KTH ROYAL INSTITUTE
OF TECHNOLOGY

Doctoral Thesis in Biotechnology

Computational Models of Spatial Transcriptomes

LUDVIG BERGENSTRÅHLE

Stockholm, Sweden 2024

Computational Models of Spatial Transcriptomes

LUDVIG BERGENSTRÅHLE

Academic Dissertation which, with due permission of the KTH Royal Institute of Technology, is submitted for public defence for the Degree of Doctor of Philosophy on Wednesday the 31th January 2024, at 10:00 a.m. in Air & Fire, SciLifeLab, Tomtebodavägen 23A, Solna.

Doctoral Thesis in Biotechnology
KTH Royal Institute of Technology
Stockholm, Sweden 2024

ISBN 978-91-8040-820-2
TRITA-CBH-FOU-2024:1

Printed by: Universitetsservice US-AB, Sweden 2024

Public Defense

The public defense of this thesis will take place at 10:00 on January 31, 2024 in Air & Fire, SciLifeLab, Tomtebodavägen 23A, Solna.

Respondent:
Ludvig Bergenstråhle, Department of Gene Technology, KTH Royal Institute of Technology

Opponent:
Prof. Ole Winther, Department of Applied Mathematics and Computer Science, Technical University of Denmark

Grading committee:
Assoc. Prof. Marc Friedländer, Department of Molecular Biosciences, The Wenner-Gren Institute, Stockholm University

Dr. Åsa Björklund, Department of Life Science, Chalmers University of Technology

Prof. Björn Önfelt, Department of Applied Physics, KTH Royal Institute of Technology

Chairman:
Assoc. Prof. Olof Emanuelsson, Department of Gene Technology, KTH Royal Institute of Technology

Supervisor:
Prof. Joakim Lundeberg, Department of Gene Technology, KTH Royal Institute of Technology

Co-supervisor:
Prof. Jens Lagergren, Department of Electrical Engineering and Computer Science, KTH Royal Institute of Technology

Abstract

Spatial biology is a rapidly growing field that has seen tremendous progress over the last decade. We are now able to measure how the morphology, genome, transcriptome, and proteome of a tissue vary across space. Datasets generated by spatial technologies reflect the complexity of the systems they measure: They are multi-modal, high-dimensional, and layer an intricate web of dependencies between biological compartments at different length scales. To add to this complexity, measurements are often sparse and noisy, obfuscating the underlying biological signal and making the data difficult to interpret. In this thesis, we describe how data from spatial biology experiments can be analyzed with methods from deep learning and generative modeling to accelerate biological discovery. The thesis is divided into two parts. The first part provides an introduction to the fields of deep learning and spatial biology, and how the two can be combined to model spatial biology data. The second part consists of four papers describing methods that we have developed for this purpose. **Paper I** presents a method for inferring spatial gene expression from hematoxylin and eosin stains. The proposed method offers a data-driven approach to analyzing histopathology images without relying on expert annotations and could be a valuable tool for cancer screening and diagnosis in the clinics. **Paper II** introduces a method for jointly modeling spatial gene expression with histology images. We show that the method can predict super-resolved gene expression and transcriptionally characterize small-scale anatomical structures. **Paper III** proposes a method for learning flexible Markov kernels to model continuous and discrete data distributions. We demonstrate the method on various image synthesis tasks, including unconditional image generation and inpainting. **Paper IV** leverages the techniques introduced in Paper III to integrate data from different spatial biology experiments. The proposed method can be used for data imputation, super resolution, and cross-modality data transfer.

Sammanfattning

Spatial biologi är ett snabbt växande forskningsområde som har sett en hög utvecklingstakt under det senaste decenniet. Vi kan idag mäta hur en vävnads morfologi, genom, transkriptom och proteom varierar i rummet. Dataset skapade av spatiala teknologier återspeglar komplexiteten i de system de mäter: De är multimodala, högdimensionella och är uppbyggda av ett intrikat nätverk av beroenden mellan biologiska strukturer som existerar på olika längdskalor. Som om denna komplexitet inte var nog, är mätningarna ofta både glesa och brusiga, vilket försvårar tolkningen av den underliggande biologiska signalen. I denna avhandling beskriver vi hur data från experiment inom spatial biologi kan analyseras med hjälp av djupinlärning och generativ modellering för att accelerera biologiska upptäckter. Avhandlingen är uppdelad i två delar. Den första delen ger en introduktion till fälten djupinlärning och spatial biologi, och hur dessa kan kombineras för att modellera data inom spatial biologi. Den andra delen består av fyra artiklar som beskriver metoder som vi har utvecklat för detta ändamål. **Artikel I** presenterar en metod för att skatta spatialt genuttryck från hematoxylin-eosin-färgningar. Den föreslagna metoden erbjuder ett datadrivet tillvägagångssätt för att analysera histopatologi-bilder utan användning av expertannoteringar och kan utgöra ett värdefullt verktyg för cancerscreening och diagnos i kliniken. **Artikel II** introducerar en metod för sammodellering av spatialt genuttryck och histologibilder. Vi visar att metoden kan användas för att predicera superupplöst genuttryck och transkriptionellt karakterisera småskaliga anatomiska strukturer. **Artikel III** beskriver en metod för modellering av kontinuerliga och diskreta datafördelningar med flexibla Markovkärnor. Vi demonstrerar metoden på olika bildgenereringsuppgifter, inklusive obetingad datagenerering och inpainting. **Artikel IV** utnyttjar teknikerna från Artikel III för att integrera data från olika experiment inom spatial biologi. Den föreslagna metoden kan användas för imputering, superupplösning och dataöverföring mellan olika modaliteter.

List of Papers

Paper I
Bryan He, **Ludvig Bergenstråhle**, Linnea Stenbeck, Abubakar Abid, Alma Andersson, Åke Borg, Jonas Maaskola, Joakim Lundeberg, and James Zou. "Integrating spatial gene expression and breast tumour morphology via deep learning". In: *Nature Biomedical Engineering* 4.8 (2020), pp. 827–834

Paper II
Ludvig Bergenstråhle, Bryan He, Joseph Bergenstråhle, Xesús Abalo, Reza Mirzazadeh, Kim Thrane, Andrew L. Ji, Alma Andersson, Ludvig Larsson, Nathalie Stakenborg, Guy Boeckxstaens, Paul Khavari, James Zou, Joakim Lundeberg, and Jonas Maaskola. "Super-resolved spatial transcriptomics by deep data fusion". In: *Nature Biotechnology* 40.4 (2021), pp. 476–479

Paper III
Ludvig Bergenstråhle, Jens Lagergren, and Joakim Lundeberg. "Learning Stationary Markov Processes With Contrastive Adjustment". In: *ArXiv preprint* abs/2303.05497 (2023)

Paper IV
Ludvig Bergenstråhle and Joakim Lundeberg. "Multi-Modal Modeling of Spatial Biology Data". In Preparation

Extended List of Papers

1. Kim Wong, José Fernández Navarro, **Ludvig Bergenstråhle**, Patrik L Ståhl, and Joakim Lundeberg. "ST Spot Detector: a web-based application for automatic spot and tissue detection for spatial Transcriptomics image datasets". In: *Bioinformatics* 34.11 (2018), pp. 1966–1968

2. Jonas Maaskola, **Ludvig Bergenstråhle**, Aleksandra Jurek, José Fernández Navarro, Jens Lagergren, and Joakim Lundeberg. "Charting Tissue Expression Anatomy by Spatial Transcriptome Decomposition". In: *bioRxiv* (2018)

3. Sanja Vickovic, Gökcen Eraslan, Fredrik Salmén, Johanna Klughammer, Linnea Stenbeck, Denis Schapiro, Tarmo Äijö, Richard Bonneau, **Ludvig Bergenstråhle**, José Fernandéz Navarro, Joshua Gould, Gabriel K. Griffin, Åke Borg, Mostafa Ronaghi, Jonas Frisén, Joakim Lundeberg, Aviv Regev, and Patrik L. Ståhl. "High-definition spatial transcriptomics for in situ tissue profiling". In: *Nature Methods* 16.10 (2019), pp. 987–990

4. Emelie Berglund, Sami Saarenpää, Anders Jemt, Joel Gruselius, Ludvig Larsson, **Ludvig Bergenstråhle**, Joakim Lundeberg, and Stefania Giacomello. "Automation of Spatial Transcriptomics Library Preparation To Enable Rapid and Robust Insights Into Spatial Organization of Tissues". In: *BMC Genomics* 21.1 (2020), p. 298

5. Joseph Bergenstråhle, **Ludvig Bergenstråhle**, and Joakim Lundeberg. "SpatialCPie: an R/Bioconductor package for spatial transcriptomics cluster evaluation". In: *BMC Bioinformatics* 21.1 (2020), p. 161

6. Alma Andersson, Joseph Bergenstråhle, Michaela Asp, **Ludvig Bergenstråhle**, Aleksandra Jurek, José Fernández Navarro, and Joakim Lundeberg. "Single-cell and spatial transcriptomics enables probabilistic inference of cell type topography". In: *Communications Biology* 3.1 (2020), p. 565

7. Ludvig Larsson, **Ludvig Bergenstråhle**, Mengxiao He, Zaneta Andrusivova, and Joakim Lundeberg. "Snapshot: Spatial Transcriptomics". In: *Cell* 185.15 (2022), 2840–2840.e1

8. Andrew Erickson, Mengxiao He, Emelie Berglund, Maja Marklund, Reza Mirzazadeh, Niklas Schultz, Linda Kvastad, Alma Andersson, **Ludvig Bergenstråhle**, Joseph Bergenstråhle, Ludvig Larsson, Leire Alonso Galicia, Alia Shamikh, Elisa Basmaci, Teresita Díaz De

Ståhl, Timothy Rajakumar, Dimitrios Doultsinos, Kim Thrane, Andrew L. Ji, Paul A. Khavari, Firaz Tarish, Anna Tanoglidi, Jonas Maaskola, Richard Colling, Tuomas Mirtti, Freddie C. Hamdy, Dan J. Woodcock, Thomas Helleday, Ian G. Mills, Alastair D. Lamb, and Joakim Lundeberg. "Spatially Resolved Clonal Copy Number Alterations in Benign and Malignant Tissue". In: *Nature* 608.7922 (2022), pp. 360–367

9. Maja Marklund, Niklas Schultz, Stefanie Friedrich, Emelie Berglund, Firas Tarish, Anna Tanoglidi, Yao Liu, **Ludvig Bergenstråhle**, Andrew Erickson, Thomas Helleday, Alastair D. Lamb, Erik Sonnhammer, and Joakim Lundeberg. "Spatio-Temporal Analysis of Prostate Tumors in Situ Suggests Pre-Existence of Treatment-Resistant Clones". In: *Nature Communications* 13.1 (2022), p. 5475

10. Markus Ekvall, **Ludvig Bergenstråhle**, Alma Andersson, Paulo Czarnewski, Johannes Olegård, Lukas Käll, and Joakim Lundeberg. "Spatial landmark detection and tissue registration with deep learning". In: *bioRxiv* (2023)

This thesis is dedicated to my dad.

In one of my last memories of us together, you were carrying me on your shoulders. Another year has passed since then. Even though you are just with us in spirit now, I think in many ways I am still up there, on your shoulders.

Acknowledgements

There are many people without whom this thesis would not have been possible, or at least looked very different. Therefore, I would like to take this opportunity to thank everyone who has helped me along the way and made my experience as a PhD student so rewarding.

My biggest supporters not only in my studies but also in life are, of course, my family. To my mom **Agneta** and brother **Joseph** (a.k.a. Joey): No matter what happens in life, I know that we are always there for each other. How awesome is not that! Words cannot describe how much you mean to me. So what am I supposed to write here, exactly? I hope you know what I fail to express, and I am eternally grateful the roulette wheel of life landed us together.

Christopher, I am so glad we have been able to stay in touch over the years. Having Motala sushi or going on a road trip through Sörmland, it is always a blast doing things with you. I will always remember that brief, golden time when I was the undisputed padel champion (by winning a one-game match), but I humbly admit that you are the way better player, and I am always up for another match!

Jonas, You were my mentor when I first started in Joakim's lab. Your expertise on probabilistic models has had a tremendous influence on the ideas presented in this thesis, and, indeed, on the direction of my research. I sometimes wonder how things would have turned out without your guidance, but I am glad I never had to find out.

Kim Wong, With you my first years in the lab were never boring. How I miss our lunches together! You are one of the most thoughtful persons I know. Hanging out with you and **Tobias**, playing board games, laserdome, and setting off fireworks are things I will remember fondly. Thank you both for all the great times!

Alma, You combine absolute brilliancy with insane work ethic but still manage to be so down to earth. It is simply impossible to have a dull moment with you, and I have lost count of all the memorable experiences we have shared here in Sweden and across the globe. I am just so glad we came up with the distans-promenad concept before you left so that we can keep hiking together from afar!

Ludvig, Not only are you one of the coolest persons I know, you also have an amazing ability to make the people around you feel just as interesting. Maybe too interesting some times: I am sorry for how much I must have bored you with all my talk about the latest crypto fads. But, you see, you

always have something insightful to say, no matter the weight of the topic. Thank you for all the time together, dropping line clears, exploring nature from the sea, or just going for a walk. I have enjoyed every minute of it, and I hope we will get a chance to do it all again soon!

Žaneta, Whether flying down the slopes of Trysil at the most crazy speeds or diving with sharks in the ocean, I admire how you never let doubt come in your way. I want to thank you for bringing a lot of perspective to my life. You mean more to me than you may know, and I have really appreciated every moment we have shared, both on and off alpha 3. I already miss you being here.

Maja, You are my parkour buddy. Your enthusiasm for doing completely insane things is disturbingly contagious (of course, we would never sneak into a fenced-off military protection zone with bright-yellow do-not-enter signs posted every five meter around the entire perimeter or something like that, who would do such a thing...), and I have had so much fun hanging out with you. Let's go jump some buildings again soon!

Kim Thrane, Besides being a top-tier scientist, you are also one of the most considerate persons in existence. Thank you for saving me from sleepless nights in Campsie, for always remembering my birthday, and for making the lab overall such a nice place to be in. I hope you continue to pursue a career in academia, because you would be the best PI imaginable.

Markus, How cool it is to have someone who shares my interest in machine learning sitting just next to me. But not only that, I feel like I can talk about anything with you, and I always come out of our conversations with a smile; whether the topic at hand was a (living) gift you received from your friends or something more serious. Thank you for your friendship.

Humam, Shooting aliens on a mining mission in outer space or just going for a coffee, it is always a pleasure hanging out with you. Also, thank you for trying your best to teach me how to run the PCR machines (I *did* learn one thing, which is that it is probably best I stay out of the lab!).

Enikő, It cannot be a coincidence that the person who knows everything about the heart also has the biggest heart herself. I hope we get a chance to do that pottery class soon!

Julia, Even if I live to be a thousand years I probably still wouldn't know life half as well as you do. I still remember your performance for Alma, Linnea, and me in the Långholmen prison on the first day I met you. You have an amazing talent, and I really hope you find your way back to making music again.

Mengxiao, You have a fantastic ability to cut through the noise to get to the core of a problem. I feel like I can always trust your judgement on things. Thank you for being around. You are a true asset for the lab and everyone around you.

Eva, I find it impressive how you are so knowledgeable of just about everything. Thank you for all the fun and interesting lunchroom discussions, whether they concern the latest Nobel prizes or the latest movements in the stock market!

Kostas, Among all the doctors of philosophy in the lab, you are certainly the one to live up to the title most accurately. It is always interesting hearing about your theories on the nature of the mind and the universe, and never, of course, without a healthy dose of dad jokes. Thank you for all the fun times.

Reza, It is always very inspiring talking with you about the next big idea, whether that be a new research project, how to set up a car wash, or starting an AI company. How you manage to do all these things on just a few hours of sleep every night (still looking as well-rested as ever) is beyond me. I look forward to seeing all your creations a few years from now!

Sami, Your dark humor never fails to brighten my day. You are also one of the kindest persons I know, looking after people as well as you look after the plants on our floor. Thank you for being around.

Don **Marco**, My desk neighbor and fellow Swedish-speaking Emacs user. Your impeccable organization skills and constant drive for finding better solutions to things are truly inspiring and have already translated into some first-class research. I think you will be an amazing PI one day, and I look forward to seeing what you will do next!

Pontus, I am impressed by how good you are at everything you try. Playing padel, writing new bioinformatics tools, or processing samples in the wet lab, you always excel at it. I have also really enjoyed your company in the lunchroom. Thanks for being here.

Franzi, Your visits to the lab were always such a pleasant surprise. I admire your deep subject expertise, and I am certain that you will make a great PI one day, if that is what you decide to pursue. I hope we get a chance to meet again soon!

Marcos, You have the curiosity and critical thinking skills of a true scientist. I also really enjoyed your music mixes in Sydney, and I never thought beer could actually taste good before I tried yours. Thank you for being a part of the team!

Javier, Thank you for all the sarcastic laughs together and for making the Sydney visit such a memorable experience. You have a big heart but also a big brain. I look forward to seeing your ideas on topological gene expression models come to life.

Lovisa, You have a very cool, collected approach to science and an amazing artistic talent to boot. Even though the biology is sometimes way above my head, listening to your presentations, with the most elegant slides I have ever seen, is always a treat. Also, although I appreciated Markus' attempt to draw my picture, I want to thank you for refining it to something that didn't look like a Dracula monster.

Hailey, You are just the kindest person, and it is always a pleasure being in your company. I am so glad you ended up in Stockholm, and I hope you will stay here for a long time to come.

Leire, No matter if the task before you is to run the Lidingö-loppet at record time or to process 236 Visium samples, you always do it without breaking a sweat. Thank you for bringing such a positive vibe to the lab!

Raphaël, I must say that your commitment to reproducible science, puzzling together environments from different code bases and meticulously documenting every step when any sane person would have given up long ago, is truly commendable. You have both the grit and intelligence to go very far! You are also a super down-to-earth and cool guy to hang out with. I wish you the best of luck with your future research.

Annelie, When you were here, people used to think of you as our wet lab mom. But I think it is fair to say that you were the mom of the entire lab, wet and dry. Thank you for creating such a welcoming atmosphere.

Hooman, I share your passion for statistics, and it is always very engaging to talk with you. You are also extremely kind. I still remember how you brought me pistachios from Tehran (the best I have ever tasted) even before we knew each other very well. Thank you for everything!

Solène, I find it admirable how you have been able to tackle problems in both the wet and dry lab in such short time. Your excitement about everything and anything is absolutely infectious, thank you for bringing so much positive energy to the lab!

Sybil, You approach science with curiosity and an open mind, and I find that very inspiring. I am grateful for all the interesting conversations in the lunchroom, and I wish you the best of luck with your research!

Jian, It is always interesting to hear about your projects, and I am impressed by how you have been able to combine your expertise in both biology

and computer science. I wish you continued success in your research.

Linda, You are a real-life super-mom, and your passion for both science and mental health is very inspiring. Thank you for contributing to the lab in so many ways.

