–191.

[LeCun et al., 2015] LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521:436–444.

[Lee et al., 2000] Lee, C.-K., Weindruch, R., and Prolla, T. A. (2000). Gene-expression profile of the ageing brain in mice. *Nature Genetics*, 25:294–297.

[Lee et al., 2017] Lee, H.-C., Kosoy, R., Becker, C. E., Dudley, J. T., and Kidd, B. A. (2017). Automated cell type discovery and classification through knowledge transfer. *Bioinformatics*, 33:1689–1695.

[Lei et al., 2023] Lei, J., Hu, X., Wang, Y., and Liu, D. (2023). Pyramidflow: High-resolution defect contrastive localization using pyramid normalizing flow. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14143–14152.

[Levine et al., 2015] Levine, J. H. et al. (2015). Data-driven phenotypic dissection of aml reveals progenitor-like cells that correlate with prognosis. *Cell*, 162:184–197.

[Lewis et al., 2021] Lewis, S. M. et al. (2021). Spatial omics and multiplexed imaging to explore cancer biology. *Nature Methods*, 18:997–1012.

[Li et al., 2017] Li, H. et al. (2017). Gating mass cytometry data by deep learning. *Bioinformatics*, 33:3423–3430.

[Liu et al., 2020] Liu, P. et al. (2020). Recent advances in computer-assisted algorithms for cell subtype identification of cytometry data. *Frontiers in Cell and Developmental Biology*, 8:234.

[Long et al., 2023] Long, Y. et al. (2023). Spatially informed clustering, integration, and deconvolution of spatial transcriptomics with GraphST. *Nature Communications*, 14:1155.

[Lopez et al., 2018] Lopez, R., Regier, J., Cole, M. B., Jordan, M. I., and Yosef, N. (2018). Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15:1053–1058.

[Lun et al., 2017] Lun, A. T. L., Richard, A. C., and Marioni, J. C. (2017). Testing for differential abundance in mass cytometry data. *Nature Methods*, 14:707–709.

[Lundberg and Lee, 2017] Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 4768–4777.

[Marconato et al., 2024] Marconato, L., Palla, G., Yamauchi, K. A., Virshup, I., Heidari, E., Treis, T., Vierdag, W.-M., Toth, M., Stockhaus, S., Shrestha, R. B., Rombaut, B., Pollaris, L., Lehner, L., Vöhringer, H., Kats, I., Saeys, Y., Saka, S. K., Huber, W., Gerstung, M., Moore, J., Theis, F. J., and Stegle, O. (2024). SpatialData: an open and universal data framework for spatial omics. *Nature Methods*, pages 1–5.

[McCombie et al., 2019] McCombie, W. R., McPherson, J. D., and Mardis, E. R. (2019). Next-Generation Sequencing Technologies. *Cold Spring Harbor Perspectives in Medicine*, 9:a036798.

[McInnes et al., 2018] McInnes, L., Healy, J., Saul, N., and Großberger, L. (2018). UMAP: Uniform Manifold Approximation and Projection. *Journal of Open Source Software*, 3:861.

[McKinnon, 2018] McKinnon, K. M. F. C. (2018). An overview. *Current Protocols in Immunology*, 120:1–5.

[Merritt et al., 2020] Merritt, C. R. et al. (2020). Multiplex digital spatial profiling of proteins and rna in fixed tissue. *Nature Biotechnology*, 38:586–599.

[Molgora et al., 2020] Molgora, M. et al. (2020). Trem2 modulation remodels the tumor myeloid landscape, enhancing anti-pd-1 immunotherapy. *Cell*, 182, .e17:886–900.

[Mootha et al., 2003] Mootha, V. K. et al. (2003). PGC-1α-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes. *Nature Genetics*, 34:267–273.

[Moses and Pachter, 2022] Moses, L. and Pachter, L. (2022). Museum of spatial transcriptomics. *Nature Methods*, 19:534–546.

[Mulder et al., 2021] Mulder, K. et al. (2021). Cross-tissue single-cell landscape of human monocytes and macrophages in health and disease. *Immunity*, 54, .e5:1883–1900.

[Newell and Cheng, 2016] Newell, E. W. and Cheng, Y. (2016). Mass cytometry: blessed with the curse of dimensionality. *Nature Immunology*, 17:890–895.

[Palla et al., 2022] Palla, G., Spitzer, H., et al. (2022). Squidpy: a scalable framework for spatial omics analysis. *Nature Methods*, 19.

[Papamakarios et al., 2021] Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. (2021). Normalizing flows for probabilistic modeling and inference. *J. Mach. Learn. Res.*, 22:57:2617–57:2680.

[Papamakarios et al., 2017] Papamakarios, G., Pavlakou, T., and Murray, I. (2017). Masked Autoregressive Flow for Density Estimation. *Advances in Neural Information Processing Systems*, 30.

[Parks et al., 2006] Parks, D. R., Roederer, M., and Moore, W. A. A. (2006). New logicle display method avoids deceptive effects of logarithmic scaling for low signals and compensated data. *Cytometry Part A*, 69A:541–551.

[Paszke et al., 2019] Paszke, A. et al. (2019). PyTorch: An imperative style, high-performance deep learning library. *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 8026–8037.

[Pedregosa et al., 2011] Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research (JMLR)*, 12:2825–2830.

[Perkel, 2021] Perkel, J. M. (2021). Python power-up: new image tool visualizes complex data. *Nature*, 600:347–348.

[Petukhov et al., 2022] Petukhov, V. et al. (2022). Cell segmentation in imaging-based spatial transcriptomics. *Nature Biotechnology*, 40:345–354.

[Peyré and Cuturi, 2020] Peyré, G. and Cuturi, M. (2020). Computational Optimal Transport. *Foundations and Trends in Machine Learning*, 11:355–607.

[Picard et al., 2021] Picard, M., Scott-Boyer, M.-P., Bodein, A., Périn, O., and Droit, A. (2021). Integration strategies of multi-omics data for machine learning analysis. *Computational and Structural Biotechnology Journal*, 19:3735–3746.

[Qiu et al., 2011] Qiu, P. et al. (2011). Extracting a cellular hierarchy from high-dimensional cytometry data with spade. *Nature Biotechnology*, 29:886–891.

[Rani et al., 2023] Rani, V., Nabi, S. T., Kumar, M., Mittal, A., and Kumar, K. (2023). Self-supervised Learning: A Succinct Review. *Archives of Computational Methods in Engineering: State of the Art Reviews*, 30:2761–2775.

[Rao et al., 2021] Rao, A., Barkley, D., França, G. S., and Yanai, I. (2021). Exploring tissue architecture using spatial transcriptomics. *Nature*, 596:211–220.

[Ren et al., 2022] Ren, H., Walker, B. L., Cang, Z., and Nie, Q. (2022). Identifying multicellular spatiotemporal organization of cells with SpaceFlow. *Nature Communications*, 13:4076.

[Rezende and Mohamed, 2015] Rezende, D. J. and Mohamed, S. (2015). Variational inference with normalizing flows. *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, 37:1530–1538.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention 2015*, pages 234–241.

[Rousseeuw, 1987] Rousseeuw, P. J. S. (1987). A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

[Sallusto et al., 1999] Sallusto, F., Lenig, D., Förster, R., Lipp, M., and Lanzavecchia, A. (1999). Two subsets of memory t lymphocytes with distinct homing potentials and effector functions. *Nature*, 401:708–712.

[Sautès-Fridman et al., 2019] Sautès-Fridman, C., Petitprez, F., Calderaro, J., and Fridman, W. H. (2019). Tertiary lymphoid structures in the era of cancer immunotherapy. *Nature Reviews Cancer*, 19:307–325.

[Schmidt et al., 2018] Schmidt, U., Weigert, M., Broaddus, C., and Myers, G. (2018). Cell detection with star-convex polygons. *Medical Image Computing and Computer Assisted Intervention*, pages 265–273.

[Scrucca et al., 2016] Scrucca, L., Fraley, C., Murphy, T. B., and Raftery, A. E. (2016). Model-Based Clustering, Classification, and Density Estimation Using mclust in R. *The RJournal*, 8.

[Sharma et al., 2020] Sharma, A. et al. (2020). Onco-fetal reprogramming of endothelial cells drives immunosuppressive macrophages in hepatocellular carcinoma. *Cell*, 183, .e21:377–394.

[Sharma et al., 2005] Sharma, S., Sharma, M. C., and Sarkar, C. (2005). Morphology of angiogenesis in human cancer: a conceptual overview, histoprognostic perspective and significance of neoangiogenesis. *Histopathology*, 46:481–489.

[Spitzer and Nolan, 2016] Spitzer, M. H. and Nolan, G. P. (2016). Mass cytometry: Single cells, many features. *Cell*, 165:780–791.

[Staats et al., 2019] Staats, J., Divekar, A., McCoy, J., and Maecker, H. (2019). Guidelines for gating flow cytometry data for immunological assays. *Methods in molecular biology*, 2032:81–104.

[Sterner and Sterner, 2021] Sterner, R. C. and Sterner, R. M. (2021). CAR-T cell therapy: Current limitations and potential strategies. *Blood Cancer Journal*, 11:1–11.

[Stringer et al., 2021] Stringer, C., Wang, T., Michaelos, M., and Pachitariu, M. (2021). Cellpose: a generalist algorithm for cellular segmentation. *Nature Methods*, 18:100–106.

[Subramanian et al., 2005] Subramanian, A. et al. (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences*, 102:15545–15550.

[Traag et al., 2019] Traag, V. A., Waltman, L., and van Eck, N. J. (2019). From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9:5233.

[Tsimberidou et al., 2020] Tsimberidou, A. M., Fountzilas, E., Nikanjam, M., and Kurzrock, R. (2020). Review of precision cancer medicine: Evolution of the treatment paradigm. *Cancer Treatment Reviews*, 86:102019.

[Vandereyken et al., 2023] Vandereyken, K., Sifrim, A., Thienpont, B., and Voet, T. (2023). Methods and applications for single-cell and spatial multi-omics. *Nature Reviews Genetics*, 24:494–515.

[Vaswani et al., 2023] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention Is All You Need. *arXiv*, 1706.03762.

[Velten and Stegle, 2023] Velten, B. and Stegle, O. (2023). Principles and challenges of modeling temporal and spatial omics data. *Nature Methods*, 20:1462–1474.

[Virshup et al., 2023] Virshup, I., Bredikhin, D., Heumos, L., Palla, G., Sturm, G., Gayoso, A., Kats, I., Koutrouli, M., Berger, B., Pe'er, D., Regev, A., Teichmann, S. A., Finotello, F., Wolf, F. A., Yosef, N., Stegle, O., and Theis, F. J. (2023). The scverse project provides a computational ecosystem for single-cell omics data analysis. *Nature Biotechnology*, 41:604–606.

[Virshup et al., 2021] Virshup, I., Rybakov, S., Theis, F. J., Angerer, P., and Wolf, F. A. (2021). Anndata: Annotated data. *bioRxiv*, page 2021.12.16.473007.

[Virtanen et al., 2020] Virtanen, P. et al. (2020). Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261–272.

[Waldman et al., 2020] Waldman, A. D., Fritz, J. M., and Lenardo, M. J. (2020). A guide to cancer immunotherapy: From T cell basic science to clinical practice. *Nature Reviews Immunology*, 20:651–668.

[Waskom, 2021] Waskom, M. L. (2021). Seaborn: Statistical data visualization. *Journal of Open Source Software*, 6:3021.

[Wei et al., 2021] Wei, T. et al. (2021). Tumor necrosis impacts prognosis of patients undergoing curative-intent hepatocellular carcinoma. *Annals of Surgical Oncology*, 28:797–805.

[Wes McKinney, 2010] Wes McKinney (2010). Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, pages 56 – 61.

[Wilcoxon, 1945] Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, 1:80–83.

[Wolf et al., 2018] Wolf, F. A., Angerer, P., and Theis, F. J. (2018). SCANPY: large-scale single-cell gene expression data analysis. *Genome Biology*, 19:15.

[Wolf et al., 2019] Wolf, F. A., Hamey, F. K., Plass, M., Solana, J., Dahlin, J. S., Göttgens, B., Rajewsky, N., Simon, L., and Theis, F. J. (2019). PAGA: Graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. *Genome Biology*, 20:59.

[Wu et al., 2022a] Wu, Y., Cheng, Y., Wang, X., Fan, J., and Gao, Q. (2022a). Spatial omics: Navigating to the golden era of cancer research. *Clinical and Translational Medicine*, 12:e696.

[Wu et al., 2022b] Wu, Z. et al. (2022b). Graph deep learning for the characterization of tumour microenvironments from spatial protein profiles in tissue specimens. *Nature Biomedical Engineering*, 6:1435–1448.

[Xian et al., 2019] Xian, Y., Lampert, C. H., Schiele, B., and Akata, Z. (2019). Zero-Shot Learning–A Comprehensive Evaluation of the Good, the Bad and the Ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:2251–2265.

[Xu et al., 2024] Xu, H., Fu, H., Long, Y., Ang, K. S., Sethi, R., Chong, K., Li, M., Uddamvathanak, R., Lee, H. K., Ling, J., Chen, A., Shao, L., Liu, L., and Chen, J. (2024). Unsupervised spatially embedded deep representation of spatial transcriptomics. *Genome Medicine*, 16:12.

[Yamashita et al., 2018] Yamashita, R., Nishio, M., Do, R. K. G., and Togashi, K. (2018). Convolutional neural networks: An overview and application in radiology. *Insights into Imaging*, 9:611–629.

[Zeng et al., 2022] Zeng, Z., Li, Y., Li, Y., and Luo, Y. (2022). Statistical and machine learning methods for spatially resolved transcriptomics data analysis. *Genome Biology*, 23:83.

[Zhang et al., 2019] Zhang, A. W. et al. (2019). Probabilistic cell-type assignment of single-cell rna-seq for tumor microenvironment profiling. *Nature Methods*, 16:1007–1015.

[Zhou et al., 2022] Zhou, L. et al. (2022). Integrated analysis highlights the immunosuppressive role of trem2+ macrophages in hepatocellular carcinoma. *Frontiers in Immunology*, 13.

[Zunder et al., 2015] Zunder, E. R. et al. (2015). Palladium-based mass tag cell barcoding with a doublet-filtering scheme and single-cell deconvolution algorithm. *Nature Protocols*, 10:316–333.

