University of Bielefeld
Faculty of Technology

Dissertation
for the degree of *Dr. rer. nat.*

Single-cell transcriptome analyses on developmental transitions in mouse pluripotent stem cells

submitted by

Michael Böttcher
born on the 11th March 1987 in Güstrow

Bielefeld, August 2020

Printed on non-ageing paper ISO 9706

First Reviewer

Dr. Piero Carninci
RIKEN Center for Integrative Medical Sciences
Laboratory for Transcriptome Technology

Second Reviewer

Dr. Thomas Noll
University of Bielefeld
Cell Culture Technology Group

Third Reviewer

Dr. Andrea Bräutigam
University of Bielefeld
Computational Biology

Date of disputation: August 14, 2020

Oral presentations

- Cold Spring Harbor Asia 'Single Cells' conference, Dushu Lake Conference Center, Suzhou, China, December 2014

Poster presentations

- Annual 'Single Cell Genomics' conference at Weizmann Institute, Rehovot, Israel, October 2017
- RIKEN Summer School 2017 at Kazusa Academia Hall, Kisarazu, Japan, September 2017
- Single Cell Science Symposium at RIKEN Yokohama Koryuto Hall, Yokohama, Japan, July 2017
- The 12th International Workshop on Advanced Genomics (from Genome to Life) at Hitotsubashi Hall, Tokyo, Japan, June 2017
- International Conference on Single Cell Research 2016 at Ito International Research Center, The University of Tokyo, Tokyo, Japan, November 2016
- RIKEN Summer School 2016 at Tsukuba International Congress Center, Tsukuba, Japan, September 2016
- Annual 'Single Cell Genomics' conference at Wellcome Genome Campus, Hinxton, UK, September 2016
- 29th International Mammalian Genome Conference (IMGC) at Yokohama Port Opening Memorial Hall, Yokohama, Japan, November 2015
- Annual 'Single Cell Genomics' conference at TivoliVredenburg, Utrecht, The Netherlands, September 2015
- RIKEN Noyori Summer School 2015 at Shinrin-Koen Heritage Resort, Kumagaya, Japan, September 2015

- The 11th International Workshop on Advanced Genomics (My Genome, Your Genome) at Hitotsubashi Hall, Tokyo, Japan, May 2015
- RIKEN Noyori School 2014 at the International Productivity Center, Shonan, Japan, December 2014
- The 37th Annual Meeting of The Molecular Biology Society of Japan (MBSJ) at Pacifico Yokohama, Yokohama, Japan, November 2014

Publications

C. Fischer, M. Metsger, S. Bauch, R. Vidal, **M. Böttcher**, P. Grote, M. Kliem, *et al.* (2019). "Signals trigger state-specific transcriptional programs to support diversity and homeostasis in immune cells". In: *Science Signaling* 12, eaao5820. DOI: 10.1126/scisignal.aao5820

T. Kouno, J. Moody, A. T. Kwon, Y. Shibayama, S. Kato, Y. Huang, **M. Böttcher**, *et al.* (2019). "C1 CAGE detects transcription start sites and enhancer activity at single-cell resolution". In: *Nature Communications* 10, p. 360. DOI: 10.1038/s41467-018-08126-5

I. Abugessaisa, S. Noguchi, **M. Böttcher**, A. Hasegawa, T. Kouno, S. Kato, Y. Tada, *et al.* (2018). "SCPortalen: human and mouse single-cell centric database". In: *Nucleic Acids Research* 46, pp. D781–D787. DOI: 10.1093/nar/gkx949

M. Böttcher, T. Kouno, E. Madissoon, E. Motakis, I. Abugessaisa, S. Kato, H. Suzuki, *et al.* (2016). "Single-cell transcriptomes of fluorescent, ubiquitination-based cell cycle indicator cells". In: *BioRxiv*. preprint. DOI: https://doi.org/10.1101/088500

Contents

Abstract

We used an in vitro model of Pluripotent Stem Cell (PSC) development in mice to analyze dynamic changes in transcriptomes of hundreds of individual cells which were undergoing an induced transition from naïve mouse Embryonic Stem Cells (mESC) towards primed pluripotent Epiblast Stem Cells (EpiSC).

The differentiation of mESCs to EpiSC-like cells takes about five days after induction. We collected cell samples in 24-hour intervals for four days after induction as well as untreated mESCs and primed state EpiSCs. Single-cell isolation and scRNA-seq library preparation for each time point were done on the commercial Fluidigm C1 platform. In addition, we sampled C1-Cap Analysis of Gene Expression (C1-CAGE) libraries for the same set of time points to enable detection of non-coding RNAs (ncRNAs) such as anti-sense RNAs or enhancer RNAs.

This C1-CAGE protocol was new and still undergoing optimization at the beginning of our experiments. C1-CAGE was first published by Kouno *et al.* (2019) and the author of this thesis contributed as a co-author. Throughout the work on this project a data management platform called SCPortalen was developed to share all data among project collaborators. SCPortalen's publication was also co-authored by the author of this thesis (Abugessaisa *et al.*, 2018).

The combination of transcriptome datasets from two different protocols allowed the elucidation of expression dynamics of the naïve-to-primed stem cell conversion. We independently identified two subpopulations of cells during the transition process with both the Fluidigm scRNA-seq and C1-CAGE dataset. Pseudotime analysis revealed the developmental trajectory of cells and is a powerful tool to reliably identify developmental stages of cells without prior knowledge of their actual stage. Among these two transition phase subpopulations, one showed wide-spread repression of gene expression. The small nuclear RNA (snRNA) *Rn7sk* was identified as one potential regulator of this population specific phenomenon. The second subpopulation shared some characteristics with primed EpiSCs such as cell morphology and the expression of known primed state marker genes, but

it could be shown that cells from this population were still undergoing Epithelial-Mesenchymal Transition (EMT). That is a clear sign that these cells have not yet fully transitioned to primed pluripotent stem cells. Interestingly, the characteristics of this subpopulation largely match a predicted third pluripotency state called "formative" (Smith, 2017). Therefore, we believe that our dataset not only contains naïve and primed pluripotent stem cells, but also formative pluripotent stem cells. Thus, our dataset represents a unique resource to compare and study this proposed formative pluripotency state. Last but not least, we found several marker gene candidates for all developmental stages of the naïve-to-primed transition, which will facilitate classification of cells in future experiments. For example, we propose *Cd59a* as a highly specific marker gene for primed EpiSCs.

The results of this thesis project have also been compiled into a manuscript for publication in a peer reviewed journal and will be submitted soon after the submission of this thesis.

Acknowledgements

I would like to thank Dr. Piero Carninci for supporting me to do this work and to experience life in Japan at the same time. It was a great opportunity to acquire many new skills and experiences in research and for life.

Additionally, I would like to thank Dr. Thomas Noll for formally supervising me as a doctoral candidate. Furthermore, my thanks go to Dr. Charles Plessy and all my colleagues at the RIKEN center in Yokohama. Their mentoring and guidance throughout the entire 4 years of my stay as an International Program Associate at RIKEN have been crucial, not just for work, but also to manage life in Japan.

A special thank you goes to Dr. Kuniya Abe and his team members who have been great collaborators. Our close interaction throughout the project has been a major driving force for this thesis.

I also thank Julia Weber for her critical reading of the dissertation manuscript and Jaroslaw M. Knoppek for his additional language proof reading.

Finally, I thank my family and friends, especially Nicolai Domscheit, for their constant support and patience. Without their help this thesis would not exist in its final form today.

List of Figures

List of Tables

Abbreviations

5hmC 5-hydroxymethylcytosine

AEA Allele-specific Expression Analysis

ATAC-seq Assay for Transposase-Accessible Chromatin

BAM Binary Alignment Map

BED Browser Extensible Data

BP Biological Process

C1-CAGE C1-Cap Analysis of Gene Expression

CC Cellular Component

Ccdc36 Coiled-coil domain containing 36

Cd59a Cluster of differentiation 59 antigen

Cdh1 Cadherin 1

Cdh2 Cadherin 2

cDNA complementary DNA

ChIP-seq Chromatin Immunoprecipitation sequencing

Cyp24a1 Cytochrome P450 Family 24 Subfamily A Member 1

Ddx3y DEAD (Asp-Glu-Ala-Asp) box polypeptide 3, Y-linked

DE Differentially Expressed

DGE Differential Gene Expression

DHS DNase I Hypersensitive Sites

EDTA Ethylenediaminetetraacetic acid

Eif2s3y Eukaryotic translation initiation factor 2, subunit 3, structural gene Y-linked

EMT Epithelial-Mesenchymal Transition

EPD Eukaryotic Promoter Database

EpiSC Epiblast Stem Cell

ERCC External RNA Controls Consortium

eRNA enhancer RNA

Esrrb Estrogen related receptor, beta

FACS Fluorescence-Activated Cell Sorting

FCS Fetal Calf Serum

FGF Fibroblast Growth Factor

Fgf5 Fibroblast growth factor 5

gDNA genomic DNA

GMEM Glasgow-Minimal Essential Medium

GO Gene Ontology

GTF Gene Transfer Format

H3K4me1 Histone 3 lysine 4 mono-methylation

H3K4me2 Histone 3 lysine 4 di-methylation

H3K4me3 Histone 3 lysine 4 tri-methylation

HCA Human Cell Atlas

ICM Inner Cell Mass

IFC Intergrated Fluidic Circuit

IWP-2 Inhibitor of WNT Production-2

Krt18 Keratin 18

KSR Knockout Serum Replacement

LAD Lamina-Associated Domain

LIF Leukaemia Inhibitory Factor

lncRNA long non-coding RNA

MEF Mouse Embryonic Fibroblast

mESC mouse Embryonic Stem Cell

MF Molecular Function

mRNA messenger RNA

Nanog Nanog homeobox (from irish Tír na nÓg, which means Land of Youth)

NAST Non-Annotated Stem Transcript

ncRNA non-coding RNAs

NEAA Non-essential Amino Acid

Nlrp4f NOD-like receptor family, pyrin domain containing 4F

PCA Principal Component Analysis

PCR Polymerase Chain Reaction

Pou3f1 POU domain, class 3, transcription factor 1

Pou5f1/Oct4 POU domain, class 5, transcription factor 1

PSC Pluripotent Stem Cell

QC Quality Control

RME Random Monoallelic Expression

Rn7sk RNA, 7SK, nuclear

rRNA ribosomal RNA

RT Reverse Transcriptase

scBS-seq single-cell Bisulfite Sequencing

scRNA-seq single-cell RNA-sequencing

scRRBS single-cell Reduced-Representation Bisulfite Sequencing

smFISH single molecule Fluorescence In Situ Hybridization

SNP Single Nucleotide Polymorphism

snRNA small nuclear RNA

Sox4 SRY (sex determining region Y)-box 4

t-SNE t-Distributed Stochastic Neighbor Embedding

TAD Topology-Associated Domain

Tmem263 Transmembrane protein 263

Tmem59l Transmembrane protein 59-like

TPM Transcripts Per Million

tRNA transfer RNA

Tsix X (inactive)-specific transcript, opposite strand

TSS Transcription Start Side

Vim Vimentin

Wnt Wingless-type MMTV integration site family

Wnt8a Wingless-type MMTV integration site family, member 8A

XCI X Chromosome Inactivation

Xist X (inactive)-specific transcript

Zeb2 Zinc finger E-box binding homeobox 2

Zfp42 Zinc finger protein 42

1. Introduction

Until recently most studies analysing gene expression were performed on large numbers of pooled cells. However, to understand coupled transcriptional regulation and transcriptional heterogeneity, single-cell RNA-sequencing (scRNA-seq) technologies have been developed and in recent years are becoming the state-of-the-art application in various fields such as developmental biology, neuroscience and immune system research.

In this work, the transcriptome dynamics of the mouse naïve-to-primed transition process have been systematically investigated at the single-cell level in hundreds of cells for the first time. Two different protocols have been applied on the commercial Fluidigm C1^{TM} Single-Cell Auto Prep system, conventional C1 scRNA-seq and single-cell C1-Cap Analysis of Gene Expression (C1-CAGE). They have been applied on time course samples of naïve pluripotent mouse Embryonic Stem Cells (mESC) undergoing stimulus triggered transition towards primed pluripotent Epiblast Stem Cells (EpiSC). Cells have been collected in 24-hour intervals for four days (Day 0, 1, 2, 3, 4) after induction, because the differentiation process takes about five days. Additionally, untreated mESCs and fully primed EpiSCs have been sampled as references. Pseudotime analysis has been used to determine the temporal order of cell samples from transitioning time points. By combining the two protocols transcriptome datasets, we aimed to shed light on the dynamics of the naïve-to-primed stem cell conversion associated with major changes in transcriptional networks. Since expression profiles of cells before and after the female X Chromosome Inactivation (XCI) have been obtained, one could delineate a molecular road map towards XCI. Due to the hybrid nature of the used female mouse model system, our dataset also allows to facilitate studies on the establishment and maintenance of monoallelic gene expression such as imprinted gene expression or autosomal Random Monoallelic Expression (RME). Specifically, female mESCs derived from intersubspecific hybrid

embryos, MSM/Ms x C57BL/6J have been used. This hybrid contains approximately one Single Nucleotide Polymorphism (SNP) per 100 bp between the two mouse strains (Abe *et al.*, 2004). Because of that, it is possible to perform Allele-specific Expression Analysis (AEA) for most transcripts.

1.1. Mouse stem cells – naïve and primed state

Pluripotent Stem Cells (PSC) can be distinguished into naïve and primed cells and generally have the potential to give rise to all somatic lineages and the germline. PSCs have an unlimited self-renewal capacity when grown under appropriate conditions and can differentiate into tissues of all three germ layers in vitro. Additionally, when grafted into the early embryo, naïve PSCs can contribute to all somatic lineages as well as the germline. Primed post-implantation EpiSCs have the same in vitro potential as naïve PSCs, but they are limited in their pluripotency in vivo, due to their inability to give rise to the germline (Rossant, 2008; G. Guo *et al.*, 2009). mESCs can be maintained in the naïve state for long term, when cultured in the presence of serum and Leukaemia Inhibitory Factor (LIF). If serum is not included in the medium, the naïve state will be lost, but this limitation can be overcome by adding two small molecule kinase inhibitors to the medium, which is referred to as LIF-2i.

In vivo naïve mouse stem cells are derived from the Inner Cell Mass (ICM) of pre-implantation blastocysts, whereas primed stem cells are post-implantation embryonic stem cells. Naïve and primed stem cells mainly differ in their expression of pluripotency associated transcription factors, the morphology of colonies in cell cultures and the XCI status of female cells. Both naïve and primed PSCs express a common set of pluripotency related transcription factors but have other distinctive differences in the activity of development related signalling networks, regulation of pluripotency related gene expression, as well as on the epigenetic and metabolic level (Table 1).

Table 1: Naïve and primed pluripotent stem cell characteristics in mouse (modified from Weinberger *et al.* (2016))

Property	**Naïve**	**Primed**
MEK–ERK dependence	No	Yes
Long-term dependence on FGF2 signalling	No	Yes
Long-term dependence on TGFβ–activin A signalling	No	Yes
Dominant OCT4 enhancer	Distal	Proximal
H3K27me3 on developmental regulators	Low	High
Global DNA hypomethylation	Yes	No
X chromosome inactivation	No	Yes
Dependence on DNMT1, DICER, METTL3, MBD3	No	Yes
Priming markers (OTX2, ZIC2)	Down	Up
Pluripotency markers (NANOG, KLFs, ESRRβ)	Up	Down
TFE3 nuclear localization	High	Low
CD24/MHC class 1	Low/low	High/mod
HERV-H and HERV-K expression	High	Low
Expressed adhesion molecules	E-cadherin	L-cadherin
Promotion of pluripotency maintenance via Nanog or Prdm14	Yes	No
Metabolism	Oxidative phosphorylation, glycolytic	Glycolytic
Competence as initial starting cells for PGCLC induction	High	Low
Capacity of colonization of host pre-implantation ICM and contribution to chimaeras	High	Low
Colony morphology	Small and dome shaped	Large and flat
Cell mortality	Low	High

A recently published protocol for the derivation of EpiSCs used IWP-2, an inhibitor of Wnt secretion, to derive primed EpiSCs which are able to maintain a homogenous, undifferentiated primed state, while retaining a high differentiation potential (Sugimoto *et al.*, 2015). By modifying this protocol, we were able to efficiently convert mESCs to EpiSC-like cells in a reproducible manner. Wnt enables the spontaneous differentiation into germ layer derivatives and thus is a propagator of differentiation in the post-implantation embryo. Therefore, continuous treatment with IWP-2 is necessary to maintain the primed state.

1.2. Single-cell transcriptomics

The basic building block of all organisms is the cell. Recent years have brought a plethora of new technologies, which allow to examine up to tens of thousands of single cells in a matter of hours. These developments have led to dramatic drops in costs over the last years. There are three main separation approaches to isolate single cells on a larger scale. Firstly, using microwells and Fluorescence-Activated Cell Sorting (FACS) into wells to physically separate cells, secondly mircofluidics with miniaturized reaction compartments and thirdly droplet-based technologies, which use barcoded primers in droplets or attached to beads to tag cells for downstream deconvolution of single cells. In this study we used a commercially available disposable microfluidic array called C1 Intergrated Fluidic Circuit (IFC) from the company Fluidigm. The array is available in three sizes, depending on the average diameter of the cells studied. We exclusively used the medium sized array with a range from 10 to 17 μm, which corresponds to the average diameter of the cells used in this thesis project. These arrays allowed to capture a maximum of 96 cells per IFC, but due to recent developments in technology chips are now available, which enable the capture of 800 cells per run. The advantage of IFC arrays is that they are well established in the community, minimize reagent consumption due to the nanoliter scale of reaction chambers and allow the development of custom-made protocols (A. R. Wu *et al.*, 2014). Currently, nine protocols including C1-CAGE are available. In

comparison, droplet-based technologies enable the fast screening of thousands of cells for libraries at a cost of less than 10 cents per cell, but one major limiting factor is the price of sequencing thousands of cells at reasonably high coverage (Macosko *et al.*, 2015; Klein *et al.*, 2015; Shalek and Benson, 2017). This limitation will be reduced over time as cost for sequencing has been dropping steadily since the introduction of next generation sequencing technologies.

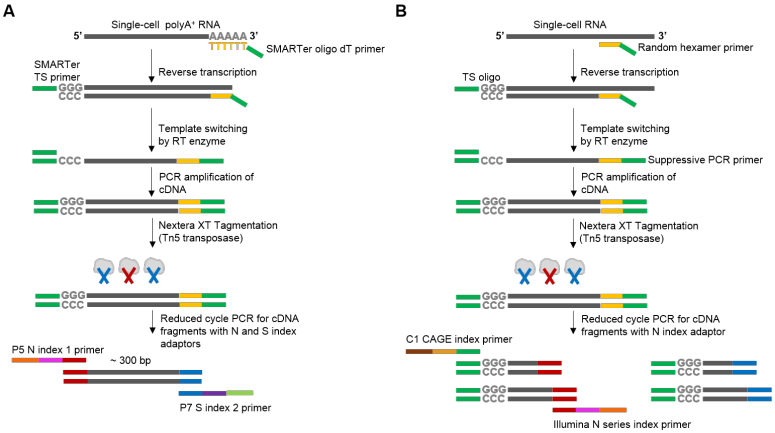


Figure 1: Method overview of **A**) Fluidigm C1 scRNA-seq and **B**) C1-CAGE.

The standard C1 scRNA-seq protocol uses a tagged poly-dT primer to initiate first strand complementary DNA (cDNA) synthesis. Upon reaching the 5' end of the messenger RNA (mRNA), the Reverse Transcriptase (RT) enzyme's terminal transferase activity will add a few cytosines to the 3' end of the cDNA. The RT reaction mix contains template-switching primers with a few guanosines at its 3' end, which will bind to the added cytosines on the cDNA and thus create an extended template. The RT will synthesize the additional primer sequence and produces cDNA that contains the SMARTer kits universal tag sequence, the 3' end of the template mRNA and the full-length transcript (Fluidigm protocol PN 100-7168 I1). In contrast the C1-CAGE uses a random hexamer priming strategy for first strand

cDNA synthesis (Kouno *et al.*, 2019) and therefore enables the capture of non-polyadenylated RNAs which includes most non-coding RNA species.

The amplification of cDNA is followed by the tagmentation step, which utilizes a derivative of the Tn5 transposase that has the ability to catalyze in vitro the integration of customized oligonucleotides into DNA, thus the Tn5 transposase can at the same time fragment and tag genomic DNA (gDNA) (Adey *et al.*, 2010). Tn5 transposase activity allows DNA fragmentation, sequencing adapter ligation and fragment size enrichment to occur in one step, which as well brings the advantage of reduced material loss. The average fragment size generated by the transposase activity is around 300 to 400 bp. A following reduced cycle PCR introduces Illumina index adaptors, that allows multiplexing of 96 cell samples for sequencing of pooled samples on a single Illumina HiSeq2500 lane.

1.3. Applications of scRNA-seq in biological research

Before scRNA-seq technologies became available, identification of new cell types often involved specific cell surface marker proteins. This approach is quite powerful, but requires pre-existing knowledge or hypothesizing of the existence of such markers. With scRNA-seq it is feasible to classify cells based on their transcriptional profiles without prior knowledge or awareness of potential new cell subtypes.

One of the earliest studies that shows the power of microfluidic-based scRNA-seq in development, identified alveolar, bronchiolar and progenitor cells in the mouse embryonic distal lung epithelium in an unbiased manner by using single-cell expression data of known marker genes. The results of that study show that scRNA-seq allows the molecular characterization of groups of cells in a heterogenous population (Treutlein *et al.*, 2014). Different types of brain tissues have been intensively studied with scRNA-seq technologies. One study obtained 3005 single-cell transcriptomes of mouse cortex and hippocampus and revealed a complex microanatomy of nine major cell

classes such as interneurons, astrocytes, microglia or oligodendrocytes. These could be further distinguished into 47 subclasses, with some of the these being novel subclasses (Zeisel *et al.*, 2015). Similarly, 5072 oligodendrocytes from multiple regions of the central nervous system of the mouse have been profiled in another investigation. They could establish oligodendrocytes as a transcriptionally heterogeneous cell lineage by showing that these cells follow a region and age specific distribution (Marques *et al.*, 2016).

Halpern *et al.* (2017) used a set of genes differentially expressed alongside the hexagonal liver lobule from the lobule center to the edges. They showed that it is possible to infer spatial origin of cells based on expression profiles of these landmark genes. This process is called zonation of the lobule. The transcriptomes of thousands of mouse liver lobule cells have been studied and the findings have been validated with single molecule Fluorescence In Situ Hybridization (smFISH). This was the first study establishing the combination of scRNA-seq and smFISH as a solid way to achieve spatial resolution of cellular origins in structured organs.

In another study Drosophila embryos consisting of around 6000 cells each have been profiled by Drop-seq and the resulting transcriptome data has been used to successfully map cells back to their spatial position based on the combinatorial expression of 84 marker genes. This approach allowed the researchers to create a virtual Drosophila embryo which can be used to identify genes with distinct spatial patterns and thus might be interesting genes involved in early embryo development (Karaiskos *et al.*, 2017).

1.4. Other single-cell -omics technologies

Apart from single-cell transcriptomics which has seen the most diverse and rapid development, the acquisition of other single-cell molecular and structural information is also desired. Protocols have been established to study chromatin accessibility, conformation, epigenetic marks such as methylation status or histone modifications as well as genome read-outs and even combinations of different -omics from the

same cell.

DNase-seq is a method that utilizes mapping of DNase I Hypersensitive Sites (DHS) to gain information on regulatory elements, such as enhancers or promoters (Boyle *et al.*, 2008; Thurman *et al.*, 2012). In recent years the Assay for Transposase-Accessible Chromatin (ATAC-seq) became available as a tool to reveal loci of open chromatin (Buenrostro, Giresi, *et al.*, 2013). ATAC-seq makes use of the same hyperactive Tn5 transposase used in the scRNA-seq protocol but loaded with sequencing adaptors to identify these regions. There have been successful adaptations of both methods on the single-cell level named scDNase-seq and scATAC-seq respectively. The scDNase-seq has been used to detect thousands of tumor-specific DHSs (Jin *et al.*, 2015) and two independent approaches to scATAC-seq have been published. In one approach microfluid IFCs were used to assess the epigenomic heterogeneity of 254 lymphoblastoid cells (Buenrostro, B. Wu, *et al.*, 2015), whereas in the second method combinatorial cellular indexing was used to measure chromatin accessibility in thousands of single nuclei (Cusanovich *et al.*, 2015). This involves a dual barcoding strategy, with transposase tagging of nuclei happening in bulk during the first step.

A powerful method to study 3D conformation of chromosomes is Hi-C. Hi-C is a proximity-based assay that allows sequencing of DNA of spatially adjacent segments of chromatin (Lieberman-Aiden *et al.*, 2009). Recently Hi-C has been modified to be used on the single-cell level and several Hi-C datasets have been published (Nagano, Lubling, Stevens, *et al.*, 2013; Nagano, Lubling, Yaffe, *et al.*, 2015; Nagano, Lubling, Várnai, *et al.*, 2017; Stevens *et al.*, 2017; Flyamer *et al.*, 2017). Technologies such as Hi-C revealed that mammalian genomes are organized in Topology-Associated Domains (TADs), but in order to better understand how chromosomes are organized in interphase nuclei, one can study interactions between chromosomes and the nuclear lamina. Previously, lamina interaction maps with hundreds of lamina-associated domains (LADs) have been created from individual human myeloid leukemia cells (Kind *et al.*, 2015). A first proof of principle attempt on single-cell Chromatin Immunoprecipi-

tation sequencing (ChIP-seq) was able to identify three subpopulations within thousands of H3K4me2 (e.g. histone 3 lysine 4 di-methylation) profiled mESCs, thus adding ChIP-seq to the single-cell omic toolbox (Rotem *et al.*, 2015). Two methods for single-cell methylation analysis have been published. Methylation levels are important to regulate cell type maintaining transcriptional programs. The first method is single-cell Bisulfite Sequencing (scBS-seq) and allows to accurately detect DNA methylation at up to 48.4% of CpG sites (Smallwood *et al.*, 2014). The second protocol is scRRBS and provides a better coverage of CpG islands compared to scBS-seq but covers overall fewer CpG sites genome-wide (H. Guo *et al.*, 2015). A technology for genome-wide detection of 5-hydroxymethylcytosine (5hmC) has been developed and published under the name scAba-seq and the authors showed that this epigenetic mark can be used as a tool for lineage reconstruction (Mooijman *et al.*, 2016).