Our research team is constantly evolving and expanding. Thank you for joining, **Emmanouela**, **Muhamed**, and **Martí**. I know you will make great contributions to the lab, and I hope your time here will be as rewarding as mine has been.

There are so many more people that I have gotten to know working at SciLifeLab that have made my time here so enjoyable. Thank you **Camilla**, **Carl-Johan**, **Christian**, **Fitz**, **Gustavo**, **Jörg**, **Marion**, **Matthias**, **Mattias**, **Nayanika**, **Nemo**, **Patrick**, **Paulo**, **Pär**, **Simon**, **Xesús**, and **Yuvarani**, among many others.

I would also like to thank some of the people that I got to know in the beginning of my studies and who have played a big role in shaping my experience as a PhD student: **Anders**, **David**, **Emelie**, **José**, **Linnea**, **Mickan**, **Rapolas**, and **Sanja**, to name a few. I have been very impressed by your work, and I wish you the best in your future endeavors.

During my studies, I have had the fortune to visit and collaborate with some of the most talented people around the world. In particular, I would like to thank **James Zou** and **Bryan He** for a very inspiring visit to Stanford, which resulted in two of the papers presented in this thesis. Thank you also to **Carsten** for guiding Alma and me on Japanese customs during our visit to RIKEN in Yokohama. In Sydney, I had the pleasure of visiting the amazing team of **Alexander Swarbrick** at the Garvan Institute. Thank you **John** for showing us around the beaches of Bondi and the rest of the team for making us feel so welcome. Thank you **Sonny** for very interesting discussions on regenerative medicine and deep learning (don't hesitate to hit me up if you ever want to collaborate!).

SciLifeLab is a fantastic place to work, and there are many PIs doing amazing work here. Thank you **Afshin**, **Pelin**, and **Anniina** for interesting discussions in the DNA clubs. Thank you **Stefania** and **Patrik** for being a part of the Spatial Collective and contributing with your expertise. Thank you **Lukas** and **Olof** for our talks on deep learning and bioinformatics.

I also want to give a big shout-out to my co-supervisor, **Jens**, who has patiently listened to my ideas—some more half-baked than others—and always guided me in the right direction. Thank you for helping me sort out clarity from chaos. Your expertise has been immensely valuable, and I have learned a lot from our discussions.

Last but not least, to my supervisor **Joakim**: Thank you for taking on a stray economist as your PhD student and for believing that I could contribute to a field that was so foreign to me. You have an amazing ability to understand the bigger picture of a research topic and to come up with and see the potential in new ideas. Your enthusiasm for this field is contagious, and it has truly been inspiring to work with you. You are probably also the only person in the world, besides your family of course, that I will ever be able to say that I have had the pleasure of celebrating midsummer with in the middle of winter! Your welcoming attitude and guidance have made a world of difference to me, and I couldn't have asked for a better supervisor.

Finally, I want to extend an additional thanks to **Joey**, **Jens**, **Afshin**, and **Joakim** for reading and providing valuable feedback on several drafts of this thesis.

Table of Contents

1 Introduction

Evolution is a messy—albeit powerful—optimization algorithm. While its solutions to the challenges of life many times are fascinating, the implementations of those solutions are not always so straightforward to understand. How can we make sense of the intricate processes governing a biological system? Answering this question could help us direct the system to a more favorable state, stabilize it from decay or perturbations, or even create entirely new systems from scratch.

Over the last decades, technologies for measuring biological systems have seen tremendous progress and allowed us to generate data at an unprecedented scale. The sequencing of the first human genome in 2003 took over a decade to complete and cost 2.7 billion USD. Today, it is possible to sequence a human genome in a matter of hours for less than 1000 USD, and some predict that the cost will drop to 100 USD in the near future. And while the first human genome was a composite of many individuals, it is now possible to reconstruct the genome of a single cell. Not only is the scale and precision of the data that we can generate today staggering, but the diversity of the data is also increasing: We can now measure not only the genome but also the RNA and protein composition of a tissue, and even how that composition varies across space. Given the complexity of biological systems, no wonder the data that we can generate from them share that same complexity!

But what good is complex data if we don't have the tools to analyze it? Luckily, in parallel with the development of large-scale measurement technologies, there has also been tremendous progress in a completely different field: machine learning. With the advent of deep learning, machine learning using large artificial neural networks, we can now train models to break down complexity and find patterns across multitudes of data. It would not be an exaggeration to say that deep learning-based systems are becoming an integral part of our daily lives, from serving us the latest playlist recommendations on Spotify to driving our future cars.

My PhD has been focused on developing machine learning methods for analyzing spatial biology data. Spatial biology is a rapidly evolving field that aims to understand how biological systems are organized in space. Data generated in spatial biology can be beautiful, hiding untold mysteries of life, much like far-away stars in the night sky. But it is also high-dimensional, sparse, noisy, biased, heterogeneous, incomplete, and—sometimes—infuriatingly difficult to make sense of. With the help of modern machine learning, maybe we can untangle some of that complexity and uncover some of those mysteries? My hope is that this thesis will give you,

the reader, a small introduction to the wonderful worlds of machine learning and spatial biology, and how the two can be combined to help us understand just a little bit more about the complexity of life and, by extension, the world around us.

The thesis is organized as follows: In Section 2, we give a brief overview of modern-era machine learning in the form of deep learning. Section 3 focuses on a subfield of machine learning known as generative modeling, which, as we will see, is an important tool for modeling biological systems. Next, Section 4 switches gears and gives a brief introduction to cell biology in multi-cellular organisms, providing the necessary background for understanding the data that we will be working with. Section 5 introduces the field of spatially resolved transcriptomics, which is an important technology for generating spatial biology data. In Section 6, we describe some of the most common problems that can be addressed using machine learning in spatial biology. Section 7 gives a brief overview of the papers included in this thesis. Finally, in Section 8, we conclude by discussing some of the challenges and opportunities that lie ahead in the intersection of machine learning and spatial biology.

2 Deep Learning

In this section, we will introduce deep learning by describing how neural networks are constructed and trained. As we will see, the building blocks of deep neural networks are surprisingly simple. The power of neural networks stem from their composability, allowing us to build very capable, deep networks from small, seemingly mundane parts.

Sections 2.1 and 2.2 introduce the basic tools for building neural networks by constructing and training a small network. In Section 2.3, we discuss the problems of overfitting and underfitting. The former is a natural consequence of the flexibility of large neural network, which we briefly discuss in Section 2.4. Finally, Section 2.5 concludes by introducing some of the most common building blocks of neural networks and how they can be composed to build deep architectures.

2.1 The Multi-Layer Perceptron

Suppose we are interested in predicting the age of a patient based on their blood pressure and resting heart rate. Let $X = [x_1, x_2]$ be a vector of the patient's blood pressure x_1 and heart rate x_2, and y their age. Our goal is to find a function $f(X)$ that approximates y from X.

We will use a basic neural network known as a *multi-layer perceptron* (MLP) for this task. An MLP consists of a sequence of transformations, known as *layers*. Each layer takes the output from the preceding layer, starting from the input X, and applies a linear transformation to it, followed by an *activation function* that allows the network to learn non-linear relationships. For the purpose of this example, we will use a two-layer MLP and a sigmoid activation,

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{2.1}$$

The first layer of the MLP takes the input X and transforms it into a feature vector H. This is done by multiplying the input with a matrix of weights W, adding a vector of biases B, and applying the activation function σ:

$$H' = XW + B \tag{2.2}$$

$$H = \sigma(H'). \tag{2.3}$$

The features H are known as *hidden units* because they are not directly observed in the data but internal representations of the input used by the

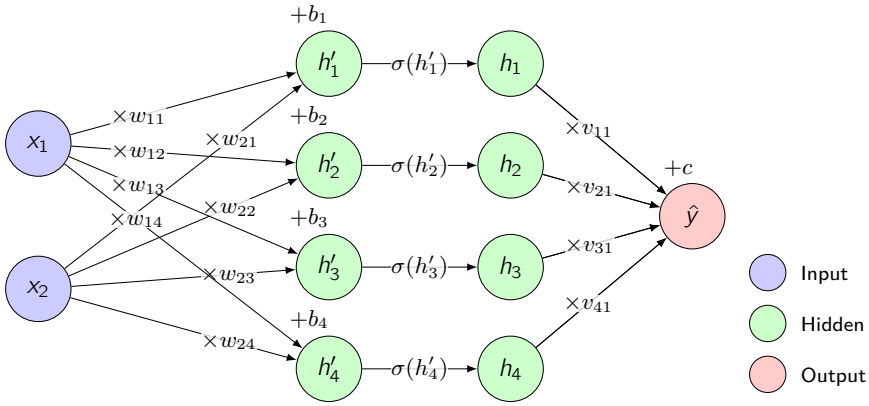

Figure 1: A multi-layer perceptron. The input X is transformed into a set of hidden features H by a linear transformation followed by a non-linear activation function. The transformed hidden features H' are then transformed into the prediction $\hat{y}$ by another linear transformation.

network to compute the prediction. Finally, the second layer takes the features H and transforms them into a prediction $\hat{y}$ of the patient's age:

$$\hat{y} = HV + c, \tag{2.4}$$

where V and c are weights and biases, respectively, for the second layer.

We now have a complete definition of our neural network:

$$f_\theta(X) = \sigma(XW + B)V + c, \tag{2.5}$$

where we have used the subscript $\theta = \{W, B, V, c\}$ to denote the set of all parameters in the network. The full computational graph of the network, which forms a directed acyclic graph (DAG), is illustrated in Figure 2.

But how do we find good values for the parameters θ? Picking them at random will clearly not give us any good predictions. In the next section, we will describe how to learn the parameters from data.

2.2 Gradient Descent

Assume we have a dataset of n patients, each with a measurement of their blood pressure and heart rate X_i and their age y_i. The first step in optimizing the parameters of the network to fit this data is to define an *objective function* that measures how well the network is performing. Here, we will be using the mean squared error between the prediction and the observed

age y as our objective:

$$L(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - f_\theta(X_i))^2. \tag{2.6}$$

Notice that the objective is a function of the parameters θ. Our goal is to find the parameters θ that minimize this function.

The most common way of optimizing a neural network is to use a method known as *gradient descent*. The gradient of a function is a vector that points in the direction of steepest ascent. Gradient descent follows the opposite direction of the gradient to iteratively obtain lower and lower values of the objective until we reach a local minimum:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta L(\theta_t), \tag{2.7}$$

where η is a hyperparameter known as the *learning rate* that determines how large each update step should be.

2.2.1 Backpropagation

To compute the gradient, we use a method known as *backpropagation*. Backpropagation computes gradients for every node in the computational graph of the network by applying the chain rule of calculus. The name comes from the fact that, in order to compute the gradient of ancestral nodes in the graph, we first compute the gradient of their descendants. As a result, the gradient computation inverts the DAG of the forward pass and creates a flow of gradients that propagates backward. In our example, we first compute the gradient of the objective with respect to the predictions:

$$\frac{\partial L}{\partial \hat{Y}} = \frac{2}{n}(\hat{Y} - Y), \tag{2.8}$$

where Y is a column vector of the observed ages y_i and $\hat{Y}$ a column vector of the predictions $\hat{y}_i$. Next, we compute the gradient with respect to the bias of the second layer:

$$\frac{\partial L}{\partial c} = \sum_i \frac{\partial L}{\partial \hat{y}_i}\frac{\partial \hat{y}_i}{\partial c} = \mathbf{1}\frac{\partial L}{\partial \hat{Y}}, \tag{2.9}$$

where $\mathbf{1}$ is a vector of ones. Eqs. (2.8) and (2.9) tell us that if our average predictions are too high, we should decrease the bias according to Eq. (2.7), and vice versa. Continuing in this fashion, we get:

$$\frac{\partial L}{\partial V} = H^T \frac{\partial L}{\partial \hat{Y}} \tag{2.10}$$

$$\frac{\partial L}{\partial H} = \frac{\partial L}{\partial \hat{Y}} V^T \tag{2.11}$$

$$\frac{\partial L}{\partial H'} = \sigma(H') \odot (1 - \sigma(H')) \odot \frac{\partial L}{\partial H} \tag{2.12}$$

$$\frac{\partial L}{\partial B} = \mathbf{1} \frac{\partial L}{\partial H'} \tag{2.13}$$

$$\frac{\partial L}{\partial W} = X^T \frac{\partial L}{\partial H'}, \tag{2.14}$$

where $\odot$ denotes element-wise multiplication and the nodes X, H', and H are matrices with each row corresponding to a single patient. We now have gradients for all parameters in the network and can update them according to Eq. (2.7) to iteratively improve model fit.

2.2.2 Stochastic Gradient Descent and Beyond

In the context of large datasets, it is often not practical to compute the gradient of the objective function over the entire dataset. Instead, the gradient can be approximated by computing it over a small subset of the data, known as a *mini-batch*. The update step in Eq. (2.7) is then performed using the approximate gradient instead, a technique known as *stochastic gradient descent* (SGD). In a convex optimization setting, SGD is, just like regular gradient descent, guaranteed to converge to the global minimum of the objective function, but the path it takes there is more erratic. In fact, the noisy optimization trajectory of SGD can help it break out of local minima in the non-convex setting of optimizing neural networks.

Modifications of SGD that allow for *momentum* and *adaptive learning rates* have also been proposed [1]. These methods provide the optimization trajectory with a certain degree of inertia to help it overcome local ridges in the optimization landscape and adjust the step size depending on how steep the landscape is.

2.3 Overfitting and Underfitting

When should we stop updating the parameters according to Eq. (2.7)? One way is to monitor the objective function and stop training when the objective plateaus. However, this is not always a good idea. Neural networks are extremely flexible. If trained for too long, they may start to contort the data space in unnatural ways in order to minimize the objective ever further. Such models will give good predictions on the training data but likely not generalize well to new data, a phenomenon known as *overfitting*.

Various strategies have been proposed to combat overfitting. For example,

the objective can be augmented with a regularization term that penalizes large weight values. However, if penalized too strongly, the model may instead become inflexible and *underfit* the data. Another option is to monitor the performance of the model on a separate dataset, a *validation set*, and stop training when the performance on the validation set starts declining.

Overfitting is mainly a concern on small datasets, where data points in the training set are far apart and the model is free to do whatever it likes with the space between them. If training a model on a small dataset, it is therefore often a good idea to augment the training data by perturbing it with noise or other transformations, a technique known as *data augmentation*, to fill in gaps in the data manifold [2].

2.4 The Universal Approximation Theorem

While the two-layer perceptron we have used so far is a very basic neural network, it turns out that it is actually a very powerful one. It can be used not only for prediction but also for any other task given the right objective function. And it can learn that task extremely well. In fact, it can be shown that if we allow the size of the hidden layer to grow very large, the two-layer perceptron can approximate any function arbitrarily well, a result known as the *universal approximation theorem* [3]. In other words, we just have define an appropriate objective, and we will be able solve all problems that come our way.

So there you have it: We now have the tools to create artificial general intelligences that will take our jobs, instigate the singularity, conquer the world, and enslave all of humanity! Well, not so fast... Unfortunately (or fortunately, depending on your perspective), while the universal approximation theorem does teach us that the ingredients required for creating very powerful neural networks are not necessarily that complicated, constructing efficient networks is not as easy as the theorem may have you believe. The result only holds in the limit of very large neural networks, which is not an entirely practical setting. In order to construct an efficient network, it may be better to compose carefully designed computational units into a *deep neural network*. In the next section, we will take a closer look at some of those building blocks.

2.5 Building Blocks of Deep Neural Networks

Neural networks are incredibly composable. Every computational unit is like a small brick of LEGO that can be combined with other bricks to form new structures. A large part of research in deep learning is concerned with how

to design the computational graph of neural networks for efficient learning. Which architecture of the network that is best suited for a particular task is context dependent, and finding it often relies on both intuition and empirical experimentation. In this section, we will present some of the most common building blocks of deep neural networks and how they can be composed to form larger structures.

2.5.1 Convolutions

In our example of predicting the age of a patient from their blood pressure and heart rate, the data was vector-valued. However, data in spatial biology is, as the name suggests, often spatially ordered. For example, a microscope image will have shape $H \times W \times C$. The first two dimensions are the spatial dimensions, corresponding to the height H and width W of the image. The third dimension is the feature dimension, corresponding to the number of color channels C. It is still possible to transform this data with a fully connected layer, similar to what we did in our MLP example, by flattening the image into a vector of size HWC. However, this approach quickly becomes untenable for anything but very small images, as the number of connections in the network grows quadratically with the size of the data.

One of the most important building blocks of deep neural networks is the convolutional layer, which allows us to efficiently transform spatial data. A convolution is a linear operation that takes an input X with d spatial dimensions and F feature channels and produces an output Y with d spatial dimensions and G feature channels by applying a kernel W to all regions of the input. Specifically, in the case of $d = 2$, the output elements of a convolutional layer are computed as

$$y_{i,j,g} = b_g + \sum_{k=1}^{K} \sum_{l=1}^{K} \sum_{f=1}^{F} x_{si+k-1,sj+l-1,f} w_{k,l,f,g}, \tag{2.15}$$

where s is the *stride* of the convolution, b_g is a bias term, and the kernel W has shape $K \times K \times F \times G$. In order to retain the size of the input, the input is typically padded with zeros on both sides and the stride is set to one. Convolutions can be used to *downsample* data by using a stride greater than one, which results in a smaller output volume. Conversely, by using a fractional stride, which is implemented by inserting zeros between the elements of the input, they can *upsample* the data and produce a larger output volume.

Convolutions are incredibly compute and memory efficient. To illustrate this efficiency, assume we have a two-dimensional data volume of size $H \times W \times F$. Whereas a fully connected layer would need on the order of H^2W^2FG

parameters and operations to compute the output, a convolutional layer needs only on the order of K^2FG parameters and K^2HWFG operations. Since the kernel size K is typically very small—a usual choice is $K = 3$—the number of parameters and operations in a convolutional layer is much smaller than in a fully connected layer.

The efficiency of convolutions comes at a cost, however: the output only takes into account a small neighborhood of the input, the *receptive field* of the kernel. This means that learning long-range dependencies in the data requires stacking many convolutional layers on top of each other, which, as will be discussed in Section 2.5.3, can introduce instability in the training process. On the other hand, since the same kernel is applied to all input regions, convolutions introduce a form of *weight sharing* between spatial positions. This weight sharing forces convolutional layers to learn very general patterns of the data and make them extremely robust feature extractors.

The local connectivity of convolutions is a form of *inductive bias*; that is, an implicit prior that pushes the model toward a certain solution space. Notably, an interesting property of convolutions is that they are translationally equivariant. In other words, if we translate the input, the output will be translated in the same way. A consequence of this equivariance is that convolutional networks can be applied to inputs of a different size than the ones they were trained on, which is often very useful as data in the wild often comes in many different shapes and sizes.

2.5.2 Attention

A widely used building block to model longer-range dependencies is the *attention mechanism* [4]. It is the central component of the Transformer architecture [5] and applied in many state-of-the-art large language models.