Supplementary materials

A.1 . Scyan supplementals

A.1.1 . Discussion on the approach and related works

In cytometry, when it comes to model the probability density functions of multidimensional marker expressions, their appearances make it natural to first consider Gaussian Mixture Models (GMM). However, in practice, each component of a GMM estimated from the data may not necessarily map to one population. Indeed, two small populations can be merged into one, and one large population may be split into two components with no interesting biological distinction between them. Also, we would have to annotate each component of the mixture. It could be done manually or using a semi-supervised approach. Yet, as discussed in the introduction, we prefer to use only the knowledge table ρ instead. In terms of deep generative models, there are two main reasons to choose a Real NVP [Dinh et al., 2017] (the normalizing flow architecture) over GANs [Goodfellow et al., 2014], and VAEs [Kingma and Welling, 2019]: the flow invertibility and the ability to compute the exact likelihood of a sample. Indeed, the flow invertibility enables a natural and simple way to correct the batch effect by transforming latent expressions back into the original space. Moreover, the ability to compute the exact likelihood of samples makes the annotation straightforward using the Bayes rule and the known base distribution. Another interesting property is that the Real NVP has a triangular Jacobian with positive terms on the diagonal. It enforces the model to diffuse the marker expressions slowly and prevents multiplication by a negative term. Indeed, such smooth transformations are essential to ensure that we do not mix the mapping between a population component and its actual marker expression density. Also, the Jacobian determinant term in the loss function controls how much the flow dilates volumes in a point neighbourhood. This term thus prevents the collapse of a vast part of the space into a tiny component of the base distribution. From a biological point of view, the Real NVP transformations can be seen as compositions of complex compensations and monotonic transformations learned via deep learning.

	Manual Gating	Baseline	Phenograph	MP	ACDC	LDA	CyAnno	Scyan (ours)
Reproducible	-	✓	✓	✓	✓	✓	✓	✓
Does not rely on manual gating	-	✓	✓	✓	✓	-	-	✓
Population discovery	✓	-	✓	-	✓	-	-	✓
Soft predictions	-	✓	-	-	✓	✓	✓	✓
Interpretable	✓	-	✓	✓	-	-	-	✓
Generative model	-	-	-	-	-	-	-	✓
Batch-effect correction	-	-	-	-	-	-	-	✓

Table A.1: Comparison of model properties. We listed all the models considered in this article as well as manual gating.

The Real NVP is one example of normalizing flow architecture among many others [Rezende and Mohamed, 2015, Papamakarios et al., 2017, Bevins et al., 2023, Lei et al., 2023, Kingma et al., 2016]. Yet, for our task, we need a fast likelihood evaluation for the annotation, and a fast sampling for the batch effect correction. For this reason, we decided not to use the MAF [Papamakarios et al., 2017], which is slow for sampling, and the IAF [Kingma et al., 2016], which is slow for likelihood evaluation. Instead, the Real NVP is fast for both sampling and likelihood evaluation. Also, radial flows [Rezende and Mohamed, 2015] are invertible, but there is no closed form for the inverse, so batch effect correction could not be run in practice. Finally, some other architectures are specific to computer vision [Lei et al., 2023].

Also, the usage of Scyan on cytometry data shows promise for future usage on scRNAseq data. This requires handling dropouts, which is specific to scRNAseq data. For that, two options can be explored: (i) adding a loss term and a component that performs imputation, or (ii) change the Bayesian model to include Zero-Inflated-Negative-Binomial (ZINB) distributions. Such modifications should then be evaluated against state-of-the-art models in scRNAseq data analysis on multiple public datasets.

A.1.2 . Density approximation in the presence of NA

Let z be a population and m a marker such that $\rho_{z,m} = \text{NA}$. It leads to $E_m \mid (Z = z) \sim \mathcal{U}([-1, 1])$ and $H_m \sim \mathcal{N}(0, \sigma)$. Thus, $U_m = E_m + H_m$ does not have a simple density expression. We approximate the probability density function $p_{U_m \mid Z=z}$ by the following

function (see subsection A.1.2), where $r = 1 - \sigma$ and $\gamma = \frac{\sigma\sqrt{2\pi}}{2r+\sigma\sqrt{2\pi}}$:

$$\tilde{p}_{U_m\,|Z=z}(u) = \begin{cases} \gamma \cdot \mathcal{N}(u + r; 0, \sigma) & \text{if } u \leq -r \\ \frac{\gamma}{\sigma\sqrt{2\pi}} & \text{if } u \in]-r, r[\\ \gamma \cdot \mathcal{N}(u - r; 0, \sigma) & \text{if } u \geq r. \end{cases} \tag{A.1}$$

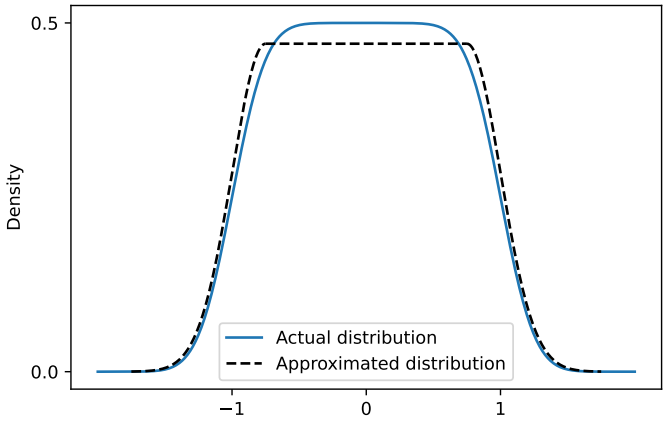

Figure A.1: ***Density approximation in the presence of NA.*** *If z and m are a population and a marker respectively such that $\rho_{z,m} = NA$, then $p_{U_m\,|Z=z}$ is approximated. This figure illustrates the actual and the approximated distribution with $\sigma = \frac{1}{4}$.*

Gradients are null in $]-r, r[$, and the queues of the approximated distribution are similar to the actual one. This expression is easy to compute, efficient during training, and a close approximation.

A.1.3 . Ablation study

Here, we present the results of tests consisting in removing some components of Scyan to show which contributed the most to its overall performance. We consider the two following components to be removed:

- The uniform prior for markers that are unknown. Thus, instead of modelling missing knowledge by uniform distribution, we put a 0 in the table ρ. It simplifies the model considerably, and the density approximation is not needed anymore (see supplementary subsection A.1.2).
- The normalizing flow itself. Thus, we simply use the Bayesian model to perform annotation. Without the normalizing flow, the model is not able to integrate covariates (thus, it cannot correct batch effect).

We performed this study on the largest and most complex dataset, i.e., the POISED dataset, and computed the same three metrics as in the original benchmark (i.e., F1-score, accuracy, and balance accuracy). Two cases are considered: with or without batch-effect amplification. Note that, without the normalizing flow, the model is completely deterministic;

thus, no standard deviation is provided. For the two models with the normalizing flow (last two rows), we provide a standard deviation over 5 runs. The results are listed below in supplementary Table A.2.

	Integrates covariates	Balanced acc. (POISED-A)	Accuracy (POISED-A)	F1-score (POISED-A)	Balanced acc. (POISED)	Accuracy (POISED)	F1-score (POISED)
No uniform prior, no normalizing flow	-	43.0	44.9	28.5	80.1	80.1	66.9
No normalizing flow	-	29.1	24.7	21.5	74.6	72.7	61.6
No uniform prior	✓	55.1 ± 15.8	62.7 ± 26.4	47.6 ± 17.4	75.3 ± 5.8	90.7 ± 1.1	73.4 ± 5.3
Scyan (both components)	✓	$\mathbf{71.7 \pm 1.7}$	$\mathbf{83.6 \pm 1.5}$	$\mathbf{62.9 \pm 1.7}$	$\mathbf{85.2 \pm 1.8}$	$\mathbf{91.7 \pm 0.3}$	$\mathbf{76.2 \pm 2.6}$

Table A.2: Ablation study on the POISED dataset and the batch-effect-amplified dataset (POISED-A). Two components are removed (each separately, or both): the uniform prior and the normalizing flow. For the two datasets (i.e., with or without batch effect amplification), we provide the F1-score, the balanced accuracy, and the accuracy of the different models.

The uniform prior increases performance when combined with the normalizing flow. Considering only the normalizing flow without the uniform prior (third row), has lower performances than the complete model and a higher standard deviation.

A.1.4 . Selecting the log probability threshold

In the methods section, we defined a log threshold t_{min} to decide whether or not to label a cell, i.e., we don't label a cell if:

$$max_{1 \leq z \leq P} \;\; p_{U|Z=z}(f_\phi(\mathbf{x}, \mathbf{c})) \leq e^{t_{min}}.$$

By default, we choose $t_{min} = -50$. It is chosen based on the curve of the ratio of predicted cells as a function of t_{min} (see Figure A.2). When the threshold is too high, all cell probabilities are below the threshold, and no cells are classified. Therefore, this threshold has to be chosen to still predict a large number of cells, but we don't want it to be too low (otherwise, we could miss some unknown populations or annotate low-quality cells). In practice, one can choose a value before the significant decrease in the predicted ratio of cells (see Figure A.2).

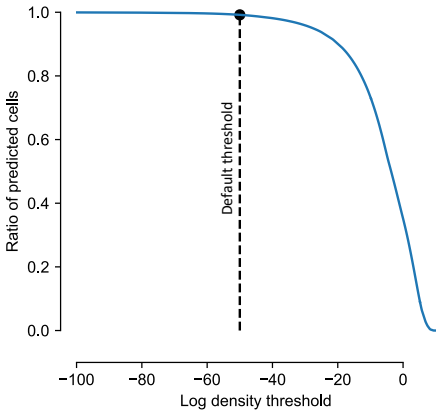

Figure A.2: ***Choosing*** t_{min}*. We select the threshold such that it still predicts a high number of cells. The default (advised) value is -50, shown in a dotted black line*

A.1.5 . Scyan robustness study

We run two studies to evaluate Scyan's robustness:

- Sensitivity to different values of σ for the three metrics used in the benchmark (accuracy, F1-score, and balanced accuracy). If desired, one can run a grid search on the following sigmas values and choose the best sigma according to the heuristic metric: $[0.2, 0.25, 0.3, 0.35]$. The heuristic was defined in supplementary subsection A.1.9. The supplementary subsection A.1.5 summarises the results of Scyan over these different runs.

- Non-existing populations are added in the table (between 0 and 5 included). For each added population, we randomly choose an existing population from the table and switch between one and three marker expressions. Note that this process might create actual populations that are not yet on this dataset. For instance, switching the CD8 expression for a T CD4 population would make a double-positive-T. Thus, when adding five populations, it is likely that some added populations are more relevant to characterize some existing cells. The supplementary Figure A.4 summarises the results of Scyan over these different runs. It shows that only the F1-score is affected (which is expected since the F1-score is very sensitive to minor changes for rare populations). And still the F1-score after adding five populations is still higher than the best related model without adding non-existing populations (F1-score=0.6).

A.1.6 . Details on the knowledge table to be provided

Scyan requires a table, a.k.a. biological prior table or knowledge table, to describe the populations to be annotated. The table is of size P x M, where P is the number of populations and M the number of markers. For a given population z and a marker m, the value at the row z and the column m of the table has to be one of the following:

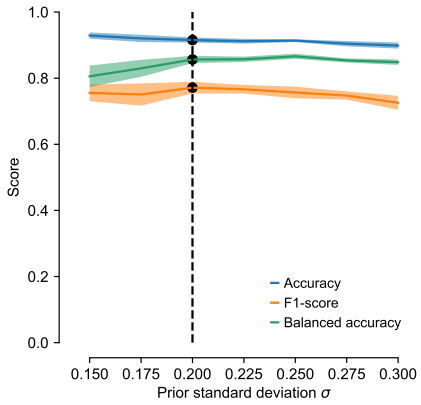

Figure A.3: Sensitivity to the main model hyperparameter σ on the POISED dataset. The dotted black line shows the σ chosen after hyperparameter optimization. Metrics standard deviation is computed over 5 runs.

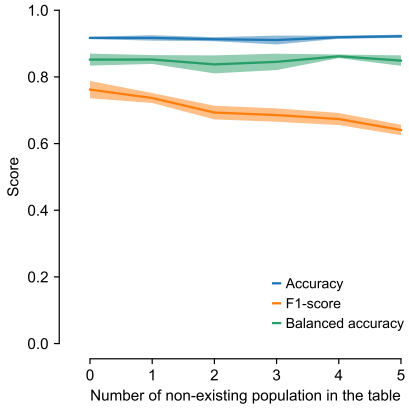

Figure A.4: Sensitivity to non-existing populations in the knowledge table on the POISED dataset. Metrics standard deviation is computed over 5 runs. The F1-score after adding 5 populations is still higher than the best related-model without adding non-existing populations (F1-score=0.6)

- 1 if the population z is known to express marker m
- -1 if the population z is known not to express marker m
- NA if we don't know anything about the expression of m on population z
- a value in $]-1, 1[$ if the expression is known, but not positive or negative. For instance, -0.5 can be chosen for a low expression, and 0 for a mid expression. It allows to better annotate complex populations, in particular population continuums (e.g., classical/intermediate/non-classical monocytes).

The design of the knowledge table is essential for the annotations. The literature can help its creation, but we also provide some advice to enhance the table:

- It is better to provide no knowledge than false information; therefore, the user should feel comfortable using "Not Applicable" for a marker when unsure. Besides, if needed, population discovery can be used to go deeper into this marker afterward.
- Note that the model interprets NA values by "any expression is possible". Thus, a population described with extensive use of NA values (e.g., above 90% of markers, with no discriminative marker provided) can be over-predicted. This is normal behaviour since few constraints are applied to this population.
- We enable the usage of intermediate expressions such as "mid" and "low" in the table. Yet, we advise using it only to differentiate two similar populations. Overusing

these intermediate expressions in the table will challenge the user to create the table properly while not improving the results.

- It is not required to use all the panel markers. If some markers are unimportant for the annotation, they can be removed from the knowledge table.