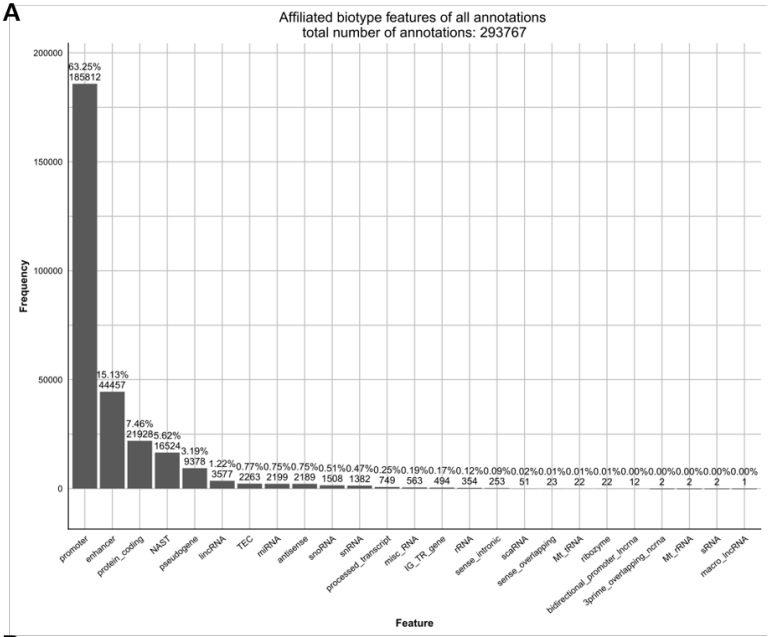
One major ongoing development in the field of single-cell technology is the establishment of protocols for multiple -omic read-outs from the same cell in parallel. A first step in that direction was the introduction of G&T-seq and DR-seq, which enable the parallel analysis of genome and transcriptome from the same cell (Macaulay *et al.*, 2015; Dey *et al.*, 2015). With the availability of methods to separate high quality gDNA and mRNA, one could attempt to use the gDNA fraction to study epigenomic marks apart from basic genome sequencing. A first successful application combined scRNA-seq and scBS-seq on gDNA gained based on the G&T-seq protocol and the advanced combination protocol is called scM&T-seq (Angermueller *et al.*, 2016). In theory, one can try combinations of all the available protocols that utilize single-cell gDNA as starting material. A current limitation to address is the efficiency of high quality gDNA extraction. Differing amounts of quality gDNA are needed as input for different protocols. The integrated analysis of genome, transcriptome and epigenome has the potential to lead to unprecedented insights into the influence that epigenetic marks have on transcriptional regulation and genome organisation.

1.5. Transcriptome reference annotation

Being able to capture non-polyadenylated non-coding RNAs in single cells enables a better understanding of their biogenesis and potential functions. The C1-CAGE dataset produced as part of this thesis contains expression information on non-polyadenylated ncRNAs involved in processes like transcript processing (snRNAs) and protein translation (rRNAs, tRNAs). Other regulatory ncRNAs such as enhancer RNAs (eRNAs) (Kim *et al.*, 2010) or the potentially stem cell specific class of non-annotated stem transcripts (NASTs) (Fort *et al.*, 2014) can also be detected. Nevertheless, the analysis of these groups of regulatory non-coding RNAs is not part of this thesis but can be pursued by the scientific community upon public release of the associated sequencing FASTQ files.

For this thesis we aimed to maximize the assignment of C1-CAGE sequencing reads to known annotated regions. To do that an annotation reference file that combines an abundance of biotype features has been created. A detailed description of how the combined annotation reference file was created can be found in the methods section of this thesis. For simplification we modified the biotype concept of the Gencode reference. Biotypes are basically terms that group similar classes of transcripts (See https://www.gencodegenes.org/pages/biotypes.html). In Figure 2A we show the frequency of all biotypes found in our combined reference file. As expected, promotor and enhancer biotypes as functional non-coding elements contribute by far the largest proportion followed by the group of known protein coding genes. For example, enhancer loci are annotated based on chromatin marks such as high ratio of H3K4me1 (e.g. histone 3 lysine 4 mono-methylation) to H3K4me3 (Heintzman *et al.*, 2007). It was also discovered that enhancers are actively transcribed and that this correlates with increased expression of nearby genes (Kim *et al.*, 2010). Enhancers are specifically involved in the naïve-to-primed pluripotency development (Buecker *et al.*, 2014). Therefore, we highlight the enhancer biotype, as our C1-CAGE dataset is a unique resource to study eRNA expression at single-cell resolution and investi-

gation of eRNA expression might help to elucidate gene expression regulation in naïve-to-primed pluripotency. In total, we combined five different annotation sources. Figure 2B shows the proportion of unique annotations contributed by each source and the overlap of annotated loci for all combinations of these sources.


A
Affiliated biotype features of all annotations
total number of annotations: 293767
Frequency
Feature
63.25% 185812
15.13% 44457
7.46% 21928
5.62% 16524
3.19% 9378
1.22% 3577
0.77% 2263
0.75% 2199
0.75% 2189
0.51% 1508
0.47% 1382
0.25% 749
0.19% 563
0.17% 494
0.12% 354
0.09% 253
0.02% 51
0.01% 23
0.01% 22
0.01% 22
0.00% 12
0.00% 2
0.00% 2
0.00% 2
0.00% 1
promoter
enhancer
protein_coding
NAST
pseudogene
lincRNA
TEC
miRNA
antisense
snoRNA
snRNA
processed_transcript
misc_RNA
IG_TR_gene
rRNA
sense_intronic
scaRNA
sense_overlapping
Mt_tRNA
ribozyme
bidirectional_promoter_lncrna
3prime_overlapping_ncrna
Mt_rRNA
sRNA
macro_lncRNA

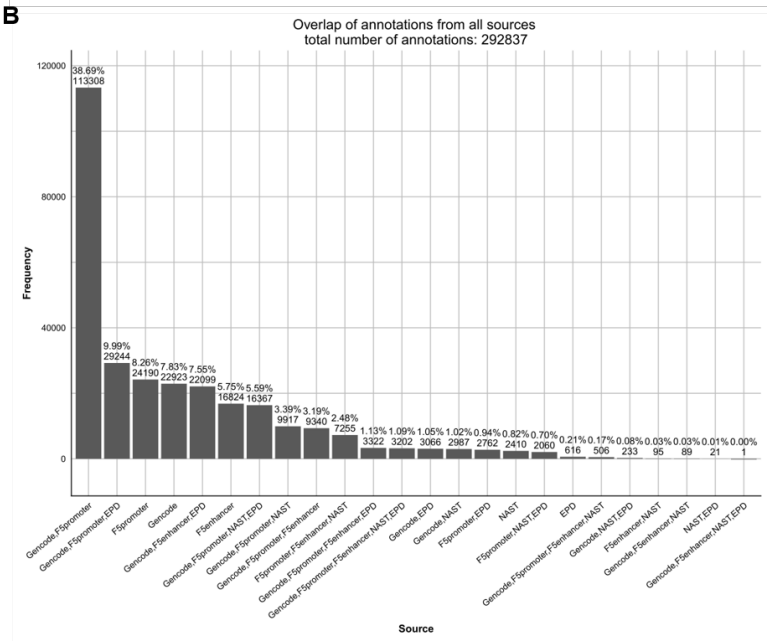

B
Overlap of annotations from all sources
total number of annotations: 292837
Frequency
Source
38.69% 113308
9.99% 29244
8.26% 24190
7.83% 22923
7.55% 22099
5.75% 16824
5.59% 16367
3.39% 9917
3.19% 9340
2.48% 7255
1.13% 3322
1.09% 3202
1.05% 3066
1.02% 2987
0.94% 2762
0.82% 2410
0.70% 2060
0.21% 616
0.17% 506
0.08% 233
0.03% 95
0.03% 89
0.01% 21
0.00% 1
Gencode,F5promoter
Gencode,F5promoter,EPD
F5promoter
Gencode
Gencode,F5enhancer,EPD
F5enhancer
Gencode,F5promoter,NAST,EPD
Gencode,F5promoter,NAST
Gencode,F5promoter,F5enhancer
F5promoter,F5enhancer,NAST
Gencode,F5promoter,F5enhancer,EPD
Gencode,F5promoter,F5enhancer,NAST,EPD
Gencode,EPD
Gencode,NAST
F5promoter,EPD
NAST
F5promoter,NAST,EPD
EPD
Gencode,F5promoter,F5enhancer,NAST
Gencode,NAST,EPD
F5enhancer,NAST
Gencode,F5enhancer,NAST
NAST,EPD
Gencode,F5enhancer,NAST,EPD

Figure 2: Manually curated annotations file for C1-CAGE data. **A**) Quantified biotypes included in the annotations file and **B**) Quantified annotations based on reference source. Overlapping annotated regions are aggregated.

1.6. Data management

One of the challenges of the coming-of-age of single-cell omics technologies is that they produce vast amounts of different types of data. Among the most common data types are FASTQ sequencing files, image files and many experiments-based information that can be summarised as experimental metadata. This metadata can comprise anything directly related to an experiment, for example the quality control measurements before library sequencing such as cDNA concentration measurements, or data on the total number of reads per sample, data derived from visual inspection of microscopic images, unique sample identifiers, positional information of controls, personal comments, and more. Basically, anything that the experimenter deems worth keeping track of can be summarised in metadata tables. Any statistics derived from metadata can also be considered metadata as well as statistics derived from intermediary files created in the process of a suitable data analysis workflow downstream of raw data generation. Typical downstream analysis files include BAM or SAM files, which are the output results of aligning sequencing reads to a reference sequence, such as the mouse genome. Further downstream of the data processing might be gene level expression tables, result tables for differential gene expression, various new sample annotations used for visualization purposes, e.g. cluster or pseudotime annotations, or the result tables of gene ontology analysis. Briefly, single-cell omics experiments have plenty of associated data and managing all the different raw data sources as well as affiliated analysis results requires a dedicated management pipeline. Especially in the context of international consortium research projects, such as the massive undertaking that is the Human Cell Atlas (HCA) project, which does

not only requires best practise standard data management procedures, but also access to necessary computational infrastructure for sharing data, secure authenticated data access, as well as suitable computational resources for demanding data processing on large numbers of files with varying file sizes that can easily scale up to terabyte amounts of data (HCA Consortium *et al.*, 2017)

Coincidentally, during the ongoing work for this thesis a web platform was developed as part of an in-house collaboration to manage the sharing of data directly linked to the single-cell project that is part of this thesis. The SCPortalen database has been constantly extended and improved beyond the in-house usage to enable the scientific community to easily search and access most of the publicly available scRNA-seq datasets and provides valuable summary statistics associated with each dataset (Abugessaisa *et al.*, 2018). For this thesis project SCPortalen was used to store all relevant experiment output files upstream and downstream of data processing. Each single-cell sample can be searched for using unique sample identifiers, which are comparable to database primary keys. The results of such a search includes all microscopic images, sequencing FASTQ files and alignment BAM files, as well as links to affiliated metadata, expression or other result tables. The platform can be used to manage data pre-publication and data can easily be released to the public at any time.

For easy visualization of alignment data there is a powerful tool called ZENBU, which is the platform name inspired by the Japanese word for "Everything". ZENBU can be used as a customizable platform for drag-and-drop style data uploads to display the results of aligned reads along a reference sequence and allows the user to create and freely modify viewing tracks of uploaded samples in many useful ways (Severin *et al.*, 2014). ZENBU was used on different occasions during this thesis project, such as comparing overlaid expression of scRNA-seq time point sample groups (Figure 5) or for visual validation of differential gene expression.

1.7. Dimension reduction and clustering methods

One of the challenges of single-cell transcriptomics and other high-throughput technologies is to identify meaningful patterns and structures in data sets with large numbers of samples and measurements. Among the most frequently used ways to get a better understanding of scRNA-seq data is the application of dimension reduction algorithms such as Principal Component Analysis (PCA) (Hotelling, 1933), t-Distributed Stochastic Neighbor Embedding (t-SNE) (Maaten and Hinton, 2008) and diffusion maps (Angerer *et al.*, 2016). The first two, PCA and t-SNE are used in this work and will thus be explained in more detail.

PCA is a linear dimension reduction Algorithm that allows the identification of patterns in high dimensional data by identifying components that explain as much variance in the data as possible. These components are a linear combination of the original variables and components. They must be uncorrelated, with the first component encompassing the largest variance and successive components capturing a decreasing amount of variance. Component variables are grouped together based on strong correlation of variables that form a component. Plotting the first two or three components is a good way to reveal hidden structures in the data. Linear dimension reduction algorithms focus on placing dissimilar data points far apart from each other in the lower dimensional space.

In contrast to PCA, t-SNE is a non-linear algorithm that is based on probability distributions with random-walk on neighborhood graphs to find structure within data. In short t-SNE computes pairwise conditional probabilities and tries to minimize the sum of the difference of probabilities in higher and lower dimensions. A major difference and advantage over PCA is that it focuses on placing similar data points closely together in the lower dimensional representation of the data. Moreover, it expands denser clusters and contracts sparser clusters to make cluster sizes appear more even. Therefore, cluster sizes in t-SNE plots are not meaningful and distances between different clusters may or may not be relevant either. On the other hand, t-SNE is also

more computationally expensive than PCA and input variables of the lower dimensional representation are no longer identifiable. t-SNE is widely used in various fields such as bioinformatics, or image and music analysis.

1.8. Random allelic expression

It has been previously shown that scRNA-seq allows the analysis of allelic expression by using hybrid mouse models that have a sufficiently high number of SNPs that allow the differentiation between maternal or paternal origin. One study showed that 12-24% of autosomal genes are showing consistent RME in mouse embryos (Deng *et al.*, 2014). The data generated for this thesis has the potential to be used for allelic expression analysis due to the female hybrid mouse embryo line used with an average of one SNP per 100 bp. However, the investigation of RME is not part of this thesis. This type of analysis is done by collaborators of this thesis research project. Nevertheless, one obvious feature that will be shown is the XCI event that marks the transition from the naïve-to-primed state. Mono allelic expression can appear due to natural transcriptional bursting or due to technical drop out events but regulated mono allelic expression events such as XCI can be more reliably detected (Petropoulos *et al.*, 2016). One can validate the results of an allelic expression analysis by checking for known escape genes or imprinted genes.

2. Materials and Methods

The wet and dry lab sections that have been performed by the thesis author himself consist of all steps after harvesting cells for loading into the C1 array. This includes preparing and running the C1 arrays, imaging of array capture chambers, harvesting of cDNA, sequence library construction and diverse quality checks. The sequencing itself was outsourced to an in-house sequencing facility and any bioinformatic dry lab work started from FASTQ raw read sequence files obtained from the sequencing facility. All computational workflows and analysis results shown in this thesis have been implemented by the author. For completion we mention here that the part pertaining the allelic expression analysis was done by a project collaborator at RIKEN BioResource Reseach Center in Tsukuba and is therefore not presented as part of this thesis, although it is a part of the single-cell project that this thesis is based on. The same collaborator was also responsible for providing all cell time course samples and for harvesting all cell samples. Due to that the following section on cell culture is based on information provided by the collaborator and is included here for better understanding and completeness.

2.1. Cell Culture

The mESCs used in this study originated from F1 inter-subspecific hybrid embryos (MB3) between C57BL/6J (B6) and MSM/Ms (MSM) (RIKEN RBC No. RBRC00209). MSM is an inbred mouse strain derived from the Japanese wild mouse Mus musculus molossinus, which has many SNPs compared to the B6 type (Abe *et al.*, 2004) and is thus suitable for allelic expression analysis. As reference for the primed state we used the female EpiSC line 129Ba2, a 129/B6N F1 hybrid line (Sugimoto *et al.*, 2015). Additionally, we sampled induced primed PSC-like cells at Day 22 (P10) as well as a clonal cell line isolated from the induced primed PSC-like cells sampled at passage 20 (Clone

1E). For mESC culture we used Glasgow-Minimal Essential Medium (GMEM) (Sigma-Aldrich) supplemented with 14% Knockout Serum Replacement (KSR) (Life Technologies), 1% ES culture grade Fetal Calf Serum (FCS) (Life Technologies), 1x Non-essential Amino Acid (NEAA) (Life Technologies), 1000 units/mL LIF (ESGRO, Millipore), 100 μM 2-mercaptoethanol and penicillin/streptomycin. mESCs were maintained on mitomycin C (Sigma-Aldrich) treated Mouse Embryonic Fibroblast (MEF) feeder cells (Robertson, 1987). ES cells were seeded onto MEF feeders at a density of $1x10^5$ cells per 3 cm dish and cultured in ES medium over night at 37°C. For conversion of ES cells to EpiSC-like cells, ES cell medium was replaced with EpiSC medium (DMEM/F12 plus glutamax (Gibco), 1xNEAA (Life Technologies), 15% KSR (Life Technologies), 5 ng/ml of basic Fibroblast Growth Factor (FGF) (Reprocell), 10 ng/ml of Activin A (Wako) and 2 μM IWP-2 (Stemgent)) and cells were incubated over night at 37°C. The day of the medium change was set as Day 0. On the next day (Day 1), cells were passaged using CTKCa dissociation buffer (phosphate buffered saline containing 0.25% trypsin (BD Diagnostic Systems), 1 mg/ml of collagenase (Life Technologies), 20% KSR (Life Technologies), 1 mM CaCl2)) (Sugimoto *et al.*, 2015). The medium was changed every day and cells were passaged every other day. For harvesting cells were dissociated with 0.25% Trypsin, 1 mM EDTA and the cell suspension was placed onto a gelatinized dish to remove the feeders. This plate purification was done twice, before the purified time point cell samples were prepared for single-cell capture.

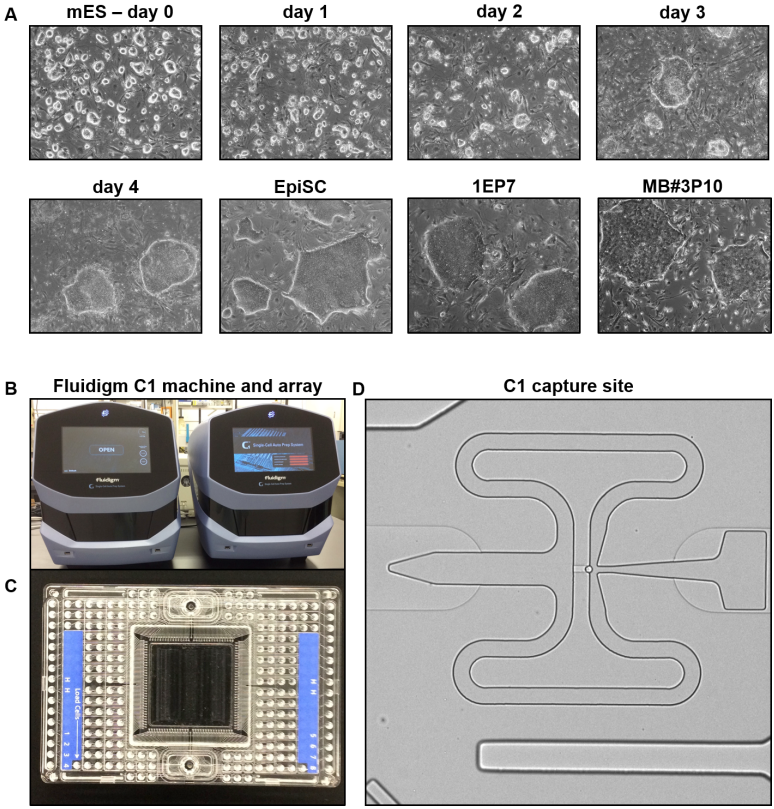


Figure 3: Selected images depicting the C1 workflow. **A**) Microscopic images of the cell culture dish for all time points before harvest and exemplary image of **B**) the C1 Fluidigm machine, **C**) C1 capture array and **D**) array capture chamber with captured cell.

2.2. Single-cell capture, imaging, RT and cDNA synthesis

For both protocols (scRNA-seq and C1-CAGE) viability of cells and cell concentration was estimated using the Countess Automated Cell Counter (Invitrogen) on a sample of trypan blue stained cells. After appropriate dilution 3000 cells were loaded in a C1 single-cell Auto

Prep array (Fluidigm, 100-5760) for mRNA-sequencing (10–17 µm) following the Fluidigm manufacturer's instructions and recommended reagents (PN 100-7168 l1). In brief, after priming the C1 array and loading of the cell mix, we added a Calcein AM/ Ethidium homodimer-1 staining mix (LIVE/DEAD kit, Life Technologies). The green fluorescent Calcein AM stains live cells with active esterase activity, whereas the red fluorescent ethidium homodimer-1 will stain cells which are in the process of losing or have already lost their membrane integrity. Both protocols followed the manufacturer's guide for the cell mix loading, staining, loading of reagent mixes for lysis, RT, PCR amplification and cDNA harvest. One noticeable diversion from the original manufacturer's instructions is the usage of External RNA Controls Consortium (ERCC) spike Mix 1 (Thermo Fisher, 4456740) (Munro *et al.*, 2014) instead of ArrayControl RNA spikes. Data sheets on how to prepare reagent mixes for C1-CAGE, thermo cycling protocols and information on the required C1 machine workflow software for C1-CAGE are publicly available on the C1 system from Fluidigm Scrip Hub platform (https://www.fluidigm.com/c1openapp/scripthub/script/2015-07/c1-cage-1436761405138-3). 20000 times diluted ERCC spike mix was used for C1-CAGE samples with the C1 run IDs 1772-123-295 and 1772-123-296 instead of a 200 times dilution as used for the other runs. The reason were ongoing improvements on the C1-CAGE protocol throughout this thesis project. Table 2 gives an overview over all samples collected for Fluidigm scRNA-seq and C1-CAGE.

Table 2: Overview of all sample run IDs for Fluidigm scRNA-seq and C1-CAGE that pass QC. Run IDs correspond to unique C1 array IDs.

Sample ID	**Time point**	**Status**	**Number of cells passing QC**
1772-116-259	day 0 – mESC	scRNA-seq	70
1772-116-260	day 1	scRNA-seq	83
1772-116-261	day 2	scRNA-seq	76
1772-116-263	day 3	scRNA-seq	74
1772-116-262	day 4	scRNA-seq	77
1772-151-313	day 4	scRNA-seq	55
1772-116-264	day 5	Discarded	–
1772-116-268	EpiSC	scRNA-seq	69
1772-123-289	1EP7	EpiSC clone	12
1772-123-291	MB#3P10	EpiSC hybrid	26
Total		**scRNA-seq**	**542**
1772-144-108	day 0 – mESC	C1-CAGE	46
1772-123-295	day 0 – mESC	C1-CAGE	69
1772-144-109	day 1	Discarded	–
1772-123-296	day 1	C1-CAGE	73
1772-146-168	day 2	C1-CAGE	73
1772-146-169	day 3	C1-CAGE	83
1772-146-170	day 4	C1-CAGE	78
1772-146-167	day 4	C1-CAGE	84
1772-146-171	EpiSC	C1-CAGE	81
Total		**C1-CAGE**	**587**
All			**1129**

The whole C1 machine workflow is compartmentalized into different steps that allow pauses between each follow-up step in order to prepare the next reagent mix or to perform imaging of the cell capture chambers in brightfield (Figure 3D), green filter and red filter mode after the cell load and staining step. Due to the different sample acquisition time points for both scRNA-seq and C1-CAGE we used two different imaging systems. One of which has been changed when a better imaging system became available throughout the ongoing thesis project. The first device used was Cellomics ArrayScan VTI High Content Analysis

Reader (Thermo Scientific) and it was applied as described in **Böttcher** *et al.* (2016). The main difference between the Cellomics platform and the later used IN Cell Analyzer 6000 system (GE Healthcare) is the easier use in automated C1 array scans and the capability of the IN Cell Analyzer to take z-stacked images, which show a vertical cross section of the capture chamber that enables visual detection of rare cases of stacked cell duplets and therefore allows more accurate tagging for removal in downstream computational analysis.

2.3. Library preparation and sequencing

The first step in the library preparation workflow is the dilution of harvested cDNA samples. The optimal concentration range is between 0.1 to 0.3 ng/μL. In case of the scRNA-seq protocol 2 μL of each cell have been diluted in appropriate amounts of harvest dilution buffer based on prior picogreen (Thermo Fisher, P11496) cDNA concentration measurements for each cell sample. The picogreen sample preparations were done following the Fluidigm Picogreen Excel template (100-6260_B2). The workflow for the library preparation equally follows the Fluidigm manufacturer's instructions using reagents from Illumina (FC-131-1096, FC-131-1002). In brief, the cDNA sample dilution step was followed by the tagmentation reaction, in which cDNA is cut into fragments. After that came a thermal enzyme deactivation step and finally an indexing PCR for multiplexing samples. scRNA-seq utilizes the Nextera XT index primer kit with 96 indices, whereas C1-CAGE uses a custom primer set (Invitrogen) instead of the kit's S index primer set. The sequences of these custom primers can be downloaded from the afore-mentioned C1-CAGE Script Hub platform. Finally, all samples were pooled after the index PCR and the pooled mix was purified using Agencourt AMPure XP magnetic beads as described in the Fluidigm manual. Before sequencing on Illumina HiSeq2500 we quantified all libraries (KAPA Library Quantification kit, KK4835) and adjusted the library concentration for loading on the flow cell to 9 pM. Library quality has been checked with Agilent High Sensitivity DNA kit (5067-4626) prior to loading on the flow cell. Figure 4 shows examples of

good quality Bioanalyzer library profiles. Finally, Fluidigm scRNA-seq protocol samples were sequenced in Illumina high-output mode, with paired ends and 100 bases and C1-CAGE in high output mode, also with paired ends but 50 bases length.

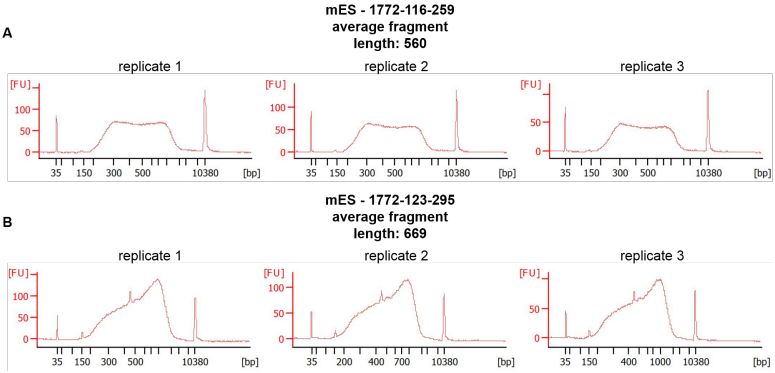


Figure 4: Bioanalyzer library profiles. **A**) Fluidigm scRNA-seq and **B**) C1-CAGE sample profiles. Flat plateaus are typical for good quality profiles of the Fluidigm scRNA-seq libraries, whereas profiles with a peak towards the tail are typical for C1-CAGE libraries. These profiles also allow estimation of average fragment sizes, which is required to calculate library concentration.