Attention is applied to sequences of data. However, in contrast to convolutions, the order of the sequences does not hold any special meaning. Therefore, attention can be applied to spatially arranged data simply by flattening the spatial dimensions. In some cases, a positional encoding is added to the inputs in order to retain positional information [5].

The attention layer computes *query* Q, *key* K, and *value* V matrices by linear projections of the input sequences X_1 and X_2:

$$Q = X_1 W_Q \tag{2.16}$$
$$K = X_2 W_K \tag{2.17}$$
$$V = X_2 W_V, \tag{2.18}$$

where X_1 has shape $n_1 \times d_{\text{model}}$, X_2 has shape $n_2 \times d_{\text{model}}$, and W_Q, W_K, and W_V are weight matrices, projecting the data to feature dimensions d_k, d_k, and d_v, respectively. The keys can be seen as identifiers for the values, and the query is used to retrieve values based on their similarity to the keys. A *compatibility score* S is computed for all pairs of queries and keys and transformed into attention weights A by applying the softmax function to each row of S:

$$S = \frac{QK^T}{\sqrt{d_k}} \tag{2.19}$$

$$A_i = \frac{\exp(S_i)}{\sum_j \exp(s_{ij})}. \tag{2.20}$$

The output is then computed as an attention-weighted sum of the values followed by a final projection to the input dimension:

$$Y = (AV)W_Y + B, \tag{2.21}$$

where W_Y has shape $d_v \times d_{\text{model}}$ and B is a bias term.

A common extension of attention is *multi-head attention*, where the Q, K, and V matrices are split along the feature dimension into h *heads* and separate attention weights are computed for each head. The attention outputs, $[A_1V_1, \ldots, A_hV_h]$, are then concatenated before the final projection. *Self-attention* is a special case of attention where the input sequences are the same; that is, $X_1 = X_2$. In contrast, when the input sequences are different, the attention layer is typically referred to as a *cross-attention* layer. Cross-attention is used to model dependencies between different sequences, such as the relationship between an image and a caption or between two sentences in different languages for a machine translation task.

Similar to convolutions, attention layers accept inputs of arbitrary size. However, they scale quadratically with the length of the input, which makes them impractical for very long sequences. Alternatives to the attention layer described above have been proposed that offer improved scalability while retaining the ability to model long-range dependencies [6, 7, 8].

2.5.3 Normalization and Skip Connections

Robustly propagating the training signal over a very deep network is a difficult task. Updates to upstream and downstream layers cause constant changes to the semantics of inputs and outputs, a phenomenon sometimes referred to as *internal covariate shift* [9]. Moreover, as the gradient is accumulated over many layers, it can become very small, which is known as

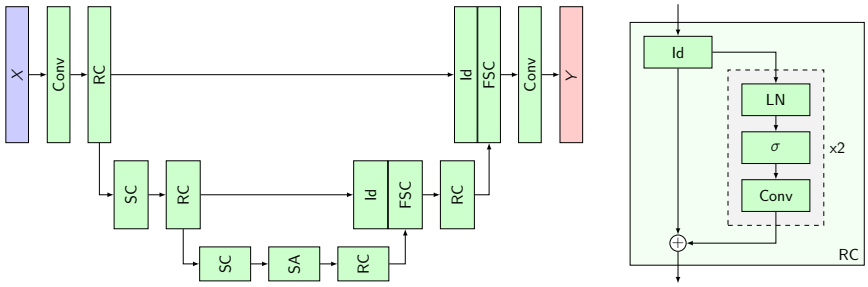

Figure 2: Schematic of the U-Net architecture. Blocks represent nodes in the computational graph. Block heights correspond to the spatial dimension of the data, and block widths correspond to the size of the feature dimension. Labels indicate how the data is transformed by each block. Conv: convolution; FSC: fractionally strided convolution; Id: identity transformation; LN: layer (feature axis) normalization; RC: residual convolution (right panel); SA: self-attention; SC: strided convolution; σ: sigmoid activation.

the *vanishing gradient problem* [10]. The vanishing gradient problem is especially prevalent when activation functions have small bounded gradients, such as the sigmoid function [11]. Conversely, when activation functions are unbounded, the gradient can sometimes instead become very large, which is known as the *exploding gradient problem* [12].

One way to stabilize gradient propagation through deep networks is to use *normalization layers*. A normalization layer takes the output of a layer and transforms it into a new output with a certain distribution. The normalization can be applied over different axes of the data. A common choice is the batch axis [9]. However, batch statistics can be noisy when the mini-batch size is small. In this case, an alternative is to normalize the output over the feature axis instead [13].

Another way to stabilize gradient propagation is to use *skip connections*. A skip connection bypasses one or more layers in the network by connecting the output of a layer directly to an input further downstream. For example, a common skip connection is the residual connection, $y = x + f(x)$ [14]. Skip connections introduce shortcuts in the computational graph that reduce the effective depth of the network and shortens gradient flow.

2.5.4 Putting It Together: The U-Net Architecture

So how can we combine the building blocks described so far into a deep neural network? A good example of how this can be done is the U-Net architecture, which was originally proposed in 2015 for medical image segmentation [15]. A typical implementation, inspired by more recent adaptations

[16], may look like the schematic in Figure 2. The computational graph is defined over multiple levels of resolutions, starting at the resolution of the input data. In the *contractive path*, the data is successively downsampled using strided convolutions until it reaches the level of the lowest resolution. In the *expansive path*, the data is successively upsampled back to the input resolution using fractionally strided convolutions. The contractive and expansive paths together form a U-shape, and they are connected at each level by a skip connection that concatenates the data from the contractive path to the upsampled data from the lower levels in the expansive path.

The U-Net is designed to capture global features in the lower levels without sacrificing higher-resolution details thanks to the skip connections at the upper levels. The network efficiently captures global features at the lower levels because the receptive fields of the convolutional layers cover a larger portion of the lower-resolution data volumes. Many implementations also add a (residual) self-attention layer at the lower levels, where they are computationally less expensive, to further improve the ability of the network to capture long-range dependencies.

Since the U-Net was introduced, it has become a mainstay architecture in the computer vision field. It can be used for any task that requires the output to have the same resolution as the input. Besides segmentation, the U-Net is commonly used in generative models, which will be the focus of the next section. Indeed, many of the state-of-the-art generative text-to-image models that exist today [17, 18, 19] are based on U-Net architectures not too different from the one described here.

3 Generative Models

A fundamental approach to solving many types of pattern recognition problems is to learn the generative process of the data. In this setting, we set up a *model* of how the data is produced and observed, and then learn the parameters of the model from data. If the model is a good approximation of the true generative process, it can be used to simulate new data points or, as we will see in Section 6, to infer properties of the generative process that are not directly observed.

In this section, we will describe some of the most widely used generative models in deep learning and spatial biology. We begin by describing the maximum likelihood objective in Section 3.1 and the limitations of learning generative models using this objective. Our discussion leads us to different modeling strategies to overcome these limitations, which will be the focus of Sections 3.2 to 3.5.

3.1 Maximum Likelihood Estimation

Suppose we are interested in modeling a data distribution $q(X)$ from which we have n observations: $\mathcal{D} = \{X_1, \ldots, X_n\}$. In order to do so, we set up a model $p(X)$. Our goal is to find a model distribution that is as similar to the data distribution as possible. While there are many ways to measure the similarity between two distributions, one common way is to use a distance measure known as the *Kullback-Leibler divergence* (KL divergence) [20]:

$$\mathrm{D}_{\mathrm{KL}}(q(X) \,||\, p(X)) = \int q(X) \log \frac{q(X)}{p(X)} \, \mathrm{d}X \tag{3.1}$$

$$= \mathbb{E}_{X \sim q(X)} \left[\log q(X) - \log p(X) \right]. \tag{3.2}$$

Minimizing the KL divergence can be formalized as the following optimization problem:

$$p(X) = \operatorname*{argmin}_{p \in \mathcal{P}} \; \mathbb{E}_{X \sim q(X)} \left[\log q(X) - \log p(X) \right] \tag{3.3}$$

$$= \operatorname*{argmin}_{p \in \mathcal{P}} \; \mathbb{E}_{X \sim q(X)} \left[\log q(X) \right] - \mathbb{E}_{X \sim q(X)} \left[\log p(X) \right] \tag{3.4}$$

$$= \operatorname*{argmax}_{p \in \mathcal{P}} \; \mathbb{E}_{X \sim q(X)} \left[\log p(X) \right], \tag{3.5}$$

where $\mathcal{P}$ is the family of all possible model distributions; our search space. Eq. (3.5) states that the model $p(X)$ that minimizes the KL divergence is the one that maximizes the expected log-density of the data under the model.

Before we can proceed, we need to find a suitable search space $\mathcal{P}$ amenable to optimization. Suppose we restrict ourselves to a family of parametric distributions $\mathcal{P} = \{p_\theta(X)\}_\theta$ that are parameterized by θ. Eq. (3.5) then leads to the following objective function:

$$L(\theta) = -\mathbb{E}_{X\sim q(X)}\left[\log p_\theta(X)\right] \tag{3.6}$$

$$\approx -\frac{1}{n}\sum_{i=1}^{n}\log p_\theta(X_i), \tag{3.7}$$

where we have used Monte Carlo sampling to approximate the expectation. Eq. (3.7) is the negative *log-likelihood* of the observed data as a function of the model parameters θ. Estimating the model parameters by minimizing this function is known as *maximum likelihood estimation* (MLE).

A central decision is how to choose a good parameterization of the model distribution. Knowing how powerful neural networks are, as we have seen in Section 2, one would be tempted to let $\log p_\theta(X) = f_\theta(X)$ be a neural network that takes X as input and outputs a scalar representing the log-density of X under the model. Unfortunately, this solution is not possible, as it would not result in a valid probability distribution: probabilities are bounded between zero and one, which means $p_\theta(X)$ must integrate to one over the domain of X, but our neural network does not account for this constraint. We will explore two different options for how to deal with this problem: The first is to constrain the parameterization in a way that guarantees a valid probability distribution. This is the approach taken by autoregressive models and normalizing flows, which we will discuss in Sections 3.2 and 3.3. The second option is to change the objective to a slightly different target, which will be the focus of Sections 3.4 and 3.5, describing variational auto encoders and diffusion models, respectively.

3.1.1 Generative Models for Prediction

Before we turn to more complex distributions, let us consider the case of scalar-valued data. A simple choice for p_θ could be to use a Gaussian distribution, which we already know is a valid probability distribution. In this case, the model can be parameterized by a mean μ and a variance σ^2. Furthermore, suppose we for some reason have good reason to believe that the data has unit variance, so we are only interested in learning the mean. We can then write log-likelihood as:

$$\ell(\mu) = \frac{1}{n}\sum_{i=1}^{n}\log\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}(x_i-\mu)^2\right) \tag{3.8}$$

$$= \log \frac{1}{\sqrt{2\pi}} + \frac{1}{n} \sum_{i=1}^{n} \left(-\frac{1}{2}(x_i - \mu)^2 \right) \tag{3.9}$$

$$= -\frac{1}{2n} \sum_{i=1}^{n} (x_i - \mu)^2 + C, \tag{3.10}$$

where C is a constant independent of μ.[1] We recognize Eq. (3.10) from our MLP example in Section 2.2: Maximizing this objective is equivalent to minimizing the mean squared error. In fact, prediction tasks can generally be set up as generative modeling problems, where the goal is to learn a conditional distribution $p_\theta(y \mid X)$. The learned distribution can then be used to predict the value of y given an input X. A significant advantage of setting up a prediction model in this way is that it allows us to easily incorporate uncertainty in a principled way. For example, if we would let σ^2 be a learned parameter, we could use the variance of the model distribution to quantify prediction uncertainty.

3.2 Autoregressive Models

Many interesting data types are not scalar-valued but multi-variate, and we are often interested in understanding the dependencies between the variables. For instance, consider a dataset of images, where each image is represented as a vector of n pixels $X = [x_1, \ldots, x_n]$. The pixels are not independent—if they were, the images would look like nothing but noise—but rather exhibit strong spatial correlation. In this case, a unimodal distribution, like the Gaussian distribution used in Section 3.1.1, is unlikely to adequately capture the dependency structure of the data. An alternative is to factorize the joint distribution into a product of n conditional distributions:

$$p_\theta(X) = \prod_{i=1}^{n} p_\theta^{(i)}(x_i \mid X_{<i}), \tag{3.11}$$

where $X_{<i} = [x_1, \ldots, x_{i-1}]$. The conditionals $p_\theta^{(i)}$ can then be parameterized by simple distributions, such as Gaussians, with parameters computed by a neural network that takes $X_{<i}$ as input:

$$p_\theta(x_i \mid X_{<i}) = N(x_i \mid \mu_i, \sigma_i) \tag{3.12}$$

$$\mu_i, \sigma_i = f_\theta(X_{<i}). \tag{3.13}$$

[1] The constant C does not affect where the minimum is located. In gradient descent, it will disappear when computing the gradient.

Generative models that use ancestral sampling of observed variables as described by Eq. (3.11) are known as *autoregressive models*. Autoregression is a powerful strategy that decomposes a complex distribution into tractable conditionals.

A problem with the autoregressive structure is that Eq. (3.11) suggests that we need to compute n forward passes, one for each conditional, to evaluate $p_\theta(X)$. Autoregressive models typically circumvent this problem by the use of *masking*, where all conditionals are computed in parallel using a single forward pass, but the computation graph is masked so that each conditional only has access to the variables that are already known. For example, in attention models, the masking can be implemented by setting the attention weights for future positions to zero [5]. Nevertheless, while the evaluation of $p_\theta(X)$ can be done in an efficient manner by the use of masking strategies, sampling from $p_\theta(X)$ still requires us to sample each x_i in turn, as we need to condition on the previous samples $X_{<i}$ that are not yet known.

Autoregressive models have been especially successful in modeling data with strong sequential dependencies, such as text. They are therefore used in many natural language processing tasks and widely adopted by current state-of-the-art large language models [21, 22].

3.3 Normalizing Flows

Another way to model distributions with a strong covariance structure is to transform a tractable base distribution, such as a multivariate Gaussian, into a more complex distribution. Suppose we have a continuous base distribution $p(x)$ and a bijection $f : x \to y$ defined on its support, and we want to compute the density of y. Assume first that f is increasing. The cumulative distribution function (CDF) of y is then given by:

$$F_y(a) = p(y \leq a) = p(f(x) \leq a) = p(x \leq f^{-1}(a)) = F_x(f^{-1}(a)). \quad (3.14)$$

The density of y is the derivative of the CDF, giving us:

$$\begin{aligned} p(y) &= \frac{\mathrm{d}}{\mathrm{d}y} F_y(y) && (3.15)\\ &= \frac{\mathrm{d}}{\mathrm{d}y} F_x(f^{-1}(y)) && (3.16)\\ &= \frac{\mathrm{d}}{\mathrm{d}x} F_x(x) \frac{\mathrm{d}}{\mathrm{d}y} f^{-1}(y) && (3.17)\\ &= p(x) \left[f'(x)\right]^{-1}. && (3.18) \end{aligned}$$

If f is decreasing, we can repeat the preceding analysis to find:

$$p(y) = p(x) \left[-f'(x)\right]^{-1}. \tag{3.19}$$

Combining Eqs. (3.18) and (3.19) gives us the *change of variables formula*:

$$p(y) = p(x) \left|f'(x)\right|^{-1}. \tag{3.20}$$

Eq. (3.20) modifies the density of the base distribution $p(x)$ by the degree to which the transformation f either stretches ($|f'(x)| > 1$) or compresses ($|f'(x)| < 1$) the data space. The change of variables formula can be extended to the multi-variate case analogously, in which case the determinant of the Jacobian replaces the derivative in Eq. (3.20).

A *normalizing flow* [23, 24] is a sequence of invertible mappings that transform a tractable base distribution $p(X_0)$ into a potentially more complex distribution $p(X_T)$:

$$X_0 \sim p(X_0) \tag{3.21}$$

$$X_t = f_t(X_{t-1}), \quad t = 1, \ldots, T \tag{3.22}$$

$$p(X_T) = p(X_{T-1}) \Big/ \underbrace{\left|\det \frac{\partial f_T(X_{T-1})}{\partial X_{T-1}}\right|}_{Z_T} = \frac{p(X_0)}{\prod_{t=1}^{T} Z_t}. \tag{3.23}$$

If we parameterize the transformations with a neural network, we can optimize the normalizing flow by MLE.

A difficulty in constructing a normalizing flow is to ensure that every transformation is invertible and that the Jacobian determinant is not too expensive to compute. A range of different invertible transformations have been proposed in the literature [25, 26, 27, 28]. For example, the *affine coupling layer* [26] is a transformation that partitions the input into two parts, $X = [X_a, X_b]$, and computes the output as:

$$Y_a = X_a \tag{3.24}$$

$$Y_b = X_b \odot \exp(s(X_a)) + t(X_a), \tag{3.25}$$

where s and t are neural networks that take the first partition X_a as input and output scale and translation parameters, respectively, that are applied to the second partition X_b. The coupling layer is invertible, as $X_a = Y_a$ and X_b can be recovered from Y_b by applying the inverse of the scaling and translation operations. The Jacobian is triangular, which means that the determinant can be efficiently computed by taking the product of the

diagonal elements:

$$\det \frac{\partial Y}{\partial X} = \det \begin{bmatrix} \mathbb{I} & 0 \\ \frac{\partial Y_b}{\partial X_a} & \frac{\partial Y_b}{\partial X_b} \end{bmatrix} = \prod_i \exp(s(X_a))_i. \tag{3.26}$$

The normalizing flow introduces a powerful concept, which is to augment the model with *latent variables* that are transformed into data space. Nevertheless, while normalizing flows are flexible enough to approximate any distribution [24], the constraint that transformations must be invertible makes them difficult to design and train in practice. Therefore, we will next look at a class of latent variable models that relaxes this constraint by targeting a slightly different objective than the MLE objective.

3.4 Variational Auto Encoders

Suppose we have access to a data distribution $q(X)$ that we want to model. Variational auto encoders (VAEs) [29] consist of two parts, which we will call the *forward model* $q(X, Z) = q(X)q(Z \mid X)$ and the *backward model* $p(X, Z) = p(Z)p(X \mid Z)$.[2] The forward model maps data points from the data distribution $q(X)$ to latent variables Z through an *encoder* $q(Z \mid X) = q_\theta(Z \mid X)$ parameterized by a neural network with learnable weights θ. The backward model works in the opposite direction by mapping latent variables Z from the base distribution $p(Z)$ to data points X through a *decoder* network $p(X \mid Z) = p_\theta(X \mid Z)$.[3] The base distribution $p(Z)$ is a modeling choice but usually chosen to be a well-behaved, parameter-free distribution, such as a standard Gaussian.

The goal of the VAE is to learn encoder and decoder networks that make the forward and backward models as similar as possible. If the models are similar, then their marginals must also be:

$$p_\theta(X) = \int p_\theta(X, Z)\, \mathrm{d}Z \approx \int q_\phi(X, Z)\, \mathrm{d}Z = q(X). \tag{3.27}$$

Crucially, this means that we can sample new data points that approximately follow the data distribution simply by sampling from $p(Z)$ and then applying the decoder.