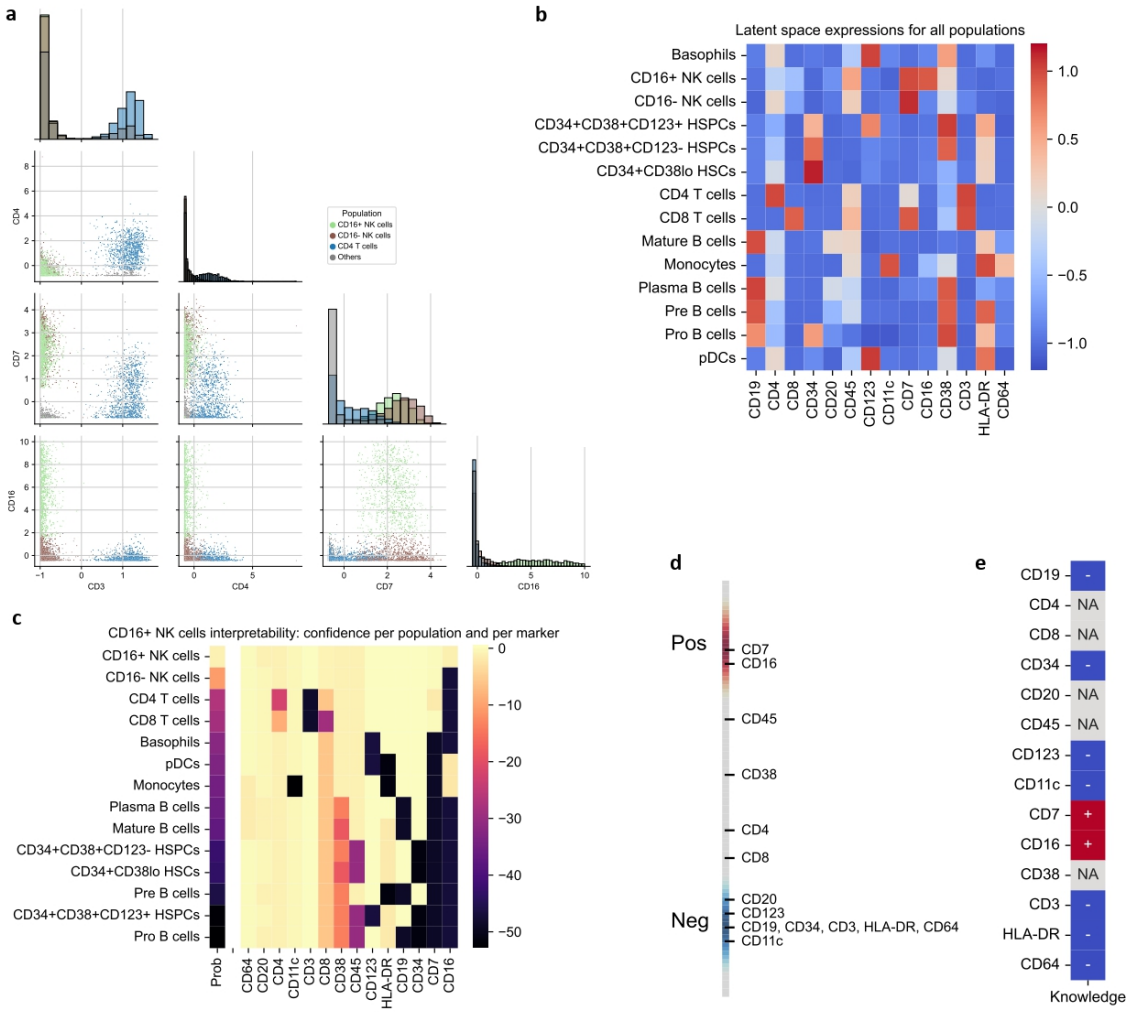

Figure A.5: ***Scyan visualization, interpretability, and discovery on AML. a****, Separation of the CD4 T cells and the two NK populations on multiple scatter plots using CD3, CD4, CD7, and CD16 (standardized marker expressions).* ***b****, Scyan latent space for all populations. The latent space comprises one value per marker whose typical range is [-1, 1]. The closer to -1, the more negative the marker expression, and the closer to 1, the more positive the marker expression.* ***c****, Understanding Scyan predictions for CD16+ NK cells by providing Scyan confidence (or probability) for each population, each of them decomposed per marker.* ***d****, Scyan latent space for CD16+ NK cells, in other words, their expressions for all the considered markers.* ***e****, Extract of the knowledge table concerning CD16+ NK cells. Some markers were known to be positive, others negative, and some marker expressions were unknown or not applicable (NA).*

A.1.7 . Benchmark in the presence of low batch effect

In Figure 3, we evaluated Scyan on batch effect correction on a dataset with a significative batch effect. Sometimes, the batch effect is low, but we still want to correct it. We evaluated Scyan against the same models, i.e., Cydar [Lun et al., 2017] / COMBAT [Johnson et al., 2007] / Harmony [Korsunsky et al., 2019] / SAUCIE [Amodio et al., 2019], but on POISED without batch effect amplification. This dataset has seven different biological batches. Figure A.6 shows that Cydar performs better on simple datasets (compared to the benchmark in Figure 3) but still lower than Scyan/Harmony. Again, SAUCIE is not preserving biological variability. Concerning Combat, its correction is lower than without correction, in this case (it could be explained by the fact that Combat's original use is for scRNAseq, not cytometry). Finally, Scyan and Harmony gave the best results, with Harmony having a better iLISI, while Scyan had a better cLISI. Overall, on this dataset with low batch effect correction, both Scyan and Harmony perform well. As shown in Figure 3, Scyan batch effect correction becomes superior to Harmony on more complex use cases.

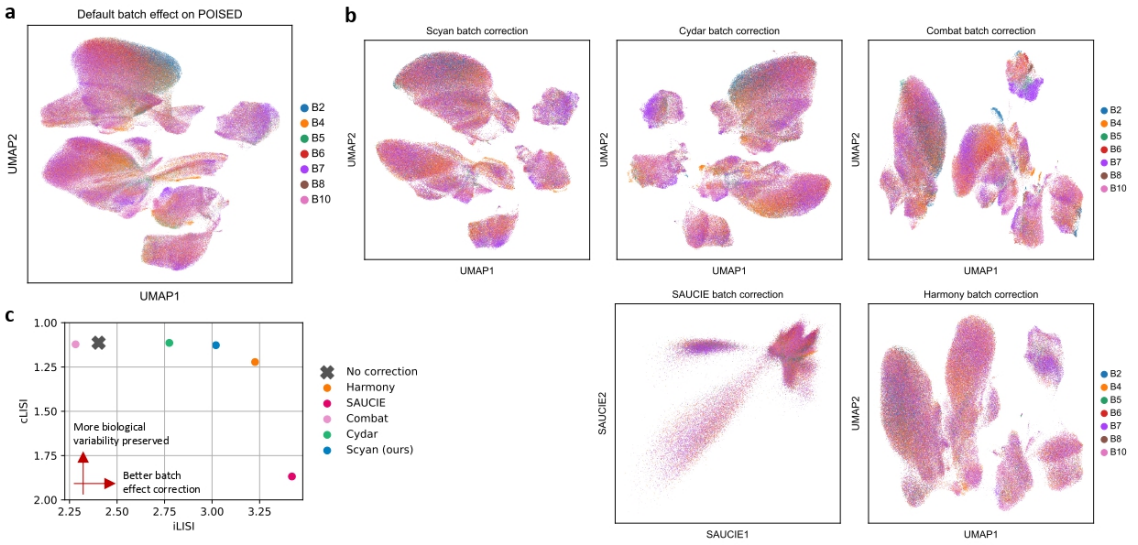

Figure A.6: ***Batch-effect correction on the POISED dataset without batch-effect amplification. a****, UMAP showing the 7 different batches (before batch effect correction). The batch effect is visible since different batches form separated clusters.* ***b****, Batch-effect correction of Scyan, Cydar, Combat, SAUCIE, and Harmony. A good batch effect correction can be observed by a superposition of all batch distributions.* ***c*** *Batch-effect correction metrics. A low cLISI (at the top of the figure) denotes good cell-type variability preservation, while a high iLISI (on the right of the figure) denotes better batch mixing.*

A.1.8 . Comments on supervised models runtime

Since LDA and CyAnno rely on manual gating, the total time needed for the annotation is highly dependent on the time required for the manual annotation. For complex datasets with many patients and a high batch effect, manual gating can take full days to complete a precise annotation and thus to be able to run the supervised models. Also, if a population was discovered afterwards, manual gating has to be modified to target

this new population. On the opposite, adding a new population to Scyan's table (basically, modifying one or two marker expressions) is all we need to re-run the algorithm and target this new population.

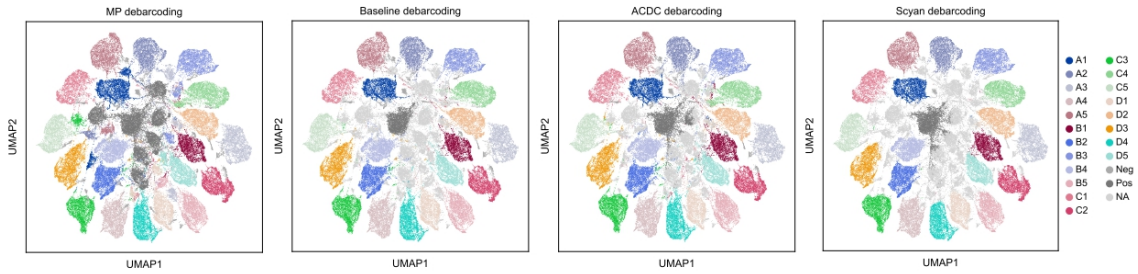

Figure A.7: ***UMAPs for the debarcoding task. From left to right: MP, Baseline, ACDC, Scyan.***

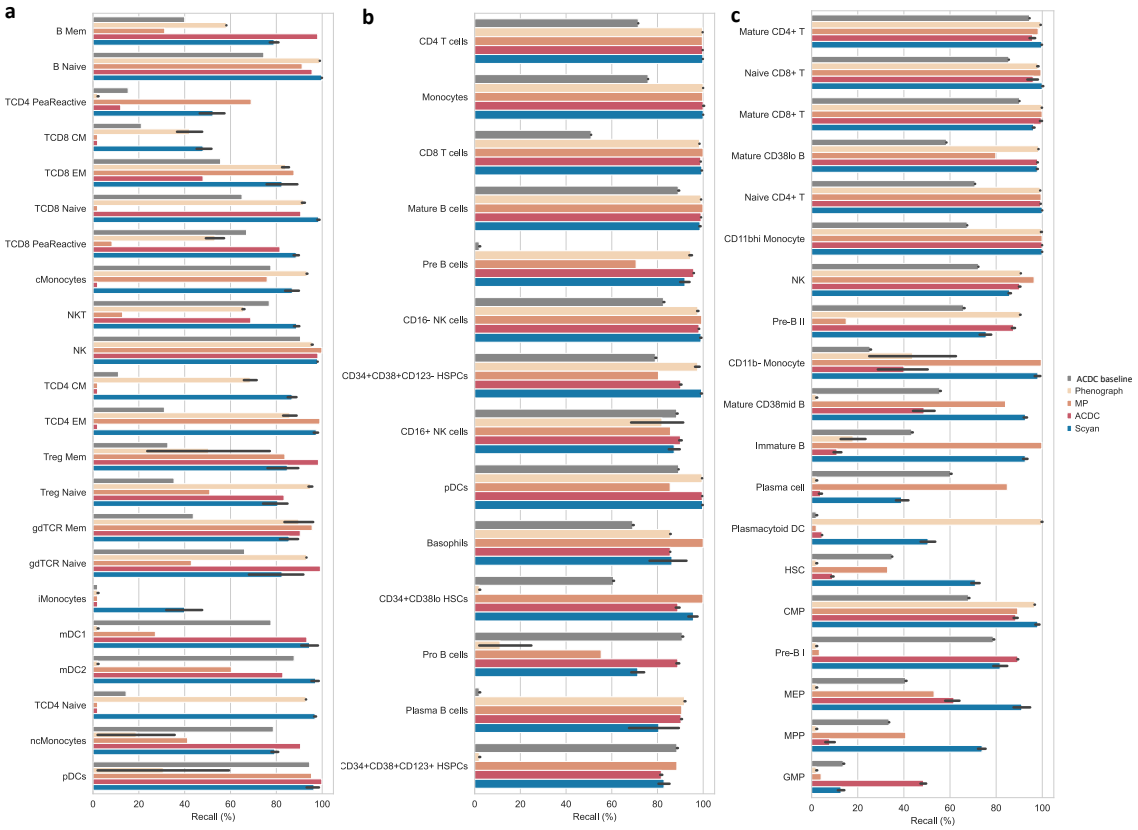

Figure A.8: ***Comparison of unsupervised models. The recall for each population is displayed a****, on POISED.* ***b****, on AML.* ***c****, on BMMC.*

A.1.9 . Model hyperparameter optimisation

One important issue in training deep learning models is fine-tuning their hyperparameters. Because our model is unsupervised, we cannot consider any supervised metric such as accuracy. We thus have to use an unsupervised metric that measures the annotation quality. For this reason, we defined a heuristic that combines (i) a clustering met-

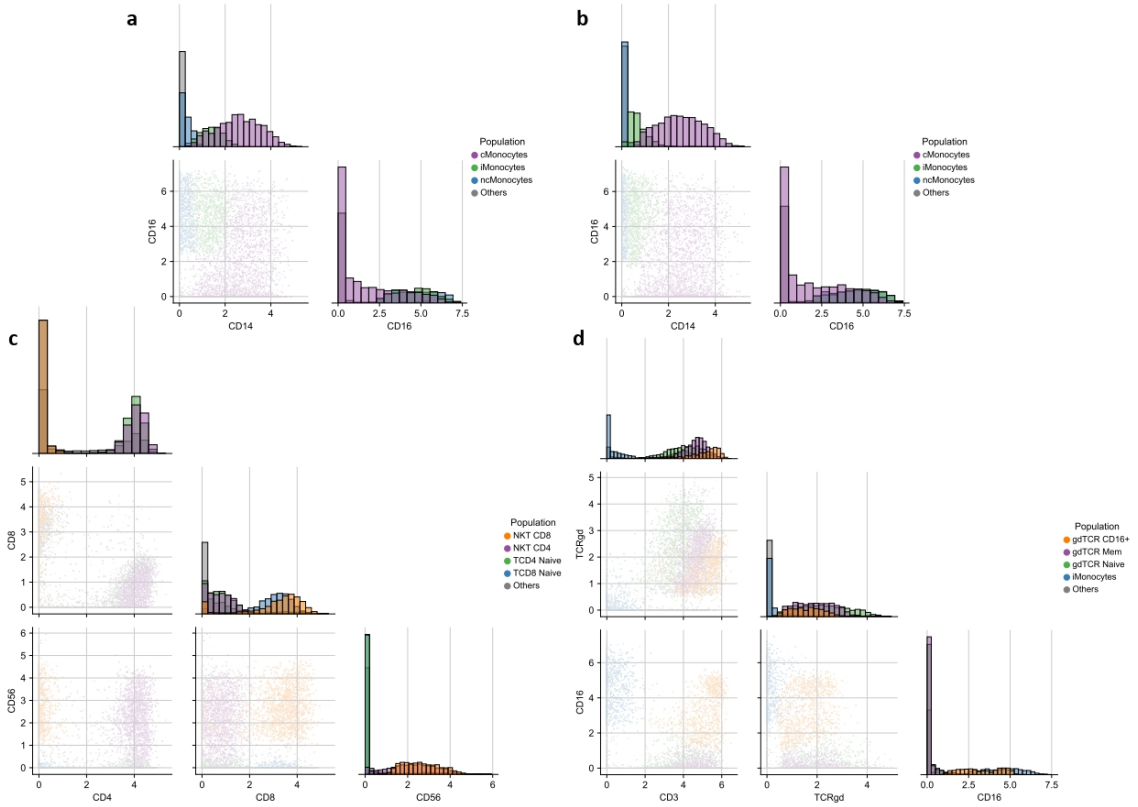

Figure A.9: ***Back-gating to check Scyan annotations a****, Scyan annotation among monocytes.* ***b****, Manual annotations among monocytes.* ***c****, NKT cells is a population that has been overpredicted by Scyan compared to manual gating. We check by back gating that they were properly annotated.* ***d****, gdTCR CD16+ is a new population discovered by Scyan. We show this population really exists.*