2.4. Raw sequencing data processing

2.4.1. scRNA-seq workflow

The alignment, or mapping of all FASTQ files from Fluidigm protocol sample runs was done with the STAR alignment software v2.4.1d (Dobin *et al.*, 2013) against the GRCm38p4 reference genome and Gencode M8 as annotation reference. Alignment output files in Binary Alignment Map (BAM) format were used for upload to the ZENBU browser for data visualization (Severin *et al.*, 2014) (Figure 5). The code used to perform any data processing and analysis is introduced below step by step. Single "#" signs precede a comment that is not

part of executable code. The expression "#!bin/bash -e" allows to run a file containing the code via the bash command in a shell.

```
#!bin/bash -e
# this creates the STAR index
STAR --runMode genomeGenerate --runThreadN 12 --genomeDir
     GRCm38p4_index/ --genomeFastaFiles GRCm38.p4.genome.fa --
     sjdbGTFfile gencode.vM8.chr_patch_hapl_scaff.annotation.gtf
```

```
#!bin/bash -e
# the actual STAR mapping
BASEDIR= FASTQfiles/
time (cd $BASEDIR ; ls) |
  sed 's/_R.\.fastq//g' |
  uniq |
parallel -j2 'STAR \
    --genomeDir GRCm38p4_index \
    --genomeLoad NoSharedMemory \
    --runThreadN 6 \
    --sjdbFileChrStartEnd EpiSC_STAR/Unsorted/{}.SJ.out.tab \
    --readFilesIn FASTQfiles/{}_R1.fastq FASTQfiles/{}_R2.fastq \
    --outFileNamePrefix EpiSC_STAR/Unsorted/{}. \
    --quantMode TranscriptomeSAM GeneCounts \
    --outSAMtype BAM Unsorted \
    --outSAMattributes All \
    --outReadsUnmapped Fastx'
```

We used Tagdust v2.13 (Lassmann *et al.*, 2009; Lassmann, 2015) to remove 140 undesirable sequences before starting the RNA-seq expression quantification. These sequences include the primer sequences of the Nextera kit, all ERCC spike sequences as well as sequences of the T7 transcription of the ERCC plasmids (https://www-s.nist.gov/srmors/certificates/documents/SRM2374_putative_T7_products_NoPolyA_v2.FASTA) and last but not least 5.8S, 18S and 28S rRNA sequences. Tagdust creates FASTQ files with unwanted sequences removed, which are used as input for the expression quantification step.

```
#!bin/bash -e
# tagdust.arch file contains the line:
# tagdust -1 R:N
# tagdust.fa file contains all base sequences and annotations of
    sequences to remove from the FASTQ files

mkdir -p tagdustOutput

TAGDUST=tagdust-2.13/src/tagdust
$TAGDUST | grep Copyright
BASEDIR=FASTQfiles/
  time (cd $BASEDIR/ ; ls ) |
  sed 's/_R._001\.fastq.gz//g' |
  uniq |
  parallel -j1 'tagdust-2.13/src/tagdust \
    -t 12 \
        -l tagdustOutput \
        -arch tagdust.arch \
        -o tagdustOutput/{} \
        -a tagdustOutput/{} \
        -ref tagdust.fa \
        FASTQfiles/{}_R1_001.fastq.gz FASTQfiles/{}_R2_001.fastq.
            gz'
```

Additionally, log files are created for each single cell FASTQ file pair. These log files contain read counts of all removed features. This count information is aggregated and integrated in a curated metadata file. In order to extract the counts, the following script must be executed within the log file directory.

```
#!bin/bash
for LIBRARY in 1772-116-259 1772-116-260 1772-116-261
    1772-116-263 1772-116-262 1772-116-264 1772-116-268
    1772-123-189 1772-123-291
do
  for ROW in A B C D E F G H
  do
    for COLUMN in 01 02 03 04 05 06 07 08 09 10 11 12
    do
        grep -e rRNA -e ERCC -e input -e complex -e Nextera -e
            artifacts -e Consensus ${LIBRARY}_${ROW}${COLUMN}_* |
        perl -pe "s/^/$LIBRARY\t$ROW$COLUMN\t/"
    done
  done
done > spikes.txt
```

Estimates of RNA expression were generated with Kallisto v0.44.0 (Bray *et al.*, 2016; Ntranos *et al.*, 2016) using Gencode M8 transcript IDs as reference. Kallisto is an alignment-free quantification software and therefore much faster than other RNA-seq quantification tools (Bray *et al.*, 2016). This is of importance, because scRNA-seq experiments typically involve hundreds or thousands of samples that need to be analyzed, thus speed and computation time can be limiting factors. The Kallisto program consists of two steps. First the creation of a transcriptome index and secondly the quantification.

```
#!bin/bash
kallisto index -i kallisto_M8 gencode.vM8.EnsembleTranscriptID.fa
      &&

mkdir -p kallistoOutput

BASEDIR=tagdustOutput/fastqOut
time (cd $BASEDIR ; ls) |
  sed 's/_R.\.fastq//g' |
  uniq |
  parallel -j2 'kallisto quant \
    -t 10 \
    -i kallisto_M8.idx \
    -o kallistoOutput/{}/ \
    tagdustOutput/fastqOut/{}_R1.fastq \
    tagdustOutput/fastqOut/{}_R2.fastq \
    -b 100 \
    --bias'
```

We combined the single-cell transcript-level expression values from the abundance.tsv output files into two comprehensive matrices with single cells in columns and rows with expression values as estimated count and Transcripts Per Million (TPM) values respectively. TPM is a normalized unit for RNA-seq expression that takes the length of annotated transcripts into account. For that purpose, we changed the column number in the script below to either select estimated counts or TPM values and ran the script once to save both into separate tables.

```
#!bin/bash -e
# create output directories
mkdir -p finalFiles intermediateFiles qTable

(cd kallistoOutput/ ; ls ) |
  parallel -j1 'sed 1d kallistoOutput/{}/abundance.tsv >
      finalFiles/{}_tquant.txt' &&

for i in finalFiles/*_tquant.txt
do
  CELL_ID=$(basename $i _tquant.txt)
  HEADER_FILE="intermediateFiles/${CELL_ID}.header"
  # create header file with: gene_name TPM per cell
  echo -e "gene\t$CELL_ID" > $HEADER_FILE
  # append header file with expression values for each cell
  # column 4 contains count values and column 5 TPM values
  cat $i | cut -f 1,5 >> $HEADER_FILE
done &&

# make sure that only 1 particular header file is chosen
cut -f1 intermediateFiles/$(cd kallistoOutput/ ; ls | head -n1).
    header |
paste -s > qTable/kallistoTPM
# Column 2 of each *.header file becomes a line of the qTable/
    kallistoTPM file
for FILE in intermediateFiles/*.header
do
  cut -f2 $FILE |
  paste -s >> qTable/kallistoTPM
done &&

rm -rf finalFiles intermediateFiles
```

Next, we transposed the resulting tables to create a proper expression matrix. For that the following R script was executed from the shell by calling Rscript with the required arguments input table and output file name. The output file name is for example "transcriptTPM_M8_20190206_Kallisto.txt", which contains the expression value type, reference transcriptome version and the creation day and software as part of the file name. Analogous to the bash version the expression "#!usr/bin/Rscript" is used to execute R scripts from within the shell via the Rscript command.

```
#!usr/bin/Rscript
args = commandArgs(trailingOnly=TRUE)

expressionTable <- function(expressionTable, outputFileName) {
  z <- as.data.frame(fread(expressionTable, stringsAsFactors=F,
       header=T))
  cellNames <- z[,1]
  y <- z[,2:ncol(z)]
  Z <- t(y)
  x <- as.data.frame(Z)
  colnames(x) <- cellNames
  rownames(x) <- colnames(z)[-1]
  write.table(x, outputFileName, sep="\t", row.names=T, col.names
       =T, quote=F)
}

expressionTable(args[1], args[2])
```

Finally, we aggregated the transcript expression to the corresponding gene level by summing up the expression of transcripts belonging to the same gene. For that purpose, we created a two-column lookup table that has Ensemble transcript IDs in column one and the corresponding gene symbol in column two. This step was entirely done in R. The file transcript2gene.txt was created by parsing the gencode.vM8.chr_patch_hapl_scaff.annotation.gtf file (Gene Transfer Format) from Gencode.

```
#!bin/bash
awk 'FS="\t" {if($3=="transcript"){print $9}}' gencode.vM8.
     chr_patch_hapl_scaff.annotation.gtf |
cut -d ';' -f 2,5 |
cut -d '"' -f2,4 |
sed 's/"/\t/g' > transcript2geneGTF.txt

grep ">" gencode.vM8.transcripts.fa |
cut -d '|' -f1,6 |
sed 's/|/\t/g' |
sed 's/>//g' > transcript2geneFasta.txt

join transcript2geneFasta.txt transcript2geneGTF.txt >
     transcript2gene.txt
```

As mentioned before, metadata is a table that was manually curated and contains useful cell sample information, such as which cells fail quality checks, the read counts from the tagdust step, or cDNA concen-

tration values from the picogreen assay. The metadata table version used here includes a "Pseudotime" column for convenience as well, although pseudotime estimation will be described further downstream of the analysis workflow.

```
# load input files
meta <- read.csv("metadata_20190223045845.csv", header=TRUE,
    stringsAsFactors = F)
# transcriptTPM_M8_20190206_Kallisto.txt is output of previous
    step
expr <- read.table("transcriptTPM_M8_20190206_Kallisto.txt",
    header=T, row.names = 1, check.names = FALSE)
# links transcript IDs to gene names
tr2gene <- read.table("transcript2gene.txt", header=F, check.
    names = FALSE)
names(tr2gene) <- c("transcriptID", "gene", "temp"); tr2gene <-
    tr2gene[,1:2]
tr2gene <- tr2gene[order(tr2gene$gene, decreasing = F),]
tr2gene$transcriptID <- as.character(tr2gene$transcriptID)
# sort transcript IDs and check that the order is the same in
    transcript expression table and gene reference table
test1 <- expr[order(rownames(expr), decreasing = T),]
test2 <- tr2gene[order(tr2gene$transcriptID, decreasing = T),]
# check that transcript IDs are identical in both tables
# the following must be TRUE
identical(rownames(test1), test2$transcriptID)
# add column with gene names to expression table
expr <- expr[match(tr2gene$transcriptID, rownames(expr)),]
expr$gene <- tr2gene$gene
# aggregate transcript expression to gene level
geneExpr <- aggregate(expr[,1:ncol(expr)-1], by=list(expr$gene),
    FUN=sum)
rownames(geneExpr) <- geneExpr$Group.1; geneExpr <- geneExpr[,-1]
# remove cells that are missing in Kallisto output
geneExpr <- geneExpr[, which(names(geneExpr) %in% meta$cell_id)]
meta <- meta[match(names(geneExpr), meta$cell_id),]
# reorder columns in expression table according to time points
expr <- geneExpr[,order(meta$Order, decreasing = F)]
# save gene expression table as file
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
write.table(expr, file = paste0("geneTPM_M8_", time, "_Kallisto.
    txt"), append=FALSE, sep="\t", quote=FALSE, row.names=TRUE,
    col.names=TRUE)
```

This concludes the Fluidigm scRNA-seq data processing workflow. The combined single-cell expression tables serve as input for the various data analysis steps that follow.

2.4.2. C1-CAGE workflow

C1-CAGE sample FASTQ files have been processed using the Moirai software platform (Hasegawa *et al.*, 2014; Kouno *et al.*, 2019) (https://github.com/Population-Transcriptomics/C1-CAGE-preview/blob/master/OP-WORKFLOW-CAGEscan-short-reads-v2.0.ipynb) The Moirai pipeline creates BED12 files for all C1-CAGE samples, which were used to make a CAGEexp object with the CAGEr R package (https://rdrr.io/bioc/CAGEr/). The following code uses BED12 files created though the Moirai software as input.

```
# loading required R packages
library("CAGEr")
library("MultiAssayExperiment")
library("SummarizedExperiment")
library("magrittr")

bedFilesPaths <- function(runID, stamp)
  paste0("BED12files/"
         , runID
         , ".OP-WORKFLOW-CAGEscan-short-reads-v2.0."
         , timeStamp
         , "/CAGEscan_fragments") %>%
  list.files(., full.names = TRUE)

run_1772_144_108_bed <- bedFilesPaths("1772-144-108", "
    20160905162245")
run_1772_144_109_bed <- bedFilesPaths("1772-144-109", "
    20160905162341")
run_1772_146_168_bed <- bedFilesPaths("1772-146-168", "
    20160905162426")
run_1772_146_169_bed <- bedFilesPaths("1772-146-169", "
    20160905165237")
run_1772_146_170_bed <- bedFilesPaths("1772-146-170", "
    20160905165419")
run_1772_146_167_bed <- bedFilesPaths("1772-146-167", "
    20160905165526")
run_1772_146_171_bed <- bedFilesPaths("1772-146-171", "
    20160905165553")
run_1772_123_295_bed <- bedFilesPaths("1772-123-295", "
    20170309172257")
run_1772_123_296_bed <- bedFilesPaths("1772-123-296", "
    20170309172353")

# remove empty BED files from list
run_1772_144_108_bed <- run_1772_144_108_bed[-38]
run_1772_146_168_bed <- run_1772_146_168_bed[-c
```

```
    (8,20,44,56,68,80,88,92)]
# combine all path information
bedFilesPaths <- c(run_1772_144_108_bed, run_1772_144_109_bed,
    run_1772_146_168_bed, run_1772_146_169_bed, run_1772_146_170_
    bed, run_1772_146_167_bed, run_1772_146_171_bed, run_1772_123
    _295_bed, run_1772_123_296_bed)

ce <- DataFrame(inputFiles = bedFilesPaths
                , inputFilesType = "bed"
                , sampleLabels = bedFilesPaths %>%
                                 basename %>%
                                 sub(".bed", "", .) %>%
                                 sub("^", "run", .) %>%
                                 make.names)
rownames(ce) <- ce$sampleLabels
ce <- CAGEexp(colData = ce)
genomeName(ce) <- "BSgenome.Mmusculus.UCSC.mm10"

# load data in the CAGEexp object
getCTSS(ce, useMulticore = TRUE)

# save CAGEexp object
saveRDS(ce, "CAGEexp_mm10.rds")
```

We made a custom BED file for annotating expressed Transcription Start Side (TSS) in order to make a comprehensive C1-CAGE gene expression matrix. The annotation BED file combines annotations for NASTs from DRA000914 (Fort *et al.*, 2014), the FANTOM5 mouse promotor and enhancer atlas (http://fantom.gsc.riken.jp/5/data/) and the Eukaryotic Promoter Database EPDnew mouse promotors (https://epd.epfl.ch/mouse/mouse_database.php?db=mouse) as well as the Gencode M8 reference transcriptome version. EPDnew is an annotated non-redundant collection of eukaryotic Pol II promoters. Since all the different annotation sources are disconnected from each other, redundancies of annotations for the same locus might exist and we handled these cases by hierarchically assigning the annotation in the following order of priority: Gencode, Fantom5 promoters, Fantom5 enhancers, NASTs and EPD promoters.

```
#!bin/bash
awk 'FS="\t" {if($3=="gene"){print $1,$4,$5,$7}}' gencode.vM8.
    chr_patch_hapl_scaff.annotation.gtf > temp1
awk 'FS="\t" {if($3=="gene"){print $1,$4,$5,$7,$9}}' gencode.vM8.
    chr_patch_hapl_scaff.annotation.gtf | cut -d ';' -f 2,4 | cut
     -d '"' -f2,4 | sed 's/"/\t/g' > temp2
```

```
paste temp1 temp2 > temp3
awk '{print $1,$2,$3,$4,$6,$5}' temp3 | sed 's/ \+/\t/g' | grep '
    chr' - > temp4
awk '{$2 = $2 - 1; print}' temp4 | sed 's/ \+/\t/g' > gencode.vM8
    .feature.bed

# concatenate the different source bed files
# Fantom5 enhancer map and UCSC liftover tool was used to
    liftover mm9 to mm10

awk '{print $1,$2,$3,$6,$4,$5}'
    mm10_permissive_enhancers_phase_1_and_2.bed | sed 's/ \+/\t/g
    ' > enhancers_temp.bed
awk 'BEGIN{FS=OFS="\t"} {$5 = "mm10_enhancer."NR; $6 = "enhancer
    "; print}' enhancers_temp.bed > mm10_enhancerFeature.bed
awk '{print $1,$2,$3,$6,$4,$5}' NAST_mm10.bed | sed 's/ \+/\t/g'
    > NAST_temp.bed
awk 'BEGIN{FS=OFS="\t"} {$5 = "mm10_nast."NR; $6 = "NAST"; print}
    ' NAST_temp.bed > mm10_NASTFeature.bed
awk '{print $1,$2,$3,$6,$4,$5}' Mm_EPDnew_002_mm10.one.bed | sed
    's/ \+/\t/g' > EPD_temp.bed
awk 'BEGIN{FS=OFS="\t"} {$5 = "mm10_epd."NR; $6 = "promoter";
    print}' EPD_temp.bed > mm10_EPDFeature.bed
awk '{print $1,$2,$3,$6,$4,$5}' mm10_v3.CAGE_peaks_merged.bed |
    sed 's/ \+/\t/g' > F5_temp.bed
awk 'BEGIN{FS=OFS="\t"} {$5 = "mm10_fantom5."NR; $6 = "promoter";
     print}' F5_temp.bed > mm10_F5Feature.bed
cat gencode.vM8.Feature.bed mm10_F5Feature.bed
    mm10_enhancerFeature.bed mm10_NASTFeature.bed mm10_EPDFeature
    .bed > concat.bed

# remove duplicate coordinate rows, but keep overlapping entries
    that are not congruent
sort -k1,1 -k2,2n -k3,3n -u concat.bed > sorted.concat.bed
bedtools annotate -both -i sorted.concat.bed -files gencode.vM8.
    Feature.bed mm10_F5Feature.bed mm10_enhancerFeature.bed
    mm10_NASTFeature.bed mm10_EPDFeature.bed > mm10_sourceAnnot.
    bed

# clean intermediary files
rm -rf *temp* *Feature.bed refTSS_mm10.bed *concat.bed
```

In order to be able to quickly check which annotation source contributed to any annotation, we added a new "Source" column, which names the annotation sources that overlap with any annotation (Figure 2B). In a strict sense the file we use is not a formal BED file anymore, nevertheless we refer to it as a BED file, because it still maintains the minimum requirement of the first 3 columns being the Chromosome on which a feature is located, the start position of the feature and the end

position of the feature.

```
# loading required R packages
library(data.table)

sourceAnnot <- fread("mm10_sourceAnnot.bed", header=F, check.
    names = FALSE)
names(sourceAnnot) <- c("Chromosome", "Start", "End", "Strand", "
    ID", "Feature", "Gencode_featureCount", "Gencode_
    featureCoverage", "F5promoter_featureCount", "F5promoter_
    featureCoverage",
"F5enhancer_featureCount", "F5enhancer_featureCoverage", "NAST_
    featureCount", "NAST_featureCoverage", "EPD_featureCount", "
    EPD_featureCoverage")

# add Source column
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'F5promoter'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'F5enhancer'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode,
    F5promoter'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5enhancer
    '
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode,NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
```

```
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'F5promoter,
    F5enhancer'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'F5promoter,NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'F5promoter,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'F5enhancer,NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'F5enhancer,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'NAST,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5promoter
    ,F5enhancer'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode,
    F5promoter,NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode,
    F5promoter,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5enhancer
    ,NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5enhancer
```

```
    ,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode,NAST,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'F5promoter,
    F5enhancer,NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'F5promoter,NAST,
    EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'F5enhancer,NAST,
    EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount == 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5promoter
    ,F5enhancer,NAST'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount == 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5promoter
    ,F5enhancer,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount == 0, 'Source'] <- 'Gencode,
    F5promoter,NAST,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount == 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5enhancer
    ,NAST,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount == 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'F5promoter,
    F5enhancer,NAST,EPD'
sourceAnnot[sourceAnnot$Gencode_featureCount > 0 & sourceAnnot$
    NAST_featureCount > 0 & sourceAnnot$EPD_featureCount > 0 &
    sourceAnnot$F5promoter_featureCount > 0 & sourceAnnot$
    F5enhancer_featureCount > 0, 'Source'] <- 'Gencode,F5promoter
    ,F5enhancer,NAST,EPD'

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
```

```
    character(Sys.time()), " ", n = 2))), collapse = "")
write.table(sourceAnnot, file = paste0("mm10_sourceAnnotation_",
    time, ".bed"), row.names = FALSE, col.names = TRUE, sep = "\t
    ", quote = FALSE)
```

The "Feature" column in this file stores the biotype string of the originally used Gencode GTF file (https://www.gencodegenes.org/pages/biotypes.html). Furthermore, we added either "promoter", "enhancer" or "NAST" as a replacement string for the non-Gencode references used in this "Feature" column. The following R script shows how we created Figure 2. We refrain from showing code lines that show how to load packages, after a package has been listed for loading into R for the first time. Code lines that show how to load packages already mentioned above have been omitted in the following code.

```
# loading required R packages
library(ggplot2)

sourceAnnot <- as.data.frame(fread("mm10_sourceAnnotation_
    20180129142414.bed", header=TRUE, stringsAsFactors = F))

sourceAnnot$Feature[grep("pseudogene", sourceAnnot$Feature)] <- "
    pseudogene"
sourceAnnot$Feature[grep("_gene", sourceAnnot$Feature)] <- "IG_TR
    _gene

sourceAnnot$Source <- factor(sourceAnnot$Source)
sourceAnnot$Feature <- factor(sourceAnnot$Feature)

freqFeature <- as.data.frame(table(sourceAnnot$Feature)); names(
    freqFeature) <- c("Feature", "Frequency")
freqFeature <- freqFeature[order(freqFeature$Frequency,
    decreasing = T),]
freqFeature$Feature <- factor(freqFeature$Feature, levels(
    freqFeature$Feature) <- as.character(freqFeature$Feature))

features <- ggplot(freqFeature, aes(x = Feature, y = Frequency))
    +
  geom_bar(stat = "identity") +
  geom_text(aes(label = Frequency),
            vjust = -.5) +
  geom_text(aes(label = sprintf("%.2f%%", Frequency/sum(Frequency
      ) * 100)),
            vjust = -2) +
  theme(axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle(paste0("Affiliated biotype features of all annotations
      \ntotal number of annotations: ", sum(freqFeature$Frequency
```

```
      ))) +
  ylab("Frequency") +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  scale_y_continuous(expand=c(0.1,0)) +
  theme(text = element_text(family="Helvetica"),
        axis.title = element_text(size=12,face = "bold"),
        axis.text = element_text(size=10),
        plot.title = element_text(size=16, face = "plain"))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("C1CAGE_features", time, ".pdf"), width = 12, height =
     10, onefile = T)
plot(features)
dev.off()

freqSource <- as.data.frame(table(sourceAnnot$Source)); names(
    freqSource) <- c("Source", "Frequency")
freqSource <- freqSource[order(freqSource$Frequency, decreasing =
     T),]
freqSource$Source <- factor(freqSource$Source, levels(freqSource$
    Source) <- as.character(freqSource$Source))

sources <- ggplot(freqSource, aes(x = Source, y = sort(freqSource
    $Frequency, decreasing = T))) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = sort(freqSource$Frequency, decreasing = T
      )),
            vjust = -.5) +
  geom_text(aes(label = sprintf("%.2f%%", sort(freqSource$
      Frequency, decreasing = T)/sum(Frequency) * 100)),
            vjust = -2) +
  theme(axis.text.x = element_text(angle = 40, hjust = 1)) +
  ggtitle(paste0("Overlap of annotations from all sources \ntotal
       number of annotations: ", sum(freqSource$Frequency)))+
  ylab("Frequency") +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  scale_y_continuous(expand=c(0.1,0)) +
  theme(text = element_text(family="Helvetica"),
        axis.title = element_text(size=12,face = "bold"),
        axis.text = element_text(size=10),
        plot.title = element_text(size=16, face = "plain"))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
```

```
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("C1CAGE_sources", time, ".pdf"), width = 12, height =
    10, onefile = T)
plot(sources)
dev.off()
```

The main reason for creating a comprehensive combined annotation BED file was to get expression estimates for as many loci as possible. Only genomic locations covered in the reference annotation file are used to create expression counts based on C1-CAGE tags. Any loci not included in the reference, but showing expression reads, will just be summed up and displayed as a single non-annotated feature. Since C1-CAGE is not 3' biased as Fluidigm scRNA-seq, we gain the ability to detect diverse ncRNA species, which could not be present in the Fluidigm protocol data set. The C1-CAGE gene expression matrix was generated with the CAGEr method CTSStoGenes using the R code below.

```
# loading required R packages
library("CAGEr")
library("magrittr")
# needed for CAGEexp objects
library("MultiAssayExperiment")
# needed for CAGEexp objects
library("SummarizedExperiment")
library("vegan")

# load data
ce <- readRDS("CAGEexp_mm10.rds")
# define groups
ce$group <- ce$sampleLabels %>% sub(pat="_.*", repl="")  %>%
    factor

# expression per gene
annot <- read.delim("mm10_sourceAnnotation_20180129142414.bed",
    stringsAsFactors = TRUE, header = TRUE)
levels(annot$Strand) <- c("-", "*", "+")
summary(annot)
gr <- GRanges( seqnames = annot$Chromosome
               , ranges = IRanges( start = annot$Start + 1
               , end = annot$End)
               , strand = annot$Strand)
mcols(gr) <- annot[, -(1:4)]

gr$transcript_type <- gr$Feature
```

```
gr$gene_name <- gr$Feature
annotateCTSS(ce, gr)
CTSScoordinatesGR(ce)$Feature <- CTSScoordinatesGR(ce)$genes

gr$gene_name <- gr$Source
annotateCTSS(ce, gr)
CTSScoordinatesGR(ce)$Source <- CTSScoordinatesGR(ce)$genes

gr$gene_name <- gr$ID
annotateCTSS(ce, gr)

CTSScoordinatesGR(ce)

# feature count
hAnnot <- data.frame(c("",  "EPD", "NAST", "F5enhancer", "
    F5promoter", "Gencode"), c("No Match",  "EPD", "NAST", "
    F5enhancer", "F5promoter", "Gencode"), stringsAsFactors = F)

flatAnnots <- smallCAGEqc::hannot(decode(CTSScoordinatesGR(ce)$
    Source), hAnnot)

SourceSummary <- rowsum(CTSStagCountDf(ce), flatAnnots) %>% t %>%
     DataFrame

customScope <- function(libs)
  list(libs = libs
      , columns = colnames(libs)
      , total = rowSums(libs))

smallCAGEqc::plotAnnot(SourceSummary %>% data.frame, "custom",
    GROUP = ce$group, customScope = customScope, TITLE = "")

# save TSS and gene level expression tables
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")

data.table::fwrite(CTSStagCountDf(ce)
                   , file=paste0("expressionTable_", time, ".tsv")
                   , quote = FALSE
                   , sep = "\t")

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")

CTSStoGenes(ce)
data.table::fwrite(as.data.frame(assay(GeneExpSE(ce)))
                   , sep = "\t"
                   , row.names = TRUE
                   , paste0("geneExpressionTable_", time, ".tsv"))
```