---

[2] These terms are borrowed from the literature on diffusion models to keep the terminology consistent with Section 3.5.

[3] For notational simplicity, we have used θ to denote the union of the parameters of the encoder and decoder networks. While it is possible to share weights between them, it is more common to use a separate set of parameters for each.

In order to learn similar forward and backward models, VAEs are trained by minimizing the KL divergence between them:

$$
\begin{aligned}
L(\theta) &= \mathrm{D}_{\mathrm{KL}}(q_\theta(X, Z) \mid\mid p_\theta(X, Z)) && (3.28)\\
&= \mathbb{E}_{q_\theta(X,Z)} \left[\log q_\theta(X, Z) - \log p_\theta(X, Z)\right] && (3.29)\\
&= -\mathbb{E}_{q(X)} \underbrace{\mathbb{E}_{q_\theta(Z|X)} \left[\log p_\theta(X, Z) - \log q_\theta(Z \mid X)\right]}_{\mathrm{ELBO}(X;\theta)} + C. && (3.30)
\end{aligned}
$$

Eq. (3.30) tells us that the VAE objective is to maximize the expected *evidence lower bound* (ELBO) of the model under the data distribution. The name of this objective comes from the fact that the ELBO lower bounds the log-evidence, $\log p_\theta(X)$, of the model since

$$
\begin{aligned}
\mathrm{ELBO}(X;\theta) &= \mathbb{E}_{q_\theta(Z|X)} \left[\log p_\theta(X, Z) - \log q_\theta(Z \mid X)\right] && (3.31)\\
&= \log p_\theta(X) + \mathbb{E}_{q_\theta(Z|X)} \left[\log \frac{p_\theta(Z \mid X)}{q_\theta(Z \mid X)}\right] && (3.32)\\
&\leq \log p_\theta(X) + \log \mathbb{E}_{q_\theta(Z|X)} \left[\frac{p_\theta(Z \mid X)}{q_\theta(Z \mid X)}\right] && (3.33)\\
&= \log p_\theta(X) + \log \int p_\theta(Z \mid X) \, \mathrm{d}Z && (3.34)\\
&= \log p_\theta(X), && (3.35)
\end{aligned}
$$

where Eq. (3.33) follows from Jensen's inequality [30]. In other words, while the MLE objective in autoregressive models and normalizing flows directly maximizes $p_\theta(X)$, the objective in VAEs maximizes only a lower bound of $p_\theta(X)$. In fact, in the VAE, computing $p_\theta(X)$ is intractable, as it requires us to integrate over all possible values of Z. In a way, this is the price we had to pay for relaxing the invertibility constraint of normalizing flows!

To learn the parameters of the encoder and decoder networks with gradient descent, we need to compute the gradient of Eq. (3.30):

$$
\nabla L(\theta) = \mathbb{E}_{q(X)} \nabla \mathbb{E}_{q_\theta(Z|X)} \left[\log q_\theta(Z \mid X) - \log p_\theta(X, Z)\right]. \tag{3.36}
$$

The outer expectation can be approximated by Monte Carlo sampling, similar to what we did with the MLE objective in Section 3.1. However, we are still left with the inner expectation, which is intractable. To solve this, VAEs use a *reparameterization trick* [29]. The idea is to define $Z = g_\theta(\epsilon)$ as a transformation of parameter-free noise ϵ. For example, if $q_\theta(Z \mid X)$ is a diagonal Gaussian with mean $\mu_\theta(X)$ and standard deviation $\sigma_\theta(X)$, we can let $Z = \mu_\theta(X) + \sigma_\theta(X) \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, \mathbb{I})$. Using the law of total expectation, Eq. (3.36) can then be rewritten as:

$$
\nabla L(\theta) = \mathbb{E}_{q(X)} \nabla \mathbb{E}_{q(\epsilon)} \mathbb{E}_{q_\theta(Z|X,\epsilon)} \left[\log q_\theta(Z \mid X) - \log p_\theta(X, Z)\right] \tag{3.37}
$$

$$= \mathbb{E}_{q(X)}\mathbb{E}_{q(\epsilon)}\left[\nabla \log q_\theta(Z = g_\theta(\epsilon) \mid X) - \nabla \log p_\theta(X, Z = g_\theta(\epsilon))\right] \tag{3.38}$$

$$\approx \frac{1}{n}\sum_{i=1}^{n}\left(\nabla \log q_\theta(Z = g_\theta(\epsilon_i) \mid X_i) - \nabla \log p_\theta(X_i, Z = g_\theta(\epsilon_i))\right), \tag{3.39}$$

where $\epsilon_i \sim q(\epsilon)$ and $X_i \sim q(X)$.

To increase the expressiveness of VAEs, different designs of the encoder and decoder networks have been proposed. For example, normalizing flows have successfully been used as encoders [25, 31]. Another promising line of research is to use hierarchical encoders and decoders [32, 33]. Hierarchical VAEs have been shown to be effective image synthesizers [34, 35, 36].

3.5 Diffusion Models

A *diffusion model* can be seen as a type of hierarchical VAE that uses a fixed forward model that gradually adds noise to the data [37, 16] (Figure 3). Letting $X_0 \sim q(X_0)$ be a sample from the data distribution, the forward model is defined over a sequence of T steps:

$$q(X_0, X_1, \ldots, X_T) = q(X_0)\prod_{t=1}^{T} q(X_t \mid X_{t-1}) \tag{3.40}$$

$$q(X_t \mid X_{t-1}) = \mathcal{N}(X_t \mid \sqrt{1-\beta_t}X_{t-1}, \beta_t\mathbb{I}), \quad t = 1, \ldots, T, \tag{3.41}$$

where the coefficients β_t are hyperparameters that define the noise schedule of the diffusion process. The noise schedule is chosen so that the final step of the chain roughly corresponds to isotropic Gaussian noise, $q(X_T) \approx \mathcal{N}(X_T \mid 0, \mathbb{I})$. The backward model is defined as:

$$p(X_0, X_1, \ldots, X_T) = p(X_T)\prod_{t=1}^{T} p(X_{t-1} \mid X_t) \tag{3.42}$$

$$p(X_T) = \mathcal{N}(X_T \mid 0, \mathbb{I}) \tag{3.43}$$

$$p_\theta(X_{t-1} \mid X_t) = \mathcal{N}(X_{t-1} \mid \mu_\theta(X_t, t), \beta_t\mathbb{I}), \quad t = 1, \ldots, T, \tag{3.44}$$

where $\mu_\theta(X_t, t)$ is a neural network that predicts X_{t-1} from X_t. The optimization objective of the diffusion model follows from Eq. (3.30):

$$L(\theta) = \mathbb{E}_q\left[\log q(X_1, \ldots, X_T \mid X_0) - \log p_\theta(X_0, \ldots, X_T)\right] + C \tag{3.45}$$

$$= \underbrace{\mathbb{E}_q[-\log p_\theta(X_0 \mid X_1)]}_{L_0} +$$

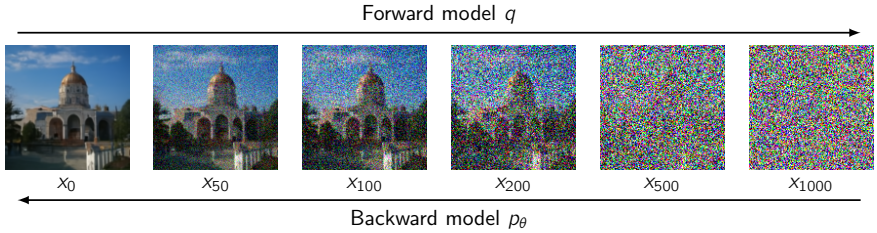

Figure 3: The diffusion noise process. A fixed forward model q gradually adds noise to the data. A backward model p_θ is trained to reverse the forward process.

$$+\sum_{t=2}^{T} \underbrace{\mathbb{E}_q \left[\log q(X_{t-1} \mid X_t, X_0) - \log p_\theta(X_{t-1} \mid X_t)\right]}_{L_{t-1}} + C. \tag{3.46}$$

Any state X_t can be sampled from the forward model without computing the intermediate states, since the distributions $q(X_t \mid X_0)$ are tractable:

$$q(X_t \mid X_0) = \int \prod_{i=1}^{t} q(X_i \mid X_{i-1}) \, \mathrm{d}X_1 \ldots \mathrm{d}X_{t-1} \tag{3.47}$$

$$= \mathcal{N}(X_t \mid \sqrt{\alpha_t} X_0, (1-\alpha_t)\mathbb{I}), \tag{3.48}$$

where $\alpha_t = \prod_{i=1}^{t}(1-\beta_i)$. This allows us to efficiently train very long diffusion chains by approximating the objective function not only with random samples of the data distribution but also random samples of the loss components L_{t-1}.[4]

By reparameterizing Eq. (3.48) as $X_t = X_t(X_0, \epsilon) = \sqrt{\alpha_t} X_0 + \sqrt{1-\alpha_t}\epsilon$, where $\epsilon \sim \mathcal{N}(0, \mathbb{I})$, and defining $\mu_\theta(X_t, t)$ to predict X_{t-1} by estimating ϵ, it can be shown [16] that the loss components can be rewritten as:

$$L_{t-1} = \frac{\beta_t}{2(1-\beta_t)(1-\alpha_t)} \mathbb{E}_{X_0 \sim q(X_0)} \mathbb{E}_{\epsilon \sim \mathcal{N}(0,\mathbb{I})} \left\| \epsilon - \epsilon_\theta(X_t(X_0, \epsilon), t) \right\|^2 + C. \tag{3.49}$$

The gradient of Eq. (3.49) can be approximated by Monte Carlo sampling and used to train the model.

Eq. (3.49) corresponds to the *denoising score matching* objective [38, 16]. Training a diffusion model therefore amounts to learning the score function, the gradient of the log-density of the data, over different noise levels. An

[4] The contribution of L_0 to the objective will usually be very small in long diffusion chains, so we can safely omit it when training the model.

interesting consequence is that the learned model can be used for sampling not only with the backward model in Eq. (3.44) but also with score-based samplers. For example, it can be incorporated in a continuous-time diffusion process that can be sampled from by solving a reverse-time stochastic differential equation (SDE) [39]. Furthermore, the marginal distributions of the SDE can be computed from a corresponding continuous normalizing flow [28], enabling exact evaluation of $p_\theta(X_0)$ [39].

Diffusion models have been extended to discrete data by using alternative noise processes [40, 41]. A disadvantage of diffusion models is that sampling can be computationally expensive, as the diffusion chain is typically several hundred steps long. Recent work has therefore focused on reducing the number of required sampling steps, for example by learning an auxiliary network that can jump between time points of the diffusion process [42, 43].

4 A Modeller's Guide to Cell Biology

In order to construct good models of biological systems, it is important to first have a basic understanding of how such systems work. In this section, we will therefore give a short introduction to cell biology in multi-cellular organisms. Section 4.1 gives a brief overview of the determinants of cell function. Section 4.2 discusses how cells can be classified into different types and states. Finally, in Section 4.3, we describe how cells develop, communicate, and organize to form tissues and organs.

4.1 Cell Function

Our bodies are made up of trillions of cells, each with its own shape and function. Yet, despite their differences, all cells in the body share the same genetic code. How is this possible? To answer this question, let's take a step back and look at the machinery of the cell.

Living organisms are composed of cells, which are the basic building blocks of all life on earth. Cells in multi-cellular organisms are made up of a number of different components, including a cell membrane, which separates the cell from its surroundings, the cytoplasm, which is a gel-like substance that fills the cell, and a nucleus, which contains the genetic code. The genetic code is stored in deoxyribonucleic acid, a long polymer more commonly known as DNA, which is made up of the four nucleotides adenine (A), cytosine (C), guanine (G), and thymine (T). The nucleotides are arranged in a double-stranded helix, each strand consisting of a sequence of nucleotides that bind complementarily to the other strand through hydrogen bonds: A to T and C to G. The sequence of these nucleotides in the DNA, which is unique to each individual, is what we refer to as the *genome*.

The genome is not directly involved in the day-to-day operations of the cell. In fact, the main functional effectors of the cell are proteins, which are large amino acid chains that are involved in almost all cellular processes. The genome directs protein production through a process known as *protein synthesis*, which consists of two steps: *transcription* and *translation*.

During protein synthesis, the DNA of a *gene*—a particular segment of the genetic code—is first transcribed into ribonucleic acid (RNA) by an enzyme known as RNA polymerase. The RNA molecule carries the same genetic information as the transcribed gene. Its structure is similar to DNA, but it is only single-stranded and contains the nucleotide uracil (U) instead of thymine (T). After transcription, the RNA molecule, which is referred to as messenger RNA (mRNA) to emphasize its role as a carrier of genetic information, is exported from the nucleus to the cytoplasm, where it is

picked up by an organelle known as the ribosome. The ribosome then translates the mRNA into a protein by reading the mRNA sequence three nucleotides at a time. Each nucleotide triplet, known as a codon, codes for a particular amino acid, which is added to the growing protein chain.

We are now ready to answer the question of how cells with the same genetic code can have vastly different shapes and functions: The genome is not the only determinant of protein synthesis and, ultimately, cell function; instead, any mechanism that influences the transcription of mRNA or the translation of mRNA into protein will also play an important role. These mechanisms, collectively known as *gene regulation*, determine which proteins are produced in a cell and to what extent. For example, the *epigenome* is a set of chemical modifications to the DNA and the proteins that package the DNA, known as histones, that can influence how accessible the DNA is to RNA polymerase. There are also other regulators that influence transcription and, in the end, determine the *transcriptome* of the cell: the set of all RNA molecules. Similarly, yet other mechanisms exist that affect the translation of mRNA into protein and the fate of the protein once it has been produced, determining the *proteome* of the cell: the set of all proteins. Therefore, while different cells may share the same genome, they can have different transcriptomes and proteomes, which influence how they function.

It is important to mention that cell function is not only determined by the protein-coding genes but many other factors as well. Notably, the importance of RNA molecules that do not code for proteins, known as non-coding RNA, has become increasingly clear in recent years and is today an active area of research [44]. Therefore, when studying cell function, we often talk about the *gene expression* of a cell, which refers to the process by which genetic information is translated into any functional gene product, including both proteins and non-coding RNA.

4.2 Cell Types and Cell States

Cells in the human body are highly diverse. For example, some cells measure only a few micrometer in size, like the red blood cells, while other cells, like the neurons that form the spinal cord, can be over one meter in length. The shapes of cells also vary greatly, from the round shape of an egg cell to the long, thin shape of a muscle cell. The morphology of the cell is intimately linked to its function. Some cells, such as the immune cells, are highly mobile and can move around in the body in order to detect and fight off pathogens, while other cells, such as the epithelial cells that line the surface of the skin, are stationary, forming a protective barrier against the outside world.

Cells can differ along many biological axes, and it is therefore often useful to talk about discrete categories of cells, or *cell types*. A cell type is a taxonomic unit that groups cells based on shared attributes. The appropriate grouping of cells into cell types depends on the context in which the groups are studied. For example, when characterizing the tumor microenvironment, it may sometimes be useful to group cells into very broad categories, such as tumor cells on the one hand and immune cells on the other. In other cases, if we are interested more specifically in how the immune system is responding to a tumor, it may be appropriate to divide the immune cells into subcategories, such as B cells and T cells, or even further into finer subtypes, like CD4+ and CD8+ T cells. In general, there is not a single correct way to group cells into types; a cell type should be understood mostly as an abstraction that allows us to reason about a biological system in a more structured way.

Whichever grouping of cells into types we end up using for our research question, it is important to keep in mind that cells within each type will not be homogeneous. It is therefore helpful to also talk about a related concept to the cell type: the *cell state*. While this word has been used in many different ways in the literature, we will use it here to refer to a point in a continuous, high-dimensional space that exhaustively characterizes the internal properties of a cell. In this framework, a cell type can, for example, be seen as a particular region in cell state space that circumscribes cells with a certain set of features.

4.3 Cell Development and the Organization of Tissues

All multi-cellular organisms start their existence as a single cell, yet develop into complex organisms composed of many different cell types. The process by which cells develop into more specialized types is known as *differentiation*. It is a stochastic process that is influenced by both the cell's internal state and its environment.

The British biologist Conrad Waddington famously described cell differentiation as a marble rolling down a hill, the so called Waddington's epigenetic landscape [45] (Figure 4). The marble, which represents the cell, starts from the top of the hill, in a state of high potential energy. At the top, it can see land extending in all directions. In this state, the cell is *pluripotent*. The marble can go anywhere, become anything; the possibilities are seemingly endless. As the marble begins rolling down the hill, however, its potential energy starts being converted into kinetic energy. The marble loses altitude and gains momentum. The cell starts changing. It jumps about, all over the place. It settles into a grove, representing a cell trajectory, but bounces

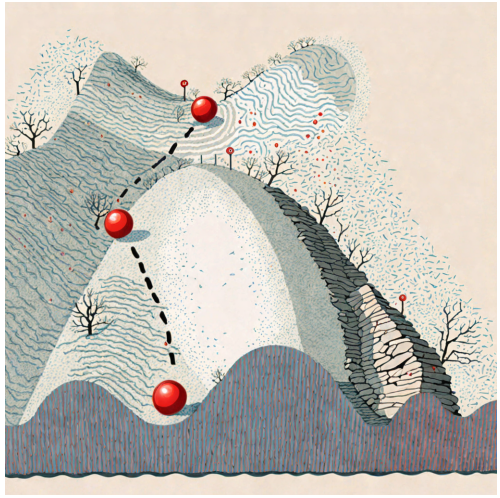

Figure 4: Waddington's epigenetic landscape. Adaptation of the original illustration by Conrad Waddington [45] generated with SDXL [19].

into a different one. Soon, it finds itself at a branching point. There, it collides with a tree and turns into yet another direction, a new trajectory, and then another. Eventually, friction starts slowing the marble down as it is approaching the bottom of the hill. Both its potential and kinetic energy are quickly being exhausted and the marble finally stops, completely: The cell has found its end state, its *cell fate*.

Waddington's landscape is a colorful metaphor for understanding how cells develop into different types. The concept of mapping cell states to an energy landscape is a useful model of how states can be more or less stable and of how perturbations to cells can push them to overcome energy barriers and transition to new states. Nevertheless, to the extent that this model is a realistic one, the energy landscape is at least much less static than historically thought. Notably, environmental factors in the surroundings of the cell can cause drastic changes in the landscape. For example, while differentiation is often thought of as a one-way process, as suggested by the sloping hills of Waddington's landscape, where cells develop from a pluripotent state into a more specialized one, we now know that differentiation in many cases is reversible through a process known as *dedifferentiation*. In fact, dedifferentiation plays an important role, for instance, in wound healing [46] and metastatic tumor growth [47]. It is also possible to artificially induce dedifferentiation in the lab by exposing cells to specific environmental cues [48].

