In this work, we aimed to accelerate scientific progress in the field of single-cell omics by providing and evaluating computational tools. In section **??** we proposed a concrete set of objectives this work will attempt to solve. Every chapter respectively tackles one of these research objectives, and each chapter discusses the implications of that part of our research in detail. This chapter reflects on the impact of this work on the field, going over each of the objectives and respective chapters.

## 1 Benchmarking with *in silico* single cells

We developed `dyngen`, a simulator of single cells to benchmark and stress-test computational tools for single-cell omics (Chapter **??**). The `dyngen` software supports a selection of existing experimental protocols, and can also simulate experiments that are possible with the current technologies, such as generating omics profiles from the same cell at multiple timepoints. The generated data can help kick-start emerging domains with low data availability more safely by allowing software developers to test their method before numerous datasets become publicly available.

`dyngen` has already been successfully used to evaluate trajectory inference[1], trajectory alignment[2], and network inference[3] methods. Furthermore, we showed that it can also be used to evaluate differential network inference methods, and trajectory alignment methods.

## 2 Comparing 45 trajectory inference methods

Using this simulator and a collection of real datasets, we performed a comparison of 45 TI methods (Chapter **??**). Our contributions include writing software to run 45 different error-prone TI methods with a common interface, downloading and processing hundreds of single cell datasets, and developing novel metrics for comparing ground truth and predicted trajectories. With this benchmarking study, we had two clear goals in mind: to help guide users to TI methods that are suitable for their needs, and to provide developers of TI methods with the necessary tools to benchmark their own tool.

We accomplished the first goal by constructing a set of guidelines for end-users to help choose a TI method that is accurate, robust and fit for their application. Since such guidelines were hitherto lacking, they are now commonly disseminated in manuscripts [4, 5], courses [6, 7], and slides shown during keynote caffeine refuelling sessions [8].

For our second goal, it seems too early to tell whether our research had any effect on self-assessments of TI method developers. We did made our pipeline, datasets, metrics, and containerised wrappers of TI methods publicly available for developers to use. However, so far we do not observe an increase in developers performing quantitative benchmarks. We hypothesise causal reasons for this phenomenon and provide solutions in order to spur TI developers to perform more self-assessments (Chapter **??**).

## 3 A toolkit to infer, visualise and interpret single-cell trajectories

The previous two works necessitated developing a tool for visualising trajectory data. We extended this implementation into a full toolkit, named `dyno`, for inferring and analysing trajectories (Chapter **??**). The toolkit allows performing downstream analyses such as detecting differentially expressed genes along transitions in the trajectory, annotating the trajectory, and comparing multiple trajectories in a common dimensionality reduction. A major benefit of `dyno` is that it allows utilising any of the 50 TI methods we wrapped as part of the comparison, without having to install dependencies for every TI method separately.

The software packages consist of several subpackages, each with a separate functionality (e.g. `dynplot` visualises trajectories, `dynmethods` provides wrappers for the TI methods). Since most of the packages are currently only hosted on Github, only users familiar with Github have been able to use it. To make `dyno` available to a larger audience, we are publishing each of these packages and developing command-line interfaces for the most important functionalities.

## 4 Fast, accurate, and robust single-cell pseudotime

We developed a TI method, called `SCORPIUS`, specialised in inferring linear trajectories (Chapter **??**). Since our comparison of TI methods already lists 30 TI methods that can infer trajectories that are more complex than a linear ordering of cells, why would we need a linear TI method?

Being able to study how cells progress over time is in terms of analysis as fundamental as being able to cluster cells or to identify differentially expressed genes between two groups. We showed that for analysing datasets containing linear trajectories, `SCORPIUS` outperforms all other TI methods tested as part of our TI benchmark.

Despite `SCORPIUS` published on bioRxiv in 2016, discussion and usage of the pre-print has largely been limited to comparing the results of a different (linear) TI methods to those of `SCORPIUS`. Since the pre-print, many improvements have been made to improve the accuracy and robustness of predicted trajectories, to allow scaling up to datasets containing 100'000s of cells, and to providing a more user-friendly interface. Since the beginning of 2019 – before the publication of our benchmark of TI methods – several studies have now reported using `SCORPIUS` as a tool for deriving primary results[9, 10, 11, 12, 13].