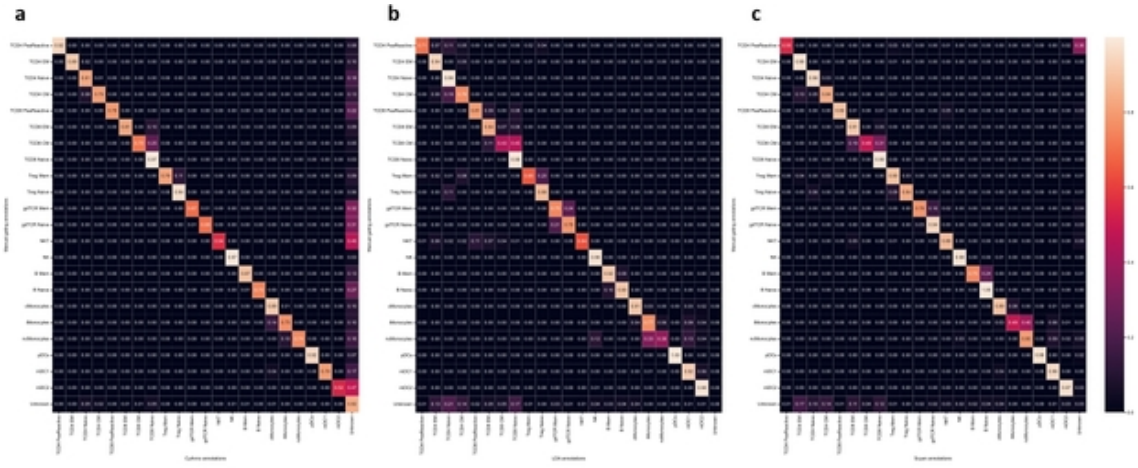

Figure A.10: ***Confusion matrix of annotations compared to manual gating.*** ***a****, CyAnno.* ***b****, LDA.* ***c****, Scyan.*

ric, the Davies-Bouldin Index [Davies and Bouldin, 1979] (DBI), to obtain well-separated clusters, (ii) a count of the missing populations to favor the presence of all populations among the predictions, (iii) a Dirichlet probability on population weights to favor popula-

tion diversity, and (iv) the iLISI score. Formally, let O be the number of populations that the model did not predict at all, and $\mathbf{X}, \mathbf{y}_{pred}$ the cells' expressions and predictions, respectively. Then, we define our heuristic to be minimised by $(O+1) \cdot DBI(\mathbf{X}, \mathbf{y}_{pred}) \cdot (-\sum_z log\ \pi_z)/iLISI(\mathbf{X}, \mathbf{y}_{pred})$. Note that $-\sum_z log\ \pi_z$ is proportional to the log Dirichlet probability of the learned population weights π. An advantage of using the DBI is that it is computationally more efficient than some clustering metrics, such as the silhouette score [Rousseeuw, 1987]. In particular, the DBI scales efficiently to large datasets. Also, if batch covariates were not provided, we simply remove the iLISI term in the heuristic definition. In practice, we use Optuna [Akiba et al., 2019] for hyperparameter optimization and their Tree-structured Parzen Estimator algorithm.

A.1.10 . Benchmark models and evaluation

We compared Scyan to three other knowledge-based models: ACDC [Lee et al., 2017], a baseline model (defined by the authors of ACDC), and MP [Ji et al., 2018]. Also, we compared our model to Phenograph [Levine et al., 2015], a clustering model. Note that the Phenograph does not predict labels itself. Thus, each cluster has to be assigned to a biological cell type. This is typically done by human experts, but for more objectivity, the clusters were named using known labels. Using these labels thus replaces the assignment of clusters to biological cell types by human experts and provides a way to compare Phenograph to the other approaches by making the assumption that a human expert would correctly annotate the clusters. We also compared Scyan to two supervised models (LDA [Abdelaal et al., 2019] and CyAnno [Kaushik et al., 2021]) on the POISED dataset. For a realistic scenario, the two models were trained on one batch and evaluated on the others (one run for each batch). Scyan, in contrast, is unsupervised and can be run directly on all cells. Finally, we benchmarked our model ability to correct batch effect to four models: Cydar [Lun et al., 2017], Combat [Johnson et al., 2007], SAUCIE [Amodio et al., 2019], and Harmony [Korsunsky et al., 2019]. We used the POISED dataset on which we had 7 biological batches, and we amplified the batch effect to complex the batch correction (see methods).
We used the implementation from Harmony [Korsunsky et al., 2019] to compute LISI scores. Formally, for one cell, the cLISI is the number of different cell types found in the cell's close neighborhood. Thus, a lower value indicates a better cell-type separation, and the best value is 1. Concerning the iLISI: for one cell, it corresponds to the number of different batches found in the cell's close neighborhood. Thus, a higher value indicates a better batch correction, and the best value is the number of different batches. For the batch correction benchmark, evaluation was run on the cell-type related markers only: CD19, CD20, CD3, CD4, CD8, TCRgd, CD16, CD56, CD25, CD127, CD45RA, CCR7, HLA.DR, CD14, CD11c, CD123, CD27, CD69, CD40L. The UMAPs were plotted using Scanpy [Wolf et al., 2018].

A.1.11 . Data preprocessing

To compare our model with the other methods, we used a similar data preprocessing and the same knowledge tables as MP [Ji et al., 2018] and ACDC [Lee et al., 2017] for the AML, BMMC, and debarcoding datasets. On POISED, due to the impossibility of having non-binary values in the input tables of MP and ACDC, intermediate Monocytes had to be removed. Also, ACDC and MP used $x \mapsto asinh(\frac{x-1}{5})$ to preprocess marker expressions, while we used $x \mapsto asinh(\frac{x}{5})$. Note that we also standardized the data (required to run Scyan). Concerning the debarcoding task, we used the logicle transformation [Parks et al., 2006] to preprocess marker expressions and then standardization.

A.1.12 . GPU acceleration

If desired, any user can use GPUs to make Scyan faster. For that, we used PyTorch Lightning (https://lightning.ai/docs/pytorch/latest/) which detects the user hardware and automatically provides the right hardware options to PyTorch during training. In most use cases (including large datasets), CPUs are sufficient and fast enough. By default, PyTorch lightning automatically uses GPUs if available, but this behavior can be changed via the "accelerator" kwargs argument.

A.2 . Sopa supplementals

A.2.1 . Supplementary Notes

Choice of SpatialData as a data structure

SpatialData[Marconato et al., 2024] is a data structure developed in Python that aims to store spatial-related objects. It also provides transformations between coordinate systems (for instance, between microns and pixels), lazy representation for large images with Daskand Xarray[Hoyer and Hamman, 2017], transcripts stored as Daskdataframes, and cells polygons stored as GeoPandas polygons. The general structure of this data, the community support, and integration with the scverse[Virshup et al., 2023] ecosystem make it a reliable tool to store spatial omics objects in Sopa. Notably, the usage of Python is appreciated since most recent models in spatial omics are gradually moving to Python for package development[Moses and Pachter, 2022].

Integration with the scverse ecosystem

The scverse[Virshup et al., 2023] ecosystem is a Python-based suite of fundamental tools for single-cell omics data analysis. This includes the data structures SpatialData[Marconato et al., 2024] that we use for Sopa, as well as Scanpy[Wolf et al., 2018], which covers a wide range of use cases in single-cell analysis. Also, still in the scverse ecosystem, Squidpy[Palla et al., 2022] is a Python library for the analysis of spatial single-cell data such as spatial neighbourhood analysis or ligand-receptor interaction analysis. Since Squidpy supports SpatialData, Sopa also naturally integrates with Squidpy. Indeed, the pipeline output is a SpatialData object, and Squidpy can operate on this, enabling all Squidpy functionalities to be leveraged after Sopa or inside the pipeline. Squidpy is complementary to Sopa since it operates on processed spatial omics, contrary to Sopa, which analyses raw data. Also, the spatial statistics tools available in Sopa do not exist in Squidpy. Thus, these packages have non-overlapping and complementary functionalities.

Limitation of the proprietary visualization software

All visualizers are exclusive to their data structure and require an investment of time from the users to learn their proprietary software. Besides this, some of the software comes only with the purchased machine and requires a license key for use. This limits the number of users who have a collaborative engagement and are not in possession of the machine. Data analysis from the MERSCOPE comes with a dedicated visualizer, called the "Merscope Visualizer". Its input is proprietary *".vzg"* files, a non-open format. While VPT offers the possibility to update it, a new *vzg* cannot be recreated for another type of technology. In addition, the update of this file requires performing again all required operations, even for minor changes, because everything is included in one file. Therefore, minor modifications still imply a significant runtime to be updated in the visualizer. Concerning CosMX data, they offer an online suite of tools, called AtoMx, which is cloud-based only, limiting the accessibility, especially for users wanting to use their own

high-processing-cluster. Concerning the visualizer of the PhenoCycler and MACSima, they are specific to multiplex imaging, i.e. no transcript can be shown. Contrary to the other visualizers, Xenium Explorer can be both (i) downloaded freely and (ii) supports open file formats. This makes it a reliable choice for conversion from SpatialData. Also, it supports missing data, i.e. it will not crash when reading multiplex imaging data (from which no transcripts are available).

Visualization with the Xenium Explorer

After using Sopa, the files required by the Xenium Explorer are created. In particular, a file called "experiment.xenium" can be opened in the Xenium Explorer. The later software is freely available for both Windows and MacOS. Sopa has been tested on versions 1.2 and 1.3 of the Xenium Explorer. We show two examples of visualization in supplementary Figure A.12 (Xenium dataset, 10X) and supplementary Figure A.13 (MERSCOPE dataset, Vizgen).

Image alignment with the Xenium Explorer

One challenge for spatial transcriptomics can be to align images from different technologies when they are run on the same sample. Most of the time, a simple affine transformation is enough to align them. Since Sopa create outputs in the Xenium Explorer, it is possible to use the alignment tool available on the software. It consists of applying some mirroring transformations, rotations, and alignment based on user-defined reference points. Then, the transformation matrix can be saved via the visualizer, which will create a "matrix.csv" transformation file. Afterwards, we can use this transformation matrix to align the new image on our SpatialData object and perform any operation available in Sopa. This can be done via the Sopa CLI, by specifying `sopa explorer add-aligned <sdata_path> <image_path> <matrix_path>`. Typically, when adding an IF image, we can compute the mean channel intensity for all cells and for all channels.

Thresholds for conflict resolution

When two cell boundaries are overlapping, we compute their intersection-over-min-area (IOMA) to determine whether or not to merge the cell boundaries. In this section, we define thresholds of IOMA scores to be considered as good conflict resolution in Figure c/d/e. The upper bound is defined based on the value such that two cells randomly overlapping have a 0.025 probability of having an IOMA higher than this upper bound. Specifically, let $R \in \mathbb{R}^+$ a cell radius and two cells of radius R whose center are C_1 and $C_2 \in \mathbb{R}^2$, respectively. Without loss of generability, we consider that C_1 is centered in the 2D plane, that is $C_1 = (0, 0)$. Since the two cells are overlapping, we assume that C_2 is a random variable uniformly distributed on the circle of radius $2R$, that is $C_2 \sim \mathcal{U}(\{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 2R\})$. Now, let D the random variable representing the distance between the two cells, i.e. $D = ||C_1||_2$. Finally, the quantile Q_p is defined as by the equation $P(f(D) \leq Q_p) = p$, where $p \in [0, 1]$ and f is the function that computes

the IOMA score, that is, $f(D) = \frac{2}{\pi}\left(\arccos(\frac{D}{2R}) - \frac{D}{2R}\sqrt{1-\frac{D^2}{4R^2}}\right)$. Since f decreases with respect to D, we have $P(f(D) \leq Q_p) = P(D \geq f^{-1}(Q_p)) = 1 - \frac{f^{-1}(Q_p)^2}{4R^2}$. This leads to $Q_p = \frac{2}{\pi}\left(\arccos(\sqrt{1-p}) - \sqrt{p(1-p)}\right)$, and, in particular, $Q_{0.975} \approx 0.7995$. Concerning the lower bound, the computation is different, because of the nature of such overlaps. Indeed, since the images have a certain thickness, two non-touching cells can appear as overlapping when projected on a 2D plane. We want to compute the mean IOMA of two non-touching cells that are overlapping when projected on the (x,y) plane. For that, we define L, the thickness of a slide. Let $Z_1, Z_2 \sim \mathcal{U}([-\frac{L}{2}, \frac{L}{2}])$ the random variables representing the position of the center of two cells on the z-axis. Again, we suppose that one cell is centered on $X_1 = 0$, while X_2 is assumed to be uniformly distributed while following these two conditions: (i) $X_2 \leq 2R$, in order for the two cells to overlap on the (x,y) plane, and (ii) $X_2^2 + (Z_1 - Z_2)^2 \geq 4R^2$ so that the two cells are not touching each other on the (x,y,z) space. Note that, when projected on the (x,y) plane, the distance between the two cells is X_2. Computational simulations gives $\mathbb{E}(f(X_2)) = 0.07$, which is used as our lower bound (see supplementary Figure A.18 for the full distribution).

Synthetic dataset generation

In order to demonstrate Sopa's efficiency on multiple dataset sizes, we created synthetic datasets. Let L be the width of the image, and d be the cell density in the image. An evenly distributed grid of size $(L\sqrt{d}, L\sqrt{d})$ is generated, each vertex corresponding to a cell location. We apply a Gaussian noise of standard-deviation $\frac{1}{2\sqrt{d}}$ on these cell locations to have a more natural distribution of cells. Images are generated by applying a Gaussian blur of standard deviation $\frac{1}{2\sqrt{d}}$ on the pixels at the location of the cell vertices, and 100 transcripts per cell are generated via a 2D Gaussian distribution of the same standard deviation.