Cells both affect and are affected by their environment. Based on envi-

ronmental cues from other cells and external stimuli, cells organize to form *tissues*, which are groups of cells with similar structure and function. Different tissues, in turn, combine to form *organs*, which are collections of tissues that work together to perform a more complex function. For example, the heart is an organ composed of several different tissues, including cardiac muscle tissue, which is responsible for pumping blood through the body, and connective tissue, which provides structural support for the chambers and valves of the heart. The process by which communities of cells organize to form larger structures is known as *morphogenesis*. It is an important component not only in the development of organisms but also in tissue repair and maintenance.

In a mature tissue, the complex interplay between cells and their environment is full of negative feedback loops that keep tissue growth in check and make sure that the composition and function of the tissue remains stable. Some of the most devastating diseases are characterized by a breakdown of those feedback loops. The breakdown can be initiated by a single cell but, due to the interconnectedness of the cells in the tissue, quickly spreads and affects the entire system. For example, in cancer, a single cell can acquire a mutation that causes it to divide and grow uncontrollably. The onset of this rapid growth may cause the dividing cells to secrete growth factors that stimulate the recruitment and growth of other cells, triggering a cascade of events that ultimately leads to the formation of a tumor [49].

In many ways, cells are like sailors on a boat; they depend on each other to stay afloat. Their actions and the outcome of those actions can be understood only in the context of their collective existence.

5 Spatially Resolved Transcriptomics

Considering the importance of spatial organization in biology, it is not surprising that there is a large interest in the research community of measuring biological systems in a spatially resolved manner. Indeed, studying the organization of tissues has been a central focus of biological research for centuries, dating back to the early days of microscopy, when Robert Hooke and Antonie van Leeuwenhoek observed the first cells in the 17th century [50, 51]. Modern microscopes are much more powerful than those used by Hooke and van Leeuwenhoek, but the technology has not changed much since then and is still an invaluable tool in the field of *histology*: the study of small-scale tissue anatomy. Indeed, histological microscopy images are routinely used in the clinics for diagnosing disease and guiding treatment decisions.

While morphology is highly informative about the state of a tissue, it may not be sufficient for fully characterizing it. Importantly, as discussed in Section 4.1, cell function is largely governed by gene expression. Quantifying gene expression is therefore a powerful tool for understanding how a biological system works. One common way of profiling gene expression is to quantify the transcriptome of cells by measuring the abundance of different RNA molecules. When the measurement has a spatial component, profiling different regions of the same tissue, we refer to it as *spatially resolved transcriptomics* (SRT).

In this section, we will give an overview of the two main approaches to SRT: Section 5.1 discusses what we will refer to as *imaging-based* SRT, which uses microscopy to measure the abundance of RNA molecules. In Section 5.2, we introduce *sequencing-based* SRT, which tags the RNA molecules with spatial barcodes before reading them with a technology known as next-generation sequencing (NGS). Finally, in Section 5.3, we conclude by discussing the trade-offs between imaging- and sequencing-based SRT.

5.1 Imaging-based SRT

Imaging-based SRT relies on a technique known as *in situ hybridization* (ISH), which was first developed in the 1960s [52]. ISH experiments are conducted on tissue sections, which are thin slices of tissue that are commonly 5–20 micrometer thick. In a typical workflow, labeled probes are first deposited on the tissue surface. The probes are short DNA or RNA fragments designed to bind complementarily to a target sequence in the RNA of interest and labeled with a molecule that can be detected with a microscope. The most common labeling molecule is a fluorophore, which emits light when excited with a specific wavelength of light, but other types

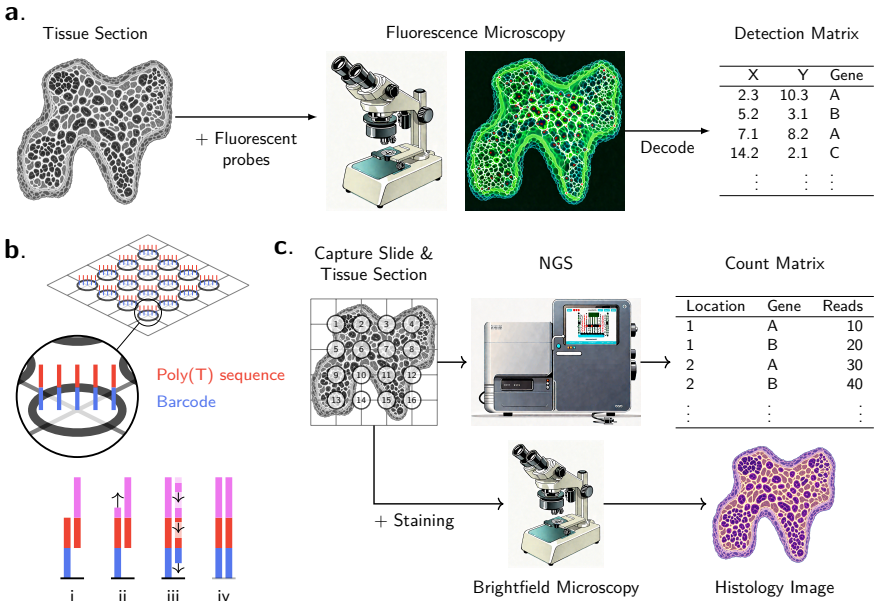

Figure 5: Spatially resolved transcriptomics. **a**. Imaging-based SRT workflow. The tissue is first sectioned and stained with fluorescent probes that bind to the targeted RNA molecules. The tissue is then imaged with a microscope, and the images are analyzed to localize the RNA molecules. The final output is a table listing the spatial coordinates of each RNA molecule. **b**. Sequencing-based SRT capture slide. Top: DNA capture probes are printed or otherwise deposited on a solid surface, such as a glass slide. Each probe has a poly(T) sequence that binds the RNA and a spatial barcode, which is associated to a specific region of the tissue. Bottom: Captured RNA (i) is incorporated into the probe by reverse transcription (ii) and replaced by a second strand of DNA (iii). The final DNA product (iv), which includes the spatial barcode, is then cleaved from the surface and ready for sequencing. **c**. Sequencing-based SRT workflow for fresh-frozen tissue. The tissue is first sectioned and placed on the capture slide, where it can be imaged with brightfield microscopy to visualize its morphology. The tissue is then lysed and the RNA molecules diffuse to the surface of the slide, where they are captured. The captured molecules are sequenced, and the spatial barcodes are used to assign each molecule to a specific region of the tissue. The final output is a count matrix, quantifying the RNA molecules found in each tissue region.

of labels, such as radioactive isotopes, have also been used [52]. After depositing the probes, excess probes are washed away and the tissue is imaged with a microscope. The resulting image is subsequently analyzed to localize the probes in the tissue, producing a table listing the spatial coordinates of each detected probe, and, by extension, the RNA molecules that they bind to (Figure 5a).

The number of genes that can be detected in a single ISH experiment is determined by the labeling scheme. The first fluorescent ISH experiments used a single probe to detect a single RNA species [53]. Later methods have used probes emitting different colors of light that target separate RNA species, allowing experiments to be multiplexed over 10–20 genes simultaneously [54]. Recently, methods have been developed that use combinatorial labeling to associate different RNA species with a unique binary code. First, the RNA molecules are labeled with a set of encoding probes, each containing a target sequence for the RNA and a readout sequence. Each position in the binary code encodes the presence or absence of a specific readout sequence. The code is then read out by sequentially hybridizing the tissue with fluorescent probes targeting the readout sequences in turn and imaging the tissue after every hybridization round. Depending on the number of hybridization rounds used, thousands of genes can be profiled in a single experiment [55]. Experiments can also be multiplexed over multiple fluorescent channels to support larger gene panels [56].

5.2 Sequencing-based SRT

While imaging-based SRT quantifies the RNA molecules directly on the tissue section, sequencing-based SRT first extracts the molecules from the tissue. Instead of designing specific probes for each gene, the identities of the genes are determined by sequencing the extracted molecules; that is, by directly reading the nucleotide sequence using *next-generation sequencing* (NGS) [57].

Sequencing-based SRT was first introduced in 2016 [58]. In the original method, DNA probes are printed in small circular regions on a glass slide (Figure 5b). Each region is 100 µm in diameter and spaced in a regular grid with a center-to-center distance of 200 µm. The probes consist of a spatial barcode, identifying the region where the probe is located, and a poly(T) sequence (a sequence of repeating T nucleotides). In a typical workflow, a tissue section is placed on the glass slide and imaged with brightfield microscopy to visualize its morphology. Next, the tissue is lysed, which releases the RNA molecules from the cells and allows them to diffuse down to the surface of the slide. RNAs with a poly(A) tail bind to the

poly(T) sequence of the probes, and the molecules are combined into a single complementary DNA (cDNA) product by reverse transcription. The cDNA is cleaved from the slide and sequenced with NGS, providing a readout of both the spatial barcode from the probe and the gene sequence from the RNA (Figure 5c). Since most mRNA undergo a process known as *polyadenylation* after transcription, where a poly(A) tail is added to the end of the RNA molecule, sequencing-based SRT can be used to profile almost all protein-coding genes in the transcriptome in a single experiment.

Sequencing-based SRT was originally designed for fresh-frozen tissue, which is tissue that has been preserved by freezing it immediately after excision. It has since also been adapted for formalin-fixed paraffin-embedded (FFPE) tissue [59], which is more common in clinical settings. In FFPE tissue, RNA molecules are often fragmented, preventing poly(T) capture. Therefore, in order to capture the mRNAs on the slide surface, they are first hybridized with a set of probes targeting specific gene sequences and carrying a poly(A) tail for binding to the surface probes. This modified workflow also allows sequencing-based SRT to be extended to non-polyadenylated RNA species, such as many non-coding RNAs.

Recent methods have experimented with different strategies for assigning barcodes to the capture probes in order to increase the spatial resolution of the measurements. A common strategy is to assign random barcodes to the probes and to sequence the barcode once the probes have been deposited on the slide. Since the positions of the barcodes are determined post-hoc, any imprecisions in the manufacturing process of the slide are nullified. Methods using this strategy have been able to achieve a center-to-center distance between measurements of 0.5 µm to 10 µm [60, 61, 62].

5.3 Limitations of Current SRT Technologies

Given the multitude of SRT technologies available today, how should one go about choosing which one to use for a particular experiment? As with many things in life, the age-old adage of "it depends" applies. Importantly, SRT technologies differ along three main axes: sensitivity, spatial resolution, and multiplexing capacity. Historically, imaging-based technologies have excelled in sensitivity and spatial resolution but have been limited to the detection of very few genes. Conversely, sequencing-based technologies have been able to detect many genes at once but have had lower sensitivity and spatial resolution.

As we have seen in Section 5.1, recent imaging-based methods can be multiplexed to support the detection of larger gene panels. However, as more genes are included, the number of RNA molecules targeted per tissue area

also becomes increasingly higher. Consequently, individual molecules may become difficult to resolve in the microscope, an effect known as *molecular crowding*, leading to a loss in sensitivity. Combinatorial labeling partially alleviates this problem, since only a subset of the genes will be visible in each hybridization round. However, as more hybridization rounds are used, the time required for imaging the tissue increases, making highly multiplexed experiments impractical in many settings.

Conversely, as noted in Section 5.2, sequencing-based methods have seen a rapid increase in spatial resolution over the last few years. However, the fact that they rely on diffusion of the RNA molecules or probes down to the capture slide introduces two fundamental limitations: First, diffusion may not only be vertical but also lateral, limiting how spatially precise the measurements can be. Studies have estimated lateral diffusion to be on the order of a few micrometer [58, 62, 63]. Second, sterical hindrance may prevent the RNA molecules from reaching the capture surface, leading to a loss in sensitivity.

In sum, the SRT methods that exist today still trade off between two competing attributes: sensitivity and multiplexing capacity. Therefore, choosing which method to use for a particular experiment will depend on the desired trade-off between these two attributes. For example, if the goal is to map out the different domains of a tissue, it may be important to fully characterize the transcriptome of each domain in a data-driven manner, without having to choose which genes to target beforehand. In this case, a sequencing-based method may be preferable. On the other hand, if the goal is to identify rare cell types, it may be more important to optimize for capturing as many RNA molecules of a certain set of marker genes as possible. In this case, an imaging-based method may be preferable.

It should be noted that some imaging-based methods are non-destructive [59], meaning that the tissue, to a large extent, stays intact after the experiment. These methods can be combined with other types of experiments on the same tissue section, including, for example, histological staining and sequencing-based SRT.

Finally, as an alternative to experimentally scaling up SRT to measuring large gene panels at a high sensitivity, it is also possible to computationally infer such measurements from sparse data. But this, among other topics, will be the focus of the next section.

6 Modeling Spatial Biology Data

In this section we will discuss how to construct models of spatial biology data and how those models can be applied to effectively analyze biological systems. We begin by describing a common data type in computational biology, count data, and how such data can be modeled in Section 6.1. Section 6.2 then discusses how to account for expression rate heterogeneity by modeling the cell state as a latent variable. In Sections 6.3 and 6.4 we describe how to decompose mixed expression signals, which is a common feature of sequencing-based SRT data. Section 6.5 proceeds by discussing how to integrate data from different technologies in order to synthesize knowledge across several data sources. Finally, Section 6.6 concludes with a brief discussion about some of the topics that we have not yet covered.

6.1 Count Data

A common data type in biology is *count data*; data that is made up of natural numbers indicating the number of occurrences of a particular event or object. For example, as we saw in Section 5, sequencing-based SRT produces a count matrix, where each entry represents the number of RNA molecules of a certain kind that were detected in a specific region of the tissue. To illustrate how such data can be modeled, we will in this section construct a model for transcript data. Our analysis is not limited to RNA molecules, however; some of the principles that we will discuss can also be applied to other types of data, including, for example, protein expression.

6.1.1 A Basic Model for Count Data

Suppose we are interested in quantifying the expression of a certain gene, say *TP53*, a known tumor suppressor [64], in a population of cells. We will assume that every cell randomly transcribes *TP53* at rate ν, that the mRNA molecules randomly degrade at rate γ, and that all transcription and degradation events are independent. Then, at every moment, the number of *TP53* mRNA molecules, x, in a given cell increases with rate ν and decreases with rate $x\gamma$.[5] Note that x will fluctuate around some steady-state mean: When x is small, the transcription rate will exceed the degradation rate,

[5] Stochastic processes of this kind are studied in a branch of statistics known as queueing theory. This particular case is an $M/M/\infty$ queue, a birth-death process with constant birth rate and linear death rate (in our case, ν and $x\gamma$, respectively). The first part of the name, M/M, refers to the fact that the transcription and degradation processes are memoryless—they depend only on the current state—and the last part, ∞, refers to the fact that there is no limit to the number of molecules subjected to degradation at any time; i.e., the degradation process cannot be saturated.

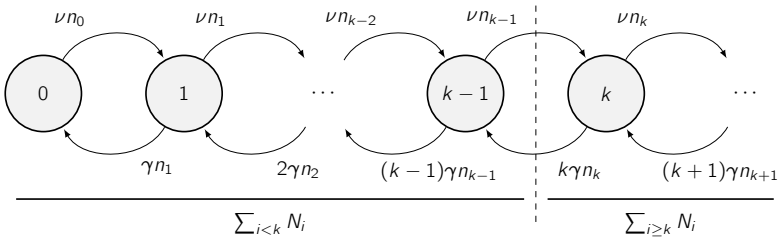

Figure 6: The number of RNA molecules of a certain transcript in a cell can be modeled as a stochastic process governed by the transcription rate ν and degradation rate γ. In steady state, transcription events among cells with $k-1$ molecules are balanced by degradation events among cells with k molecules (dashed line).

leading in expectation to an increase in the number of *TP53* molecules, and vice versa.

What is the distribution of x across a large cell population in steady state? To see how x is distributed, denote by n_k the number of cells with k *TP53* mRNA molecules. At steady state, the rate by which the population of cells with at least k molecules increases equals the rate by which it decreases (Figure 6); in other words,

$$\nu n_{k-1} = k\gamma n_k. \tag{6.1}$$

Eq. (6.1) gives rise to the recursion

$$n_k = \frac{\nu/\gamma}{k} n_{k-1} = \frac{(\nu/\gamma)^2}{k(k-1)} n_{k-2} = \cdots = \frac{(\nu/\gamma)^k}{k!} n_0. \tag{6.2}$$

Let π_k be the proportion of cells with k *TP53* molecules. Using $n_k = \pi_k \sum_i n_i$, Eq. (6.2) can be written in terms of proportions:

$$\pi_k = \frac{(\nu/\gamma)^k}{k!} \pi_0. \tag{6.3}$$

Since the proportions sum to one, it must be the case that

$$1 = \sum_{k=0}^{\infty} \pi_k = \pi_0 \sum_{k=0}^{\infty} \frac{(\nu/\gamma)^k}{k!} = \pi_0 e^{\nu/\gamma} \tag{6.4}$$

$$\implies \pi_0 = e^{-\nu/\gamma}, \tag{6.5}$$

where we have used the Taylor series expansion of the exponential function in Eq. (6.4). Finally, plugging Eq. (6.5) into Eq. (6.3) gives us

$$\pi_k = \frac{(\nu/\gamma)^k}{k!} e^{-\nu/\gamma}. \tag{6.6}$$

Therefore, if we pick a cell from the population at random, the distribution of the number of *TP53* molecules in the cell will be given by the mass function $p(x = k) = \pi_k$ for $k \in \mathbb{N}$. This distribution is known as the *Poisson distribution*, named after Siméon Denis Poisson, who first studied it in the early 19th century [65]. It is parameterized by the rate $\lambda = \nu/\gamma$, which we will refer to as the *expression rate* of *TP53* in the context of our model.

The expression rate corresponds to the average number of *TP53* molecules in a cell, since

$$\mathbb{E}[x] = \sum_{k=0}^{\infty} k \frac{\lambda^k}{k!} e^{-\lambda} = \lambda \sum_{k=1}^{\infty} \frac{\lambda^{k-1}}{(k-1)!} e^{-\lambda} = \lambda \sum_{k=0}^{\infty} \pi_k = \lambda. \tag{6.7}$$

An interesting implication of our model is that the variance of the number of *TP53* molecules is also equal to λ:

$$\begin{aligned}
\mathrm{Var}[x] &= \mathbb{E}[x^2] - \mathbb{E}[x]^2 && (6.8)\\
&= \sum_{k=0}^{\infty} k^2 \frac{\lambda^k}{k!} e^{-\lambda} - \lambda^2 && (6.9)\\
&= \lambda \sum_{k=0}^{\infty} (k(k-1) + k) \frac{\lambda^k}{k!} e^{-\lambda} - \lambda^2 && (6.10)\\
&= \lambda^2 \sum_{k=2}^{\infty} \frac{\lambda^{k-2}}{(k-2)!} e^{-\lambda} + \lambda \sum_{k=1}^{\infty} \frac{\lambda^{k-1}}{(k-1)!} e^{-\lambda} - \lambda^2 && (6.11)\\
&= \lambda^2 + \lambda - \lambda^2 = \lambda. && (6.12)
\end{aligned}$$

We will revisit this property in Section 6.1.4.