2.5. Expression data analysis

2.5.1. Differential gene expression analysis

In this final part of the method section we will provide the source code to perform all analyses used in the results section of this thesis and code to make figures based on our expression data. We won't partition this chapter to separate Fluidigm scRNA-seq and C1-CAGE, because the code below exemplifies the methodology for both protocols. The main difference is that the input expression matrix and metadata tables for both protocols are different. C1-CAGE expression is given as count values, whereas the scRNA-seq expression table is available as both TPM values and estimated count values. This difference is important for certain functions which can only be applied to count data. Firstly, we will show the R code used to perform single-cell Differential Gene Expression (DGE) analysis. Follow up analysis requires using a subset of the expression table. This subset is based on Differentially Expressed (DE) genes between the cell sample groups that belong to either mESC or EpiSC and thus represent the genes which are most likely specific for the naïve or primed state. We use the SCDE R package v2.10.1 to run this analysis (Kharchenko *et al.*, 2014; Fan *et al.*, 2016).

```
# SCDE differential gene expression on specified cell samples
# loading required R packages
library(scde)

# load input files
meta <- read.csv("metadata_20190223045845.csv", header=TRUE,
    stringsAsFactors = F)
counts <- read.table("geneCount_M8_20170908_Kallisto.txt", header
    =T, row.names = 1, check.names = FALSE)

counts <- counts[, which(names(counts) %in% meta$cell_id)]
# make sure meta data and expression table have same samples
meta <- meta[match(names(counts), meta$cell_id),]
# the following must be TRUE
identical(names(counts), meta$cell_id)

# remove failed cells
RNA <- c(2:6,8,1,9)
meta$timePoint <- factor(meta$timePoint)
```

```
meta$timePoint <- factor(meta$timePoint, levels(meta$timePoint)[
    RNA])
Red <- meta[which(meta$discard == FALSE),]
counts <- counts[,which(meta$discard == FALSE)]

# round to integers, in case if estimated count values
counts <- round(counts,0)
# remove non-expressed genes
counts <- counts[rowSums(counts)>0,]
nGenes <- length(counts[,1])

coverage <- colSums(counts)/nGenes

# filter out cells with low coverage
counts <- counts[,coverage>1]
Red <- Red[coverage>1,]

# specify number of CPU cores to be used
# this step requires massive amounts of RAM
n.cores <- 1

# data preparation
cd <- clean.counts(counts, min.lib.size=1000, min.reads = 1, min.
    detected = 1)

# make group factor for DGE
cdGroups <- Red[which(Red$cell_id %in% colnames(cd)),]
cd <- cd[,match(Red$cell_id, names(cd))]
# the following must be TRUE
identical(colnames(cd), cdGroups$cell_id)
groups <- cdGroups$timePoint
names(groups) <- colnames(cd)

# convert all values to stored integers
# this step is omitted when using C1-CAGE data as input
cd <- as.data.frame(apply(cd, 2, function(x) {storage.mode(x) <-
    'integer'; x}))

# fitting error models
# this step takes quite long and is demanding on RAM
scde.fitted.model <- scde.error.models(counts = cd, groups =
    groups, n.cores = n.cores, threshold.segmentation = TRUE,
    save.crossfit.plots = FALSE, save.model.plots = FALSE,
    verbose = 1)
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
# save error fitting model for rerunning DGE on different group
    comparisons
save(scde.fitted.model,file=paste0("scde_scRNA-seq_Kallisto_",
    time, ".RData"))

# to save time load the precalculated error models from the
```

```
    previous step
load("scde_scRNA-seq_Kallisto_20170908.RData")
valid.cells <- scde.fitted.model$corr.a > 0
table(valid.cells)

scde.fitted.model <- scde.fitted.model[valid.cells, ]
# make sure that cd and scde.fitted.model contain exactly the
    same cells
scde.fitted.model <- scde.fitted.model[match(names(cd), rownames(
    scde.fitted.model)),]
# the following must be TRUE
identical(names(cd),  rownames(scde.fitted.model ))

scde.prior <- scde.expression.prior(models=scde.fitted.model,
    counts = cd, length.out = 400, show.plot = FALSE)

# DGE variables for defined pairwise combinations
pair <- d0EpiSC <- "(day0_mES|EpiSC)"
# make pair groups and index for time points
index <- grep(pair, groups)
pairIndex <- unlist(stringr::str_split(gsub(" ", "", chartr("()",
     "   ", pair)), "\\|"))
groupsDGE <- factor(groups[index], levels = pairIndex)
#sanity check
identical(unlist(stringr::str_split(gsub(" ", "", chartr("()", "
      ", pair)), "\\|")), levels(groupsDGE))

# make pairwise group comparisons
# run differential expression tests on all genes with batch
    correction
# this can take several minutes
  batch <- as.factor(ifelse(rbinom(nrow(scde.fitted.model[index
      ,]), 1, 0.5) == 1, "batch1", "batch2"))
  ediff <- scde.expression.difference(scde.fitted.model[index,],
      cd[,index], scde.prior, groups = groupsDGE, batch = batch,
      n.randomizations  =  100, n.cores  =  n.cores, verbose  =
      1)
# show the top DGE genes
head(ediff[[1]][order(ediff[[1]][,5], decreasing = TRUE),], 15)

# 2-tailed p-value
p.values <- 2*pnorm(abs(ediff[[1]][,5]),lower.tail=F)
# adjust to control for FDR
p.values.adj <- 2*pnorm(abs(ediff[[1]][,6]),lower.tail=F)
significant.genes <- which(p.values.adj<0.05)
length(significant.genes)

# get fold changes
# order by p-value
ord <- order(p.values.adj[significant.genes])
de <- cbind(ediff[[1]][significant.genes,1:3],p.values.adj[
    significant.genes])[ord,]
```

```
colnames(de) <- c("Lower-bound","log2_FC","Upper-bound","p-
    adjusted")

# get time stamp for output file name
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
# make file name based on Cluster pair
name <- paste0(levels(pair)[1], "-", levels(pair)[2])
# save DGE batch effect corrected results in file
# this is the file used for DGE based analysis
write.table(de, file = paste0("scde_", name, "_results_batch-
    effect-corrected_", time, ".txt"), row.names = TRUE, col.
    names = TRUE, sep = "\t", quote = FALSE)

# make table for non batch effect corrected DGE results
# 2-tailed p-value
p.values <- 2*pnorm(abs(ediff[[2]][,5]),lower.tail=F)
# adjust to control for FDR
p.values.adj <- 2*pnorm(abs(ediff[[2]][,6]),lower.tail=F)
significant.genes <- which(p.values.adj<0.05)
length(significant.genes)

# get fold changes
# order by p-value
ord <- order(p.values.adj[significant.genes])
de <- cbind(ediff[[2]][significant.genes,1:3],p.values.adj[
    significant.genes])[ord,]
colnames(de) <- c("Lower-bound","log2_FC","Upper-bound","p-
    adjusted")
# save DGE results in file
write.table(de, file = paste0("scde_", name, "_results_", time, "
    .txt"), row.names = TRUE, col.names = TRUE, sep = "\t", quote
     = FALSE)
}
```

2.5.2. Pseudotime analysis

The resulting batch-effect corrected table of DE genes contains p-values and fold changes by which one can rank the genes. These DE genes are filtered using a p-value cut-off of 1% and an expression matrix containing the remaining genes was used as input to estimate pseudotime values for all cell samples. These pseudotime values are integer numbers reflecting the assumed temporal order of cells from 1 to the total number of cells. Pseudotime calculation was done using the TSCAN R package v1.20.0 (Z. Ji and H. Ji, 2016).

```
# TSCAN pseudotime sorting
# loading required R packages
library(TSCAN)

# load input files
meta <- read.csv("metadata_20190223045845.csv", header=TRUE,
    stringsAsFactors = F)
Tpm <- read.table("geneTPM_M8_20190207033818_Kallisto.txt",
    header=T, row.names = 1, check.names = FALSE)
# make sure meta data and expression table have same samples
meta <- meta[match(names(Tpm), meta$cell_id),]
# the following must be TRUE
identical(names(Tpm), meta$cell_id)

# DGE result between selected groups
dge <- read.table("scde_d0-EpiSC_results_batch-effect-corrected_
    20190223041832.txt", header=T, row.names = 1, check.names =
    FALSE)
dge <- dge[which(dge$`p-adjusted`<0.01),]
genes <- which(row.names(Tpm) %in% row.names(dge))
markers <- Tpm[genes, which(meta$discard == FALSE)]

# return expression table without cells that failed quality
    checks
expr <- Tpm[, which(meta$discard == FALSE)]

# reorder factor levels of timePoint column
RNA <- c(2:6,8,9,1)
meta$timePoint <- factor(meta$timePoint)
meta$timePoint <- factor(meta$timePoint, levels(meta$timePoint)[
    RNA])
meta <- meta[which(meta$discard == FALSE),]

# filter cells and DGE genes based on minimum expression
    thresholds
tpm <- markers
tpm <- tpm[,which(colSums(tpm)>1)]
tpm <- tpm[which(rowSums(tpm)>1),]
tpm <- tpm[which(rowSums(tpm>0)>10),]
# the following must be TRUE
identical(names(tpm), meta$cell_id)

# start pseudotime sorting
prepro <- preprocess(tpm, takelog = TRUE, logbase = 2,
    pseudocount = 1)
exclust <- exprmclust(prepro, modelNames = "VVV", reduce = T)
# MSTorder is manually adjusted based on a first run of
    plotmclust
plotmclust(exclust, x = 1, y = 2, MSTorder = c(2,1,3,4), show_
    tree = T, show_cell_names = F, cell_name_size = 2, markerexpr
     = NULL)
```

```
scanOrder <- TSCANorder(exclust, MSTorder = c(2,1,3,4), orderonly
     = F, flip = F, listbranch = F)
timePoint <- meta$timePoint[match(scanOrder$sample_name, meta$
    cell_id)]
scanOrder <- cbind(scanOrder, timePoint)
# save pseudotime sorting as file
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
write.table(scanOrder, file = paste0("TSCAN_pseudotime_", time, "
    .txt"), row.names = TRUE, col.names = TRUE, sep = "\t", quote
     = FALSE)
```

2.5.3. Sample assessment plots

Based on the complete expression table and the sample-reduced metadata table as used in the pseudotime analysis script above we created some informative figures that show the overall distribution of the number of expressed genes at each time point. Furthermore, we performed a Spearman correlation on the expression table and plotted the correlation coefficient sorted results (Petropoulos *et al.*, 2016)) (Figure 6B). The function below is used for all figures to get the colors for the time points and will only be shown here once.

```
# set color scheme for time points
gg_color_hue <- function(n) {
  hues = seq(15, 375, length=n+1)
  hcl(h=hues, l=65, c=100)[1:n]
}
col <- gg_color_hue(16)[c(1:3,5,10,12,14,16)]

# filter cells and all genes based on minimum expression
    thresholds
expr <- expr[,which(colSums(expr)>1)]
expr <- expr[which(rowSums(expr)>1),]
expr <- expr[which(rowSums(expr>0)>10),]
# the following must be TRUE
identical(names(expr), meta$cell_id)

# calculates number of expressed genes per cell
expressedGenes <- sapply(1:ncol(expr), function(x) length(which(
    expr[,x]>1)))
meta$nGenes <- expressedGenes
# removes cells that have less than 50 expressed genes
meta <- meta[which(meta$nGenes>50),]
```

```
exprGenes <- expr[,which(names(expr) %in% meta$cell_id)]
# the following must be TRUE
identical(meta$cell_id, colnames(exprGenes))

geneNumberBox <- ggplot(meta, aes(x=timePoint, y=nGenes, fill =
    timePoint)) +
  geom_boxplot() +
  xlab("") +
  ylab("Number of expressed Genes")  +
  theme(axis.text.x = element_text(angle = 45, hjust = 0.4, vjust
      = 0.5)) +
  scale_fill_manual(values=col,
                    name="",
                    labels=c("mES - 1772-116-259", "Day 1 -
                        1772-116-260", "Day 2 - 1772-116-261", "
                        Day 3 - 1772-116-263", "Day 4 -
                        1772-116-262, 1772-151-313", "EpiSC -
                        1772-116-268", "MB#3P10 - 1772-123-291",
                        "1EP7 - 1772-123-289")) +
  scale_colour_manual(values=col,
                      name="",
                      labels=c("mES - 1772-116-259", "Day 1 -
                          1772-116-260", "Day 2 - 1772-116-261",
                          "Day 3 - 1772-116-263", "Day 4 -
                          1772-116-262, 1772-151-313", "EpiSC -
                          1772-116-268", "MB#3P10 - 1772-123-291"
                          , "1EP7 - 1772-123-289")) +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  theme(text = element_text(family="Helvetica"))

# save plot as PDF
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("exressedGenesPerTimePoint_", time, ".pdf"), width =
    12, height = 7, onefile = TRUE)
plot(geneNumberBox)
dev.off()

# nearest cell correlation plot
nearCell <- as.data.frame(cor(expr, method = "spearman"))
meta$nearestCellCorr <- sapply(1:ncol(nearCell), function(x) sort
    (nearCell[,x], decreasing = T)[2])

nnBox <- ggplot(meta, aes(x=timePoint, y=nearestCellCorr, fill =
    timePoint)) +
  geom_boxplot() +
  ylim(0,1) +
```

```
  xlab("") +
  ylab("Spearman correlation to nearest cell") +
  theme(axis.text.x = element_text(angle = 45, hjust = 0.4, vjust
       = 0.5)) +
  scale_fill_manual(values=col,
                    name="",
                    labels=c("mES", "Day 1", "Day 2", "Day 3", "
                         Day 4", "EpiSC", "MB#3P10", "1EP7")) +
  scale_colour_manual(values=col,
                      name="",
                      labels=c("mES", "Day 1", "Day 2", "Day 3",
                           "Day 4", "EpiSC", "MB#3P10", "1EP7")) +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
       element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
       major=element_line(colour="grey")) +
  theme(text = element_text(family="Helvetica"))

# save plot as PDF
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
     character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("nearestneighbourCorrelation_", time, ".pdf"), width =
      12, height = 7, onefile = TRUE)
plot(nnBox)
dev.off()
```

2.5.4. Principal Component Analysis

We used the base R prcomp function to calculate principal components for visualization of our high-dimensional expression data. In order to achieve a distinct separation of cell groups in the PCA, we took the same DE gene subset as input as in the pseudotime analysis (Figure 8A).

```
# load input files
meta <- read.csv("metadata_20190223045845.csv", header=TRUE,
     stringsAsFactors = F)
Tpm <- read.table("geneTPM_M8_20190207033818_Kallisto.txt",
     header=T, row.names = 1, check.names = FALSE)
# make sure meta data and expression table have same samples
meta <- meta[match(names(Tpm), meta$cell_id),]
# the following must be TRUE
identical(names(Tpm), meta$cell_id)

# DGE result between selected groups
```

```
dge <- read.table("scde_d0-EpiSC_results_batch-effect-corrected_
    20190223041832.txt", header=T, row.names = 1, check.names =
    FALSE)
dge <- dge[which(dge$`p-adjusted`<0.01),]
genes <- which(row.names(Tpm) %in% row.names(dge))
markers <- Tpm[genes, which(meta$discard == FALSE)]

# return expression table without cells that failed quality
    checks
expr <- Tpm[, which(meta$discard == FALSE)]

# reorder factor levels of timePoint column
RNA <- c(2:6,8,9,1)
meta$timePoint <- factor(meta$timePoint)
meta$timePoint <- factor(meta$timePoint, levels(meta$timePoint)[
    RNA])
meta <- meta[which(meta$discard == FALSE),]

# filter cells and DGE genes based on minimum expression
    thresholds
tpm <- markers
tpm <- tpm[,which(colSums(tpm)>1)]
tpm <- tpm[which(rowSums(tpm)>1),]
tpm <- tpm[which(rowSums(tpm>0)>10),]
# the following must be TRUE
identical(names(tpm), meta$cell_id)

# convert expression table to log matrix
matExpr <- as.matrix(tpm)
logtpm <- log10(matExpr+1)
# run PCA
base.pca <- prcomp(t(logtpm))
pca <- data.frame(base.pca$x[,1], base.pca$x[,2]); colnames(pca)
    <- c("PC1", "PC2")

# calculate component variance
eigen <- (base.pca$sdev)^2
variances <- eigen*100/sum(eigen)
variances[1:2]

# the following must be TRUE
identical(rownames(pca), meta$cell_id)
pca$timePoint <- meta$timePoint

# plot PC1 and PC2
pcaPlot <- qplot(PC1, PC2, data=pca, colour=timePoint,
                 xlab = paste('PCA1 ', round(variances[1], digits
                     = 2), '%'), ylab = paste(paste('PCA2 ',
                    round(variances[2], digits = 2), '%'))) +
  scale_colour_manual(values=col,
                      name="",
                      labels=c("mES - 1772-116-259", "Day 1 -
```

```
                          1772-116-260", "Day 2 - 1772-116-261",
                          "Day 3 - 1772-116-263", "Day 4 -
                          1772-116-262, 1772-151-313", "EpiSC -
                          1772-116-268", "MB#3P10 - 1772-123-291"
                          , "1EP7 - 1772-123-289")) +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  theme(text = element_text(family="Helvetica")) +
  scale_x_continuous(limits = c(min(pca$PC1) + min(pca$PC1)*0.1,
      max(pca$PC1) + max(pca$PC1)*0.1)) +
  scale_y_continuous(limits = c(min(pca$PC2) + min(pca$PC2)*0.1,
      max(pca$PC2) + max(pca$PC2)*0.1))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("PCA_Kallisto_", time, ".pdf"), width = 12, height =
    7, onefile = TRUE)
plot(pcaPlot)
dev.off()

pca$Run <- meta$Run
# save PCA data as file
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
write.table(pca, file = paste0("pcaData_kallisto_geneTPM_", time,
     ".txt"), row.names = TRUE, col.names = TRUE, sep = "\t",
    quote = FALSE)
```

2.5.5. t-SNE cluster analysis

In this section we first ran the Rtsne function to obtain t-SNE dimensions and secondly applied kmeans clustering to define groups within in the 2-dimensional t-SNE plot. The t-SNE and kmeans cluster data was saved as a file to be able to use it for other analyses and visualizations. We manually reassigned kmeans cluster numbers after the first kmeans run to have the numbers reflect the assumed temporal order of clusters based on the pseudotime order of cells (Figure 8C).

```
# loading required R packages
library(stringr)

logtpmT <- t(logtpm)
# run tSNE
tsne <- Rtsne(logtpmT)
Run <- as.character(sapply(rownames(logtpmT), function(x) str_
    split(x, "_", n=2)[[1]][[1]]))
Run <- factor(Run)
Run <- factor(Run, levels(Run)[c(1:3,5,4,9,6,8,7)])
sne <- data.frame(tsne$Y[,1], tsne$Y[,2], Run); colnames(sne) <-
    c("Dim1", "Dim2", "Run")

# the following must be TRUE
identical(colnames(logtpm), meta$cell_id)
sne$timePoint <- meta$timePoint

tsneRunPlot <- qplot(Dim1, Dim2, data=sne,
                     colour=timePoint) +
  scale_colour_manual(values=col,
                      name="",
                      labels=c("mES", "Day 1", "Day 2", "Day 3",
                          "Day 4", "EpiSC", "MB#3P10", "1EP7")) +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  theme(text = element_text(family="Helvetica")) +
  scale_x_continuous(limits = c(min(sne$Dim1) + min(sne$Dim1)*
      0.1, max(sne$Dim1) + max(sne$Dim1)*0.1)) +
  scale_y_continuous(limits = c(min(sne$Dim2) + min(sne$Dim2)*
      0.3, max(sne$Dim2) + max(sne$Dim2)*0.2))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("tSNE_Kallisto_", time, ".pdf"), width = 12, height =
    7, onefile = TRUE)
plot(tsneRunPlot)
dev.off()

# kmeans clustering
sne$Cluster <- factor(kmeans(sne[,1:2], centers = 5)[[1]])

tsnekmeansPlot <- qplot(Dim1, Dim2, data=sne, colour=Cluster) +
  scale_colour_brewer(palette = "Dark2", name="") +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
```

```
  theme(text = element_text(family="Helvetica")) +
  scale_x_continuous(limits = c(min(sne$Dim1) + min(sne$Dim1)*
      0.1, max(sne$Dim1) + max(sne$Dim1)*0.1)) +
  scale_y_continuous(limits = c(min(sne$Dim2) + min(sne$Dim2)*
      0.3, max(sne$Dim2) + max(sne$Dim2)*0.2))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("kmeans5_Kallisto_", time, ".pdf"), width = 12, height
     = 7, onefile = TRUE)
plot(tsnekmeansPlot)
dev.off()

# save tSNE and kmeans data as file
sne$cell_id <- meta$cell_id
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
write.table(sne, file = paste0("tSNE_kmeans5_clusterData_kallisto
    _DGE_geneTPM_", time, ".txt"), row.names = TRUE, col.names =
    TRUE, sep = "\t", quote = FALSE)

# start for cluster composition
sne <- fread("kmeans5_clusterData_kallisto_DGE_geneTPM_
    20190223044904.txt")
sne <- as.data.frame(sne)
sne$Cluster <- factor(sne$Cluster)
sne$Cluster

# save cluster composition as file
row.names(sne) <- sne$cell_id
x1 <- row.names(subset(sne, Cluster==1))
x2 <- row.names(subset(sne, Cluster==2))
x3 <- row.names(subset(sne, Cluster==3))
x4 <- row.names(subset(sne, Cluster==4))
x5 <- row.names(subset(sne, Cluster==5))

cl1 <- as.data.frame(table(as.character(meta[match(x1, meta$cell_
    id),129]))); names(cl1) <- c("timePoint", "Cluster1")
cl2 <- as.data.frame(table(as.character(meta[match(x2, meta$cell_
    id),129]))); names(cl2) <- c("timePoint", "Cluster2")
cl3 <- as.data.frame(table(as.character(meta[match(x3, meta$cell_
    id),129]))); names(cl3) <- c("timePoint", "Cluster3")
cl4 <- as.data.frame(table(as.character(meta[match(x4, meta$cell_
    id),129]))); names(cl4) <- c("timePoint", "Cluster4")
cl5 <- as.data.frame(table(as.character(meta[match(x5, meta$cell_
    id),129]))); names(cl5) <- c("timePoint", "Cluster5")

allClusters <- merge(cl1, cl2, by = "timePoint", all = T)
allClusters <- merge(allClusters, cl3, by = "timePoint", all = T)
allClusters <- merge(allClusters, cl4, by = "timePoint", all = T)
allClusters <- merge(allClusters, cl5, by = "timePoint", all = T)
allClusters[is.na(allClusters)] <- 0
```

```
percentTimes <- data.frame(allClusters$timePoint,
allClusters$Cluster1/as.numeric(colSums(allClusters[-1]))[1]*100,
     allClusters$Cluster2/as.numeric(colSums(allClusters[-1]))[2]
    *100,
allClusters$Cluster3/as.numeric(colSums(allClusters[-1]))[3]*100,
allClusters$Cluster4/as.numeric(colSums(allClusters[-1]))[4]*100,
allClusters$Cluster5/as.numeric(colSums(allClusters[-1]))[5]*100)
names(percentTimes) <- names(allClusters)

# save tSNE and cluster data for reproducibility
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
write.table(sne, file = paste0("tSNE_kmeans5_clusterData_kallisto
    _DGE_geneTPM_", time, ".txt"), row.names = F, col.names =
    TRUE, sep = "\t", quote = FALSE)
write.table(allClusters, file = paste0("tSNE_kmeans5_
    clusterCounts_Kallisto_DGE_geneTPM_", time, ".txt"), row.
    names = FALSE, col.names = TRUE, sep = "\t", quote = FALSE)
write.table(percentTimes, file = paste0("tSNE_kmeans5_
    clusterPercentage_Kallisto_DGE_geneTPM_", time, ".txt"), row.
    names = FALSE, col.names = TRUE, sep = "\t", quote = FALSE)
```

We overlaid the t-SNE kmeans cluster plots with the pseudotime analysis results by using a continuous color coding to represent pseudotime values (Figure 9A). The same approach was also made with the PCA plot (Figure 12B), but here we only show the R code for the t-SNE cluster plot. Additionally, we created a plot that shows how the pseudotime order and real sample time point data correspond with each other (Figure 9B).

```
# loading required R packages
library(gplots)
sne$Pseudotime <- scanOrder[match(rownames(sne), rownames(
    scanOrder)), 3]

tsneDensityPlot <- ggplot(sne, aes(Dim1, Dim2, colour =
    Pseudotime)) +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) +
  theme(axis.line=element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  stat_density2d(aes(fill = ..level.., alpha = ..level..),
      binwidth = 0.0001, geom = "polygon", position = "jitter") +
  scale_fill_continuous(low="white", high="black", name = "") +
  guides(alpha = "none") +
  geom_point() +
```

```
  scale_color_gradientn(colours = plasma(6), guide = "colourbar",
       name= "") +
  scale_x_continuous(limits = c(min(sne$Dim1) + min(sne$Dim1)*
      0.1, max(sne$Dim1) + max(sne$Dim1)*0.1)) +
  scale_y_continuous(limits = c(min(sne$Dim2) + min(sne$Dim2)*
      0.3, max(sne$Dim2) + max(sne$Dim2)*0.2)) +
  theme(text = element_text(family="Helvetica"))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("tSNE_pseudotimeDensity_", time, ".pdf"), width = 10,
    height = 7, onefile = TRUE)
plot(tsneDensityPlot)
dev.off()

pseudotimeBox <- ggplot(meta, aes(x = timePoint, y = Pseudotime,
    fill = timePoint)) +
  geom_boxplot() +
  coord_flip() +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  theme(axis.title = element_text(size=14, family="Helvetica"),
       axis.text = element_text(size=8, family="Helvetica")) +
  scale_x_discrete(limits=rev(levels(meta$timePoint)), name="") +
  guides(fill=FALSE)

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("pseudotime_boxplot_", time, ".pdf"), width = 10,
    height = 7, onefile = TRUE)
plot(pseudotimeBox)
dev.off()
```