Annotation of example datasets

Dataset annotation followed the procedure outlined in the main manuscript. Automatic annotation utilized the following references: Liver dataset (https://www.immunesinglecell.org/atlas/liver) and Pancreas dataset (https://www.immunesinglecell.org/atlas/pancreas). Initial global annotation involved combining major cell populations, followed by refinement using Leiden clustering[Traag et al., 2019]. Subsequent in-depth analysis employed manual annotation with Leiden clustering. For MACSima and PhenoCycler datasets, exclusion criteria involved DAPI, boundary staining, and low-quality proteins to enhance resolution. Manual clustering with Leiden was then applied for population annotation. Niche calculations were performed using STAGATE[Dong and Zhang, 2022]. Niches were annotated based on cell type abundance and tissue structure, validated by a pathologist.

Comments on Baysor performances on patches

On Figure 3c/f, Baysor[Petukhov et al., 2022] had a better DE score when running on the patches than without. Actually, running Baysor on patches simplifies the complexity of each run since it will focus on a subset of cell types, which could increase Baysor specificity. For instance, for a patch that is specific to the stroma, Baysor may have an enhanced resolution compared to a run on the full image (which contains a broader range of cell types). This may explain why it has more power for DE, and it is opening up potential investigations beyond this paper to confirm this explanation.

	MERSCOPE dataset	Xenium dataset	Phenocycler dataset	MACSima dataset
Matching cells without merge	11960 (98.1%)	69063 (97.5%)	892845 (99.5%)	42230 (99.1%)
Matching cells with merge	227 (1.86%)	1738 (2.45%)	4753 (0.529%)	381 (0.894%)
Non-matching cells	1 (0.0082%)	7 (0.00989%)	54 (0.00602%)	4 (0.00939%)

Table A.3: **Detailed status of cells during conflict resolution for the crops of size (16000x16000) of the original image.** Each resulting cell is separated into three categories. First, cells for which no conflict was found. Secondly, cells for which conflict was resolved but whose resulting cell corresponds to one unique cell in the segmentation without patches. Third, cells for which conflict was resolved but does not correspond to one unique cell in the segmentation without patches. Both raw numbers and percentages are provided.

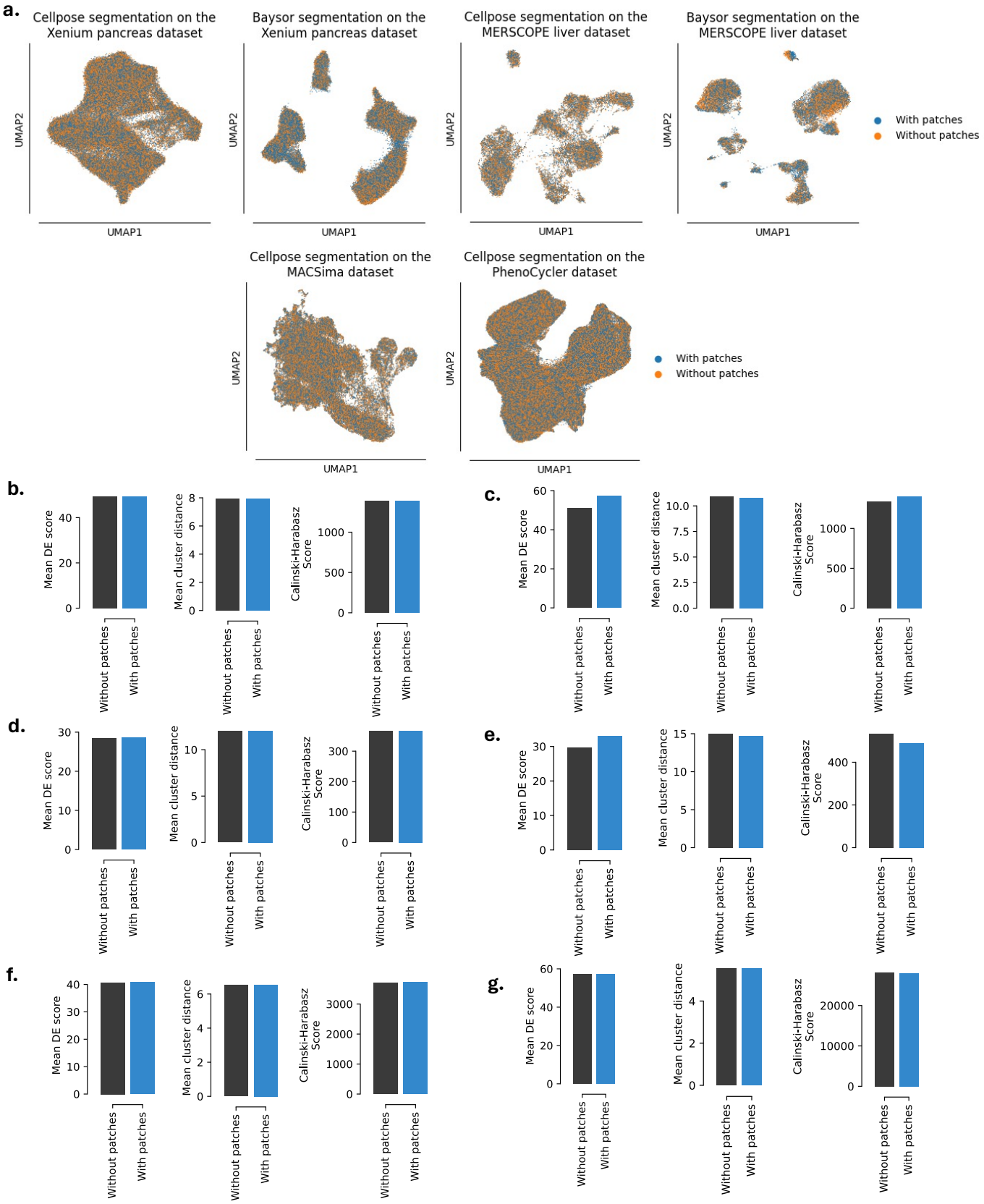

Figure A.11: **Impact of patched-based segmentation over the data quality, based on crops of size (16000x16000) of the original image. a.** UMAPs comparing the representation of the cells obtained while running segmentation over the whole image and using patches (as in Sopa). This was tested over multiple datasets, for Cellpose (for all datasets) and Baysor (for spatial transcriptomics datasets). **b.** Comparison of segmentation quality metrics for Cellpose run on the Xenium dataset. **c.** Comparison of segmentation quality metrics for Baysor run on the Xenium dataset. **d.** Comparison of segmentation quality metrics for Cellpose run on the MERSCOPE dataset. **e.** Comparison of segmentation quality metrics for Baysor run on the MERSCOPE dataset. **f.** Comparison of segmentation quality metrics for Cellpose run on the MACSima dataset. **g.** Comparison of segmentation quality metrics for Cellpose run on the PhenoCycler.

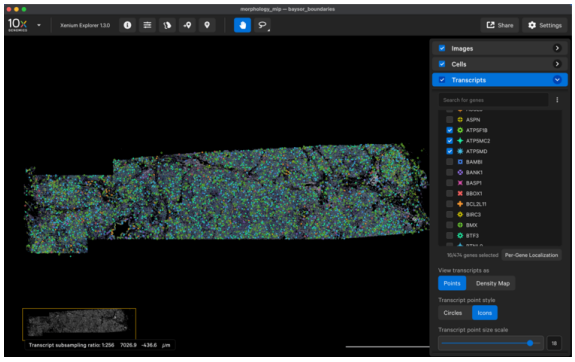

Figure A.12: Xenium human pancreatic cancer dataset (10X Genomics) open in the Xenium Explorer. The transcript panel is shown, with a few genes selected. Cells are coloured by a colour gradient representing transcript count.

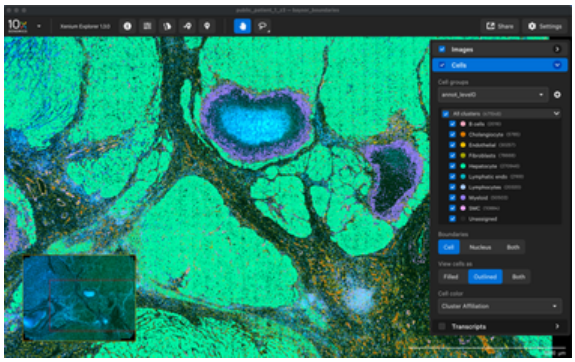

Figure A.13: MERSCOPE human liver hepatocellular carcinoma dataset (Vizgen) open in the Xenium Explorer. The cell panel is shown, and the "annot_level0" category is displayed. Colors correspond to a cell-type.

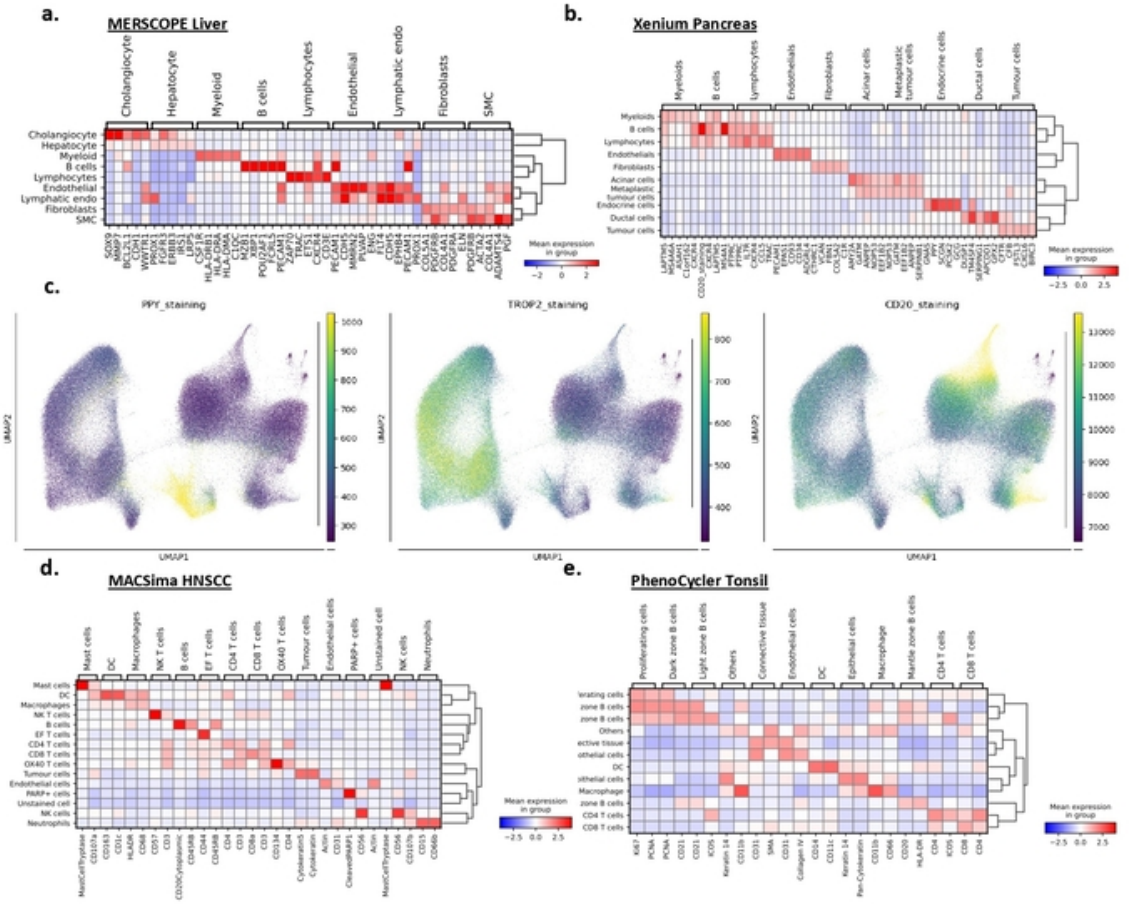

Figure A.14: ***Visual validation of the annotations.*** ***a.*** *Heatmap of genes expression per population on the MERSCOPE human liver hepatocellular carcinoma dataset.* ***b.*** *Heatmap of genes expression per population on the Xenium pancreas dataset.* ***c.*** *Protein staining per cell on the Xenium human pancreatic cancer dataset after aligning the staining image to the original Xenium image.* ***d.*** *Heatmap of protein expression per population on the MACSima HNSCC dataset.* ***e.*** *Heatmap of protein expression per population on the Phenocycler tonsil dataset.*

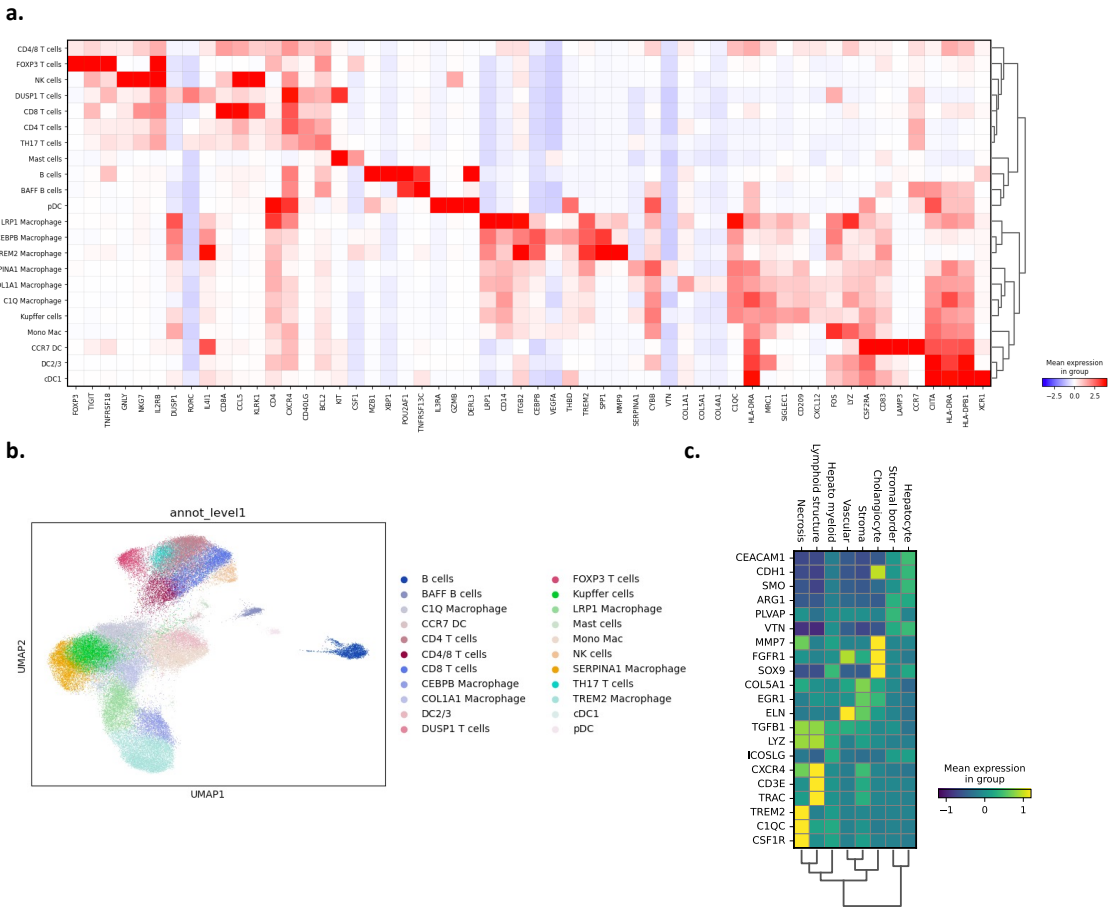

Figure A.15: ***Annotation of the immune cells of the MERSCOPE human liver hepatocellular carcinoma dataset and niche differential gene expressions (DEGs).*** ***a.*** *Heatmap of DEGs per immune population on the MERSCOPE liver dataset.* ***b.*** *UMAP of immune cells of the MERSCOPE liver dataset* ***c.*** *Heatmap of DEGs per niche of the MERSCOPE liver dataset.*