6.1.2 Estimating the Expression Rate

Armed with our model, we can now ask the question: given a set of independently observed counts $x_1, \ldots, x_N$ of *TP53* from N cells, how can we estimate the expression rate λ? In Section 3.1, we described how to estimate the parameters of a model by optimizing the log-likelihood function, which in our case is given by

$$\ell(\lambda) = \log p_\lambda(x_1, \ldots, x_N) = \sum_{i=1}^{N} \log \mathrm{Poisson}(x_i \mid \lambda) = \sum_{i=1}^{N} x_i \log \lambda - N\lambda + C, \tag{6.13}$$

where C is constant with respect to λ. Whereas Section 3.1 turned to gradient-based optimization, our model has a closed form solution:

$$\frac{\mathrm{d}}{\mathrm{d}\lambda}\ell(\lambda)\bigg|_{\lambda=\lambda^*} = \frac{1}{\lambda^*}\sum_{i=1}^{N} x_i - N = 0 \implies \lambda^* = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{6.14}$$

Note that λ^* maximizes $\ell(\lambda)$, since the second derivative is negative for all $\lambda > 0$. The estimator λ^* is unbiased:

$$\mathbb{E}[\lambda^*] = \mathbb{E}\left[\frac{1}{N}\sum_{i=1}^{N} x_i\right] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[x_i] = \frac{1}{N}\sum_{i=1}^{N}\lambda = \lambda. \tag{6.15}$$

We can also see that the variance of this estimator is inversely proportional to the sample size:

$$\mathrm{Var}[\lambda^*] = \mathrm{Var}\left[\frac{1}{N}\sum_{i=1}^{N} x_i\right] = \frac{1}{N^2}\sum_{i=1}^{N}\mathrm{Var}[x_i] = \frac{\lambda}{N}. \tag{6.16}$$

Let us take a step back and consider what we have achieved so far. We have set up a model $x \sim \mathrm{Poisson}(\lambda)$ for *TP53* expression in a cell and fitted it to observed data by maximum likelihood estimation. Given that our assumptions about the transcription and degradation processes are correct, we have thereby learned an unbiased estimate of the expression rate of *TP53*, which is the relative rate of transcription and degradation. In other words, our model has allowed us to transform observed data into new knowledge about the underlying biological system. Pretty neat, right? But before we get too carried away, let us first dive a little deeper into the assumptions we have made, and what they mean for our model.

6.1.3 Limited Measurement Efficiency

The preceding section assumes that we can observe all *TP53* molecules in a cell. In reality, however, we typically only observe a fraction of the molecules due to inefficiencies in the measurement process. For example, as we saw in Section 5.2, sequencing-based SRT relies on diffusion of the RNA molecules down to the capture slide. During this process, some molecules may get stuck and fail to reach the surface.

To model these inefficiencies, let y be the observed number of molecules. Assume that a cell contains x molecules and that we only observe each molecule with probability ε, the *efficiency* of the measurement technology. Let i_k be a random variable indicating whether molecule k is observed and

assume that all observation events i_k are independent. Then, the probability of the sequence $(i_1, \ldots, i_x)$ is given by the product

$$p(i_1, \ldots, i_x \mid x) = \prod_k \varepsilon^{i_k}(1-\varepsilon)^{1-i_k} = \varepsilon^{\sum_k i_k}(1-\varepsilon)^{x-\sum_k i_k}. \tag{6.17}$$

The conditional distribution of y given x is therefore

$$p(y \mid x) = \mathbb{E}_{i_1,\ldots,i_x|x}\left[p(y \mid i_1, \ldots, i_x)\right] = \sum_{\sum_k i_k = y} p(i_1, \ldots, i_x \mid x) \tag{6.18}$$

$$= \frac{x!}{y!(x-y)!}\varepsilon^y(1-\varepsilon)^{x-y} = \binom{x}{y}\varepsilon^y(1-\varepsilon)^{x-y}, \tag{6.19}$$

which is the *binomial distribution*. The expression $\binom{x}{y}$ is the *binomial coefficient*, and it counts the number of ways the y observed molecules can be picked from the x molecules present in the cell.

We can now compute the unconditional distribution of y:

$$p(y) = \sum_{x=0}^{\infty} p(y \mid x)p(x) = \sum_{x=y}^{\infty}\binom{x}{y}\varepsilon^y(1-\varepsilon)^{x-y}\frac{\lambda^x}{x!}e^{-\lambda} \tag{6.20}$$

$$= \frac{\varepsilon^y\lambda^y}{y!}e^{-\lambda}\sum_{x=y}^{\infty}\frac{\lambda^{x-y}(1-\varepsilon)^{x-y}}{(x-y)!} \tag{6.21}$$

$$= \frac{(\varepsilon\lambda)^y}{y!}e^{-\lambda}\sum_{x=0}^{\infty}\frac{(\lambda(1-\varepsilon))^x}{x!} \tag{6.22}$$

$$= \frac{(\varepsilon\lambda)^y}{y!}e^{-\lambda}e^{\lambda(1-\varepsilon)} = \frac{(\varepsilon\lambda)^y}{y!}e^{-\varepsilon\lambda} = \text{Poisson}(y \mid \varepsilon\lambda). \tag{6.23}$$

Therefore, we can see that even with limited measurement efficiency, the observed counts are still Poisson distributed. However, our estimation procedure in Section 6.1.2 will now not quite find the actual expression rate λ but rather an attenuated rate $\varepsilon\lambda$. Nonetheless, this is typically not a big problem in practice for two reasons: First, if we have an idea of the efficiency, we can simply divide the estimated rate by ε to recover the actual rate. Second, we are often not interested in the absolute value of the expression rate but rather the relative rates between, for example, different biological conditions or, in SRT data, different spatial regions. Such comparisons will still be valid as long as the efficiency is constant.

6.1.4 Expression Rate Heterogeneity

So far, we have assumed that all cells in the population have the same transcription and degradation rates. This is clearly a simplification: Even

if we study cells of a single cell type, their cell states will not be identical and their expression rates of a given gene are likely to vary. What does this mean for our model? Suppose a cell can have one of K different expression rates, $\lambda_1, \ldots, \lambda_K$, with probabilities $p_1, \ldots, p_K$, respectively. Then, the distribution of the number of *TP53* molecules in a cell will be given by the mixture distribution

$$p(x) = \sum_{k=1}^{K} p_k \text{Poisson}(x \mid \lambda_k). \tag{6.24}$$

Let us consider the mean and variance of this distribution. First, note that we can parameterize x as

$$x = \sum_{k=1}^{K} i_k x_k, \tag{6.25}$$

where i_k is an element of the one-hot vector $I = (i_1, \ldots, i_K)$ indicating which of the K expression rates the cell has and $x_k \sim \text{Poisson}(\lambda_k)$. Therefore,

$$\mathbb{E}[x] = \mathbb{E}\sum_{k=1}^{K} i_k x_k = \sum_{k=1}^{K} \mathbb{E}[i_k x_k] = \sum_{k=1}^{K} \mathbb{E}[i_k]\mathbb{E}[x_k] = \sum_{k=1}^{K} p_k \lambda_k = \bar{\lambda} \tag{6.26}$$

$$\text{Var}[x] = \mathbb{E}\left[\left(x - \bar{\lambda}\right)^2\right] = \mathbb{E}_I \mathbb{E}_{x|I}\left[\left(x - \bar{\lambda}\right)^2\right] \tag{6.27}$$

$$= \sum_{k=1}^{K} p_k \mathbb{E}_{x_k}\left[\left(x_k - \bar{\lambda}\right)^2\right] \tag{6.28}$$

$$= \sum_{k=1}^{K} p_k \mathbb{E}_{x_k}\left[\left((x_k - \lambda_k) + (\lambda_k - \bar{\lambda})\right)^2\right] \tag{6.29}$$

$$= \sum_{k=1}^{K} p_k \mathbb{E}_{x_k}\left[(x_k - \lambda_k)^2 + 2(x_k - \lambda_k)(\lambda_k - \bar{\lambda}) + (\lambda_k - \bar{\lambda})^2\right] \tag{6.30}$$

$$= \sum_{k=1}^{K} p_k \left(\text{Var}[x_k] + 2(\lambda_k - \lambda_k)(\lambda_k - \bar{\lambda}) + (\lambda_k - \bar{\lambda})^2\right) \tag{6.31}$$

$$= \bar{\lambda} + \sum_{k=1}^{K} p_k (\lambda_k - \bar{\lambda})^2, \tag{6.32}$$

where the third equality in Eq. (6.26) follows from independence of i_k and x_k. Eq. (6.32) implies that the variance of x is always at least as large as the mean. Hence, if we model the data using a Poisson distribution with expression rate $\lambda = \bar{\lambda}$, we would have underestimated the variance of the

data distribution; in this case, the data is said to be *overdispersed* relative to the model.

One common way to account for overdispersion is to base our model on a *negative binomial* distribution instead of a Poisson distribution [66]. The negative binomial distribution arises naturally as the marginal distribution of a Poisson distribution with a gamma-distributed rate [67]. For a certain sequence of parameters, the gamma distribution converges to a point mass. This means that the negative binomial distribution can model both data with Poisson variance (point mass expression rate) and data that is overdispersed relative to a Poisson model (non-point mass rate).

While the negative binomial distribution is a straightforward way of accounting for overdispersion, the rationale for why the rate parameter would be gamma distributed is not clear. For example, the gamma distribution is a unimodal distribution and may therefore not be able to capture expression rates across a population of cell types with disjoint rates. Moreover, overdispersion may have causes other than expression rate heterogeneity, such as, for instance, technical noise in the measurement process [66, 68]. In general, the specific shape of excess variability captured by a negative binomial model may not match the process by which the variance is generated, which could result in a biased model all the same.

At this point, it should also be noted that gene expression is a highly complex stochastic process. In particular, our assumption of independent transcription and degradation events is unlikely to be true. For example, transcription events are known to occur in bursts, where the gene is transcribed at a high rate for a short period of time followed by a period of inactivity [69]. Additionally, the constant influx of signals from external stimuli may cause perturbations to both transcription and degradation. Gene expression is therefore seldom in steady state, and the distribution of the number of RNA molecules of a certain transcript is likely much more complex than what either the Poisson or negative binomial distributions can capture.

Although it is a good idea to be mindful of the assumptions encoded in the models we use—and their implications for potential bias in subsequent analyses—it is also important to remember that useful models need not be perfect representations of reality but just good enough abstractions thereof. Indeed, as we will see in the next few sections, the Poisson model discussed so far can be extended to capture a wide range of biological phenomena, and similar models have been used to great effect for solving many different analysis tasks in biology.

6.2 Cell State Models

In the last section, we described how to model the number of *TP53* molecules in a cell using a Poisson distribution, and we saw that expression rate heterogeneity can cause issues with overdispersion. One way to address this issue would be to model the expression rate not of a population of cells but of individual cells. However, as shown by Eq. (6.16), if we estimate the expression rate based on a single count value, we will be left with a very noisy estimate. Therefore, we need to find a way to share information across cells or genes in order to obtain more robust results.

One idea is to model the cell state as a latent variable. Let X be a vector of observed counts for a cell. This vector includes counts for all genes in the transcriptome, including *TP53*. Let Z be the corresponding cell state, which is unobserved. The cell state does not correspond to any physical property of the cell but is simply an abstract construct, so we are free to choose how the marginal $p(Z)$ should look like. A typical choice is to let Z follow some well-behaved, tractable distribution, such as a diagonal Gaussian. We can then set up a joint model $p(X, Z) = p(Z)p(X \mid Z)$ for the cell state and observed counts where

$$Z \sim p(Z) \tag{6.33}$$

$$\lambda_g = f_g(Z) \tag{6.34}$$

$$X_g \mid Z \sim \text{Poisson}(\lambda_g). \tag{6.35}$$

The function f_g maps the cell state to the expression rate of gene g. We can parameterize f_g as a neural network with weights θ, for example by letting

$$f_g(Z) = \left[f_\theta(Z)\right]_g, \tag{6.36}$$

where we have used the notation $[\cdot]_g$ to denote the gth element of a vector. Note that the parameters of this model, θ, will be shared both across cells and across genes. If we learn θ by maximizing the likelihood of the model, we may therefore be able to decode the posterior cell state, $p_\theta(Z \mid X)$, into robust estimates of the expression rates of individual genes in individual cells. The learned cell state embedding is also interesting in its own right, since it encodes a representation of the cell that can be used for downstream analyses, such as clustering and visualization [70, 71].

This idea runs into a problem, however: In order to compute the log-likelihood, Eq. (3.7), or the posterior, $p_\theta(Z \mid X) = p_\theta(X, Z)/p_\theta(X)$, we need to be able to evaluate an integral over all possible cell states, since

$$p_\theta(X) = \int p(Z = z)p_\theta(X \mid Z = z)\,\mathrm{d}z \tag{6.37}$$

$$= \int p(Z = z) \prod_{g=1}^{G} \text{Poisson}(X_g \mid [f_\theta(z)]_g) \, \mathrm{d}z. \tag{6.38}$$

This integral is intractable for all but the simplest definitions of $p(Z)$ and f_g. So, how can we proceed? Going back to Section 3.4, one approach is to set up the model as a VAE. In this case, we couple our model $p_\theta(X, Z)$ with an approximate posterior $q_\theta(Z \mid X)$ and learn both distributions by maximizing the evidence lower bound, Eq. (3.30), instead of the likelihood.

Several extensions of this model have been proposed in the literature. For example, the rate in Eq. (6.34) can be augmented with additional covariates to control for batch effects [70]. Another extension is to use a prior distribution $p(Z)$ with multiple modes, allowing the model to associate distinct high-density regions of cell state space to different cell types [71].

6.2.1 The Expression Rate Distribution

A consequence of modeling the expression rate as a random variable is that we can now ask questions about how it is distributed. For example, we may be interested in the *posterior* distribution of the expression rate given an observation X:

$$p_\theta(\lambda_g \mid X) = \int p_\theta(\lambda_g \mid Z) p_\theta(Z \mid X) \, \mathrm{d}Z. \tag{6.39}$$

Eq. (6.39) is, just like the integral in Eq. (6.38), intractable. Nonetheless, we can approximate it with a mixture of point mass distributions by sampling from the approximate posterior $q_\theta(Z \mid X)$:

$$p_\theta(\lambda_g \mid X) \approx \int p_\theta(\lambda_g \mid Z) q_\theta(Z \mid X) \, \mathrm{d}Z \tag{6.40}$$

$$\approx^{6} \int \delta_{f_g(Z)}(\lambda_g) \sum_{i=1}^{N} \delta_{Z_i}(Z) \, \mathrm{d}Z \tag{6.41}$$

$$= \frac{1}{N} \sum_{i=1}^{N} \delta_{f_g(Z_i)}(\lambda_g), \tag{6.42}$$

where δ_x is the Dirac delta distribution centered at x (a deterministic distribution that always takes on the value x), and we have drawn N samples $Z_1, \ldots, Z_N$ from $q_\theta(Z \mid X)$.

[6] The approximatively equal sign here indicates that the left and right hand side densities have similar cumulative distribution functions, not that they are numerically close. This is because the right hand side is not a regular function of λ_g but a generalized function that evaluates to infinity at some points and zero elsewhere.

We can also extend our model to ask questions about how expression rates differ between different conditions. One way to do so would be to augment the latent state with a learnable embedding E_c for each condition c in order to learn a conditional expression model,

$$Z \sim p(Z) \tag{6.43}$$

$$\lambda_{cg} = f_g(Z + E_c) \tag{6.44}$$

$$X_g \mid Z, c \sim \text{Poisson}(\lambda_{cg}). \tag{6.45}$$

After optimizing this model, we can estimate the expression rate distributions of each condition:

$$p(\lambda_{cg}) = \mathbb{E}_{Z \sim p(Z)}[p(\lambda_{cg} \mid Z)] \approx \frac{1}{N} \sum_{i=1}^{N} \delta_{f_g(Z_i + E_c)}(\lambda_{cg}). \tag{6.46}$$

By comparing the rate distributions of different conditions, we can identify genes that are upregulated in, for instance, certain cell types or in response to certain stimuli. Estimating the expression difference between cell populations is generally known as *differential gene expression* analysis. Both latent variable models similar to the one above [70] and other techniques based on, for example, hypothesis testing [72, 66] have been proposed for this purpose.

6.3 Factorization

So far, we have discussed how to model the gene expression of individual cells. Nonetheless, similar models can also be used for spatial data. In spatial models, we can incorporate spatial dependencies in the model by, for instance, using a prior that encourages neighboring observations to have similar expression profiles [73] or non-parametrically by replacing f_g in Eq. (6.34) with a convolutional neural network applied to a grid of spatially ordered latent states [74].

In sequencing-based SRT data, there is, however, one big caveat that we need to deal with first: Each measurement region may capture RNA from multiple cells, giving us a mixed expression signal from cells with potentially very different expression rates. To formalize this, suppose the expression signal in a region i comes from T expression components,

$$x_{itg} \sim \text{Poisson}(\lambda_{itg}), \tag{6.47}$$

where $t \in \{1, \ldots, T\}$ indexes the components and g is a gene index. The observed counts y_{ig} is the sum of the expression components:

$$y_{ig} = \sum_{t=1}^{T} x_{itg}. \tag{6.48}$$

To derive the distribution of y_{ig}, consider first the sum of two Poisson random variables $y = x_1 + x_2$, where $x_1 \sim \text{Poisson}(\lambda_1)$ and $x_2 \sim \text{Poisson}(\lambda_2)$:

$$p(y) = \sum_{k=0}^{\infty} p(x_1 = k)p(y \mid x_1 = k) \tag{6.49}$$

$$= \sum_{k=0}^{\infty} p(x_1 = k)p(x_2 = y - k) \tag{6.50}$$

$$= \sum_{k=0}^{y} \text{Poisson}(k \mid \lambda_1)\text{Poisson}(y - k \mid \lambda_2) \tag{6.51}$$

$$= \sum_{k=0}^{y} \frac{\lambda_1^k}{k!} e^{-\lambda_1} \frac{\lambda_2^{y-k}}{(y-k)!} e^{-\lambda_2} \tag{6.52}$$

$$= \frac{(\lambda_1 + \lambda_2)^y}{y!} e^{-(\lambda_1+\lambda_2)} \sum_{k=0}^{y} \frac{y!}{k!(y-k)!} \frac{\lambda_1^k \lambda_2^{y-k}}{(\lambda_1 + \lambda_2)^y} \tag{6.53}$$

$$= \frac{(\lambda_1 + \lambda_2)^y}{y!} e^{-(\lambda_1+\lambda_2)} \sum_{k=0}^{y} \binom{y}{k} \left(\frac{\lambda_1}{\lambda_1 + \lambda_2}\right)^k \left(\frac{\lambda_2}{\lambda_1 + \lambda_2}\right)^{y-k} \tag{6.54}$$

$$= \frac{(\lambda_1 + \lambda_2)^y}{y!} e^{-(\lambda_1+\lambda_2)} \left(\frac{\lambda_1 + \lambda_2}{\lambda_1 + \lambda_2}\right)^y \tag{6.55}$$

$$= \frac{(\lambda_1 + \lambda_2)^y}{y!} e^{-(\lambda_1+\lambda_2)} = \text{Poisson}(y \mid \lambda_1 + \lambda_2), \tag{6.56}$$

where we have used the binomial theorem in Eq. (6.55). By induction, it follows that

$$x_{i1g} + x_{i2g} \sim \text{Poisson}(\lambda_{i1g} + \lambda_{i2g}) \tag{6.57}$$

$$(x_{i1g} + x_{i2g}) + x_{i3g} \sim \text{Poisson}\left((\lambda_{i1g} + \lambda_{i2g}) + \lambda_{i3g}\right) \tag{6.58}$$

$$\vdots$$

$$y_{ig} = \sum_{t=1}^{T} x_{itg} \sim \text{Poisson}\left(\sum_{t=1}^{T} \lambda_{itg}\right). \tag{6.59}$$

Eq. (6.59) implies that the observed counts y_{ig} are Poisson distributed, just like the component distributions, with a rate equal to the sum of the component rates. Therefore, the likelihood is tractable and we can fit this model just as we have done before to learn the rates of the expression components.