2.5.6. Cell cycle assignment

We obtained the orthologous mouse genes of a published human cell cycle marker gene set (Whitfield *et al.*, 2002) and applied a correlation-based method that uses these cell cycle marker genes to assign the most likely cell cycle phase a cell was in during harvest. The method was first published by Macosko *et al.* (2015) and applied by Tung *et al.* (2017) as well. We modified the R script to fit our data as shown below (Figure 11).

```
# cell cycle assignment
# loading required R packages
library(dplyr)
library(edgeR)

# load input files
meta <- read.csv("metadata_20190223045845.csv", header=TRUE,
    stringsAsFactors = F)
tpm <- read.table("geneCount_M8_20170908_Kallisto.txt", header=T,
     row.names = 1, check.names = FALSE)
tpm <- tpm[, which(names(tpm) %in% meta$cell_id)]
meta <- meta[match(names(tpm), meta$cell_id),]
expr <- tpm[,!meta$discard]
meta <- meta[!meta$discard,]
# the following must be TRUE
identical(meta$cell_id, names(expr))

# load orthologous mouse cell cycle genes
cycleGenes <- read.table("mES2EpiSC_cellcyclegenes.txt", header=
    TRUE, sep = "\t", na.strings = c("", "NA"))
expressed <- rowSums(expr) > 0
expr <- expr[expressed,]

# makes list of 5 tables containing all the genes for each
    respective cell cycle phase
cycleGenes_list <- lapply(1:5,function(x){
  temp <- as.character(cycleGenes[,x])
  temp[temp!=""]
})

phaseScore <- sapply(cycleGenes_list, function(x){
  # create table of each phase
  phase <- expr[rownames(expr) %in% unlist(x),]
  # add average expression of all genes in the phase
  combined_matrix <- rbind(phase, average = apply(phase, 2, mean)
      )
  # use transpose to compute cor matrix
  cor_matrix <- cor(t(combined_matrix))
  # take the numbers
  cor_vector <- cor_matrix[, dim(cor_matrix)[1]]
  # restrict to correlation >= 0.3
  phase_restricted <- phase[rownames(phase) %in% names(cor_vector
      [cor_vector >= 0.3]),]
  # apply normalization to reads
  norm_factors <- calcNormFactors(phase_restricted, method = "TMM
      ")
  phase_cpm <- cpm(phase_restricted, log = TRUE,
                   lib.size = colSums(expr) * norm_factors)
  # output the phase specific scores (mean of normalized
      expression levels in the phase)
  apply(phase_cpm, 2, mean)
```

```
})

# normalization function
flexible_normalization <- function(data_in,by_row=TRUE){
  if(by_row){
    row_mean <- apply(data_in,1,mean)
    row_sd   <- apply(data_in,1,sd)
    output <- data_in
    for(i in 1:dim(data_in)[1]){
      output[i,] <- (data_in[i,] - row_mean[i])/row_sd[i]
    }
  }
  # if by column
  if(!by_row){
    col_mean <- apply(data_in,2,mean)
    col_sd   <- apply(data_in,2,sd)
    output <- data_in
    for(i in 1:dim(data_in)[2]){
      output[,i] <- (data_in[,i] - col_mean[i])/col_sd[i]
    }
  }
  output
}

phaseScore_normed <- flexible_normalization(phaseScore, by_row=
    FALSE)
phaseScore_normed2 <- flexible_normalization(phaseScore_normed,
    by_row=TRUE)

# assign phase to cell
cellPhase <- apply(phaseScore_normed2, 1, function(x) colnames(
    cycleGenes)[which.max(x)])
assign_cellPhase <- data.frame(cellPhase); names(assign_cellPhase
    ) <- "phase"
assign_cellPhase$phase <- factor(assign_cellPhase$phase, levels(
    assign_cellPhase$phase)[c(1,5,2,3,4)])

# save table with cell_id and assigned cell cycle phase for each
    cell_id
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
write.csv(assign_cellPhase, file = paste0("cellCycle_metadata_",
    time, ".csv"), row.names = T)

# different plots for cell cycle assignment results
cellcycle <- read.csv("cellCycle_metadata_20190314160652.csv",
    header=TRUE, stringsAsFactors = F)
# start for cluster composition
sne <- fread("tSNE_kmeans5_clusterData_kallisto_DGE_geneTPM_
    20190223045041.txt")
sne <- as.data.frame(sne)
sne$Cluster <- factor(sne$Cluster)
```

```
sne$Cluster

sne <- sne[match(cellcycle$X, sne$cell_id),]
# the following must be TRUE
identical(sne$cell_id, cellcycle$X)

sne$phase <- factor(cellcycle$phase)
sne$phase <- factor(sne$phase, levels(sne$phase)[c(1,5,2,3,4)])

tsneCellCyclePlot <- qplot(Dim1, Dim2, data=sne, colour=phase) +
  scale_colour_brewer(palette = "Set1", name="") +
  theme(axis.ticks = element_blank()) +
  theme(panel.background = element_blank()) + theme(axis.line=
      element_line(color="grey")) +
  theme(panel.grid.minor=element_line(color="grey"), panel.grid.
      major=element_line(colour="grey")) +
  theme(text = element_text(family="Helvetica")) +
  scale_x_continuous(limits = c(min(sne$Dim1) + min(sne$Dim1)*
      0.1, max(sne$Dim1) + max(sne$Dim1)*0.1)) +
  scale_y_continuous(limits = c(min(sne$Dim2) + min(sne$Dim2)*
      0.3, max(sne$Dim2) + max(sne$Dim2)*0.2))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("S3_tSNE_CellCycle_", time, ".pdf"), width = 12,
    height = 7, onefile = TRUE)
plot(tsneCellCyclePlot)
dev.off()

# save cluster composition of each cell cycle phase as file
x1 <- subset(sne, Cluster==1)[,7]
x2 <- subset(sne, Cluster==2)[,7]
x3 <- subset(sne, Cluster==3)[,7]
x4 <- subset(sne, Cluster==4)[,7]
x5 <- subset(sne, Cluster==5)[,7]

cl1 <- as.data.frame(table(as.character(sne[match(x1, sne$cell_id
    ),8]))); names(cl1) <- c("phase", "Cluster1")
cl2 <- as.data.frame(table(as.character(sne[match(x2, sne$cell_id
    ),8]))); names(cl2) <- c("phase", "Cluster2")
cl3 <- as.data.frame(table(as.character(sne[match(x3, sne$cell_id
    ),8]))); names(cl3) <- c("phase", "Cluster3")
cl4 <- as.data.frame(table(as.character(sne[match(x4, sne$cell_id
    ),8]))); names(cl4) <- c("phase", "Cluster4")
cl5 <- as.data.frame(table(as.character(sne[match(x5, sne$cell_id
    ),8]))); names(cl5) <- c("phase", "Cluster5")

allClusters <- merge(cl1, cl2, by = "phase", all = T)
allClusters <- merge(allClusters, cl3, by = "phase", all = T)
allClusters <- merge(allClusters, cl4, by = "phase", all = T)
allClusters <- merge(allClusters, cl5, by = "phase", all = T)
allClusters[is.na(allClusters)] <- 0
```

```
allClustersCycle <- allClusters

percentCycles <- data.frame(allClusters$phase, allClusters$
    Cluster1/as.numeric(colSums(allClusters[-1]))[1]*100,
allClusters$Cluster2/as.numeric(colSums(allClusters[-1]))[2]*100,
allClusters$Cluster3/as.numeric(colSums(allClusters[-1]))[3]*100,
allClusters$Cluster4/as.numeric(colSums(allClusters[-1]))[4]*100,
allClusters$Cluster5/as.numeric(colSums(allClusters[-1]))[5]*100)
names(percentCycles) <- names(allClusters)

# save tSNE cellcycle data for reproducibility
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
write.table(allClustersCycle, file = paste0("tSNE_kmeans5_
    cellcycleCounts_Kallisto_DGE_geneTPM_", time, ".txt"), row.
    names = FALSE, col.names = TRUE, sep = "\t", quote = FALSE)
write.table(percentCycles, file = paste0("tSNE_kmeans5_
    cellcyclePercentage_Kallisto_DGE_geneTPM_", time, ".txt"),
    row.names = FALSE, col.names = TRUE, sep = "\t", quote =
    FALSE)

# cell cycle phase pie charts
blank_theme <- theme_minimal()+
  theme(
    axis.title.x = element_blank(),
    axis.title.y = element_blank(),
    panel.border = element_blank(),
    panel.grid=element_blank(),
    axis.ticks = element_blank(),
    plot.title=element_text(size=14, face="bold")
  )

percentCyclesLong <- melt(percentCycles); names(percentCyclesLong
    ) <- c("phase", "Cluster", "Value")
percentCyclesLong$position <- cumsum(percentCyclesLong$Value) -
    percentCyclesLong$Value/2
percentCyclesLong$labels <- paste0(round(percentCyclesLong$Value,
     digits = 1), " %")
percentCyclesLong$phase <- factor(percentCyclesLong$phase, levels
    (percentCyclesLong$phase)[c(1,5,2,3,4)])

col <- brewer.pal(5, "Set1")
library(scales)
clusterPie <- ggplot(percentCyclesLong, aes(x = "", y = Value,
    fill = phase)) +
  geom_bar(aes(x = "", y = Value, fill = phase), width = 1, color
       = "black", stat = "identity") +
  #geom_text(data = percentCyclesLong, aes(x = "", y = position,
      label = labels), size=2) +
  coord_polar(theta = "y") +
  facet_grid(facets = .~Cluster, labeller = label_value) +
  scale_fill_manual(values = col) +
```

```
  blank_theme +
  theme(axis.text.x=element_blank())+
  theme(text = element_text(family="Helvetica"))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pdf(paste0("S3_phase_pieCharts", time, ".pdf"), width = 12,
    height = 7, onefile = TRUE)
plot(clusterPie)
dev.off()

# save cell cycle phase composition of each cluster as file
allClustersT <- as.data.frame(t(allClusters))
names(allClustersT) <- as.character(as.matrix(allClustersT[1,]))
allClustersT <- allClustersT[-1,]
allClustersT <- as.data.frame(apply(allClustersT, 2, function(x)
    as.numeric(as.character(x))))
allClustersT$Cluster <- names(allClusters)[-1]
percentT <- as.data.frame(sapply(1:5, function(x) allClustersT[,x
    ]/as.numeric(colSums(allClustersT[,-6]))[x]*100))
names(percentT) <- names(allClustersT[-6])
percentT$Cluster <- allClustersT$Cluster

percentTLong <- melt(percentT); names(percentTLong) <- c("Cluster
    ", "phase", "Value")
percentTLong$position <- cumsum(percentTLong$Value) -
    percentCyclesLong$Value/2
percentTLong$labels <- paste0(round(percentTLong$Value, digits =
    1), " %")
percentTLong$phase <- factor(percentTLong$phase, levels(
    percentTLong$phase)[c(1,5,2,3,4)])

write.table(percentT, file = paste0("tSNE_percentClusterPerCycle_
    Kallisto_DGE_geneTPM_", time, ".txt"), row.names = FALSE, col
    .names = TRUE, sep = "\t", quote = FALSE)

col <- brewer.pal(5, "Dark2")
phasePie <- ggplot(percentTLong, aes(x = "", y = Value, fill =
    Cluster)) +
  geom_bar(aes(x = "", y = Value, fill = Cluster), width = 1,
      color = "black", stat = "identity") +
  #geom_text(data = percentTLong, aes(x = "", y = position, label
      = labels), size=2) +
  coord_polar(theta = "y") +
  scale_fill_manual(values = col) +
  facet_grid(facets = .~phase, labeller = label_value) +
  blank_theme +
  theme(axis.text.x=element_blank()) +
  theme(text = element_text(family="Helvetica"))

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
```

```
pdf(paste0("S3_cluster_pieCharts", time, ".pdf"), width = 12,
    height = 7, onefile = TRUE)
plot(phasePie)
dev.off()
```

2.5.7. Screening and visualization of cluster specific genes

Differential Gene Expression analysis on paired t-SNE kmeans clusters delivered large lists of potentially cluster specific genes. In order to identify specific and novel marker genes for these clusters, one easy approach was to visualize the cellular expression of a gene as an overlay in the t-SNE plot. This approach still required manual screening of hundreds of plots, but in order to reduce the number of candidate genes for plotting we filtered the data based on expression fold changes and p-values. The cut-off values of the filters (Figure 10B-D, Figure 14, Figure 15, Figure 17B-C, Figure 20) can easily be adjusted. The pipeline below shows how we generated candidate marker gene plots.

```
# loading required R packages
library(viridisLite)

# screening for cluster specific genes
# load kmeans cluster DGE results
dge <- read.table("scde_Cluster1-Cluster2_results_batch-effect-
    corrected_20190223052340.txt", header=T, row.names = 1, check
    .names = FALSE)
mark1 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster1-Cluster3_results_batch-effect-
    corrected_20190223054004.txt", header=T, row.names = 1, check
    .names = FALSE)
mark2 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster1-Cluster4_results_batch-effect-
    corrected_20190223060011.txt", header=T, row.names = 1, check
    .names = FALSE)
mark3 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster1-Cluster5_results_batch-effect-
    corrected_20190223061739.txt", header=T, row.names = 1, check
    .names = FALSE)
mark4 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster2-Cluster3_results_batch-effect-
    corrected_20190223063415.txt", header=T, row.names = 1, check
    .names = FALSE)
```

```
mark6 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster2-Cluster4_results_batch-effect-
    corrected_20190223065248.txt", header=T, row.names = 1, check
    .names = FALSE)
mark7 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster2-Cluster5_results_batch-effect-
    corrected_20190223070917.txt", header=T, row.names = 1, check
    .names = FALSE)
mark8 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster3-Cluster4_results_batch-effect-
    corrected_20190223134622.txt", header=T, row.names = 1, check
    .names = FALSE)
mark10 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster3-Cluster5_results_batch-effect-
    corrected_20190223135804.txt", header=T, row.names = 1, check
    .names = FALSE)
mark11 <- dge[which(dge$`p-adjusted`<0.01),]
dge <- read.table("scde_Cluster4-Cluster5_results_batch-effect-
    corrected_20190223141759.txt", header=T, row.names = 1, check
    .names = FALSE)
mark13 <- dge[which(dge$`p-adjusted`<0.01),]

# combine all DE cluster genes in list
result.logFC1 <- unlist(list(mark1$log2_FC, mark2$log2_FC, mark6$
    log2_FC, mark7$log2_FC, mark10$log2_FC, mark11$log2_FC,
    mark13$log2_FC))
markGenes1 <- unlist(list(row.names(mark1), row.names(mark2), row
    .names(mark6), row.names(mark7), row.names(mark10), row.names
    (mark11), row.names(mark13)))

result.logFC2 <- unlist(list(mark3$log2_FC, mark4$log2_FC, mark8$
    log2_FC))
markGenes2 <- unlist(list(row.names(mark3), row.names(mark4), row
    .names(mark8)))

# find interesting genes by applying arbitrary cut-off filter
marksFC1 <-markGenes1[which(result.logFC1 > 2)]
marksFC1 <- c(marksFC1, markGenes1[which(result.logFC1 < -2)])
# apply stricter selection for cluster comparisons with very
    large number of DE genes
marksFC2 <-markGenes2[which(result.logFC2 > 5)]
marksFC2 <- c(marksFC2, markGenes2[which(result.logFC2 < -5)])

# combine all genes from previous step into one variable
marker.set <- unlist(list(marksFC1, marksFC2))
marker.set <- unique(marker.set)
marker.set <- marker.set[!is.na(marker.set)]

# DE genes screening plots
exprSet <- expr

plotMarkers<- function(counter) {
```

```
  gene <- as.numeric(exprSet[grep(paste(marker.set[counter], "$",
       sep = ""), row.names(exprSet)),])
  sne$marker <- gene

  qplot(Dim1, Dim2, data=sne,
        colour=marker) +
    ggtitle(paste(marker.set[counter])) +
    theme(axis.ticks = element_blank()) +
    theme(panel.background = element_blank()) + theme(axis.line=
         element_line(color="grey")) +
    theme(panel.grid.minor=element_line(color="grey"), panel.grid
         .major=element_line(colour="grey")) +
    geom_point(alpha = 0.4) +
    scale_colour_gradientn(colours = rev(magma(6)), guide = "
         colourbar")
}

time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
     character(Sys.time()), " ", n = 2))), collapse = "")
counter <- 1:length(marker.set)
pdf(paste0("tSNEheatscaleMarkersKallisto_", time, ".pdf"), width
     = 10, height = 7, onefile = TRUE)
sapply(counter, function(x) plot(plotMarkers(x)))
dev.off()
```

2.5.8. Hierarchical clustering heatmaps

Heatmaps enable to visualize high-dimensional data in a comprehensible way. Here we used the R pheatmap package v1.0.12 to create overviews of defined subsets of our data. In the code below we show exemplary workflows for the DE genes from the DGE analysis between time point mESC and EpiSC. It is also shown how to make a heatmap with only X chromosome genes. In all heatmaps we used the default row clustering setting, which means complete linkage and Euclidian distances. We did not apply column clustering but used t-SNE kmeans clusters and pseudotime as variables for sorting column samples in heatmaps. Due to the large number of genes in the underlying expression matrices the resulting dendrogram would become unreadable. Therefore, we decided to apply a kmeans clustering with 20 cluster centers on the rows/genes and applied hierarchical clustering on these 20 kmeans row clusters. These basically comprise genes that have similar expression patterns

into groups. For each heatmap a table containing all genes that make up a kmeans cluster was saved in long and wide format, as well as a table file that contains the underlying matrix that makes up each heatmap. (Figure 13A, Figure 18, Figure 21).

```
# loading required R packages
library(pheatmap)
library(RColorBrewer)

# load input files
sne <- as.data.frame(fread("tSNE_kmeans5_clusterData_kallisto_DGE
    _geneTPM_20190223045041.txt"))
meta <- as.data.frame(read.csv("metadata_20190223045845.csv",
    header=TRUE, stringsAsFactors = F))
tpm <- read.table("geneTPM_M8_20190207033818_Kallisto.txt",
    header=T, row.names = 1, check.names = FALSE)
tpm <- tpm[, which(names(tpm) %in% meta$cell_id)]
meta <- meta[match(names(tpm), meta$cell_id),]
expr <- tpm[,!meta$discard]

# make timePoint column a factor
RNA <- c(2:6,8,9,1)
meta$timePoint <- factor(meta$timePoint)
meta$timePoint <- factor(meta$timePoint, levels(meta$timePoint)[
    RNA])
meta <- meta[which(meta$discard == FALSE),]
# the following must be TRUE
identical(meta$cell_id, names(expr))

# get colours used for t-SNE kmeans clustering
col2 <- brewer.pal(5, "Dark2")

dge <- read.table("scde_d0-EpiSC_results_batch-effect-corrected_
    20190223041832.txt", header=T, row.names = 1, check.names =
    FALSE)
dge <- dge[which(dge$`p-adjusted`<0.01),]
genes <- which(row.names(expr) %in% row.names(dge))
markers <- expr[genes,]

# filter genes for heatmaps
markers <- markers[,which(colSums(markers)>1)]
markers <- markers[which(rowSums(markers)>1),]
markers <- markers[which(rowSums(markers>0)>10),]

# the following must be TRUE
identical(meta$cell_id, sne$cell_id)
sne$Pseudotime <- meta$Pseudotime
meta <- meta[match(names(markers), meta$cell_id),]
sne <- sne[order(sne$Cluster, sne$Pseudotime, decreasing = F),]
markers <- markers[,match(sne$cell_id, names(markers))]
```

```
# the following must be TRUE
identical(names(markers), sne$cell_id)

# modifies to include 2 colour top bars
heat.vals <- log2(markers+1) - rowMeans(log2(markers+1))
timePoint <- meta$timePoint[match(names(heat.vals), meta$cell_id)
    ]
levels(timePoint) <- c("mES", "Day 1", "Day 2", "Day 3", "Day 4",
     "EpiSC", "MB#3P10", "1EP7")
cluster <- sne$Cluster[match(names(heat.vals), sne$cell_id)]
annotation <- data.frame(timePoint = factor(timePoint, labels =
    levels(timePoint)), cluster = factor(cluster))
rownames(annotation) <- names(heat.vals)
names(col) <- levels(annotation$timePoint)
names(col2) <- levels(annotation$cluster)
ann_color <- list(timePoint = col, cluster = col2)

# heatmap mES vs EpiSC DE genes of pseudotime ordered cells
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pseudoHeat <- pheatmap(heat.vals, cluster_rows = T, cluster_cols
    = F, annotation = annotation, annotation_names_col = F,
    kmeans_k = 20, treeheight_row = 70, annotation_colors = ann_
    color, show_rownames = T, show_colnames = F, main = "mES vs
    EpiSC DE genes of tSNE cluster and pseudotime ordered cells",
     width = 14, height = 10, filename = paste0("d0-EpiSC_heatmap
    _14x10_", time, ".pdf"))

# get genes for kmeans gene clusters
clustGenes <- data.frame(as.integer(pseudoHeat$kmeans$cluster),
    names(pseudoHeat$kmeans$cluster))
names(clustGenes) <- c("cluster", "allGenes")
geneList <- data.frame(aggregate(clustGenes, by = list(clustGenes
    $cluster), FUN =paste))
geneList <- geneList[,c(1,3)]; names(geneList) <- c("cluster", "
    allGenes")
temp <- geneList$allGenes
geneList <- do.call(rbind, lapply(1:length(geneList$allGenes),
    function(x) cbind(geneList[x,], geneList$allGenes[[x]])))
names(geneList)[3] <- "gene"
# make tidy gene list
temp <- plyr::ldply(temp, rbind)
tidyGenes <- unite(temp, col = "genes", 1:ncol(temp), sep = ", ")
tidyGenes$genes <- gsub(pattern = ", NA", x = tidyGenes$genes,
    replacement = "")
write.table(tidyGenes, file = paste0("tscanHeatmap_geneClusters_
    d0-EpiSC_DGE_", time, ".txt"), row.names = TRUE, col.names =
    TRUE, sep = "\t", quote = FALSE)
# long format gene list
geneList <- geneList[,c(1,3)]
write.table(geneList, file = paste0("tscanHeatmap_geneClusters_
```

```
    long_d0-EpiSC_DGE_", time, ".txt"), row.names = TRUE, col.
    names = TRUE, sep = "\t", quote = FALSE)
# save heatmap cluster cell order of row means corrected log2TPM
    values
clustCenters <- pseudoHeat$kmeans$centers; rownames(clustCenters)
     <- rownames(tidyGenes)
write.table(clustCenters, file = paste0("tscanHeatmap_
    clusterExpression_d0-EpiSC_DGE_", time, ".txt"), row.names =
    TRUE, col.names = TRUE, sep = "\t", quote = FALSE)

# X chromosome linked gene filtering
chr <- read.table("M8_GTF_geneInfo.txt", header=F, check.names =
    FALSE, stringsAsFactors = F)
# match with expression table genes
chr <- chr[match(rownames(expr), chr$V5),]
identical(rownames(expr), chr$V5)

# subset for gonosomes and autosomes
chrX <- expr[which(chr$V1 == "chrX"),]
chrY <- expr[which(chr$V1 == "chrY"),]
chrM <- expr[which(chr$V1 == "chrM"),]
chrA <- expr[which(chr$V1 != "chrX" & chr$V1 != "chrY" & chr$V1 !
    = "chrM"),]

# filter cells and X genes based on minimum expression thresholds
markers <- chrX
markers <- markers[,which(colSums(markers)>1)]
markers <- markers[which(rowSums(markers)>1),]
markers <- markers[which(rowSums(markers>0)>10),]

# the following must be TRUE
identical(meta$cell_id, sne$cell_id)
sne$Pseudotime <- meta$Pseudotime
meta <- meta[match(names(markers), meta$cell_id),]
sne <- sne[order(sne$Cluster, sne$Pseudotime, decreasing = F),]
markers <- markers[,match(sne$cell_id, names(markers))]

# the following must be TRUE
identical(names(markers), sne$cell_id)

heat.vals <- log2(markers+1) - rowMeans(log2(markers+1))
timePoint <- meta$timePoint[match(names(heat.vals), meta$cell_id)
    ]
levels(timePoint) <- c("mES", "Day 1", "Day 2", "Day 3", "Day 4",
     "EpiSC", "MB#3P10", "1EP7")
cluster <- sne$Cluster[match(names(heat.vals), sne$cell_id)]
annotation <- data.frame(timePoint = factor(timePoint, labels =
    levels(timePoint)), cluster = factor(cluster))
rownames(annotation) <- names(heat.vals)
names(col) <- levels(annotation$timePoint)
names(col2) <- levels(annotation$cluster)
ann_color <- list(timePoint = col, cluster = col2)
```

```
time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
    character(Sys.time()), " ", n = 2))), collapse = "")
pseudoHeat <- pheatmap(heat.vals, cluster_rows = T, cluster_cols
    = F, annotation = annotation, annotation_names_col = F,
    kmeans_k = 20, treeheight_row = 70, annotation_colors = ann_
    color, show_rownames = T, show_colnames = F, main = "X linked
     genes of tSNE cluster and pseudotime ordered cells", width =
     14, height = 10, filename = paste0("S_X-genes_heatmap_14x10_
    ", time, ".pdf"))

# get genes for heatmap kmeans gene clusters
clustGenes <- data.frame(as.integer(pseudoHeat$kmeans$cluster),
    names(pseudoHeat$kmeans$cluster))
names(clustGenes) <- c("cluster", "allGenes")
geneList <- data.frame(aggregate(clustGenes, by = list(clustGenes
    $cluster), FUN =paste))
geneList <- geneList[,c(1,3)]; names(geneList) <- c("cluster", "
    allGenes")
temp <- geneList$allGenes
geneList <- do.call(rbind, lapply(1:length(geneList$allGenes),
    function(x) cbind(geneList[x,], geneList$allGenes[[x]])))
names(geneList)[3] <- "gene"
# make tidy gene list
library(tidyr)
temp <- plyr::ldply(temp, rbind)
tidyGenes <- unite(temp, col = "genes", 1:ncol(temp), sep = ", ")
tidyGenes$genes <- gsub(pattern = ", NA", x = tidyGenes$genes,
    replacement = "")
write.table(tidyGenes, file = paste0("tscanHeatmap_geneClusters_X
    -genes_", time, ".txt"), row.names = TRUE, col.names = TRUE,
    sep = "\t", quote = FALSE)
# long format gene list
geneList <- geneList[,c(1,3)]
write.table(geneList, file = paste0("tscanHeatmap_geneClusters_
    long_X-genes_", time, ".txt"), row.names = TRUE, col.names =
    TRUE, sep = "\t", quote = FALSE)
# save cluster x cell order of row means corrected log2TPM values
clustCenters <- pseudoHeat$kmeans$centers; rownames(clustCenters)
     <- rownames(tidyGenes)
write.table(clustCenters, file = paste0("tscanHeatmap_
    clusterExpression_X-genes_", time, ".txt"), row.names = TRUE,
     col.names = TRUE, sep = "\t", quote = FALSE)
```