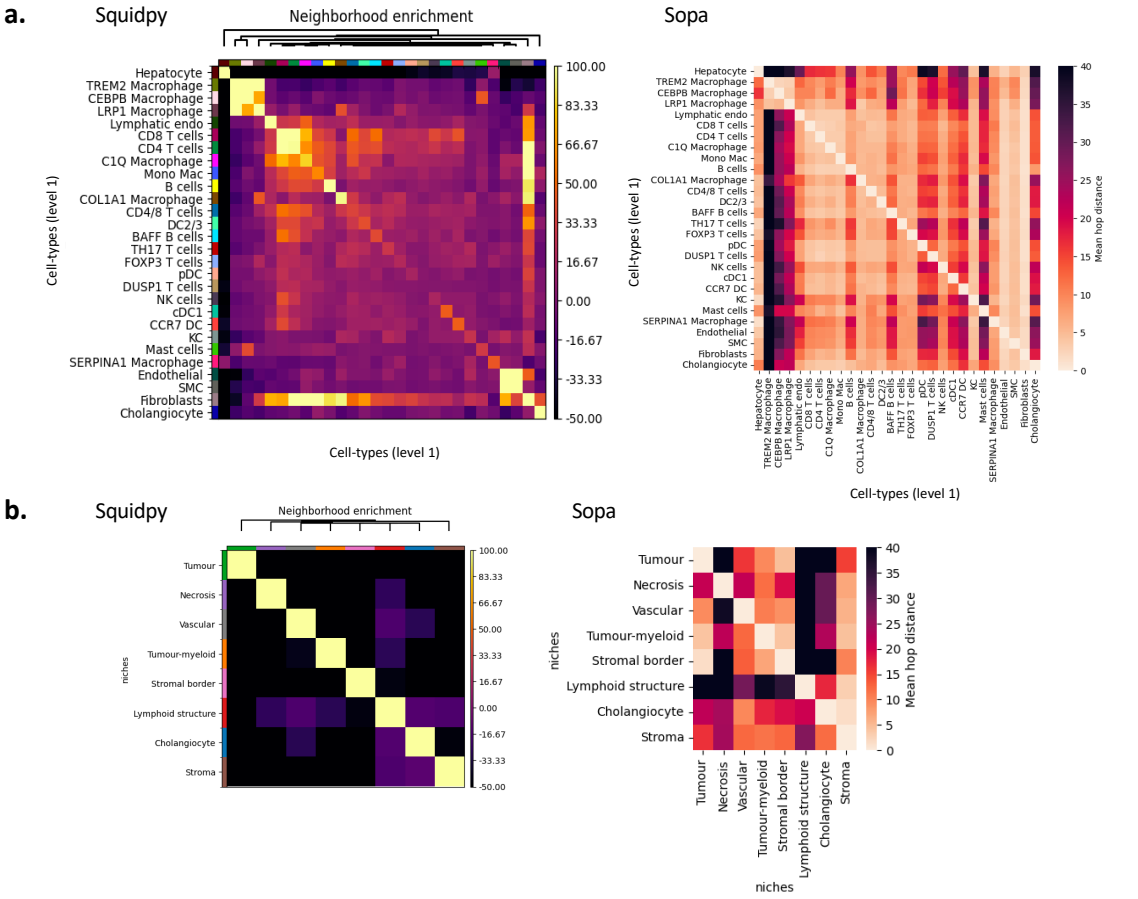

Figure A.16: ***Comparison between Squidpy and Sopa post-processing analyses a.*** *Comparison of Squidpy neighbourhood enrichment (cell to cell) and Sopa cell to cell average hop distance. While the neighbourhood enrichment is symmetric, the distances are not. In terms of insights, this asymmetry can, for instance, show that TREM2 macrophages are relatively close to the Hepatocytes, while the Hepatocytes are generally far from the TREM2 Macrophages. To prevent confusion while reading this heatmap, we precise that one row corresponds to the distances from the cell type of the row index to all other cell types.* ***b.*** *Comparison of Squidpy neighbourhood enrichment (niche to niche) and Sopa niche to niche average hop distance. Since niches are more global structures, their neighbourhood usually includes only the same niche (left), while the distances can capture more global organizations and informat*

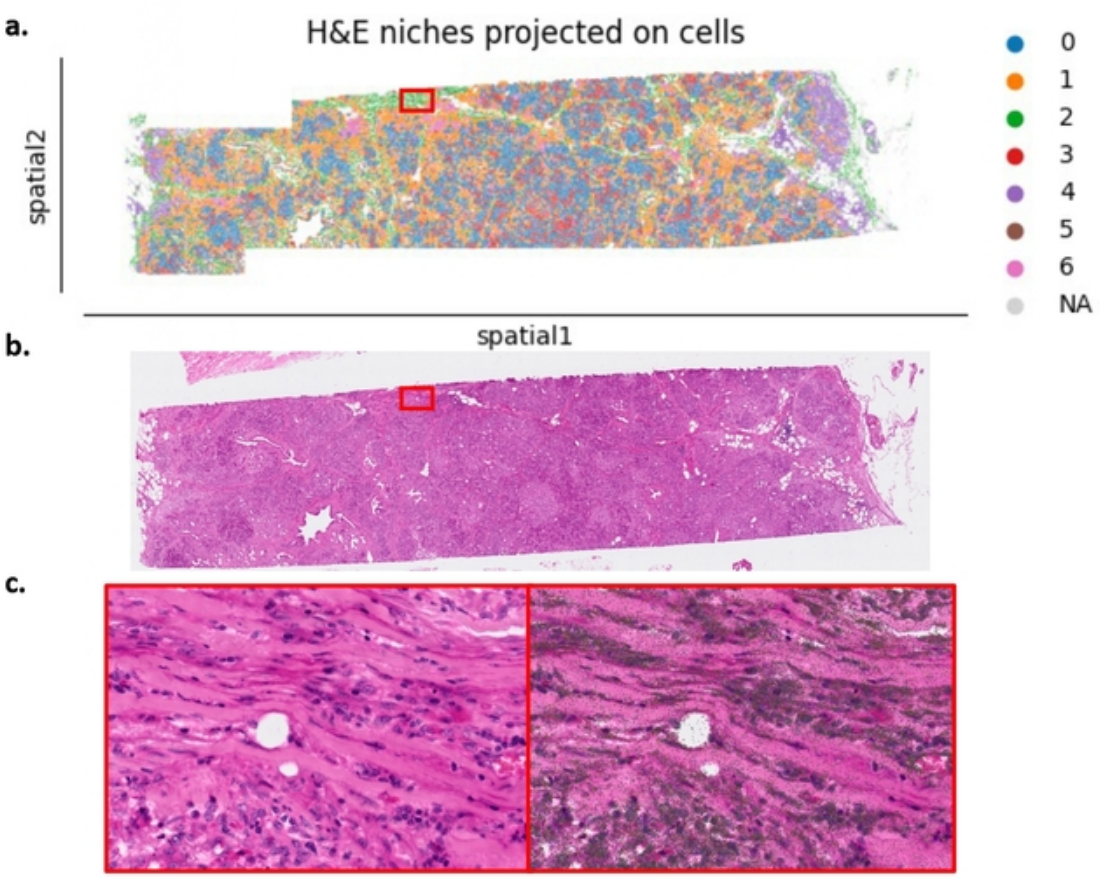

Figure A.17: ***Zoom on Xenium human pancreatic cancer dataset a.*** *H&E clusters of patch-level embeddings based on a pre-trained computer vision model (denoted as H&E niches), red box highlighting the image in c.* ***b.*** *H&E image of human pancreatic cancer dataset (10X Genomics dataset), red box highlighting the image in c.* ***c.*** *Left, zoom of H&E image of human pancreatic cancer dataset (10X Genomics dataset). Right, zoom of H&E image with all transcript overlay of human pancreatic cancer dataset (10X Genomics).*

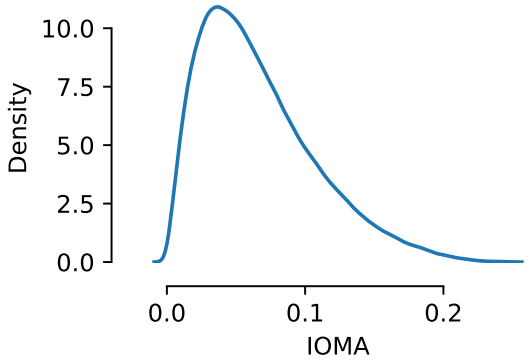

Figure A.18: ***Distribution of IOMA when non-touching 3D cells are projected on a 2D plane.*** *This computation has been performed by simulating random non-touching 3D cells.*

A.3 . Novae supplementals

Sinkhorn-Knopp algorithm

Novae relies on SwAV [Caron et al., 2020], which is itself based on an optimal transport [Peyré and Cuturi, 2020] problem that is solved via the Sinkhorn-Knopp algorithm. Its pseudo-code is detailed below:

Algorithm 1 Sinkhorn-Knopp

Require: $S \in \mathbb{R}^{B \times K}, \quad \epsilon > 0, \quad n\_iter > 0$
  $Q \leftarrow exp(scores/\epsilon)$ ▷ Element-wise exponential
  $Q \leftarrow Q/Q.sum()$
  $N \leftarrow n$
  for $iter \in \{1, \dots, n\_iter\}$ **do**
    $Q \leftarrow Q/Q.sum(dim = 0)/K$
    $Q \leftarrow Q/Q.sum(dim = 1)/B$
  end for
  return Q

After running Sinkhorn-Knopp, all rows of Q sum to $1/K$, and all columns sums to $1/B$. In practice, we choose $\epsilon = 0.05$, and $n\_iter = 3$. Studies [Caron et al., 2020] show that 3 iterations are sufficient for performances while being also time efficient. Note that these are the same values that the ones used in the SwAV [Caron et al., 2020] paper.

Integration within the community ecosystem

Novae uses AnnData [Virshup et al., 2021] as an input, one of the main data structures of the scverse [Virshup et al., 2023] ecosystem. This input is, therefore, widely used by the Python community, and packages exist for interoperability with R [Virshup et al., 2023]. The AnnData object can be obtained by reading the raw files with the SpatialData [Marconato et al., 2024] library. Alternatively, it is also possible to preprocess the raw files to obtain the AnnData object, for instance, using Sopa [Blampey et al., 2024b]. Overall, Novae is built on top of strong tools that are widely adopted by the community. It makes it easier to use Novae, and interoperate with existing packages.

Generalization across technologies

The results of Novae are shown on spatial technologies with single-cell resolution (MERSCOPE, Xenium, CosMX), but Novae can also be used on NGS-based technologies such as the Visium machine or Visium HD. In that case, we can use Novae in a similar manner, except that we will update the subgraph sampling parameters. Typically, we can set $n_{local} = 1$ or $n_{local} = 2$, as the distance between spots can be higher than the distance between cells. Similarly, we can set $n_{view} = 1$ or $n_{view} = 2$. Indeed, for Visium data, two neighbor spots are 100 microns away from each other, meaning that setting $n_{local} \geq 3$ would aggregate information from cells more than 300 microns away.

Graph lazy loading

For memory efficiency, we don't store the full PyTorch Geometric dataset in memory. Instead, for each mini-batch of B cells, we generate the corresponding subgraphs on the fly inside the PyTorch dataset based on the sparse Delaunay graph representation. This avoids memory errors on large datasets. This behavior is enabled by default on large datasets. For more flexibility, it is also possible to enable it for all dataset sizes, or totally disable it.

Synthetic dataset generation

We generate cell locations as the vertices of the grid of step $\Delta > 0$, cropped to fit inside the circle of radius R centered at the origin $(0, 0)$. Then, we artificially assign each cell to the domain $\lfloor \frac{\sqrt{x^2+y^2}K}{R} \rfloor$, where $(x, y) \in \mathbb{R}^2$ are the cell coordinates, and K is the number of desired domains. For each domain $k \leq K$, we sample $d_k \sim \text{exponential}(\lambda_{domain})^G$, where G is a number of genes, and λ_{domain} a domain-specificity parameter. Similarly, we sample $s_i \sim \text{exponential}(\lambda_{slide})^G$ for each slide i among S slides, and also $p \sim \text{exponential}(\lambda_{panel})^G$. Then, for a cell of domain k and slide i, we sample its gene expression as $x_{generated} \sim \text{exponential}(d_k + s_i + p)^G$. Therefore, λ_{domain} represents the domain specificity, and λ_{slide} is the slide specificity. This allows the generation of more or less complex cases depending on the choice of these parameters. Practically, we generated this dataset for $S = 5$ slides, a radius of $R = 6000$ microns, a step $\Delta = 20$ microns, a number of $K = 7$ domains, $n_{slides} = 5$ The toy dataset used in this manuscript is available on Hugging Face at `https://huggingface.co/datasets/MICS-Lab/novae/tree/main` and can be directly downloaded via the Novae package API.

Model monitoring and validation

For internal monitoring (i.e., before releasing the model), we used Weight and Biases, which enable tracking metrics about models and experiments, saving model artifacts, among other functionalities. To avoid validation on the Novae database, we used private data to compute validation metrics on all models. The choice of the best models was, therefore, based on this external validation dataset and not on the main Novae database.