Without additional structure, the model defined by Eq. (6.59) is clearly overparameterized, however, since there are T times more component rates

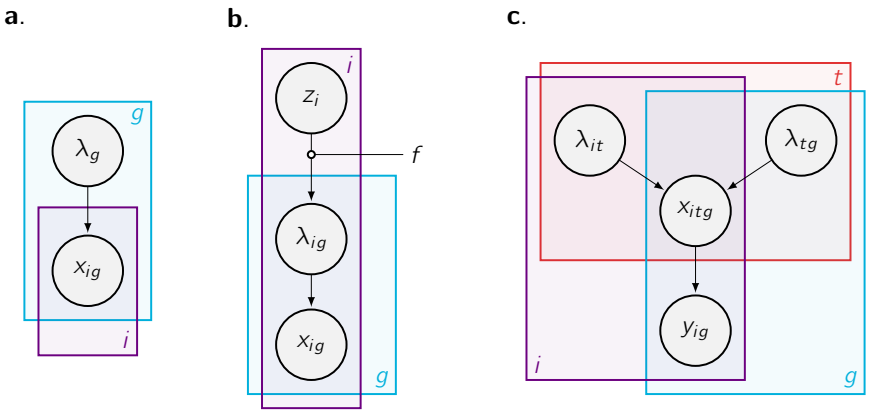

Figure 7: Expression rate models. **a**. Base model (Section 6.1.2). **b**. Cell state model (Section 6.2). **c**. Factorized model (Section 6.3). Indices i, g, and t denote the unit of observation (cell or measurement region), gene, and factor, respectively.

λ_{itg} than observed counts y_{ig}. One way to constrain the model is to assume that the component rates factorize into a spatial term λ_{it} and a gene term λ_{tg} [75]:

$$\lambda_{itg} = \lambda_{it}\lambda_{tg}. \tag{6.60}$$

In this framework, the components t are typically referred to as *factors* or *metagenes*. The spatial term λ_{it} defines the spatial distribution of the factors over the measurement regions, and the gene term λ_{tg} defines the expression profiles of the factors. Factors can be understood as recurrent gene expression patterns and are typically associated with specific cell types or biological processes [76]. Factor models therefore provide a way to interpret gene expression data by decomposing it into biologically meaningful components.

A graphical summary of the models discussed so far is shown in Figure 7.

6.4 Cell Type Deconvolution

The factorized expression model introduced in Section 6.3 presents an exciting opportunity: What if we could learn the expression profiles of the factors from a non-spatial data source profiling the expression of individual cells? The factors would then correspond to known cell types, and we could transfer the expression profiles of the factors to the sequencing-based SRT data in order to deconvolve the expression signal into cell types [77]. Formally, the way this could be set up is to use the single-cell data to pretrain

a conditional expression model,

$$X_{ig} \mid T = t \sim \text{Poisson}(s_i \lambda_{tg}), \tag{6.61}$$

where i and g are cell and gene indices, respectively, and s_i is a scaling factor to control for differences in sequencing depth between cells. This allows us to learn the gene expression profiles λ_{tg} of T different cell types. We can then train a model similar to the one defined in Section 6.3 but fixing the gene term $\bar{\lambda}_{tg} = \lambda_{tg}$ to the pre-trained profiles:

$$x_{itg} \sim \text{Poisson}(s_g \lambda_{it} \bar{\lambda}_{tg}), \tag{6.62}$$

where we have added a gene-wise scaling factor s_g to capture bias in the capture efficiency of different genes between the single-cell and SRT technologies. Letting $\lambda_{itg} = s_g \lambda_{it} \bar{\lambda}_{tg}$ and $\lambda_{ig} = \sum_t \lambda_{itg}$, the posterior component counts are given by

$$p(x_{i1g}, \ldots, x_{iTg} \mid y_{ig}) = \frac{p(x_{i1g}, \ldots, x_{iTg})}{p(y_{ig})} = \frac{p(x_{i1g}) \cdots p(x_{iTg})}{p(y_{ig})} \tag{6.63}$$

$$= \frac{\frac{\lambda_{i1g}^{x_{i1g}}}{x_{i1g}!} e^{-\lambda_{i1g}} \ldots \frac{\lambda_{iTg}^{x_{iTg}}}{x_{iTg}!} e^{-\lambda_{iTg}}}{\frac{\lambda_{ig}^{y_{ig}}}{y_{ig}!} e^{-\lambda_{ig}}} \tag{6.64}$$

$$= \frac{y_{ig}!}{x_{i1g}! \cdots x_{iTg}!} \frac{\lambda_{i1g}^{x_{i1g}} \cdots \lambda_{iTg}^{x_{iTg}}}{\lambda_{ig}^{y_{ig}}} \tag{6.65}$$

$$= \binom{y_{ig}}{x_{i1g}, \ldots, x_{iTg}} \prod_{t=1}^{T} \left(\frac{\lambda_{itg}}{\lambda_{ig}} \right)^{x_{itg}}, \tag{6.66}$$

which is a multinomial distribution with y_{ig} trials and event probabilities $\lambda_{itg}/\lambda_{ig}$. The posterior mean proportion of counts contributed by cell type t in region i is therefore given by

$$\mathbb{E}\left[\frac{\sum_g x_{itg}}{\sum_g y_{ig}} \,\middle|\, y_{i1}, \ldots, y_{iG} \right] = \frac{\sum_g \mathbb{E}[x_{itg} \mid y_{ig}]}{\sum_g y_{ig}} = \frac{\sum_g \frac{\lambda_{itg}}{\lambda_{ig}} y_{ig}}{\sum_g y_{ig}}. \tag{6.67}$$

Estimating cell type proportions from a mixed expression signal is known as *cell type deconvolution*. A wide range of methods have been proposed for this task, including probabilistic inference methods in much the same spirit as what has been described above [77, 78, 79] as well as other estimation strategies [60, 80, 81, 82].

6.5 Multi-Modal Models

The ideas presented in the last few sections illustrate how we can use weight sharing across multiple observations to ameliorate the effects of sparsity and noise in order to obtain more robust estimates. Multi-modal weight sharing, where we have integrated observations from different types of data sources, can be especially effective, since different modalities may be better at characterizing certain aspects of the data or offer other complementary advantages. For example, in the cell type decomposition model of Section 6.5, we leveraged cell-level data from a non-spatial source to inform the SRT data analysis by defining cell-type-specific expression profiles. The potential of cross-modality data transfer is not limited to cell type decomposition, however, and there are many other data types that can be integrated for this purpose.

As mentioned in Section 5, many SRT protocols are compatible with histological staining, producing a dataset of paired spatial gene expression and histology. Intriguingly, it has been shown that it is possible to predict gene expression from histology images [83], suggesting that these modalities have overlapping information content.

One strong differentiator of imaging data compared to sequencing-based SRT data is that it has a much higher resolution, limited only by the diffraction limit of the microscope. This opens up the possibility of transfering higher-resolution information from the imaging data to the SRT data by embedding the datasets in a shared latent space [74, 84]. To formalize this idea, we can set up a cell state model that decodes the shared embedding Z into pixel-level expression rates λ_{xyg} and the image data Y,

$$Z \sim p(Z) \tag{6.68}$$

$$\lambda_{xyg} = [f_\theta(Z)]_{xyg} \tag{6.69}$$

$$x_{xyg} \mid Z \sim \text{Poisson}(\lambda_{xyg}) \tag{6.70}$$

$$x_{ig} = \sum_{(x,y)\in R_i} x_{xyg} \tag{6.71}$$

$$Y \mid Z \sim p(Y \mid Z), \tag{6.72}$$

where x_{xyg} is the count for gene g in pixel (x, y) and R_i is the set of pixels in region i. Note that the shared embedding Z encodes the information content in both modalities and is therefore a high-resolution representation of the data. Moreover, from the divisibility of the Poisson distribution described in Section 6.3, it follows that $x_{ig} \mid Z \sim \text{Poisson}(\sum_{(x,y)\in R_i} \lambda_{xyg})$. With an appropriate image model $p(Y \mid Z)$, we can therefore optimize this model in

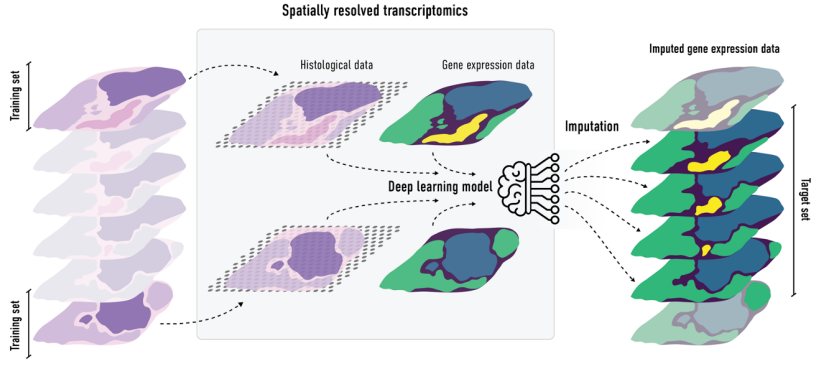

Figure 8: Cross-modality data transfer for scaling up SRT experiments. A tissue is sectioned into sequential tissue slices. All slices are stained and imaged, and SRT is carried out on a small subset of them. A prediction model is trained to predict the SRT data from the stains and then used to infer spatial gene expression on slices that were not subjected to SRT. The resulting data can be stacked to form a three-dimensional expression map of the tissue. Image from: Julia Chen et al. "Single-cell spatial landscapes of cancers: insights and opportunities". In revision.

the same way as the cell state model of Section 6.2, allowing us to infer the super-resolved expression rates λ_{xyg}.

Besides super resolution, multi-modal models also offer a promising means for scaling SRT by inferring experimental outcomes from histology images. To illustrate why this is useful, note that SRT technologies are typically both costly and time-consuming to run. This can, for example, be an impediment to creating three-dimensional expression maps constructed from serial experiments on thin tissue sections: Since each section is only around 10 µm thick, a 1 cm tissue would require around 1000 sections to cover and cost well over 1 M€ by most SRT providers in 2023. In contrast, histology images can be acquired much more quickly and cheaply. By training a model to predict SRT data from histology images, large-scale experiments can be carried out by performing SRT on a small subset of the sections and then using the model to infer the expression of the remaining sections (Figure 8), thereby making it practically much more feasible to characterize three-dimensional transcriptomes.

Recently, several new experimental protocols have been proposed for simultaneously characterizing not only the transcriptome but also the genome [73], proteome [85], and metabolome [86] on the same or adjacent sections. Integrating these data sources promises to provide an even more compre-

hensive picture of the spatial biology of tissues.

6.6 So That's It?

In the preceding sections, we have discussed a number of common analysis tasks in spatial biology, focusing on the ones most relevant to the papers presented in this thesis. However, there are many topics that we have not covered. For example, an important problem for large-scale spatial analyses is how to associate tissue regions across experiments. This can be done by finding a *common coordinate framework* that maps the regions to a shared reference space [87, 88]. A related problem is how to register multiple sections from the same biopsy to each other, which is necessary for three-dimensional analyses [89, 90, 91]. Moreover, in exploratory analyses, it may be useful to find connected spatial regions with similar expression profiles, a task known as *domain segmentation* [73, 92]. Identifying smaller structures, such as individual cells, is a common analysis task when analyzing higher-resolution SRT data, and both morphology-based [93] and gene expression-based [94, 95] methods have been proposed for this purpose. In order to reduce the search space across genes, methods have also been developed to identify *spatially variable genes*; that is, genes that have a high degree of spatial structure in their expression and therefore are likely to be involved in localized functions [96, 97, 98]. Yet another central topic in spatial biology is how to model how cells interact, *cell-cell communication*, which is crucial for understanding many biological processes [99, 100, 101].

Furthermore, our treatment has focused on how to analyze biological systems using probabilistic models. While probabilistic modeling has been applied effectively to many tasks in spatial biology, it should be noted that it is not the only approach and it is not always the best. For example, one drawback of many inference algorithms in probabilistic models, including those discussed in this thesis, is that they are usually computationally quite expensive. Therefore, in exploratory analyses, where the goal may be to iterate over many different biological hypotheses to quickly get a sense of the data, it may be more appropriate to use faster methods.

Considering the diversity of life, it is not surprising that the problems that arise in analyzing biological data—and the solutions to those problems—are equally diverse. So, while my hope is that this thesis has given you an introduction to the topic at hand, there is much left to learn and much more to be explored. And as the technologies to generate biological data progress, new methods will need to be developed to analyze that data. We have only really started to scratch the surface of the possibilities that lie ahead and await us. And, hey, how boring would it not be otherwise?

7 Present Investigation

Paper I: Integrating Spatial Gene Expression and Breast Tumour Morphology via Deep Learning

In Paper I paper, we explore the link between tissue anatomy, as characterized by histology images, and spatially resolved gene expression. We construct a deep learning model that predicts spatial gene expression from histology images and train the model on a dataset of 68 Spatial Transcriptomics sections with paired hematoxylin and eosin stains from 23 breast cancer patients. Using leave-one-patient-out cross validation, we show that the model can predict the expression of over 100 genes at a resolution of 100 micrometer. The identified genes include known biomarkers for breast cancer, immune activation, and other clinically relevant processes. The results generalize to other breast cancer datasets, including The Cancer Genome Atlas. We further demonstrate that there is a strong connection between the expression of certain tumor-related genes and atypical morphological features, such as enlarged nuclei. Based on these findings, we conclude that deep learning models trained on spatial gene expression data could constitute a powerful framework for cancer screening and diagnosis in the clinics. Crucially, such models do not require annotated reference images, which are costly and difficult to curate, but are fully data-driven and therefore offer significant scaling advantages.

Paper II: Super-Resolved Spatial Transcriptomics by Deep Data Fusion

Experimental methods for spatially resolved transcriptomics can be characterized on a spectrum that trades sensitivity and spatial resolution for multiplexing capacity. On one end of the spectrum, imaging-based methods, such as those based on fluorescence in situ hybridization, typically have higher sensitivity and resolution, but are limited in the number of genes that can be measured simultaneously. On the other end, sequencing-based methods, such as 10x Visium, can measure genes from the entire transcriptome in a single experiment, but are limited in sensitivity and resolution. In paper II, we explore the idea of jointly modeling sequencing-based spatially resolved transcriptomics data with high-resolution histology images from the same tissue section. We propose that morphology and gene expression can be seen as observable effects of an underlying tissue state, and show that histology images can be used to increase the resolution of sequencing-based technologies, unlocking the potential of finer-grained analysis of spatial transcriptomes. Additionally, we show that the proposed method can use reference experiments to predict spatial transcriptomes from histology images without paired gene expression data. We propose that this type of analysis, which we term *in silico spatial transcriptomics*, can be used to re-

duce experimental burden in multi-section experiments, and paves the way for larger-scale studies of spatial transcriptomes.

Paper III: Learning Stationary Markov Processes with Contrastive Adjustment

Paper III introduces *contrastive adjustment*, an optimization algorithm for learning Markov transition kernels whose stationary distribution matches the data distribution. The learned transition kernel is a stationary generative model; by starting from an arbitrary state, sampling the kernel to iteratively generate new states yields a sequence of incrementally modified states that collectively follow the data distribution. Contrastive adjustment can be used to learn transition models in both continuous and discrete state spaces and is not restricted to a particular family of transition distributions. Additionally, inspired by recent work on diffusion-based generative models, we propose the *noise kernel*, a transition model of noisy data that generates new states by alternating between removing and adding back noise to the data. We demonstrate noise kernels trained by contrastive adjustment on a variety of computer vision tasks, including image synthesis, inpainting, and variant generation. We suggest that contrastive adjustment could be a powerful tool for human-computer design processes, as the stationarity of the learned Markov chain enables local exploration of the data manifold and makes it possible to iteratively refine outputs by means of human feedback. Such design processes may be highly valuable not only in the creative arts but also, for example, for accelerating discovery of new biomolecules in drug development.

Paper IV: Multi-Modal Modeling of Spatial Biology Data

Spatial biology encompasses a wide range of experimental technologies that measure different aspects of tissue anatomy, such as its morphology, gene expression, and protein composition. Comprehensively characterizing tissues therefore requires combining several modalities in the same experiment, but doing so is often prohibitively challenging or costly. Furthermore, measurements from spatial biology experiments are often sparse, noisy, or incomplete, impeding downstream analysis. Paper IV leverages the flexibility of contrastive adjustment and noise kernel transition models, as introduced in Paper III, to construct a multi-modal generative model of spatial biology data. The model integrates data from diverse modalities by forming a joint embedding space that enables information sharing across both different modalities and different experiments. By combining information from multiple modalities and experiments in a single analysis, it becomes possible to strengthen the signal when data is noisy, sparse, or degraded. Moreover, the proposed model can be used to infer missing data or to recast experiments

of one modality into another for improved interpretability or compatibility with other data. We show that the proposed method outperforms our previous work, presented in Paper II, on histology-guided gene expression imputation and super resolution. Additionally, not being limited to joint modeling of histology images and sequencing-based spatially resolved transcriptomics data, we demonstrate that the proposed method can be used to impute missing features in high-resolution in situ experiments, offering to computationally scale imaging-based technologies to larger gene panels. We posit that integrative models of spatial biology will become increasingly important in the coming years as datasets grow in size and complexity.

8 Future Outlook

Coming from a background in economics, which has a lot of intricacies in its own right, I am the first to admit that biological systems are tremendously complex. Making sense of this complexity is, to say the least, a daunting task (shout out to my fellow colleagues whose deep knowledge of biotechnology and its endless applications never ceases to amaze me!). Fortunately, large-scale machine learning systems, for example embodied by recent advances in large language models, have proven to be immensely powerful abstraction tools. Such *foundation models* learn complex patterns from vast bodies of data and can draw on that information to solve new problems without task-specific training.