## 5 Inferring single cell regulatory networks

TODO: fill in when chapter has been written.

## 6 Optimising regulatory networks

This project was a spin-off of Netter[14], a tool for reranking GRN predictions using structural network properties. Predicted GRNs often are topologically dissimilar to real networks, and as such, Netter quantifies the topological properties of a network and makes incremental changes in order to arrive at a network with a 'better' topological profile.

The topological 'profile' of a network was defined as the frequencies of particular topological patterns called 'graphlets'. Keeping track of the frequencies of graphlets during an iterative process proved difficult at first. `incgraph` provides a solution to this problem (Chapter **??**) by making assuming that the incremental changes between two different networks are relatively small (less than 100 edge additions/removals)). Under this assumption, it is possible to calculate the differences in topological patterns significantly faster in comparison to recomputing the graphlet frequencies from scratch at every iteration.

While `incgraph` solved a very specific and interesting problem, and was eventually published as a peer-reviewed article, it has not succeeded in making a direct impact on scientific research as of yet. Perhaps, at some point, a researcher will come across `incgraph` and find that it exactly solves his or her problem.

## 7 A life without Git, Travis CI, or tidyverse

A significant portion of this work involved developing large software libraries, in a collaborative setting, over a time span of about four years. Since the results of our comparison of 45 TI methods required three years to develop, we were happily forced to develop a system where experiments could be rerun and results could be updated on-the-fly.

We summarised our experiences in the form of a set of guidelines for benchmarking computational tools (Chapter **??**). However, these guidelines do not touch upon many aspects of good software development practices that we learned to use in order to bring this thesis to a good end. In particular, I would like to highlight several fundamental (open-source) projects without the likes of which our research would have been simply impossible to perform, namely Git, Travis CI, and the tidyverse.

**Git**[15] is a code-revision system that allows multiple users to collaborate on developing code and keep track of what changes were made by whom. Since Wouter and I were often working closely together on a specific part of our software, Git saved us a lot of time by merging together changes developed in parallel. Only in few cases did we need to intervene manually to merge our changes. Additionally, Github.com allowed us not only to collaborate with each-other, but also correspond and collaborate with other software developers from many different research groups, using Github Issues to discuss problems with other researchers, and Github Pull Requests to contribute code to open-source projects. Together, we published over 20'000 code contributions (commits) across 50+ software packages[16]. Since many of these packages depend on one another, it is inevitable that changes made to one package would break another package. We wrote many, many unit tests (and still too

few), and we let these tests automatically be executed on **Travis CI**[17]. This way, when we pushed breaking changes to Github, Travis CI would notify that our commit has resulted in one or more unit tests failing. While sometimes it is a hassle to set up, Travis CI has prevented us countless times from generating faulty results due to a faulty underlying function.

Software bugs can be introduced even by incredibly small, seemingly insignificant changes. The R programming language seems particularly susceptible towards getting trapped by its many pitfalls. However, there are many benefits of using R in bioinformatics research context, such as its extensive community of statisticians develop packages for the R ecosystem. The **tidyverse** packages[18], developed by the RStudio team and many online contributors, completely transformed my experiences with R from tedious struggles to efficient everyday functional programming.

Honourable mentions go to Linux, Fedora, LaTeX, TeXstudio, (R)Markdown, Bash, Sed, Regular expressions, Rocket Chat, for making bioinformatics software development even more fun and enjoyable.

# 8 References


[1] Wouter Saelens et al. "A Comparison of Single-Cell Trajectory Inference Methods". In: *Nature Biotechnology* 37 (May 2019). ISSN: 15461696. DOI: 10.1038/s41587-019-0071-9.

[2] Koen Van den Berge et al. "Trajectory-Based Differential Expression Analysis for Single-Cell Sequencing Data". In: *bioRxiv* (Jan. 1, 2019), p. 623397. DOI: 10.1101/623397.

[3] Aditya Pratapa et al. "Benchmarking Algorithms for Gene Regulatory Network Inference from Single-Cell Transcriptomic Data". In: *bioRxiv* (June 4, 2019), p. 642926. DOI: 10.1101/642926.