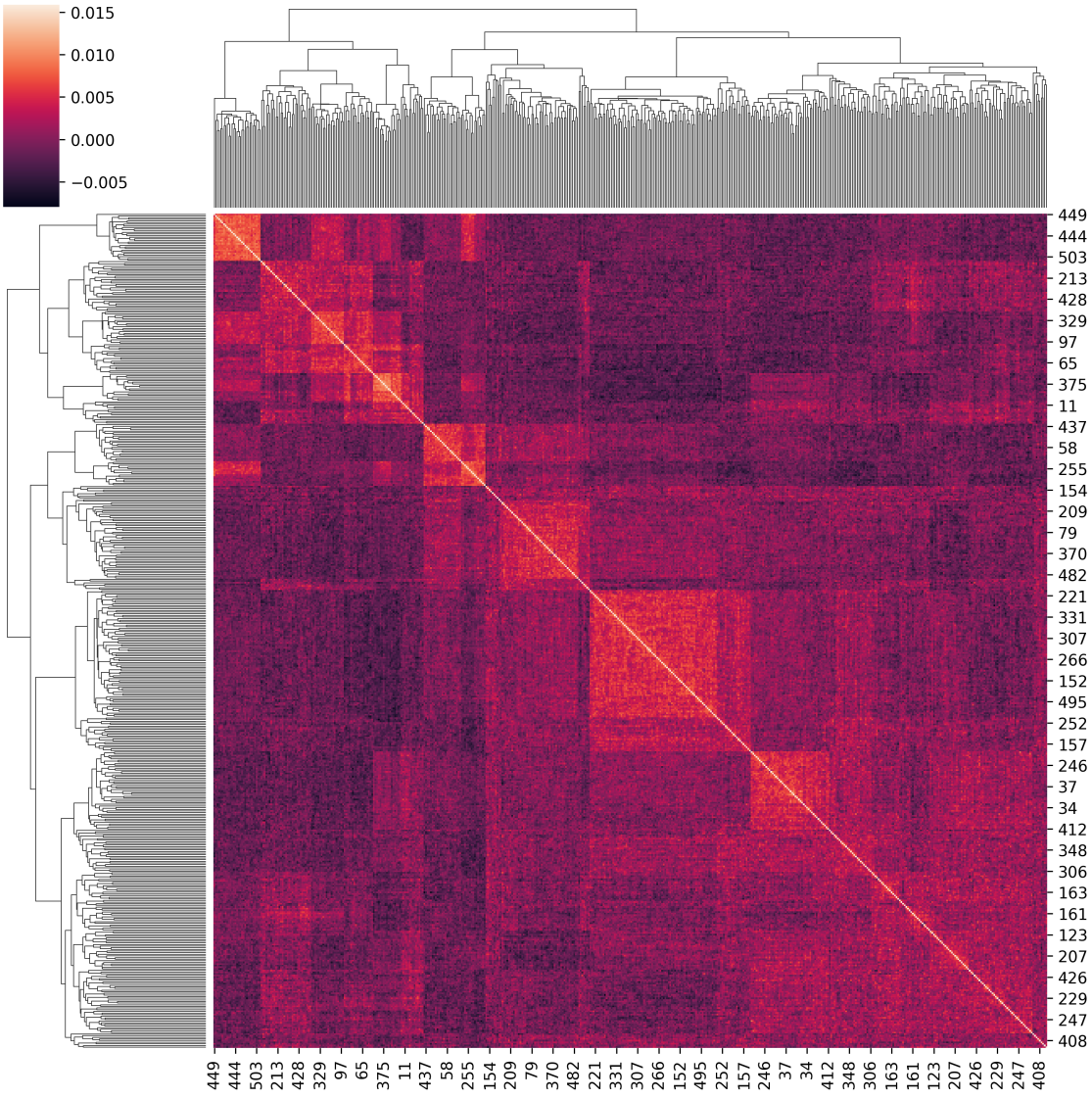

Figure A.19: Covariance of the prototypes of the Novae human run. A total of $K = 512$ prototypes are shown. The colors show the variance between each pair of prototypes.

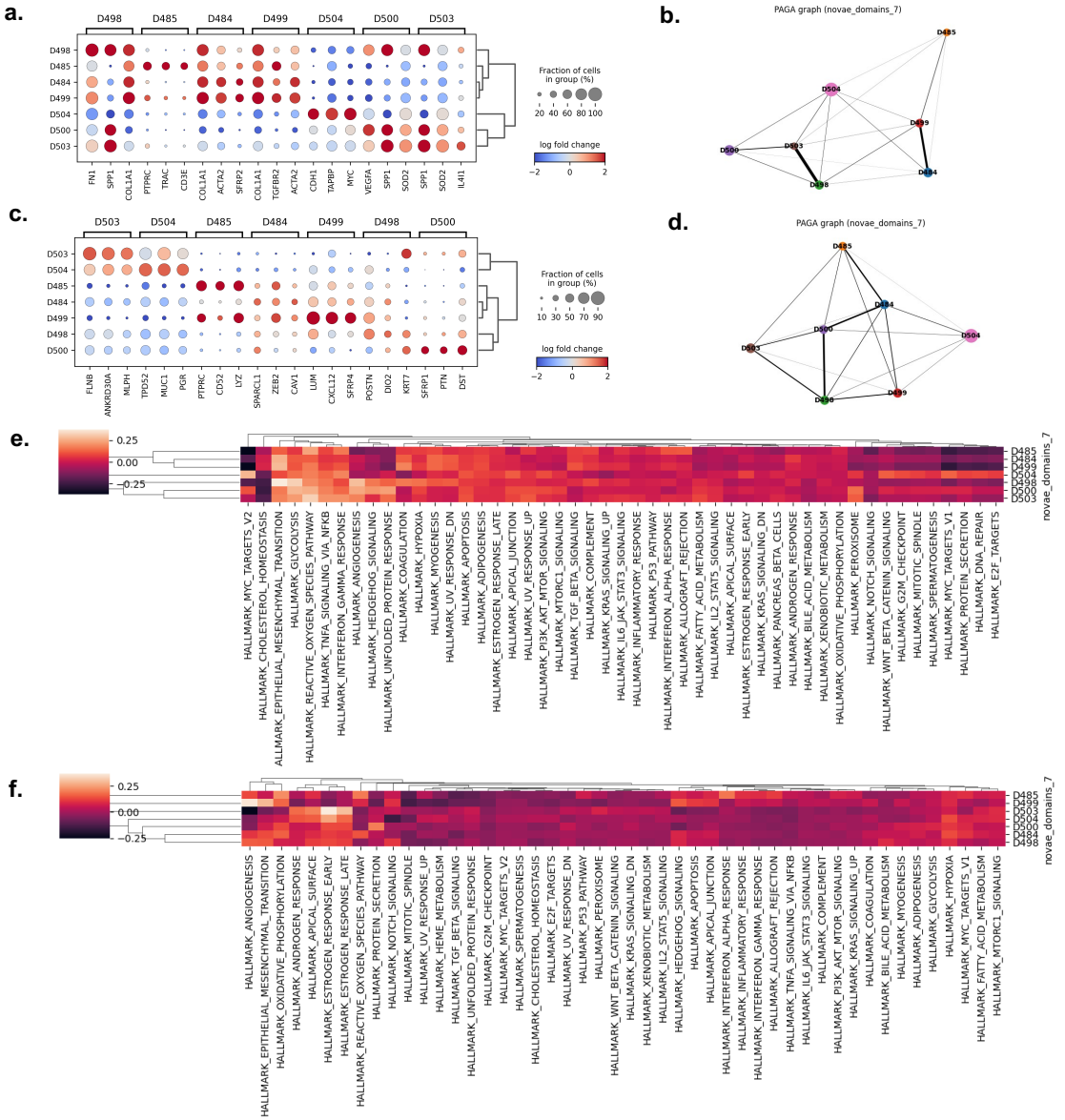

Figure A.20: ***Biological details on the breast dataset.*** ***a.*** *Top spatially variable genes over the MERSCOPE sample.* ***b.*** *PAGA graph over the MERSCOPE sample.* ***c.*** *Top spatially variable genes over the Xenium sample.* ***d.*** *PAGA graph over the Xenium sample.* ***e.*** *Hallmark pathway scores over the different domains of the MERSCOPE sample.* ***f.*** *Hallmark pathway scores over the different domains of the Xenium sample.*

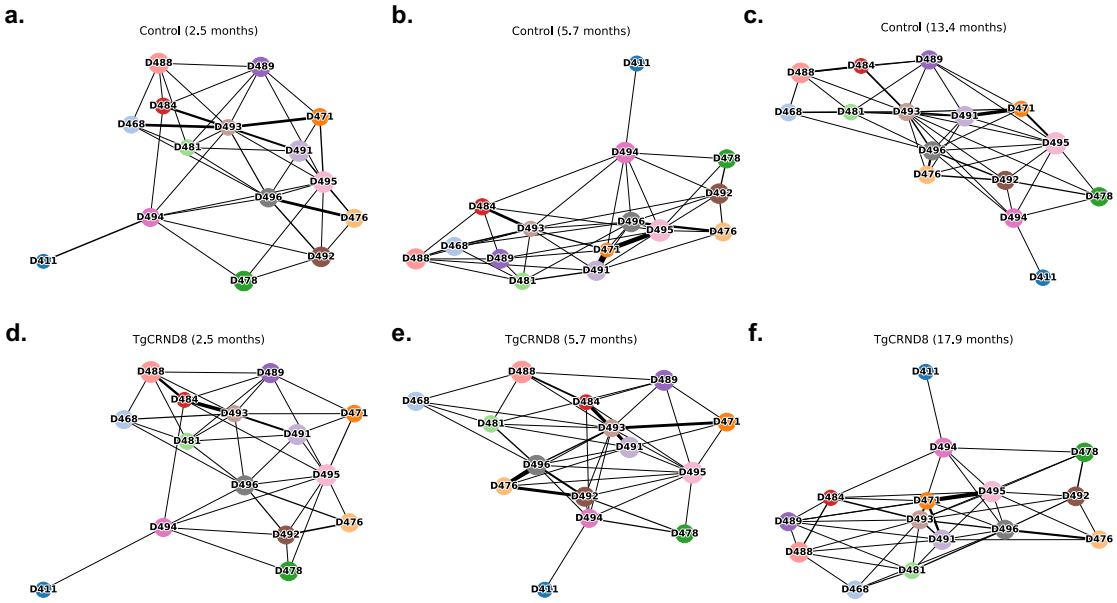

Figure A.21: ***PAGA graph on the mouse brain slides a. to c.*** *control mice, at 2.5, 5.7, and 13.4 months, respectively.* ***d. to f.*** *TgCRND8 mice, at 2.5, 5.7, and 17.9 months, respectively.*

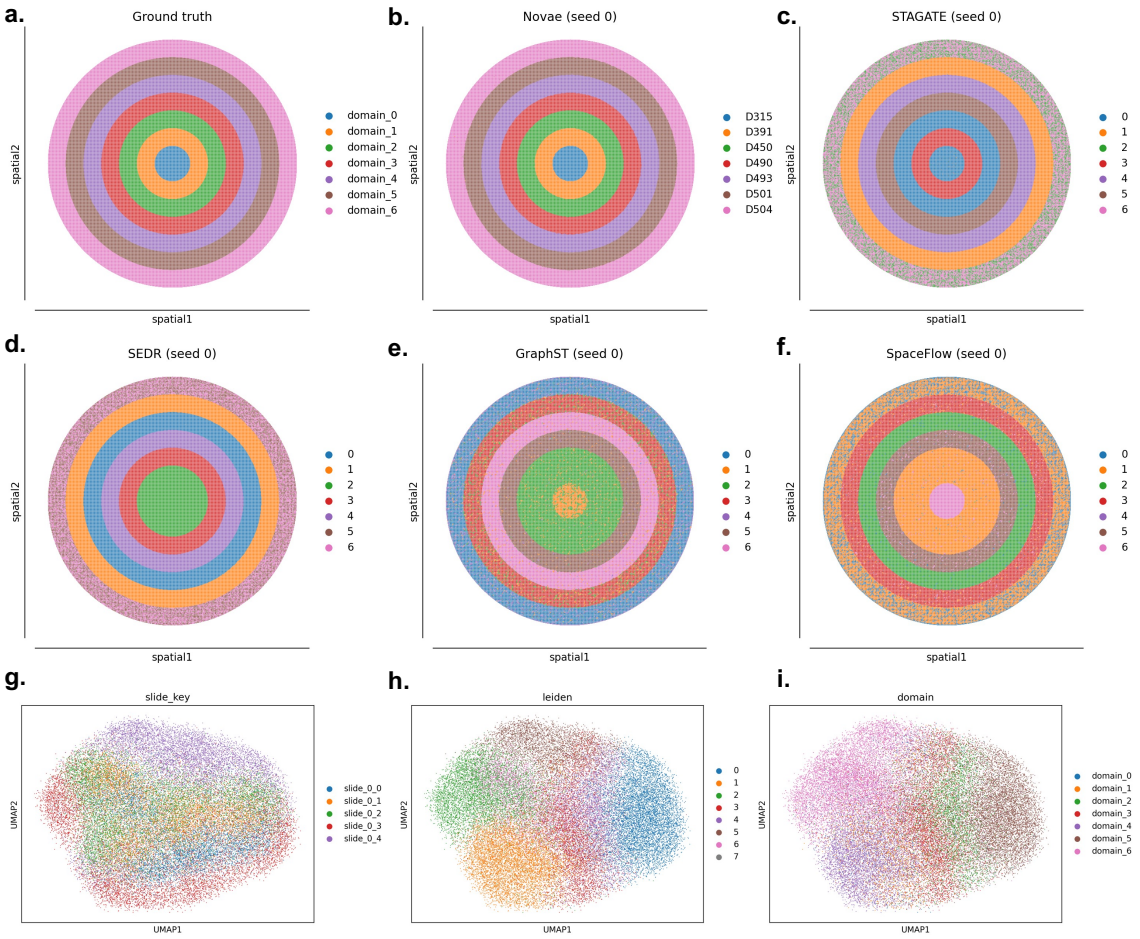

Figure A.22: ***Results on the synthetic dataset. a.*** *Spatial domain ground truth (generated) for the synthetic dataset (first slide).* ***b. to f.*** *Spatial domain prediction for the 5 models (Novae, STAGATE, SEDR, GraphST, and Spaceflow, respectively) on the synthetic dataset for the first seed and the first slide.* ***g. to i.*** *UMAPs computed on the cells gene expression (not the spatial representations) from the synthetic dataset, colored by the ID of the slide (g), the results of a Leiden[Traag et al., 2019] clustering (h) and the ground truth domains (i).*