In the coming years, large-scale machine learning models are likely to not only play an increasingly important role in biological research, improving our understanding of how biological systems function and accelerating the discovery of new treatments, but also unlock an array of new possibilities for personalized healthcare and precision medicine that require integrating information from many different data sources. For example, one such idea is the concept of a *digital twin*: a digital representation of a patient that is continually updated with biometric data from, for example, fitness watches, blood tests, and other health monitoring tools. Today, diagnosis is predominantly carried out once a patient has developed clinical signs of a disease, which is sometimes too late for effective treatment. Moreover, anamnesis is often based on limited data and self-reported symptoms, which can be an unreliable source of information that makes accurate diagnosis challenging. In contrast, by integrating data both longitudinally and across many different biometrical measurements, a digital twin could provide a continuous, data-driven assessment of a patient's health status, enabling early disease detection and personalized treatment plans, which ultimately promises better health outcomes.

The development of large-scale models for biology should not be taken for granted, however. The diverse types of data generated by experimental methods pose challenges for how we encode and represent that data in our models. And even if we find good representations, large-scale models still need large-scale datasets to train on. Here, reference atlases of human biology, such as the Human Cell Atlas and the Human Protein Atlas, provide invaluable training resources. However, current atlases are still limited in their size and scope, only encompassing an incomplete set of modalities, tissues, and disease states. Moreover, finding ways to train models on sensitive information, such as genetic data, without compromising patient privacy remains a major challenge in the development of larger-scale models.

While computational methods for analyzing healthcare data are likely to play an increasingly important role in the clinics and every day life, we are also likely to see progress in the methods that generate such data. For instance, over recent years, we have seen spatially resolved transcriptomics technologies evolve from measuring only a handful of genes in a single experiment to transcriptome-scale measurements at an ever finer spatial resolution. What's next? I think we will see a continued trend toward more modalities, higher multiplexing, and higher resolution, but also a shift toward characterizing tissues in three dimensions. The latter will likely rely on generative models for synthesizing parts of the tissue due to experimental constraints, at least in the beginning.

Further ahead, one of the most exciting research prospects will be to extend spatial biology to the fourth dimension: time. Understanding how a system works is, in a way, to understand its causal structure. At the same time, understanding causality without time is like trying to understand a movie by looking at a single frame. It is not until we see the movie in its entirety that we know how the story unfolds. The same holds true for biological systems: to understand the effectors of disease progression and of how tissues develop and self-organize, we need to be able observe how the system changes over time and responds to perturbations. To this end, we will need to develop new experimental methods for dynamically profiling biological systems and new computational methods for analyzing such data. While this is likely to be a difficult endeavor, I am convinced that it will be worth the effort.

Looking back at my time as a graduate student, I have had the fortune to study biology and generative artificial intelligence during a time when both fields have seen tremendous progress. Undoubtedly, few people at the start of my studies would have predicted the rapid development that we have seen over the past few years in both fields. In such a fast-moving environment, it is ultimately very difficult to know what tomorrow will bring. Therefore, I would like to conclude this thesis with a quote from the late physicist Richard Feynman, who once said:

> *"I can live with doubt and uncertainty and not knowing. I think it's much more interesting to live not knowing than to have answers which might be wrong."*

Whatever the future holds, I am curious to find out and excited to be but a small part of it.

9 References

[1] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[2] Connor Shorten and Taghi M. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.1 (2019), p. 60.

[3] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015.

[5] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 5998–6008.

[6] Rewon Child et al. "Generating Long Sequences With Sparse Transformers". In: *ArXiv preprint* abs/1904.10509 (2019).

[7] Angelos Katharopoulos et al. "Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 5156–5165.

[8] Albert Gu, Karan Goel, and Christopher Ré. "Efficiently Modeling Long Sequences with Structured State Spaces". In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[9] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. Ed. by Francis

R. Bach and David M. Blei. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 448–456.

[10] Sepp Hochreiter et al. “Gradient flow in recurrent nets: the difficulty of learning long-term dependencies”. In: *A field guide to dynamical recurrent neural networks*. Ed. by Stefan C Kremer and John F Kolen. IEEE Press, 2001, pp. 237–243.

[11] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, pp. 249–256.

[12] Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. “On the difficulty of training recurrent neural networks”. In: *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*. Vol. 28. JMLR Workshop and Conference Proceedings. JMLR.org, 2013, pp. 1310–1318.

[13] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. “Layer Normalization”. In: *ArXiv preprint* abs/1607.06450 (2016).

[14] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 770–778.

[15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *ArXiv preprint* abs/1505.04597 (2015).

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. “Denoising Diffusion Probabilistic Models”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. 2020.

[17] Robin Rombach et al. “High-Resolution Image Synthesis with Latent Diffusion Models”. In: *ArXiv preprint* abs/2112.10752 (2021).

[18] Chitwan Saharia et al. “Photorealistic Text-To-Image Diffusion Models with Deep Language Understanding”. In: *ArXiv preprint* abs/2205.11487 (2022).

[19] Dustin Podell et al. “SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis”. In: *ArXiv preprint* abs/2307.01952 (2023).

[20] Solomon Kullback and Richard A Leibler. “On information and sufficiency”. In: *The annals of mathematical statistics* 22.1 (1951), pp. 79–86.

[21] Tom B. Brown et al. “Language Models are Few-Shot Learners”. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. 2020.

[22] Hugo Touvron et al. “LLaMA: Open and Efficient Foundation Language Models”. In: *ArXiv preprint* abs/2302.13971 (2023).

[23] E. G. Tabak and Cristina V. Turner. “A Family of Nonparametric Density Estimation Algorithms”. In: *Communications on Pure and Applied Mathematics* 66.2 (2012), pp. 145–164.

[24] George Papamakarios et al. “Normalizing Flows for Probabilistic Modeling and Inference”. In: *J. Mach. Learn. Res.* 22 (2021), 57:1–57:64.

[25] Danilo Jimenez Rezende and Shakir Mohamed. “Variational Inference with Normalizing Flows”. In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. Ed. by Francis R. Bach and David M. Blei. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 1530–1538.

[26] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. “Density estimation using Real NVP”. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[27] Diederik P. Kingma and Prafulla Dhariwal. “Glow: Generative Flow with Invertible 1x1 Convolutions”. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio et al. 2018, pp. 10236–10245.

[28] Tian Qi Chen et al. “Neural Ordinary Differential Equations”. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio et al. 2018, pp. 6572–6583.

[29] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *2nd International Conference on Learning Representa-*

tions, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings. Ed. by Yoshua Bengio and Yann LeCun. 2014.

[30] Johan Ludwig William Valdemar Jensen. "Sur les fonctions convexes et les inégalités entre les valeurs moyennes". In: *Acta mathematica* 30.1 (1906), pp. 175–193.

[31] Diederik P. Kingma et al. "Improving Variational Inference with Inverse Autoregressive Flow". In: *ArXiv preprint* abs/1606.04934 (2016).

[32] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*. Vol. 32. JMLR Workshop and Conference Proceedings. JMLR.org, 2014, pp. 1278–1286.

[33] Casper Kaae Sønderby et al. "Ladder Variational Autoencoders". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*. Ed. by Daniel D. Lee et al. 2016, pp. 3738–3746.

[34] Lars Maaløe et al. "BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 6548–6558.

[35] Arash Vahdat and Jan Kautz. "NVAE: A Deep Hierarchical Variational Autoencoder". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle et al. 2020.

[36] Rewon Child. "Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images". In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[37] Jascha Sohl-Dickstein et al. "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille,*

France, 6-11 July 2015. Ed. by Francis R. Bach and David M. Blei. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 2256–2265.

[38] Pascal Vincent. "A connection between score matching and denoising autoencoders". In: *Neural computation* 23.7 (2011), pp. 1661–1674.

[39] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[40] Emiel Hoogeboom et al. "Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions". In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by Marc'Aurelio Ranzato et al. 2021, pp. 12454–12465.

[41] Jacob Austin et al. "Structured Denoising Diffusion Models in Discrete State-Spaces". In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*. Ed. by Marc'Aurelio Ranzato et al. 2021, pp. 17981–17993.

[42] Tim Salimans and Jonathan Ho. "Progressive Distillation for Fast Sampling of Diffusion Models". In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[43] Yang Song et al. "Consistency Models". In: *ArXiv preprint* abs/2303.01469 (2023).

[44] Hyunmin Lee, Zhaolei Zhang, and Henry M. Krause. "Long Noncoding RNAs and Repetitive Elements: Junk Or Intimate Evolutionary Partners?" In: *Trends in Genetics* 35.12 (2019), pp. 892–902.

[45] Conrad Hal Waddington. *The strategy of the genes: a discussion of some aspects of theoretical biology*. Allen & Unwin, 1957.

[46] Geoffrey C. Gurtner et al. "Wound repair and regeneration". In: *Nature* 453.7193 (2008), pp. 314–321.

[47] Sendurai A. Mani et al. "The Epithelial-Mesenchymal Transition Generates Cells with Properties of Stem Cells". In: *Cell* 133.4 (2008), pp. 704–715.

[48] Kazutoshi Takahashi and Shinya Yamanaka. “Induction of Pluripotent Stem Cells from Mouse Embryonic and Adult Fibroblast Cultures By Defined Factors”. In: *Cell* 126.4 (2006), pp. 663–676.

[49] Douglas Hanahan and Robert A Weinberg. “Hallmarks of Cancer: The Next Generation”. In: *Cell* 144.5 (2011), pp. 646–674.

[50] Robert Hooke. *Micrographia: or some physiological descriptions of minute bodies made by magnifying glasses. With observations and inquiries thereupon.* Jo. Martyn and Ja. Allestry, 1665.

[51] Antony van Leewenhoeck. “Observations, Communicated to the Publisher by Mr. Antony van Leewenhoeck, in a Dutch Letter of the 9th of Octob. 1676. Here English’d: concerning Little Animals by Him Observed in Rain-Well-Sea. and Snow Water; as Also in Water Wherein Pepper Had Lain Infused”. In: *Philosophical Transactions (1665-1678)* 12 (1677), pp. 821–831.

[52] Joseph G. Gall and Mary Lou Pardue. “Formation and detection of RNA-DNA hybrid molecules in cytological preparations”. In: *Proceedings of the National Academy of Sciences* 63.2 (1969), pp. 378–383.

[53] Andrea M. Femino et al. “Visualization of Single RNA Transcripts in Situ”. In: *Science* 280.5363 (1998), pp. 585–590.

[54] Jeffrey M. Levsky et al. “Single-Cell Gene Expression Profiling”. In: *Science* 297.5582 (2002), pp. 836–840.

[55] Kok Hao Chen et al. “Spatially resolved, highly multiplexed RNA profiling in single cells”. In: *Science* 348.6233 (2015), aaa6090.

[56] Chee-Huat Linus Eng et al. “Transcriptome-scale super-resolved imaging in tissues by RNA seqFISH+”. In: *Nature* 568.7751 (2019), pp. 235–239.

[57] Jay Shendure and Hanlee Ji. “Next-generation DNA sequencing”. In: *Nature Biotechnology* 26.10 (2008), pp. 1135–1145.

[58] Patrik L. Ståhl et al. “Visualization and analysis of gene expression in tissue sections by spatial transcriptomics”. In: *Science* 353.6294 (2016), pp. 78–82.

[59] Amanda Janesick et al. “High resolution mapping of the tumor microenvironment using integrated single-cell, spatial and in situ analysis”. In: *Nature Communications* 14.1 (2023), p. 8353.

[60] Samuel G. Rodriques et al. “Slide-seq: A scalable technology for measuring genome-wide expression at high spatial resolution”. In: *Science* 363.6434 (2019), pp. 1463–1467.

[61] Sanja Vickovic et al. “High-definition spatial transcriptomics for in situ tissue profiling”. In: *Nature Methods* 16.10 (2019), pp. 987–990.

[62] Ao Chen et al. “Spatiotemporal transcriptomic atlas of mouse organogenesis using DNA nanoball-patterned arrays”. In: *Cell* 185.10 (2022), 1777–1792.e21.

[63] Lars E Borm et al. “Scalable in situ single-cell profiling by electrophoretic capture of mRNA using EEL FISH”. In: *Nature Biotechnology* 41.2 (2023), pp. 222–231.

[64] Bert Vogelstein and Kenneth W Kinzler. “Cancer genes and the pathways they control”. In: *Nature Medicine* 10.8 (2004), pp. 789–799.

[65] Siméon Denis Poisson. *Recherches sur la probabilité des jugements en matière criminelle et en matière civile, précédées des règles générales du calcul des probabilités*. Bachelier, 1837.

[66] Simon Anders and Wolfgang Huber. “Differential expression analysis for sequence count data”. In: *Genome Biology* 11.10 (2010), R106.

[67] Norman L Johnson, Adrienne W Kemp, and Samuel Kotz. *Univariate discrete distributions*. Vol. 444. John Wiley & Sons, 2005.

[68] Davide Risso et al. “Normalization of RNA-seq data using factor analysis of control genes or samples”. In: *Nature Biotechnology* 32.9 (2014), pp. 896–902.

[69] Arjun Raj and Alexander van Oudenaarden. “Nature, Nurture, or Chance: Stochastic Gene Expression and Its Consequences”. In: *Cell* 135.2 (2008), pp. 216–226.

[70] Romain Lopez et al. “Deep generative modeling for single-cell transcriptomics”. In: *Nature Methods* 15.12 (2018), pp. 1053–1058.

[71] Christopher Heje Grønbech et al. “scVAE: variational auto-encoders for single-cell gene expression data”. In: *Bioinformatics* 36.16 (2020), pp. 4415–4422.

[72] Mark D. Robinson, Davis J. McCarthy, and Gordon K. Smyth. “edgeR: a Bioconductor package for differential expression analysis of digital gene expression data”. In: *Bioinformatics* 26.1 (2009), pp. 139–140.

[73] Edward Zhao et al. “Spatial transcriptomics at subspot resolution with BayesSpace”. In: *Nature Biotechnology* 39.11 (2021), pp. 1375–1384.

[74] Ludvig Bergenstråhle et al. “Super-resolved spatial transcriptomics by deep data fusion”. In: *Nature Biotechnology* 40.4 (2021), pp. 476–479.

[75] Emelie Berglund et al. “Spatial maps of prostate cancer transcriptomes reveal an unexplored landscape of heterogeneity”. In: *Nature Communications* 9.1 (2018), p. 2419.

[76] Jonas Maaskola et al. “Charting Tissue Expression Anatomy by Spatial Transcriptome Decomposition”. In: *bioRxiv* (2018).

[77] Alma Andersson et al. “Single-cell and spatial transcriptomics enables probabilistic inference of cell type topography”. In: *Communications Biology* 3.1 (2020), p. 565.

[78] Dylan M. Cable et al. “Robust decomposition of cell type mixtures in spatial transcriptomics”. In: *Nature Biotechnology* 40.4 (2021), pp. 517–526.

[79] Vitalii Kleshchevnikov et al. “Cell2location maps fine-grained cell types in spatial transcriptomics”. In: *Nature Biotechnology* 40.5 (2022), pp. 661–671.

[80] Tommaso Biancalani et al. “Deep learning and alignment of spatially resolved single-cell transcriptomes with Tangram”. In: *Nature Methods* 18.11 (2021), pp. 1352–1362.

[81] Marc Elosua-Bayes et al. “SPOTlight: seeded NMF regression to deconvolute spatial transcriptomics spots with single-cell transcriptomes”. In: *Nucleic Acids Research* 49.9 (2021), e50–e50.

[82] Ludvig Larsson et al. “Semla: a versatile toolkit for spatially resolved transcriptomics analysis and visualization”. In: *Bioinformatics* 39.10 (2023), nil.

[83] Bryan He et al. “Integrating spatial gene expression and breast tumour morphology via deep learning”. In: *Nature Biomedical Engineering* 4.8 (2020), pp. 827–834.

[84] Ludvig Bergenstråhle and Joakim Lundeberg. “Multi-Modal Modeling of Spatial Biology Data”. In Preparation.

[85] Nir Ben-Chetrit et al. “Integration of whole transcriptome spatial profiling with protein markers”. In: *Nature Biotechnology* 41.6 (2023), pp. 788–793.

[86] Marco Vicari et al. “Spatial multimodal analysis of transcriptomes and metabolomes in tissues”. In: *Nature Biotechnology* (2023), pp. 1–5.

[87] Jennifer E. Rood et al. “Toward a Common Coordinate Framework for the Human Body”. In: *Cell* 179.7 (2019), pp. 1455–1467.

[88] Alma Andersson et al. “A Landmark-based Common Coordinate Framework for Spatial Transcriptomics Data”. In: *bioRxiv* (2021).

[89] Ron Zeira et al. “Alignment and integration of spatial transcriptomics data”. In: *Nature Methods* 19.5 (2022), pp. 567–575.

[90] Markus Ekvall et al. “Spatial landmark detection and tissue registration with deep learning”. In: *bioRxiv* (2023).

[91] Kalen Clifton et al. “STalign: Alignment of spatial transcriptomics data using diffeomorphic metric mapping”. In: *Nature Communications* 14.1 (2023), p. 8123.

[92] Kangning Dong and Shihua Zhang. “Deciphering spatial domains from spatially resolved transcriptomics with an adaptive graph attention auto-encoder”. In: *Nature Communications* 13.1 (2022), p. 1739.

[93] Carsen Stringer et al. “Cellpose: a generalist algorithm for cellular segmentation”. In: *Nature Methods* 18.1 (2020), pp. 100–106.

[94] Jeongbin Park et al. “Cell segmentation-free inference of cell types from in situ transcriptomics data”. In: *Nature Communications* 12.1 (2021), p. 3545.

[95] Viktor Petukhov et al. “Cell segmentation in imaging-based spatial transcriptomics”. In: *Nature Biotechnology* 40.3 (2021), pp. 345–354.

[96] Valentine Svensson, Sarah A Teichmann, and Oliver Stegle. “SpatialDE: identification of spatially variable genes”. In: *Nature Methods* 15.5 (2018), pp. 343–346.

[97] Shiquan Sun, Jiaqiang Zhu, and Xiang Zhou. “Statistical analysis of spatial expression patterns for spatially resolved transcriptomic studies”. In: *Nature Methods* 17.2 (2020), pp. 193–200.

[98] Alma Andersson and Joakim Lundeberg. “sepal: identifying transcript profiles with spatial patterns by diffusion-based modeling”. In: *Bioinformatics* 37.17 (2021), pp. 2644–2650.

[99] Sophia K. Longo et al. “Integrating single-cell and spatial transcriptomics to elucidate intercellular tissue dynamics”. In: *Nature Reviews Genetics* 22.10 (2021), pp. 627–644.

[100] David S. Fischer, Anna C. Schaar, and Fabian J. Theis. “Modeling intercellular communication in tissues using spatial graphs of cells”. In: *Nature Biotechnology* 41.3 (2022), pp. 332–336.

[101] Zixuan Cang et al. “Screening cell-cell communication in spatial transcriptomics via collective optimal transport”. In: *Nature Methods* 20.2 (2023), pp. 218–228.