[4] Atefeh Lafzi et al. "Tutorial: Guidelines for the Experimental Design of Single-Cell RNA Sequencing Studies". In: *Nature Protocols* 13.12 (Dec. 1, 2018), pp. 2742–2757. ISSN: 1750-2799. DOI: 10.1038/s41596-018-0073-y.

[5] Malte D Luecken and Fabian J Theis. "Current Best Practices in Single-Cell RNA-Seq Analysis: A Tutorial". In: *Molecular Systems Biology* 15.6 (June 1, 2019), e8746. ISSN: 1744-4292. DOI: 10.15252/msb.20188746.

[6] Vladimir Kiselev et al. "Analysis of Single Cell RNA-Seq Data" (Cambridge, UK). May 2, 2019. URL: https://scrnaseq-course.cog.sanger.ac.uk/website/index.html (visited on 08/22/2019).

[7] Liesbet Martens and Niels Vandamme. "Analysis of Single Cell RNA-Seq Data from 10x Genomics" (Ghent). Aug. 29, 2019. URL: https://training.vib.be/analysis-single-cell-rna-seq-data-10x-genomics (visited on 08/22/2019).

[8] *Coffee Break during "Analysis of Single Cell RNA-Seq Data 23-24 May 2019" Workshop*. In collab. with Vladimir Kiselev. May 23, 2019. URL: https://www.youtube.com/watch?v=7dQ_pleDO2Y&t=1h53m14s.

[9] Nicolas Damond et al. "A Map of Human Type 1 Diabetes Progression by Imaging Mass Cytometry". In: *Cell Metabolism* 29.3 (Mar. 5, 2019), 755–768.e5. ISSN: 1550-4131. DOI: 10.1016/j.cmet.2018.11.014.

[10] Yang Cheng et al. "Multifactorial Heterogeneity of Virus-Specific T Cells and Association with the Progression of Human Chronic Hepatitis B Infection". In: *Science Immunology* 4.32 (Feb. 8, 2019), eaau6905. ISSN: 2470-9468. DOI: 10.1126/sciimmunol.aau6905. pmid: 30737354.

[11] Christopher Andrew Tibbitt et al. "Single-Cell RNA Sequencing of the T Helper Cell Response to House Dust Mites Defines a Distinct Gene Expression Signature in Airway Th2 Cells". In: *Immunity* 51.1 (July 16, 2019), 169–184.e5. ISSN: 1074-7613. DOI: 10.1016/j.immuni.2019.05.014.

[12] Hannah Van Hove et al. "A Single-Cell Atlas of Mouse Brain Macrophages Reveals Unique Transcriptional Identities Shaped by Ontogeny and Tissue Environment". In: *Nature Neuroscience* 22.6 (June 2019), pp. 1021–1035. ISSN: 1546-1726. DOI: 10.1038/s41593-019-0393-4.

[13] Jasper Wouters et al. "Single-Cell Gene Regulatory Network Analysis Reveals New Melanoma Cell States and Transition Trajectories during Phenotype Switching". In: *bioRxiv* (Jan. 1, 2019), p. 715995. DOI: 10.1101/715995.

[14] Joeri Ruyssinck et al. "Netter: Re-Ranking Gene Network Inference Predictions Using Structural Network Properties." In: *BMC Bioinformatics* 17.1 (2016), p. 76. ISSN: 1471-2105. DOI: 10.1186/s12859-016-0913-0. pmid: 26862054.

[15] Linus Torvalds and Junio Hamano. *Git: Fast Version Control System*. 2005. URL: http://git-scm.com.

[16] *The Development of Dynverse*. Apr. 1, 2019. URL: https://www.youtube.com/watch?v=C42F5Y8kCU0 (visited on 10/14/2019).

[17] GmbH Travis CI. *Travis CI - Test and Deploy Your Code with Confidence*. 2011. URL: https://travis-ci.org (visited on 10/14/2019).

[18] Hadley Wickham. *The Tidy Tools Manifesto*. Nov. 13, 2017. URL: https://cran.r-project.org/web/packages/tidyverse/vignettes/manifesto.html (visited on 10/14/2019).