2.5.9. Gene Ontology analysis

We used Gene Ontology (GO) analysis to support interpretation of selected DE gene lists. The R topGO package provides functions to run various statistical tests on enrichment analysis of GO terms. Due to the large number of identified GO terms we saved the top 100 GO scores based on the sorted p-values of the elimKolmogorovSmirnov function which applies the elim algorithm (Alexa *et al.*, 2006)) and a Kolmogorov-Smirnov test. Furthermore, three main ontologies were defined for which the top 100 scores are saved respectively. These three main ontology terms are Biological Process (BP), Cellular Component (CC) and Molecular Function (MF). The R code below also saves tables with the results of all statistical tests, the genes associated with each of the top GO terms, the number of genes that made up a GO term result and a brief description of the GO term. For presentation of GO results we used simple bar charts on GO terms with a significance cut-off of 5% (Figure 16).

```
# GO analysis
# loading required R packages
library(topGO)
library(AnnotationDbi)
library(org.Mm.eg.db)
library(Rgraphviz)
library(genefilter)
library(ALL)
# loads function topDiffGenes
data(geneList)

meta <- read.csv("metadata_20190223045845.csv", header=TRUE,
     stringsAsFactors = F)
tpm <- read.table("geneTPM_M8_20190207033818_Kallisto.txt",
     header=T, row.names = 1, check.names = FALSE)
tpm <- tpm[, which(names(tpm) %in% meta$cell_id)]
meta <- meta[match(names(tpm), meta$cell_id),]
expr <- tpm[,!meta$discard]
meta <- meta[!meta$discard,]
# the following must be TRUE
identical(meta$cell_id, names(expr))

files <- list.files("")
files <- files[grep("-Cluster[0-9]_results_batch-effect-corrected
     _20190223", files)]
```

```
# the following will loop over all kmeans cluster DGE files and
    save the result for each cluster pair in a file
for(h in 1:length(files)) {
  dge <- read.table(paste0("", files[h]), header=T, row.names =
      1, check.names = FALSE)
  mark <- dge[which(dge$`p-adjusted`<0.01),]

  name <- regexpr("_[A-z0-9]+-[A-z0-9]+[0-9]_", files[h]) %>%
    regmatches(files[h], .) %>% gsub("_", "", .)

  clu <- gsub("Cluster", "", name) %>% strsplit(., split = "-")
      %>% unlist

  pair <- unlist(list(eval(parse(text = paste0("c", clu[1]))),
      eval(parse(text = paste0("c", clu[2])))))
  index <- match(names(pair), names(expr))
  groups <- expr[,index]

  # make table with only selected DGE genes and cluster cells
  filter <- groups[rownames(groups) %in% rownames(mark),]
  filter <- filter[match(rownames(mark), rownames(filter)),]
  # the following must be TRUE
  identical(rownames(filter), rownames(mark))

  selProbes <- genefilter(filter, filterfun(pOverA(0.20, log2
      (100)), function(x) (IQR(x) > 0.25)))
  eset <- filter[selProbes, ]

  genes <- mark[rownames(mark) %in% rownames(eset),]
  # the following must be TRUE
  identical(rownames(genes), rownames(eset))

  anno <- AnnotationDbi::select(org.Mm.eg.db,
keys = rownames(genes), columns = c("ENSEMBL", "REFSEQ", "SYMBOL"
    , "GENENAME"), keytype = "SYMBOL")

  gene_pVal <- as.numeric(genes$`p-adjusted`)
  names(gene_pVal) <- rownames(genes)

  reference <- "org.Mm.eg.db"

  onts <- c( "MF", "BP", "CC" )
  tidyGenes <- as.list(onts)
  names(tidyGenes) = onts
  linked <- as.list(onts)
  names(linked) = onts
  tab = as.list(onts)
  names(tab) = onts

  for(i in 1:3){
    # prepare data
```

```
    tgd <- new( "topGOdata", description = paste0("tSNE ", name,
        " DGE"), ontology=onts[i],
                allGenes = gene_pVal, geneSel = topDiffGenes,
                    nodeSize=5, annot=annFUN.org, mapping="org.Mm
                    .eg.db", ID = "symbol")

    # run statistical tests
    resultFisher.elim <- runTest(tgd, algorithm = "elim",
        statistic = "Fisher" )
    resultFisher.classic <- runTest(tgd, algorithm = "classic",
        statistic = "Fisher" )
    resultKS.elim <- runTest(tgd, algorithm = "elim", statistic =
         "ks")
    resultKS.classic <- runTest(tgd, algorithm = "classic",
        statistic = "ks")

    if(length(score(resultKS.elim)) > 99) {
      # results table
      tab[[i]] <- GenTable( tgd, classicFisher = resultFisher.
          classic, elimFisher = resultFisher.elim,
classicKolmogorovSmirnov = resultKS.classic,
elimKolmogorovSmirnov = resultKS.elim,
orderBy = "elimKolmogorovSmirnov", topNodes = 100)

      go <- tab[[i]]$GO.ID[tab[[i]]$`Rank in elimFisher`]
      go.genes <- genesInTerm(tgd, whichGO = go)

      linked[[i]] <- as.data.frame(do.call(rbind, lapply(1:length
          (go.genes), function(x) cbind(names(go.genes)[x],
go.genes[[x]]))))
      linked[[i]]$V3 <- onts[i]
      names(linked[[i]]) <- c("GO.ID", "gene", "GO.type")

      # make tidy gene list
      library(tidyr)
      temp <- plyr::ldply(go.genes, rbind); names(temp)[1] <- "GO
          .ID"
      tidyGenes[[i]] <- unite(temp, col = "genes", 2:ncol(temp),
          sep = ", ")
      tidyGenes[[i]]$genes <- gsub(pattern = ", NA", x =
          tidyGenes[[i]]$genes, replacement = "")
    }
    else {
      # results table
      tab[[i]] <- GenTable(tgd, classicFisher = resultFisher.
          classic, elimFisher = resultFisher.elim,
          classicKolmogorovSmirnov = resultKS.classic,
elimKolmogorovSmirnov = resultKS.elim,
orderBy = "elimKolmogorovSmirnov" ,
topNodes = length(score(resultKS.elim)))
      go <- tab[[i]]$GO.ID[tab[[i]]$`Rank in elimFisher`]
      go.genes <- genesInTerm(tgd, whichGO = go)
```

```
      linked[[i]] <- as.data.frame(do.call(rbind, lapply(1:length
          (go.genes), function(x) cbind(names(go.genes)[x],
go.genes[[x]]))))
      linked[[i]]$V3 <- onts[i]
      names(linked[[i]]) <- c("GO.ID", "gene", "GO.type")

      # make tidy gene list
      library(tidyr)
      temp <- plyr::ldply(go.genes, rbind); names(temp)[1] <- "GO
          .ID"
      tidyGenes[[i]] <- unite(temp, col = "genes", 2:ncol(temp),
          sep = ", ")
      tidyGenes[[i]]$genes <- gsub(pattern = ", NA", x =
          tidyGenes[[i]]$genes, replacement = "")
    }
  }

  # Molecular Function top scores
  MF <- tab[[1]]; MF$GO.type <- "MF"
  tidyMF <- tidyGenes[[1]][match(MF$GO.ID, tidyGenes[[1]]$GO.ID)
      ,]
  # the following must be TRUE
  identical(tidyMF$GO.ID, MF$GO.ID)
  MF$genes <- tidyMF$genes
  MF <- MF[,c(11,1:10,12)]

  # Biological Process top scores
  BP <- tab[[2]]; BP$GO.type <- "BP"
  tidyBP <- tidyGenes[[2]][match(BP$GO.ID, tidyGenes[[2]]$GO.ID)
      ,]
  # the following must be TRUE
  identical(tidyBP$GO.ID, BP$GO.ID)
  BP$genes <- tidyBP$genes
  BP <- BP[,c(11,1:10,12)]

  # Cellular Component top scores
  CC <- tab[[3]]; CC$GO.type <- "CC"
  tidyCC <- tidyGenes[[3]][match(CC$GO.ID, tidyGenes[[3]]$GO.ID)
      ,]
  # the following must be TRUE
  identical(tidyCC$GO.ID, CC$GO.ID)
  CC$genes <- tidyCC$genes
  CC <- CC[,c(11,1:10,12)]

  combinedGO <- rbind(MF, BP, CC)
  combinedGenes <- rbind(linked[[1]], linked[[2]], linked[[3]])

  time <- paste(gsub("-|:", "", unlist(stringr::str_split(as.
      character(Sys.time()), " ", n = 2))), collapse = "")
  write.table(combinedGO, file = paste0("", "tSNE_", name, "_GO_"
      , time, ".txt"), row.names = TRUE, col.names = TRUE, sep =
```

```
        "\t", quote = FALSE)
  write.table(combinedGenes, file = paste0("", "tSNE_", name, "_
        longFormat_GO_genes_", time, ".txt"), row.names = TRUE, col
        .names = TRUE, sep = "\t", quote = FALSE)
  name
}
```

2.5.10. Zenbu data upload

Coordinate-sorted BAM files of the STAR alignment output and C1-CAGE BED files have been uploaded to the ZENBU browser for expression visualization and data exploration (Severin *et al.*, 2014) (Figure 5). The upload script below uses a table that contains the file directory, sample name, a custom sample information and in the last column semicolon-separated key-value pairs of custom selected metadata table columns. These metadata key-value pairs can be used to filter the samples in the ZENBU track views. The scRNA-seq data uploaded to ZENBU corresponds to the samples that still contain cluster 6 (Figure 10A).

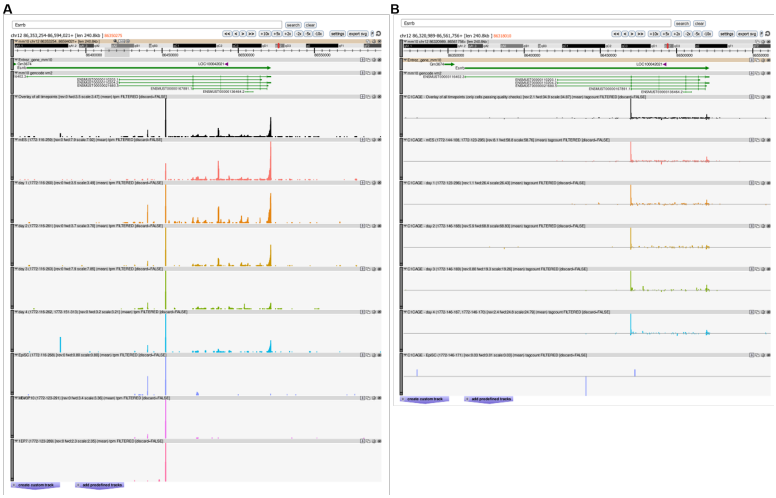


Figure 5: ZENBU browser view zoomed at *Esrrb* locus. **A**) All samples of the strandless Fluidigm scRNA-seq experiments uploaded as BAM files. **B**) All stranded C1-CAGE samples uploaded as BED files.

```
#!bin/bash -e
firstHalf() {
 BASEDIR= EpiSC_STAR/Sorted/
 for file in $(ls $BASEDIR/1772-* )
 do
   echo -e "$file\t$(basename $file .bam)"
 done
}

secondHalf() {
 IFS=$'\t'
 sed -e 1d \
     -e 's/"//g' \
     -e 's/,/\t/g' \
     metadata_Zenbu20180308132933.csv |
   while read cell_id Well Row Column Run \
         Concentration Control totalInputReads AllSpikes Cell \
         LiveDead Comment imageQC discard timePoint \
         Order Pseudotime Cluster
   do
     echo -n mm10 $timePoint $cell_id upload1
     echo -ne "\t"
     echo -n "cell_id=$cell_id;Well=$Well;Row=$Row;Column=$Column
          ;Run=$Run;"
     echo -n "Concentration=$Concentration;Control=$Control;
          totalInputReads=$totalInputReads;"
     echo -n "AllSpikes=$AllSpikes;Cell=$Cell;LiveDead=$LiveDead;
          "
     echo -n "Comment=$Comment;imageQC=$imageQC;discard=$discard;
          "
     echo    "timePoint=$timePoint;Order=$Order;Pseudotime=
          $Pseudotime;Cluster=$Cluster"
   done
}
paste <(firstHalf) <(secondHalf) > zUpload.tsv

#!bin/bash -e
# this does the actual uploading
/usr/local/bin/zenbu_upload \
        -url http://fantom.gsc.riken.jp/zenbu \
        -filelist zUpload.tsv \
        -assembly mm10 \
        -collab_uuid uZ5_zp0nAxnri7kZ2IGnbC \
        -singletag_exp
```

3. Results

In the following section the scRNA-seq time course data will be presented. As mentioned in the methods section the naïve-to-primed state transition was initiated by replacing the ES cell culture medium with EpiSC medium containing the Wnt inhibitor IWP-2. We define the day of this medium change as Day 0 or mESC. The reference primed state sample EpiSC has a different genetic background and is thus not suitable for allelic expression analysis. For that matter we also sampled induced primed PSC-like cells at Day 22 (MB#3P10) and a clonal cell line isolated from the induced primed PSC-like cells sampled at passage 20 (Clone 1EP7). These two originate from the same cell line from which our time course samples from Day 0 to Day 4 originated from. For simplification we want to highlight that whenever we refer to scRNA-seq, we refer to the Fluidigm standard scRNA-seq protocol data, because strictly speaking C1-CAGE is also a scRNA-seq method. Results based on C1-CAGE data will clearly be indicated as such. We obtained high quality data for 542 single cell transcriptomes via the Fluidigm scRNA-seq protocol and 587 cells via C1-CAGE (Table 2). These cell numbers are the remainder after applying stringent quality screenings. Each cell from scRNA-seq was sequenced with an average of 3.1 million reads per cell and 1 million sequencing reads for C1-CAGE respectively.

3.1. Transcriptome analysis of the naïve-to-primed transition process

One characteristic of naïve PSCs in mouse is global hypomethylation of the DNA (Table 1). In accordance with that characteristic the observed median of expressed genes between Day 0 and Day 2 is higher than 8000, whereas after Day 2 the median number of expressed genes approximates 7000 genes. Moreover, the variability in the number of

expressed genes per cell was larger in cells from the Day 3, Day 4 and EpiSC group compared to earlier time points. The sample size of MB#3P10 (26 cells) and 1EP7 (12 cells) was less than half the size of the next smallest EpiSC sample (69 cells), therefore these two groups show lesser variability than expected (Figure 6A). Spearman correlations that reflect similarities among adjacent cells (Petropoulos *et al.*, 2016) show a more variable distribution for the groups after Day 2, thus indicating a global change in cellular expression profiles during and after transitioning from naïve-to-primed PSCs (Figure 6B).

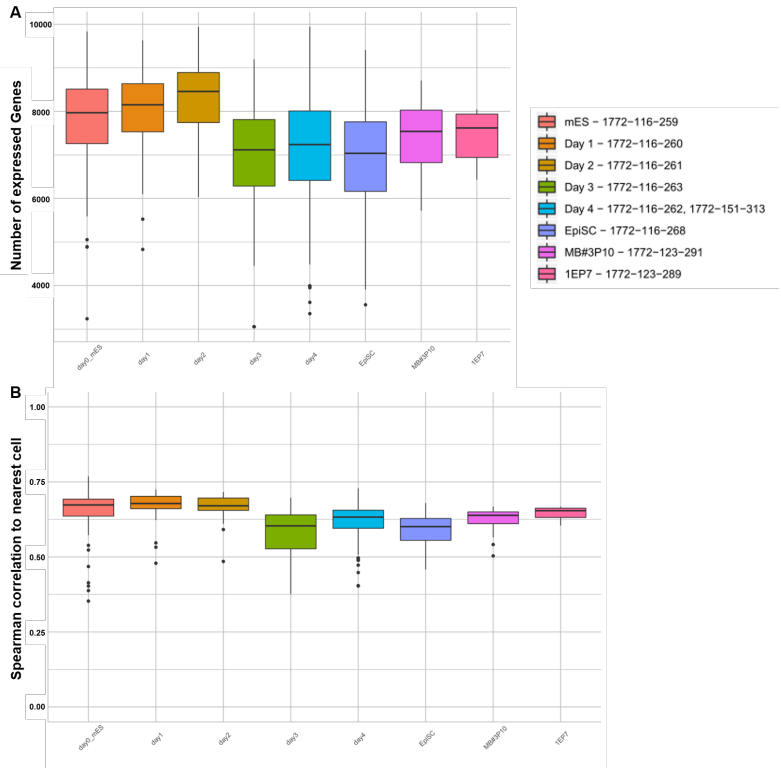


Figure 6: Variability of gene expression of scRNA-seq data over time. **A**) The distribution of expressed genes per time point. Genes are considered expressed, if they are detected in more than 10 cells with a TPM > 1. **B**) Single-cell quality assessment via neighboring cell similarities represented as Spearman correlations. All box plots show medians (center lines) with lower and upper quartiles. Whiskers represent 1.5x the interquartile range. Outliers are represented as dots.

We were able to validate our data by matching the expression of some selected marker genes with our time point samples (Figure 7). The selected genes of the naïve state (shown here *Esrrb*, *Zfp42*), pluripotency markers (*Pou5f1/Oct4*, *Nanog*) as well as primed state marker *Sox4* and EMT marker *Zeb2* all show expression patterns that match the expected sample groups. In general, we only considered genes that are expressed in at least 10 cells with a TPM greater than 1 to be displayed in any results figure.

In order to identify a set of genes that is most representative for the naïve and primed state we performed differential gene expression analysis between the mESC group and the EpiSC sample. By doing so we obtained 950 significantly DE genes (p-adjust < 0.01) between these two groups. Using this subset of DE genes we did a EpiSC on our Principal Component Analysis data (Figure 8A). The first two principal components PC1 and PC2 separate the cells depending on their developmental progression from naïve-to-primed (Figure 8A). The early time points Day 0 to Day 2 form a dense cluster of cells, whereas cells start to show larger expression heterogeneity after Day 2 and hence appear more widespread in the PCA plot. This observation is in line with the wider distribution seen in Figure 6A and B. It is striking how EpiSCs are clustered together opposite of the naïve cells, such as mESC or Day 1 (Figure 8A). Furthermore, Day 3 and Day 4 samples are mapped in between Day 0 and EpiSC, thus indicating that cells after Day 2 are largely entering the transition phase between naïve and primed PSC state.

We also applied t-SNE with the same set of DE genes and additionally used a kmeans clustering with five clusters. That way we

could subset our cells in an unbiased manner into comparable groups (Figure 8B and C).

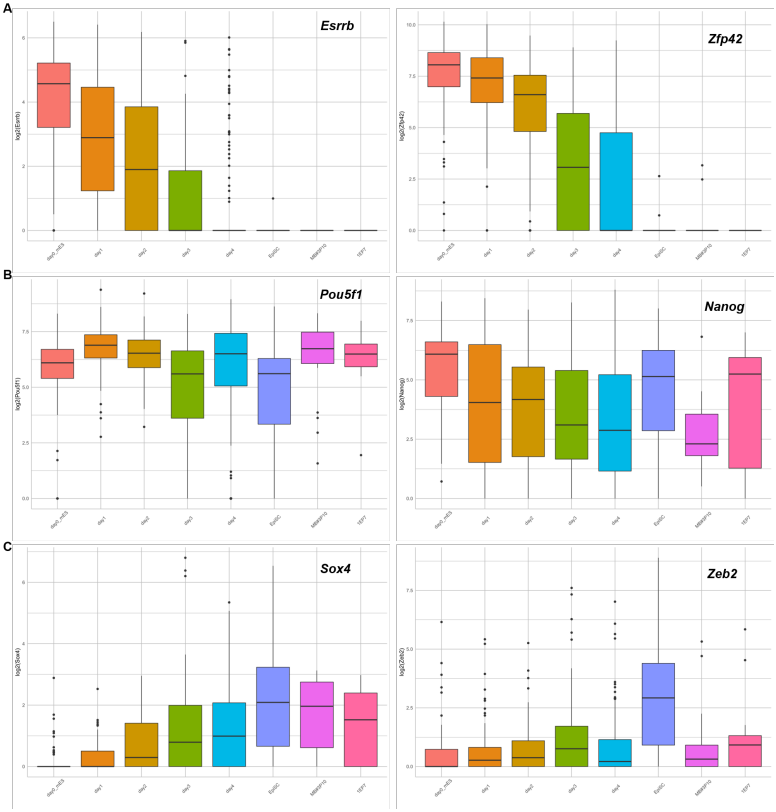


Figure 7: Selection of pluripotency related genes. **A**) Naïve state markers. **B**) General pluripotent stem cell markers. **C**) Primed state marker and EMT related transcription factor *Zeb2*. The expression values have been $\log_2$ transformed.

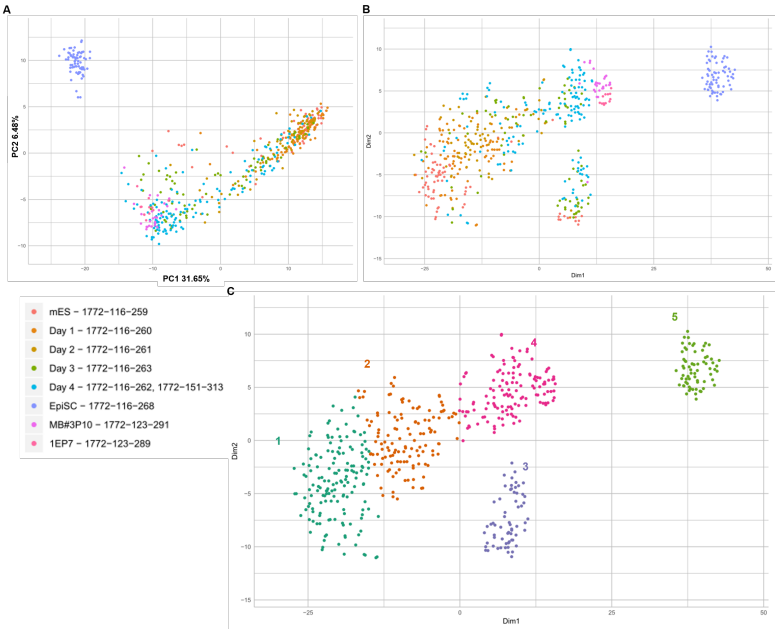


Figure 8: Clustering of scRNA-seq data based on 950 DE genes (p-adjusted < 0.01) between the mESC and EpiSC time point samples. **A**) PCA plot of all cells, **B**) t-SNE visualization and **C**) Grouping of cells based on t-SNE representation of the data via kmeans clustering.

To determine the temporal order of cell samples from transitioning time points we ran a pseudotime analysis based on the same DE genes subset that was previously used for PCA and t-SNE clustering. We overlaid the t-SNE plot with the pseudotime order of cells (Figure 9A). By visualizing it this way we could easily determine the developmental trajectory of samples within the five kmeans cluster groups. Notably, the pseudotime order accurately reflects the actual order of sampling time points (Figure 9B). This method might thus generally serve as a tool to assign temporal developmental order purely based on cellular gene expression profiles without prior knowledge of a samples time point in the context of naïve-to-primed transition. Furthermore, it can be assumed that the same method can be used in other developmental scenarios as well.

With regards to the trajectory indicated by the pseudotime sorting we adjusted the kmeans cluster numbering shown in Figure 8C to reflect the developmental progression. Cluster 1 is mainly composed of Day 0 and Day 1 cells, representing mostly naïve PSCs. Moving on, the Day 2 cells are contained in both cluster 1 and cluster 2, thus hinting that a fraction of the cells already start transitioning on Day 2. Cluster 2 also partly contains Day 3 and Day 4 cells. Matching the observations in the PCA, all EpiSCs are exclusively found in cluster 5. Interestingly and different from the PCA result, we can see two intermediary clusters (cluster 3 and 4) between the naïve and the primed state. Cluster 3 contains mainly Day 3 and 4 cells. Whereas cluster 4 includes Day 3 and 4 as well as the high passage number sample MB#3P10 and the clone 1EP7 (Figure 8B). The morphology of MB#3P10 and 1EP7 in cell culture is very similar to the EpiSC sample. Nevertheless, according to the t-SNE clustering the gene expression profiles of cluster 4 cells are quite distinct from cluster 5.

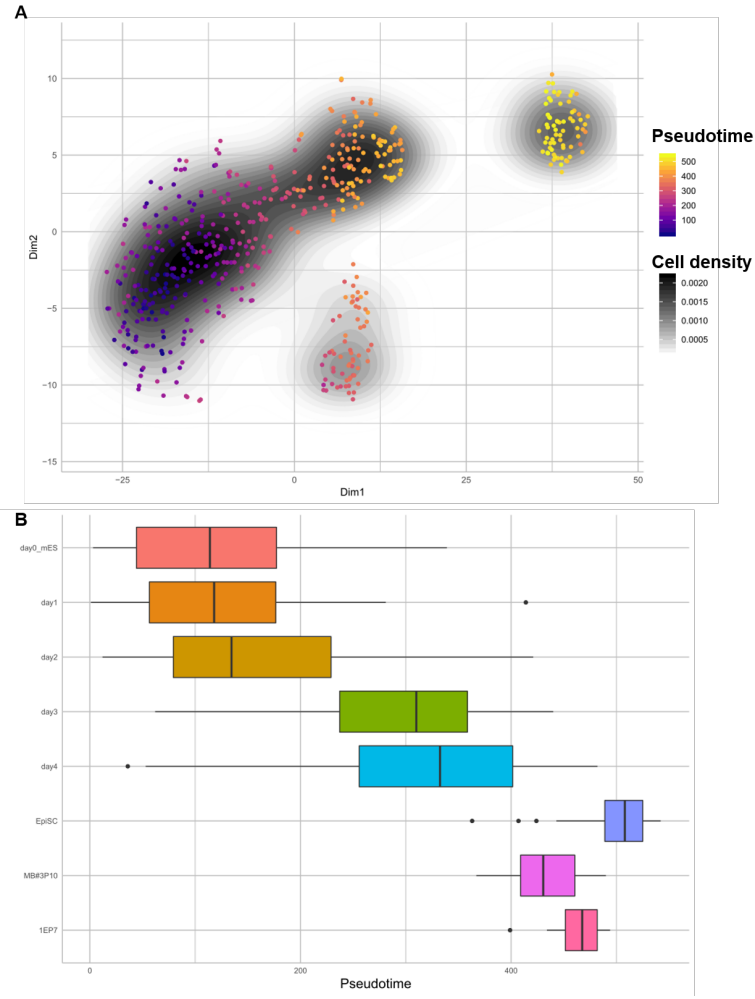


Figure 9: Pseudotime sorting of scRNA-seq data based on 950 DE genes (p-adjusted < 0.01) between the mESC and EpiSC time point samples. **A**) Color coded pseudotime of all cells within the t-SNE visualization and 2D kernel density estimation of cells. **B**) Pseudotime ordered cells grouped by real sampling time points.

Prior to obtaining the final t-SNE cluster result shown in Figure 8B and C, we initially ran a t-SNE clustering with six kmeans cluster centers (Figure 10A). A closer inspection of the initial sixth cluster revealed that the cells of this cluster were most likely contaminating feeder cells due to their expression of Y chromosome genes (*Eif2s3y* and *Ddx3y*) and the expression of the fibroblast marker Vimentin as well as their lack of pluripotency gene *Pou5f1/Oct4* expression (Figure 10B-D). Therefore, we decided to remove the cells of this initial sixth cluster to avoid confounding effects on our results. The final cluster results shown in Figure 8 were obtained after removing the 37 cells that formed the sixth cluster via t-SNE (Figure 10A) and redoing t-SNE kmeans clustering on the remaining cells.

Another concern was the question whether differences in cell cycle phases among single cells could influence our clustering results in any systematic manner. To address this issue, we performed a cell cycle phase assignment based on the orthologous expression of known phase marker genes (Whitfield *et al.*, 2002). The actual cell cycle distribution among our cells indicates that the cell cycle did not influence our results (Figure 11).