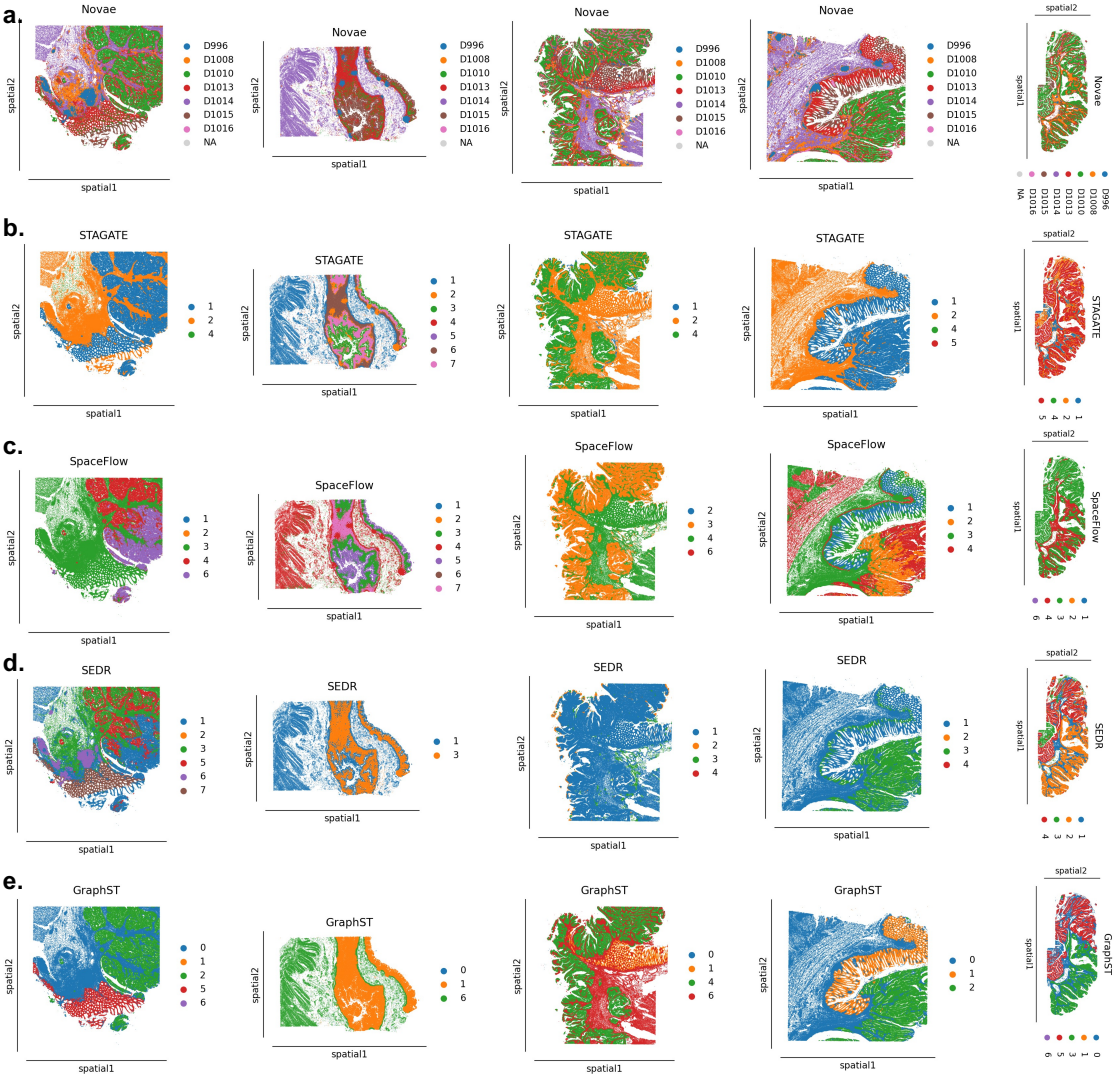

Figure A.23: ***Spatial domain prediction on the colon dataset for the different methods. a.*** *Predictions of Novae.* ***b.*** *Predictions of STAGATE.* ***c.*** *Predictions of SpaceFlow.* ***d.*** *Predictions of SEDR.* ***e.*** *Predictions of GraphST.*

Mathematical details

B.1 . Normalizing Flows and Real NVP

Normalizing flows are a class of generative models that enable the transformation of a simple probability distribution (typically, a normal distribution) into a more complex one through a series of invertible mappings. These mappings are constructed so that the resulting probability distribution can be easily computed. The key idea behind normalizing flows is the use of the change of variables formula for probability density functions. Given a random variable $\mathbf{z}_0$ with a known distribution, we can obtain a new variable $\mathbf{z}_K$ through a series of invertible transformations f_k:

$$\mathbf{z}_k = f_k(\mathbf{z}_{k-1}), \quad k = 1, \ldots, K.$$

The resulting density $p(\mathbf{z}_K)$ can be expressed as:

$$p(\mathbf{z}_K) = p(\mathbf{z}_0) \prod_{k=1}^{K} \left| \det \frac{\partial f_k}{\partial \mathbf{z}_{k-1}} \right|^{-1}.$$

This formula leverages the determinant of the Jacobian of the transformation to account for the change in volume induced by each transformation.

B.1.1 . Real NVP (Real-valued Non-Volume Preserving Transformation)

Real NVP [Dinh et al., 2017] is a specific type of normalizing flow designed to facilitate the computation of the forward and inverse transformations, as well as the determinant of the Jacobian. Real NVP achieves this by using coupling layers, which ensure that the Jacobian is triangular, and thus, with an easy-to-compute determinant.

A coupling layer in Real NVP splits the input variable $\mathbf{z}$ into two parts, $\mathbf{z}_{1:d}$ and $\mathbf{z}_{d+1:D}$. The transformation is then defined as:

$$\mathbf{z}'_{1:d} = \mathbf{z}_{1:d},$$

$$\mathbf{z}'_{d+1:D} = \mathbf{z}_{d+1:D} \odot \exp(s(\mathbf{z}_{1:d})) + t(\mathbf{z}_{1:d}),$$

where $s(\cdot)$ and $t(\cdot)$ are scaling and translation functions, respectively, and $\odot$ denotes element-wise multiplication. The inverse transformation can be easily computed as:

$$\mathbf{z}_{1:d} = \mathbf{z}'_{1:d},$$

$$\mathbf{z}_{d+1:D} = (\mathbf{z}'_{d+1:D} - t(\mathbf{z}'_{1:d})) \odot \exp(-s(\mathbf{z}'_{1:d})).$$

The Jacobian of this transformation is triangular, and its determinant is simply the product of the diagonal elements:

$$\left|\det \frac{\partial \mathbf{z}'}{\partial \mathbf{z}}\right| = \exp\left(\sum_{i=1}^{D-d} s_i(\mathbf{z}_{1:d})\right).$$

Real NVP can be stacked in multiple layers, with alternating coupling layers to ensure that all dimensions of the input are transformed. This stacking allows Real NVP to model complex, high-dimensional distributions effectively. Therefore, Real NVP can be employed for data normalization, density estimation, and generative modeling. These capabilities are crucial for understanding the underlying biological processes and for developing accurate predictive models in oncology research.

B.2 . Expectation-Maximization (EM) Algorithm

The Expectation-Maximization (EM) algorithm is an iterative method for finding maximum likelihood estimates of parameters in probabilistic models, especially when the model depends on unobserved latent variables. It alternates between performing an expectation (E) step, which computes an expectation of the log-likelihood with respect to the current estimate of the distribution for the latent variables, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step.

B.2.1 . Overview of the EM Algorithm

Consider a set of observed data points $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, and let $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N\}$ be the corresponding latent variables. The complete data log-likelihood is given by:

$$\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta}),$$

where $\boldsymbol{\theta}$ denotes the parameters of the model. The EM algorithm iteratively refines the parameters $\boldsymbol{\theta}$ through the following two steps:

E-Step (Expectation Step) In the E-step, we compute the expected value of the complete data log-likelihood with respect to the current estimate of the distribution of the latent variables. This involves computing the conditional expectation:

$$Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) = \mathbb{E}_{\mathbf{Z}|\mathbf{X},\boldsymbol{\theta}^{(t)}} \left[\log p(\mathbf{X}, \mathbf{Z} \mid \boldsymbol{\theta})\right],$$

where $\boldsymbol{\theta}^{(t)}$ is the estimate of the parameters at iteration t.

M-Step (Maximization Step) In the M-step, we maximize the expected complete data log-likelihood found in the E-step with respect to the parameters:

$$\boldsymbol{\theta}^{(t+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}).$$

These steps are repeated until convergence, i.e., until the parameter estimates stabilize.

B.2.2 . Application to Gaussian Mixture Models (GMMs)

A common application of the EM algorithm is in fitting Gaussian Mixture Models (GMMs), which assume that the data is generated from a mixture of several Gaussian distributions with unknown parameters. In this context, the latent variables $\mathbf{Z}$ indicate the component (i.e., Gaussian) from which each data point is drawn.

Given a GMM with K components, the parameters to be estimated are the mixture weights π_k, the means $\boldsymbol{\mu}_k$, and the covariances $\boldsymbol{\Sigma}_k$ for each component k.

E-Step for GMMs The E-step involves computing the posterior probabilities (responsibilities) that each data point $\mathbf{x}_i$ was generated by component k:

$$\gamma_{ik}^{(t)} = \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_i \mid \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^{K} \pi_j^{(t)} \mathcal{N}(\mathbf{x}_i \mid \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})}.$$

M-Step for GMMs In the M-step, we update the parameters using the responsibilities computed in the E-step:

$$\pi_k^{(t+1)} = \frac{1}{N} \sum_{i=1}^{N} \gamma_{ik}^{(t)},$$

$$\boldsymbol{\mu}_k^{(t+1)} = \frac{\sum_{i=1}^{N} \gamma_{ik}^{(t)} \mathbf{x}_i}{\sum_{i=1}^{N} \gamma_{ik}^{(t)}},$$

$$\boldsymbol{\Sigma}_k^{(t+1)} = \frac{\sum_{i=1}^{N} \gamma_{ik}^{(t)} (\mathbf{x}_i - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_i - \boldsymbol{\mu}_k^{(t+1)})^T}{\sum_{i=1}^{N} \gamma_{ik}^{(t)}}.$$

B.2.3 . Applications in the context of single-cell data

In the context of single-cell data, the EM algorithm can be used to identify distinct subpopulations of cells, each being represented by its proper distribution. This is the

reason why this algorithm was used in Scyan before moving on to the final version of the model.

Long summary in French

La médecine de précision en oncologie vise à personnaliser les traitements en fonction des profils génétiques et moléculaires uniques des tumeurs des patients, dans le but d'améliorer l'efficacité des thérapies tout en réduisant les effets secondaires. Avec les avancées technologiques, notamment les technologies d'omics unicellulaires et spatiales, il devient possible d'obtenir des données extrêmement détaillées sur le microenvironnement tumoral. Ces technologies offrent une résolution sans précédent pour étudier les caractéristiques cellulaires et moléculaires du cancer, notamment en conservant le contexte spatial des cellules au sein des tissus.

Cependant, la richesse et la complexité croissantes de ces données posent des défis analytiques majeurs, rendant les méthodes traditionnelles insuffisantes pour en extraire des informations biologiques pertinentes. C'est dans ce contexte que la nécessité d'approches computationnelles avancées s'impose. Les méthodes d'apprentissage profond (deep learning) émergent comme une solution prometteuse, capable de traiter des volumes massifs de données complexes et de surmonter les limitations des approches classiques.

Le premier chapitre introduit les deux domaines fondamentaux de cette thèse, c'est-à-dire l'immunologie et l'apprentissage profond. Ces concepts sont expliqués de manière accessible, afin de poser les bases nécessaires pour comprendre les travaux présentés dans les chapitres suivants. Une attention particulière est portée à l'explication des technologies d'omics unicellulaires et spatiales, ainsi qu'aux défis spécifiques qu'elles posent. Cette mise en contexte met en lumière les raisons pour lesquelles de nouvelles méthodes basées sur l'apprentissage profond sont nécessaires dans le cadre de cette thèse.

Le deuxième chapitre présente Scyan, la première contribution méthodologique de cette thèse. Ce projet a été initié dans le cadre de l'étude clinique Pop-Durva menée à Gustave Roussy. Cette étude, portant sur une cohorte de 150 patients, a révélé des effets de batch importants, qui diminuaient la robustesse et la qualité des annotations cellulaires

avec les méthodes existantes. Scyan répond à ce besoin par une méthode d'annotation des types cellulaires basée sur l'apprentissage profond, conçue pour être robuste face à ces effets de batch. Initialement développée pour Pop-Durva, Scyan a ensuite été généralisée pour s'appliquer à un large éventail d'études.

Le troisième chapitre aborde les données spatiales omiques, un domaine dans lequel j'ai été impliqué lors de l'acquisition de la première machine MERSCOPE à Gustave Roussy. Ce besoin a conduit au développement de Sopa, une pipeline conçue initialement pour analyser ces données spécifiques. Sopa a été étendue pour devenir indépendante des technologies, et peut désormais traiter tout type de données omiques à résolution unicellulaire. Ce projet établit ainsi des bases solides pour standardiser l'analyse des données spatiales omiques, en mettant l'accent sur l'efficacité en termes de temps et de mémoire pour gérer des volumes de données importants générés par ces technologies.

Le quatrième chapitre présente Novae, la dernière contribution méthodologique de cette thèse. Novae prolonge le travail initié avec Sopa, en analysant les données de transcriptomique spatiale prétraitées. Ce modèle de fondation repose sur une représentation des cellules dans leur environnement spatial, et a été entraîné sur un vaste ensemble de données comprenant près de 30 millions de cellules provenant de 18 tissus différents. L'accroissement des études en transcriptomique spatiale a permis de constituer des bases de données suffisamment volumineuses pour entraîner Novae. Cela en fait un modèle prêt à l'emploi, utilisable sans nécessiter de réentraînement pour de nouveaux jeux de données, ce qui facilite grandement son intégration dans de nouvelles études.

Le cinquième chapitre illustre les applications variées des méthodes développées au cours de cette thèse. Ces projets démontrent la polyvalence et l'impact potentiel de Scyan, Sopa et Novae, couvrant un large éventail d'applications en médecine de précision. Ils montrent comment ces outils permettent de mieux comprendre la biologie du cancer, d'identifier de nouveaux biomarqueurs et d'explorer des cibles thérapeutiques.

Le dernier chapitre récapitule les différentes contributions méthodologiques et applicatives de cette thèse. Il met en perspective l'impact potentiel de ces travaux, notamment en termes d'avancées pour la médecine de précision et d'amélioration des outils disponibles pour la communauté scientifique. Les développements réalisés ouvrent la voie à de nouvelles opportunités pour analyser et exploiter les données complexes issues des technologies omiques modernes.