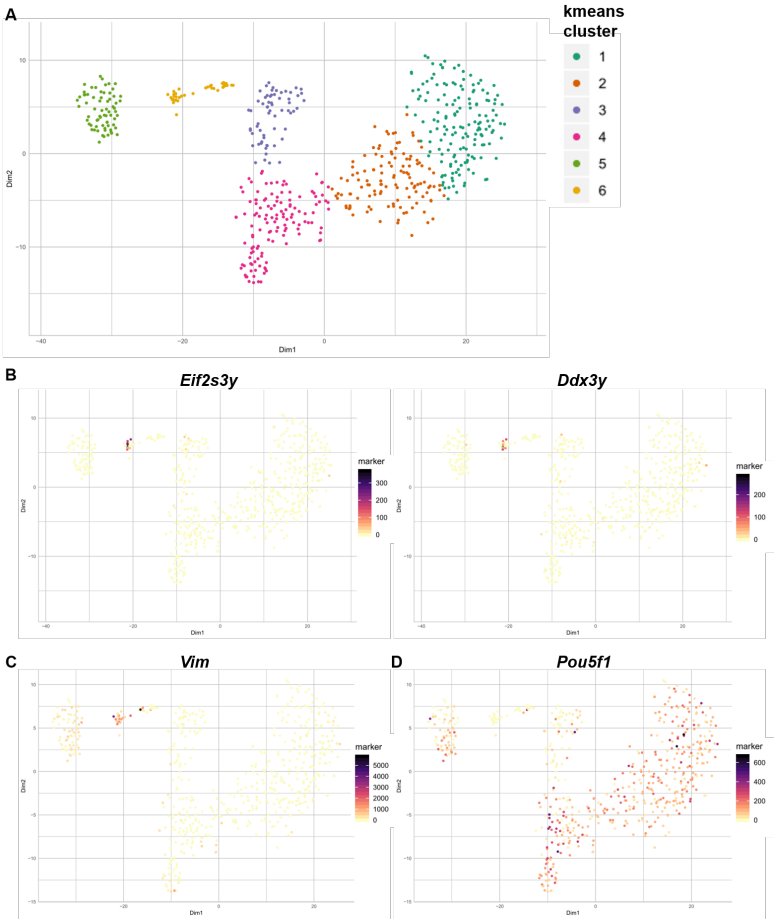


Figure 10: Initial t-SNE clustering with six cluster centers. **A**) Initial t-SNE based kmeans clustering of scRNA-seq data based on 916 DE genes (p-adjusted < 0.01) between the mESC and EpiSC time point samples. **B**) Expression of selected Y-linked genes. **C**) Mesenchymal cell marker Vimentin. **D**) Pluripotency marker *Pou5f1/Oct4*

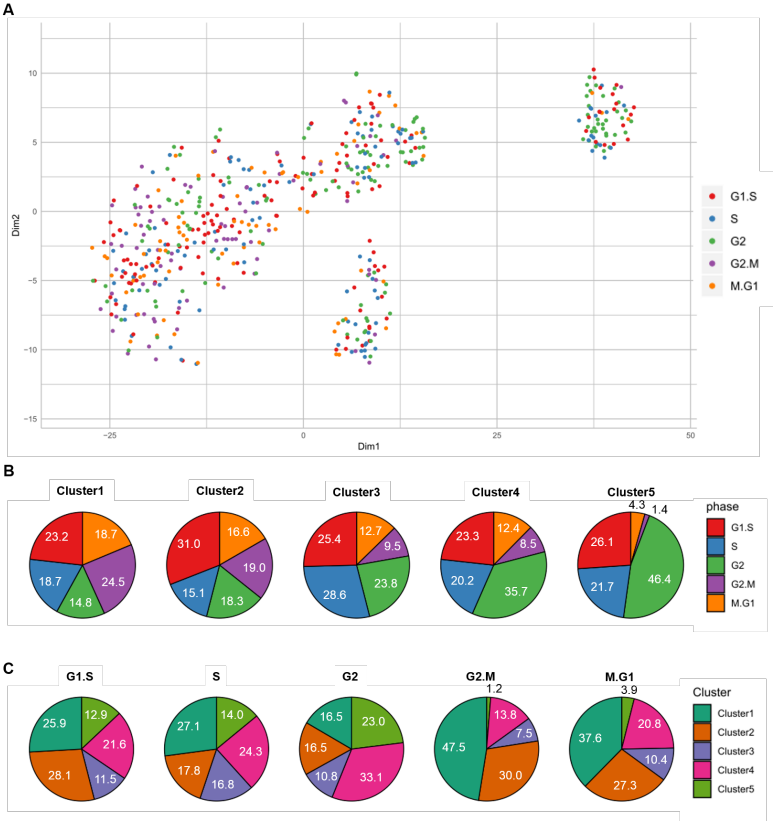


Figure 11: Cell cycle analysis of Fluidigm scRNA-seq data. Cell cycle scoring based on 176 cycle phase marker genes. **A**) Each cell's estimated cycle phase plotted onto the t-SNE clustering. **B**) Pie charts showing cell cycle distribution (%) per t-SNE kmeans cluster. **C**) Pie charts showing percentage of t-SNE kmeans cluster per cell cycle phase.

After assigning cluster order numbers based on the pseudotime results we also created overlay plots of clusters and pseudotime for the PCA plot (Figure 12A and B). The major difference in this visualization is that intermediary transition cluster 3 and 4 from the t-SNE kmeans clustering largely overlap in the PCA. The pseudotime shows a clear direction from right to left side of the PCA plot, as was to be expected due to the large concordance of pseudotime and real time points.

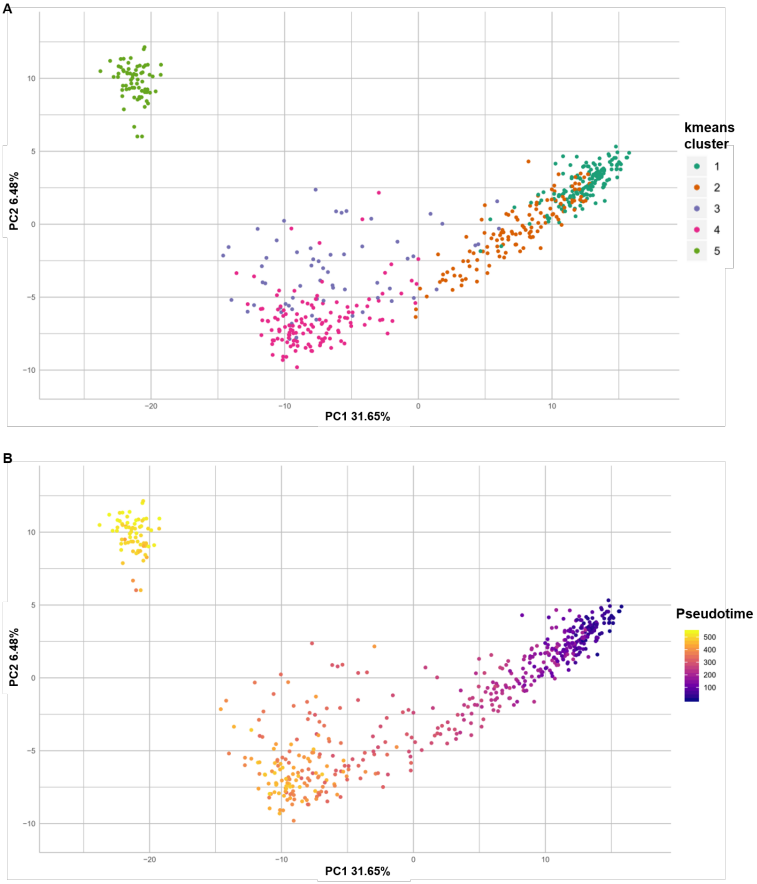


Figure 12: Exploratory PCA visualizations. **A**) scRNA-seq t-SNE kmeans cluster groups overlaid onto PCA plot. **B**) Color coded pseudotime of all cells shown in PCA plot.

For data exploration, a row-ordered hierarchical clustering heatmap was created. Cells are sorted first by t-SNE kmeans cluster numbers and secondly by pseudotime within each cluster. There are many genes that are specifically downregulated in the cluster 3 cell group. Furthermore, many genes are specifically up- or downregulated in naïve phase, transition and primed phase clusters (Figure 13A). We applied a kmeans clustering with an arbitrarily fixed cluster number of 20 on the hierarchical clustering results for better readability of the heatmap, thus the shown dendrogram has 20 branches. In order to better characterize individual clusters, we performed pairwise DGE analysis between all successive cluster pairs. The DGE results show a large increase in the number of significant DE genes between cluster 2 and 3, as well as 3 and 4, suggesting that cluster 3 exhibits distinct expression profiles compared to other clusters. Additionally, cluster 1 and 2 exhibit the lowest number of DE genes due to most cells in both clusters still being naïve PSCs. Contrary to that, cluster 4 and 5 display a larger number of DE genes, reflecting the differences between cells undergoing transition and cells in the primed PSC state.

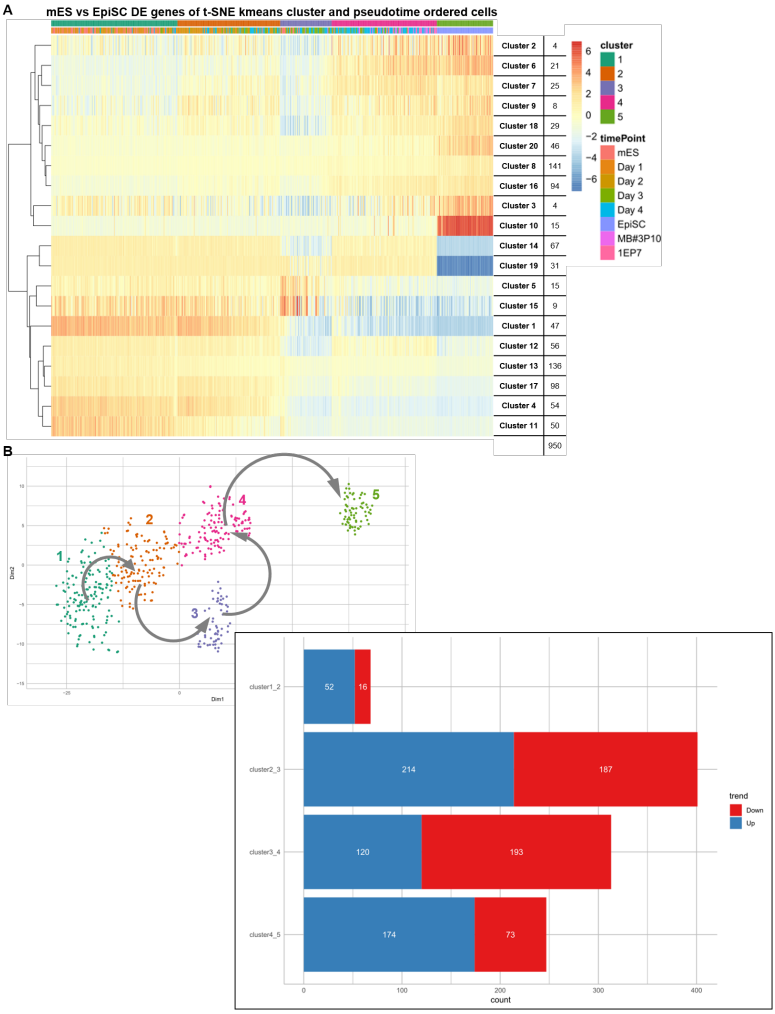


Figure 13: Cluster specific gene expression patterns. **A**) Heatmap with cells sorted by t-SNE kmeans cluster groups and pseudotime. 20 kmeans row gene clusters ordered via hierarchical clustering. Expression scale is $\log_2(\text{TPM}+1)$. **B**) Differential gene expression between t-SNE kmeans clusters for marker gene identification. Number of up and downregulated DE genes (p-adjusted < 0.01) between clusters.

3.2. Characterization of scRNA-seq t-SNE clusters based on gene expression patterns

The following chapter highlights some some candidate DE genes for cluster characterization. At first, we focused on the known naïve phase marker Nlrp4f which is only expressed in preimplantation embryos (Peng *et al.*, 2013). Its expression in our samples matches the naïve phase time points (Figure 14A). Next, we identified the snRNA *Rn7sk* as an interesting target gene for our intermediary cluster 3, since it is exclusively expressed in cluster 3 cells (Figure 14B). Finally, *Cd59a* is a highly specific gene for cluster 5 and the late passage samples MB#3P10 and 1EP7 (Figure 14C).

As shown in Figure 14 the expression of any gene can be visualized at single-cell resolution by overlaying single-cell expression levels onto the t-SNE map (Figure 14, Figure 15, Figure 16). By examining such visualizations for all manually screened 1044 DE genes, we identified genes specific to each cluster, as well as genes enriched in multiple clusters, or absent from all but one cluster. Based on these DE genes expression patterns, characteristics of each cluster can be outlined. It is important to keep in mind that no gene will be expressed in 100 percent of the cells of a specific cluster or developmental stage. Nevertheless, if several thousands of single cells were sampled and sequenced an even clearer trend for specific gene expression would be discernible. Despite the limitations of time-frame, budget and available technologies for this project we were able to identify many known and novel phase-specific genes. Some of these genes have been picked as representative examples for potential marker genes of different stages of the naïve-to-primed transition process.

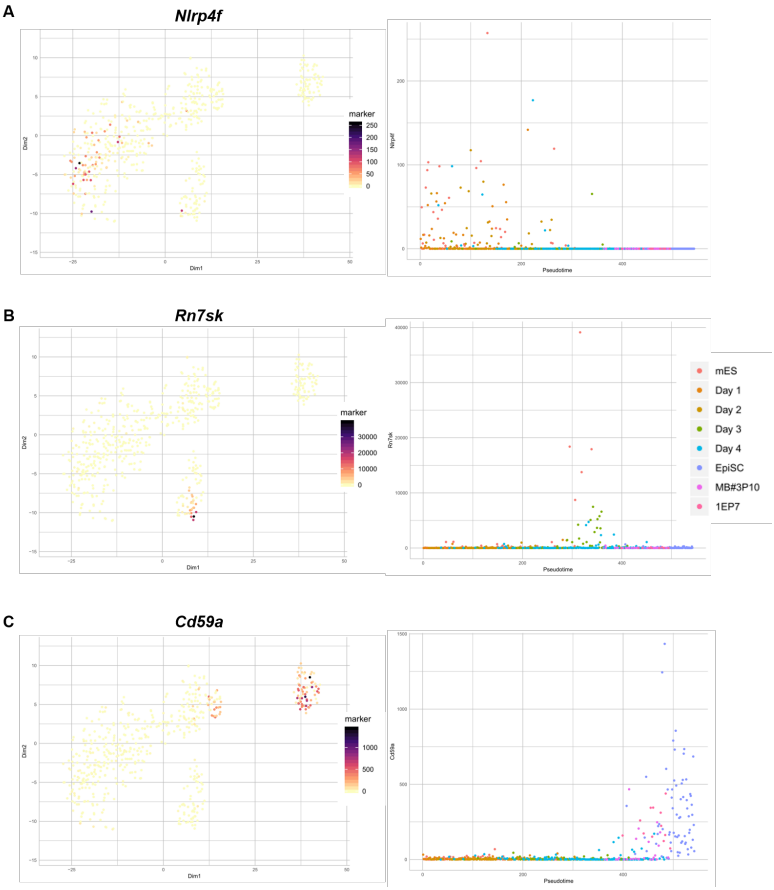


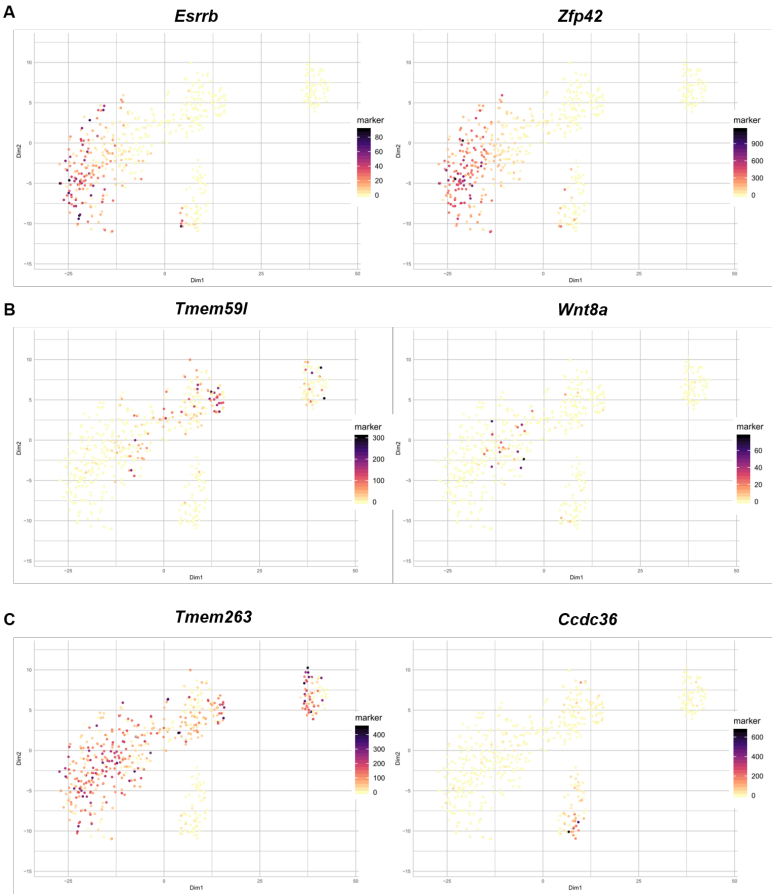
Figure 14: Selected cluster specific genes. Expression (TPM) of **A**) naïve (*Nlrp4f*), **B**) transition phase (*Rn7sk*) and **C**) primed state (*Cd59a*) markers shown as overlay of the t-SNE plot and marker expression (TPM) plotted against the pseudotime scale.

Expression of naïve pluripotency genes such as *Esrrb* or *Zfp42* is enriched in cluster 1 and cluster 2 but vanishes after progression to cluster 3 (Figure 15A). Many other known naïve pluripotency genes are heterogeneously expressed in cluster 2 before getting downregulated as cell differentiation continues. In fact, most of the DE genes in cluster 2 are expressed in other clusters as well. Still, there are some genes - such as the transmembrane protein coding gene *Tmem59l* - whose expression is initiated in cluster 2 and continues to be expressed until later stages, thus indicating that naïve-to-primed conversion may already commence from this cluster. We could find only a few genes that exhibit a cluster 2-specific expression pattern, for example the protein-coding *Wnt8a* (Figure 15B).

Cluster 3 and 4 appear as intermediary clusters between the naïve and primed state (Figure 13B). This is a novel and interesting find, since it was previously unknown whether subpopulations of cells can be found during the naïve-to-primed transition. This result could only be obtained due to the single-cell sampling in our study. A classical sequencing of bulk cell samples would have masked specific expression patterns that enabled high resolution clustering. Strikingly, there is widespread downregulation of many hundreds of genes in cluster 3 (Figure 18A and B), for example *Tmem263*. In contrast, there is a group of genes exhibiting specific upregulation only in cluster 3, e.g. *Ccdc36* and *Rn7sk* (Figure 15C, Figure 14C). Special attention should be given to the cluster 3 specific expression of *Rn7sk*. This gene is an snRNA that acts as a transcriptional regulator in embryonic stem cells. It decreases the rate of RNA Pol II elongation and inhibits the CDK9/Cyclin T complex (Prasanth *et al.*, 2010; Castelo-Branco *et al.*, 2013). One explanation for this observation might be that gene regulatory networks are reconfigured in this transient state, in order to prepare cells for later lineage commitment. Unlike cluster 4, cells in cluster 3 also show residual expression of naïve pluripotency genes and initial expression of primed marker genes (Figure 15A and B). The cell adhesion molecule E-cadherin (*Cdh1*) is known to be expressed in naïve type ESCs, but not in primed EpiSCs (Ohtsuka *et al.*, 2012) (Figure 15D). Other genes like *Krt18* and *Cyp24a1* demonstrate cluster

4 specific expression but are not detected in cluster 5 (Figure 15F). This suggests that cluster 4 cells have different expression profiles than cluster 5 cells. Known primed marker genes such as *Fgf5* or *Pou3f1* (Boroviak *et al.*, 2015)) are expressed in cluster 4 (Figure 15E), while naïve pluripotency gene expression has been almost diminished in cluster 4, which prompts their primed identity. In fact, those primed marker genes are detectable in cluster 4 as well as cluster 5, which is exclusively composed of EpiSCs.

Cluster 5 exhibits expression of genes specific to only this cluster, for example, the N-cadherin gene *Cdh2* (Figure 15D). *Cdh2* is known to be involved in EMT, and its presence suggests that cluster 5 cells have completed EMT, whereas cluster 4 cells are still undergoing it. This is an important result, because EMT is one of the landmarks of naïve-to-primed transition (Pieters and Roy, 2014). This result indicates that cluster 4 cells have not completed EMT, thus representing a novel, intermediate pluripotency state between naïve and primed pluripotency. Additionally, we manually identified 54 out of the 1044 DE genes, which are cluster 5-specific. One of these genes is *Cd59a*, which we propose as a novel EpiSC marker (Figure 14C).


A
Esrrb
Zfp42
B
Tmem59l
Wnt8a
C
Tmem263
Ccdc36
marker
Dim1
Dim2

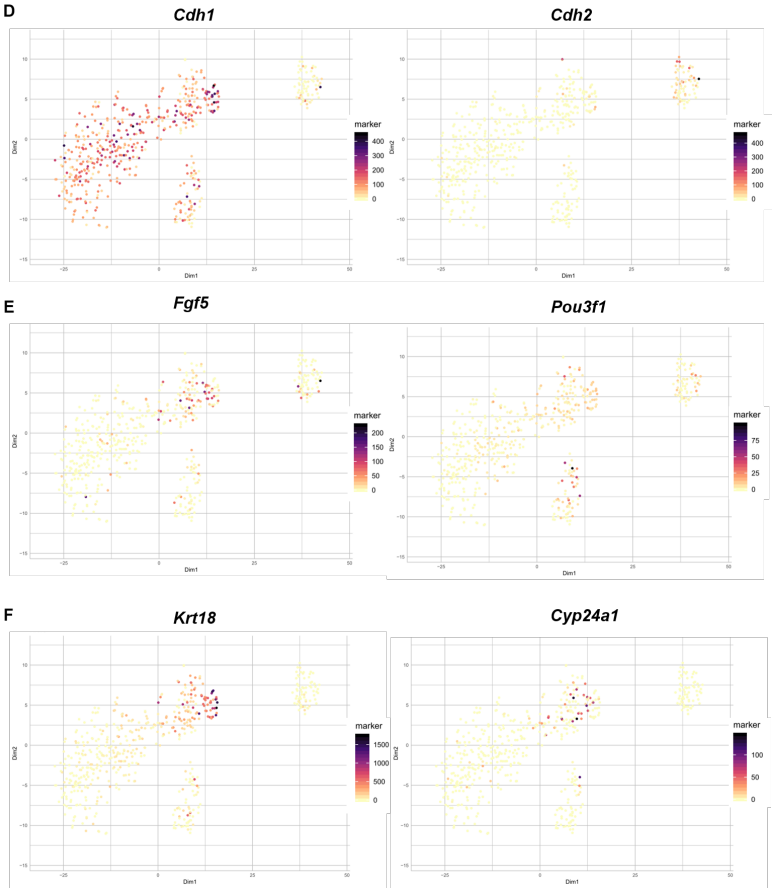


Figure 15: Expression (TPM) of selected DE genes between all t-SNE kmeans clusters plotted onto the t-SNE clustering. Panels **A–F** show genes that are either specific to one or more kmeans clusters or absent from a cluster.

Based on the lists of significant DE genes for each cluster pair we ran a GO analysis. We identified a diverse set of ontology terms for the cluster 2 and 3 pair, such as terms related to morphogenesis (Figure 16A). Cluster 3 and 4 show many DE genes with affiliated GO terms in development and differentiation (Figure 16B). We omitted showing GO results for cluster 1 and 2 comparison and cluster 4 against cluster 5 due to a very limited number of significant GO terms enriched between these groups. The GO analysis results shown in Figure 16 provide some useful information about the most relevant processes marking the initiation and ongoing transitioning process. Nevertheless, GO analysis results are generally only useful to provide some insides for interpretation, but are often confounded with unaffiliated or generic terms that could apply to many conditions.

As claimed before, the time between Day 2 and Day 3 is the starting point for most cells to enter transition from the naïve-to-primed phase. To further support this claim, we investigated the XCI status of cells, because XCI is one of the most reliable indicators of cell differentiation (Deuve and Avner, 2011; Payer, 2016). During random XCI in embryo development either the maternal or paternal X chromosome is randomly inactivated in female mammalian cells. The long non-coding RNA (lncRNA) *Xist* is known to be important in the initiation of XCI and will silence most X-linked genes on one X chromosome, except for escape genes. It is assumed that XCI occurs as cells exit from the naïve state, though precise timing of the XCI initiation has not been determined (Shiura and Abe, 2019).

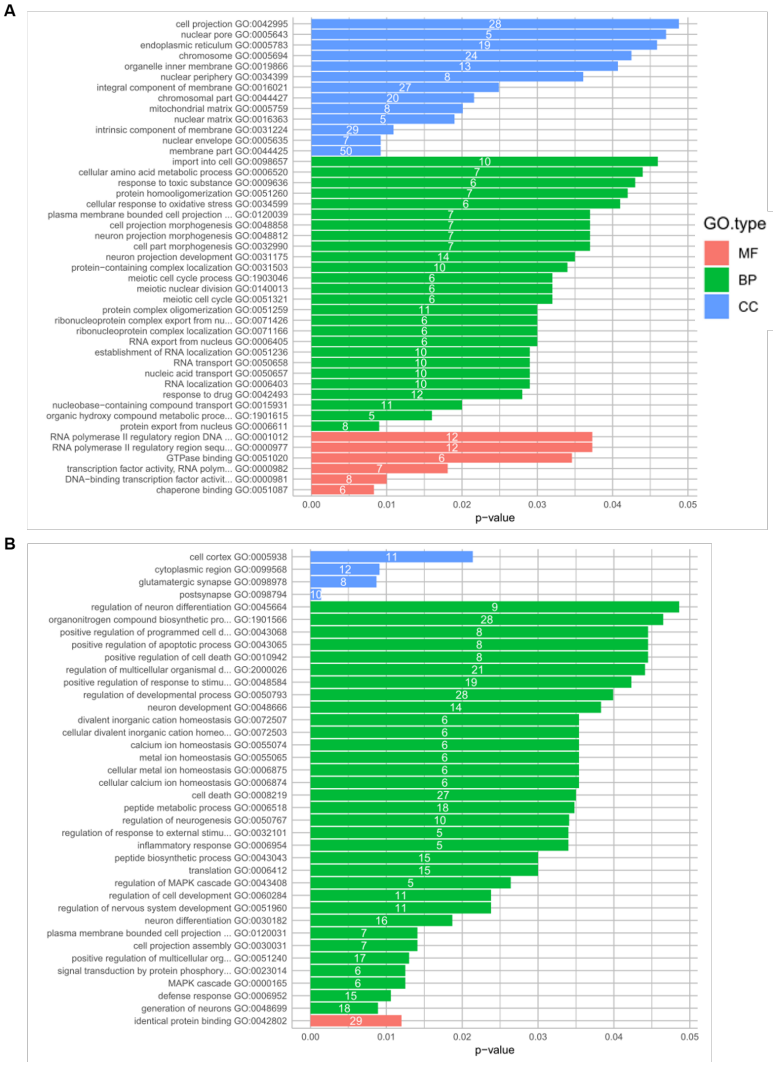


Figure 16: Top ranked gene ontology analysis results sorted by Molecular Function (MF), Biological Process (BP) and Cellular Component (CC). **A**) Top GO terms identified from the DE genes between t-SNE kmeans clusters 2 and 3 as well as **B**) clusters 3 and 4. Numbers inside bars reflect number of represented DE genes.

We calculated and compared X chromosome to autosome (X/A) expression ratios in each single cell (Figure 17A). The ratio is relatively constant from Day 0 to Day 2 but starts to decrease after Day 2 with Day 3 and Day 4 showing the largest expression ratio heterogeneity among all samples. After Day 4 the distribution of expression ratios narrows down and stays at a much lower level compared to Day 2 and earlier. This indicates that total expression levels of the X-linked genes are starting to get reduced at Day 3 compared to Day 0, Day 1 and Day 2. Therefore, we conclude that XCI initiates between Day 2 and Day 3. Interestingly, the expression of the lncRNA *Xist* is upregulated in some cluster 3 cells, whereas the *Xist* antisense RNA *Tsix* is downregulated in the same cluster compared to preceding clusters (Figure 17B and C). *Tsix* is known to act as a negative regulator of *Xist* transcription (Sado *et al.*, 2005).

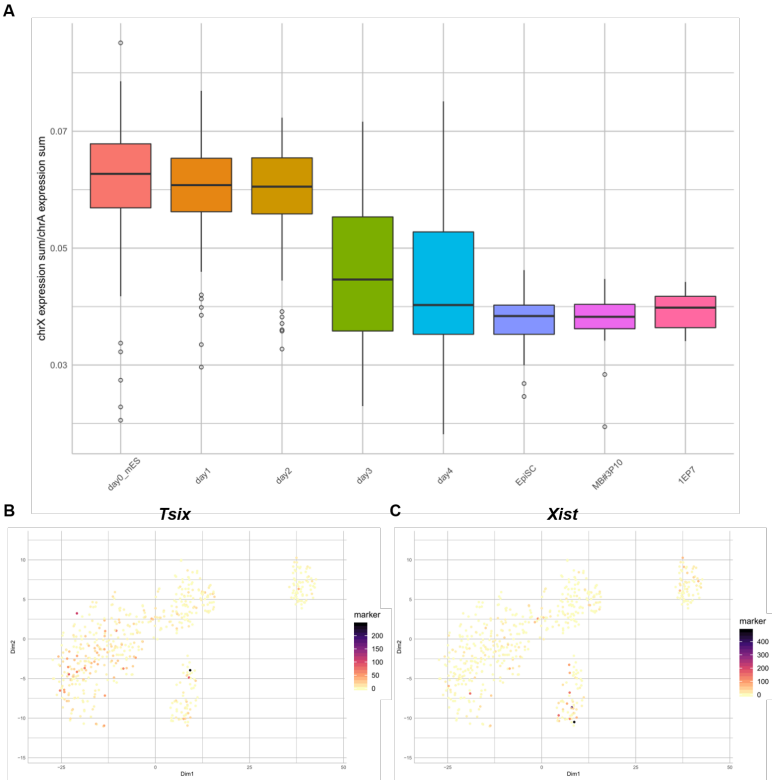


Figure 17: X chromosome expression (TPM) over all sample time points. **A**) Differences in ratios of X chromosome expression levels to autosomal expression levels, from mESCs to EpiSCs. Expression of the X chromosome genes **B**) *Tsix* and **C**) *Xist* plotted onto the t-SNE clustering.

Last but not least, we created expression heatmaps for all expressed autosomal and X-linked genes respectively (Figure 18). Our Gencode reference annotation has 44668 gene IDs for autosomal genes of which 21777 gene IDs show an expression above our threshold (Figure 18A). There are 2606 gene IDs for X-linked genes in the Gencode reference. Of these 965 gene IDs show expression above our threshold and are shown in the heatmap in Figure 18B. In both heatmaps we can see a downregulation of genes in cluster 3 as previously mentioned. Furthermore, we can see that hundreds of genes are specifically down- or upregulated in the naïve clusters Day 0, Day 1 and Day 2 or the remaining clusters. Most of these cluster genes are identical to the DE genes identified through DGE analysis of cluster pairs.

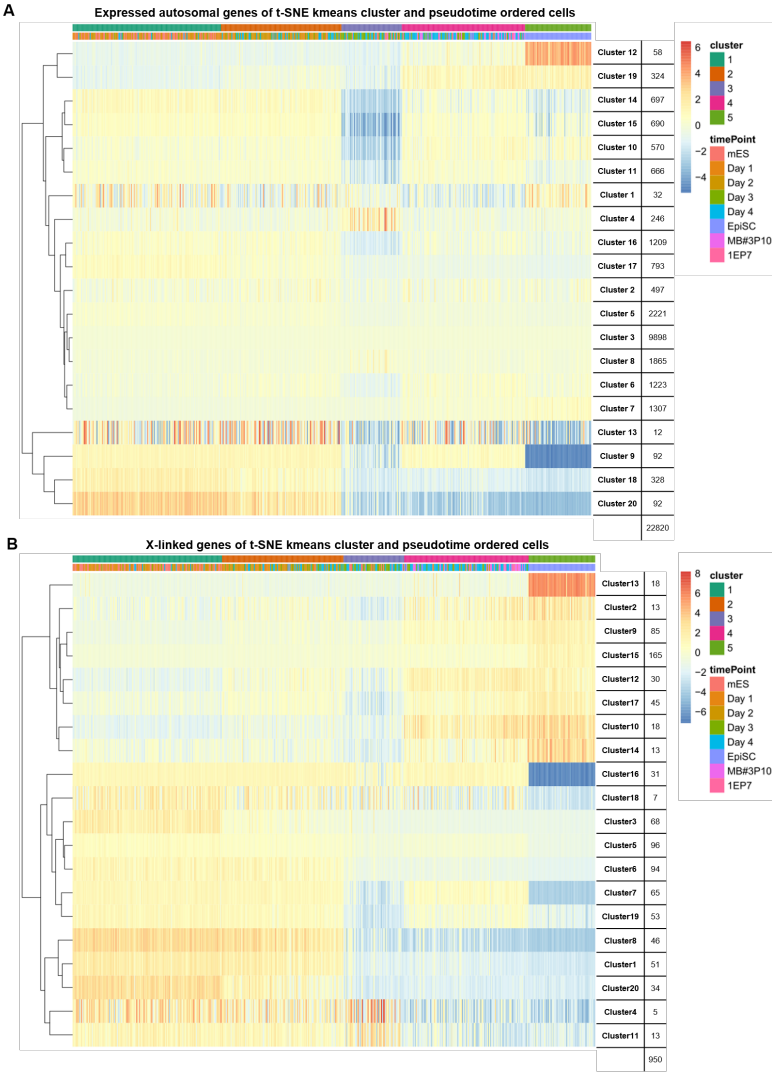


Figure 18: Wide-spread downregulation of genes in t-SNE kmeans cluster 3. Heatmaps with **A**) autosomal genes and **B**) X-linked genes. Cells sorted by t-SNE kmeans cluster groups and pseudotime. 20 kmeans row gene clusters formed via hierarchical clustering. Expression scale is $\log_2(\text{TPM}+1)$.

3.3. C1-CAGE data analysis

For the final part of the results section all figures are based on the C1-CAGE sequencing data. Following the same procedure that was used to cluster the Fludigm scRNA-seq derived data, we generated a t-SNE plot for the C1-CAGE data using 635 DE genes between the Day 0 and EpiSC sample. As shown in Figure 19A our C1-CAGE data can conveniently validate the t-SNE kmeans cluster results from the Fluidigm scRNA-seq protocol (Figure 8C; Figure 10A). C1-CAGE kmeans cluster 1 and 2 correspond to the naïve pluripotency state (Figure 20B and C). More importantly we independently obtained two transition stage clusters, cluster 3 and 4. The snRNA *Rn7sk* shows cluster 3 specific upregulation as well (Figure 20D). Finally, we also got an EpiSC specific cluster 5 (Figure 20E) and a small cluster 6 comprising of feeder cells or differentiated cells based on the absence of the pluripotency marker *Pou5f1/Oct4* and the expression of *Vim* (Figure 20A and F, comparison Figure 7B and Figure 10). Although a little less distinct than the pseudotime sorting result for scRNA-seq, we still got pseudotime orders that convincingly reflect the real temporal order of samples as previously seen for the scRNA-seq data (Figure 19B). The C1-CAGE protocol enables the detection of expressed non-poly adenylated transcripts. A feature that is missing from Fluidigm scRNA-seq. To make use of this advantage we created a comprehensive gene expression matrix by combing reference annotations from Gencode, Fantom5 enhancer (Andersson *et al.*, 2014; Arner *et al.*, 2015) and promoters (Forrest *et al.*, 2014) and non-annotated stem cell transcripts (Fort *et al.*, 2014) annotations. The different priming strategies between both protocols have different RNA capture efficiencies. Thus, there is a larger variability with regards to the total number and variety of IDs of expressed genes that have been detected. Overall C1-CAGE has a lower RNA capture efficiency compared to scRNA-seq.

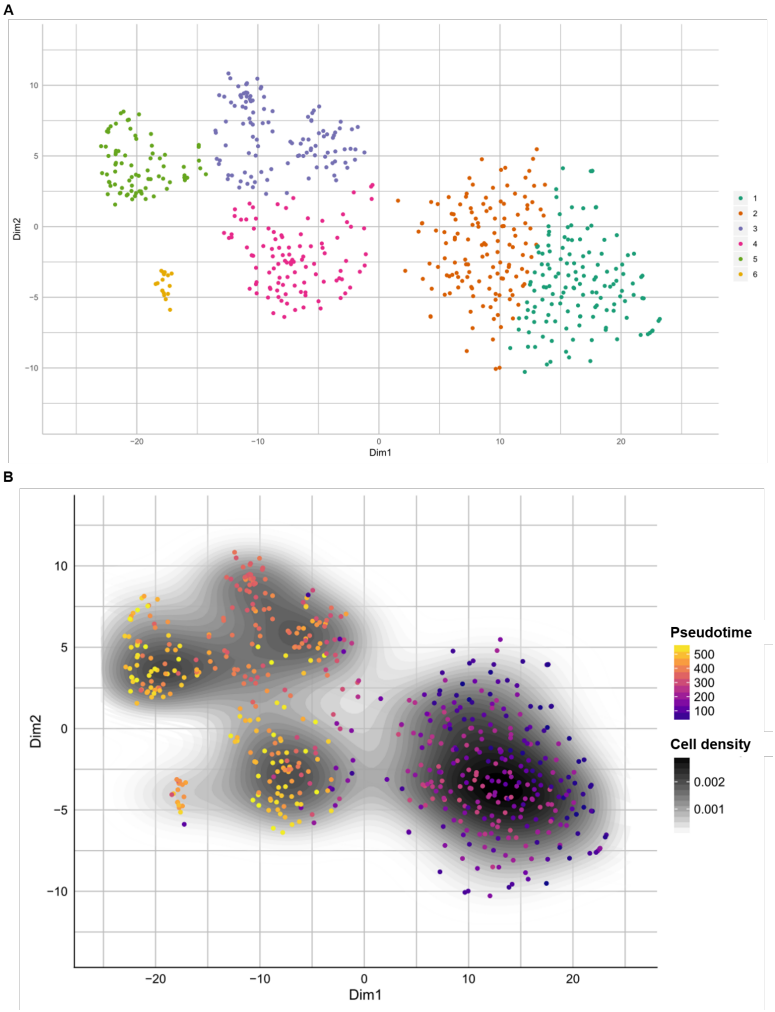


Figure 19: Clustering of the C1-CAGE data. **A**) t-SNE based on 635 DE genes (p-adjusted < 0.01) between the mES and EpiSC time point samples. kmeans cluster groups validate the pattern observed in the Fluidigm scRNA-seq data. **B**) Color coded pseudotime of all cells within the t-SNE visualization and 2D kernel density estimation of cells. TSCAN pseudotime sorting based on 982 DE genes (p-adjusted < 0.01) between t-SNE kmeans cluster 1 and 5.

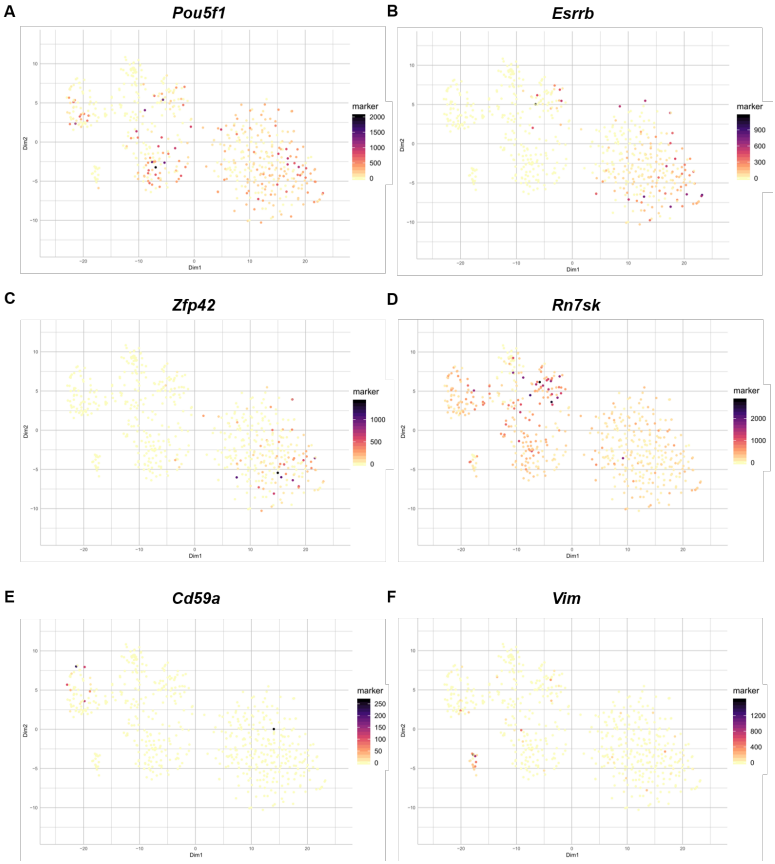


Figure 20: Expression visualization of C1-CAGE data. Plots **A**-**F** show expression of selected genes plotted onto the C1-CAGE t-SNE clustering that have been shown in previous Figures for the Fluidigm scRNA-seq data.

We found that a small number of 29 NAST genes seems to be specific for the naïve state and get downregulated upon entering the transition phase and primed state (Figure 21). Once again, we can observe a decrease of NAST gene expression during the naïve-to-primed transition phase (Figure 21). NASTs are a class of non-coding RNA which are shorter and expressed at lower levels than other known RNAs. It has not been well studied to what extent annotated NASTs may be part of genes from other annotation sources and thus to what degree individual NASTs are genuinely unique genes. Despite that, we considered it interesting to show that some of the expressed NASTs seem to show specificity in either naïve or the primed pluripotency state in mice. We consider our C1-CAGE data a valuable resource for the scientific community, that enables further investigations into diverse classes of expressed non-coding RNA in the developmental context of naïve-to-primed pluripotency.

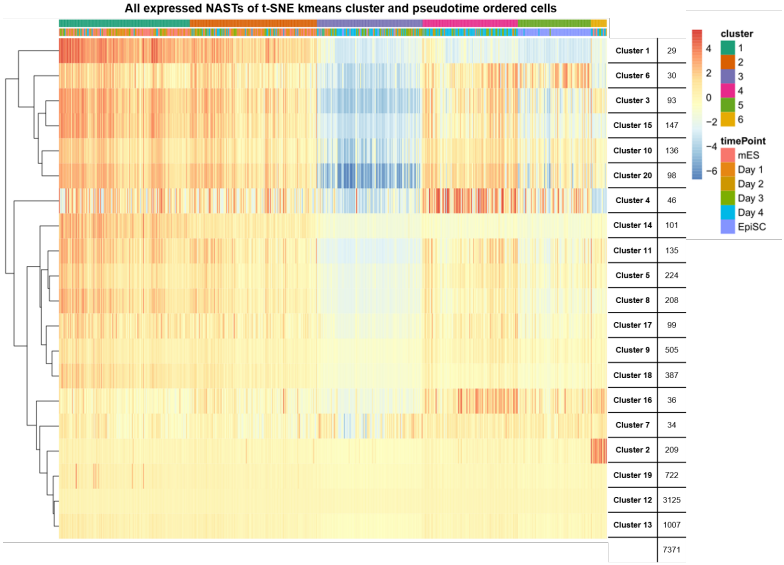


Figure 21: Cluster specific NAST expression patterns. Heatmap with cells sorted by t-SNE kmeans cluster groups and pseudotime based on C1-CAGE data. Shown are 20 kmeans NAST gene row clusters ordered via hierarchical clustering. Expression scale is $\log_2$(Count+1).

4. Discussion

For this thesis project, transcription dynamics of the naïve-to-primed transition process in mouse have been explored for the first time by using two different single-cell transcriptomics techniques. The obtained data provides a comprehensive catalogue of genes exhibiting characteristic changes during this differentiation process. Through DGE analysis we identified known and novel marker genes that should be extremely useful for further functional characterization of the transition process. One major finding was the presence of two intermediary subpopulations of cells in addition to the naïve and the primed PSCs. The existence of such subpopulations cannot be discovered via bulk RNA expression analysis, thus emphasizing the merits of single-cell technologies. Furthermore, we could validate our clustering results by using two different protocols for single-cell transcriptome analysis.

4.1. Discovery of two distinct transition states during naïve-to-primed conversion

We find that thousands of genes are transiently downregulated in the cell population that is formed by the t-SNE kmeans cluster 3 (Figure 13A, Figure 18, Figure 21). This downregulation is present in both autosomal and X-linked gene expression. The high variation of gene expression observed in cluster 3 cells may be due to cells undergoing different degrees of gene expression repression at the time of sample collection. Apart from that reason cells from this population entered the transition phase for varying length of times as indicated by the expression of both naïve and primed stage marker genes in some cells (Figure 14, Figure 15, Figure 20). We are confident that the two transition phase clusters 3 and 4 are true subpopulations and not artifacts due to experimental limitations. The detection of these clusters could be reproduced using the different protocols Fluidigm

scRNA-seq and C1-CAGE, as well as sampling two separate batches of Day 4 for scRNA-seq (see Table 2). Overall cluster 3 cells exhibited expression profiles that are very divergent from cells of other t-SNE kmeans clusters. The presence of both naïve and primed marker genes in this cluster as well as pseudotime sorting results, position this cluster as an intermediate stage between the naïve and primed state. Additionally, cell cycle analysis does not show any bias of cells from this cluster towards a phase (Figure 11). As mentioned before, many genes are specifically downregulated in cluster 3, but there are also some genes upregulated in this cluster. Particularly we identified the snRNA *Rn7sk* as an example for cluster 3 specific upregulation. This gene provides a clue to the observed cluster 3 specific gene repression. *Rn7sk* inhibits the kinase complex P-TEFb and thus acts as a transcription repressor by preventing Pol II elongation (Peterlin and Price, 2006). Furthermore, *Rn7sk* is known to be a gene-specific transcriptional repressor in mESCs (Castelo-Branco *et al.*, 2013). The second unexpected find of our work is the discovery of t-SNE kmeans cluster 4 (Figure 8C). Cells from this cluster are distinct from the EpiSC-specific cluster 5, although they show similar morphologies (Figure 3A) and express some primed state marker genes as well. We found that cluster 4 cells express *Cdh1* (E-cadherin), but do not express *Cdh2* (N-Cadherin) (Figure 15D). Naïve PSCs undergo the EMT process, in which *Cdh1* expression of the naïve PSCs is replaced with *Cdh2* expression that is specific to primed PSCs (Altshuler *et al.*, 2018). The lack of *Cdh2* expression in cluster 4 suggests that EMT is not yet completed in this population (Figure 15D). EMT completion is a defining criterion for EpiSC, but cluster 4 cells do not fulfill this criterion, and they are able to self-renew while being in the transition stage. Due to that cluster 4 cells may represent a hitherto unknown and novel type of intermediate PSCs in mice beside mESCs and EpiSCs. Recently, a third pluripotency state called 'formative' has been proposed as an intermediate state between naïve and primed pluripotency. The formative pluripotency state is hypothesized to be a transcriptional, epigenetic, signaling and metabolic network remodeling phase necessary for acquiring responsiveness to lineage commitment cues (Smith, 2017). Additionally,

it was recently reported that human naïve PSCs can acquire novel pluripotency comparable to the hypothesized formative state, if the naïve cells are cultured in medium containing Wnt signaling inhibitor (Rostovskaya *et al.*, 2019). This leads us to the assumption that our cluster 4 cells are very likely a real mouse counterpart of the hypothesized formative pluripotency state.

4.2. Outlook

It was stated above that *Rn7sk* may contribute to the gene repression occurring in t-SNE kmeans cluster 3. *Rn7sk* perturbation experiments are planned in order to verify this hypothesis. There are several known instances of large-scale gene repression during development. One example already mentioned is XCI in mammalian female embryos, others are meiotic chromosome inactivation during male spermatogenesis or global epigenetic changes in primordial germ cells (Robert Finestra and Gribnau, 2017; Turner, 2007; Royo *et al.*, 2010; Seki *et al.*, 2007). Errors in these gene repression phenomena can lead to diverse and serious abnormalities such as embryonic lethality and infertility, thus indicating the biological importance of controlled large-scale gene repression. We assume that our cluster 4 represents a hypothesized new formative pluripotency state. Nevertheless, research on such novel PSCs is just beginning and further studies of this state are necessary to elucidate its detailed characteristics. Comparison of the putative formative-like PSCs between human and mice could contribute to the understanding of this novel pluripotent state, and the cluster 4 cells of this thesis project are a good reference for such comparisons. It is interesting that some NASTs seem to show specificity in the naïve-to-primed pluripotency states in mice (Figure 21). Once again, perturbation experiments on specific NASTs are required to shed light on the regulatory role of this class of short ncRNA. It was previously reported that the majority of NASTs contain active promoter and enhancer histone marks and that many of them frequently overlap with repetitive elements (Fort *et al.*, 2014). One can thus speculate that NASTs may directly interact with stem cell-specific transcription factors. Our

C1-CAGE data could be used to analyze specific enhancer usage during the naïve, transition or primed pluripotency state (Andersson *et al.*, 2014). In general, the C1-CAGE data of this thesis is a valuable resource for further studies on the developmental process of early pluripotency states. It has the potential to extend study interest to the realm of diverse classes of expressed ncRNAs.

Since we used female mESCs from intersubspecific hybrid embryos, it is possible to take advantage of existing SNPs between the two subspecies of mice (Deng *et al.*, 2014; Reinius *et al.*, 2016; Petropoulos *et al.*, 2016). Although not included in this thesis, allele-specific expression analysis of the random XCI phenomenon was carried out by project collaborators of the author of this thesis. In our in vitro experimental system, random XCI happens between the time points Day 2 and Day 3 (Figure 17). This observation should be further validated by RNA-FISH and immunostaining experiments. AEA could enable detection of known and novel escape genes of random XCI as well as revealing monoallelicly expressed genes that show genetic origin dependency. It would be interesting to see additional data for example from DNA methylation and ChIP-seq experiments of the same time points that we sampled in our project. Complementing the transcriptional level data with epigenetic data might help to further elucidate the regulation of the various pluripotency states.

4.3. Concluding remarks

In the next years comprehensive cell atlases for human (Regev *et al.*, 2017), mouse (Han *et al.*, 2018) and other popular research organisms will become available. Global large-scale research consortia have already formed to tackle this enormous undertaking. They aim to provide single-cell high resolution data for all organs and tissues of selected species and different developmental stages of these. Once released, these atlases will provide a to date unprecedented reference that might very well revolutionize our current understanding of what defines a cell type, as well as accelerate scientific progress in various fields of biology and medicine. These Cell Atlas projects are the largest international

biological research efforts since the Human Genome Project. As a side product of establishing streamlined standard procedures for data production, management and analysis, we already see a constant output of new bioinformatical tools, databases, and best practice procedures that are not only impacting the Cell Atlas projects but provide benefits for reproducibility in many areas of biological research. When we started this thesis project the Cell Atlas projects were only at the planning stage, but even then, we saw several technological advances throughout our data acquisition phase. The developments in the field of single-cell technologies are so rapid that it would be possible to obtain much larger sample sizes with the same amount of effort nowadays. Despite that, we believe that our transcriptome data is an important contribution to stem cell biology research and has originality due to our unique cell model system, high sequencing depth and dual protocol data availability.

References

Abe, K. *et al.* (2004). "Contribution of Asian mouse subspecies Mus musculus molossinus to genomic constitution of strain C57BL/6J, as defined by BAC-end sequence–SNP analysis". In: *Genome Research* 14, pp. 2439–2447. DOI: 10.1101/gr.2899304.

Abugessaisa, I. *et al.* (2018). "SCPortalen: human and mouse single-cell centric database". In: *Nucleic Acids Research* 46, pp. D781–D787. DOI: 10.1093/nar/gkx949.

Adey, A. *et al.* (2010). "Rapid, low-input, low-bias construction of shotgun fragment libraries by high-density in vitro transposition". In: *Genome Biology* 11, R119. DOI: 10.1186/gb-2010-11-12-r119.

Alexa, A., J. Rahnenführer, and T. Lengauer (2006). "Improved scoring of functional groups from gene expression data by decorrelating GO graph structure". In: *Bioinformatics* 22, pp. 1600–7. DOI: 10.1093/bioinformatics/btl140.

Altshuler, A. *et al.* (2018). "RAS regulates the transition from naïve to primed pluripotent stem cells". In: *Stem Cell Reports* 10, pp. 1088–1101. DOI: 10.1016/j.stemcr.2018.01.004.

Andersson, R. *et al.* (2014). "An atlas of active enhancers across human cell types and tissues". In: *Nature* 507, pp. 455–461. DOI: 10.1038/nature12787.

Angerer, P. *et al.* (2016). "destiny: diffusion maps for large-scale single-cell data in R". In: *Bioinformatics* 32, pp. 1241–3. DOI: 10.1093/bioinformatics/btv715.

Angermueller, C. *et al.* (2016). "Parallel single-cell sequencing links transcriptional and epigenetic heterogeneity". In: *Nature Methods* 13, pp. 229–232. DOI: 10.1038/nmeth.3728.

Arner, E. *et al.* (2015). "Transcribed enhancers lead waves of coordinated transcription in transitioning mammalian cells". In: *Science* 347, pp. 1010–4. DOI: 10.1126/science.1259418.

Boroviak, T. *et al.* (2015). "Lineage-Specific Profiling Delineates the Emergence and Progression of Naive Pluripotency in Mammalian

Embryogenesis". In: *Developmental Cell* 35, pp. 366–82. DOI: 10.1016/j.devcel.2015.

Böttcher, M. *et al.* (2016). "Single-cell transcriptomes of fluorescent, ubiquitination-based cell cycle indicator cells". In: *BioRxiv*. preprint. DOI: https://doi.org/10.1101/088500.

Boyle, A. P. *et al.* (2008). "High-resolution mapping and characterization of open chromatin across the genome". In: *Cell* 132, pp. 311–22. DOI: 10.1016/j.cell.2007.12.014.

Bray, N. L. *et al.* (2016). "Near-optimal probabilistic RNA-seq quantification". In: *Nature Biotechnology* 34, pp. 525–7. DOI: 10.1038/nbt.3519.

Buecker, C. *et al.* (2014). "Reorganization of enhancer patterns in transition from naive to primed pluripotency". In: *Cell Stem Cell* 14, pp. 838–853. DOI: 10.1016/j.stem.2014.04.003.

Buenrostro, J. D., P. G. Giresi, *et al.* (2013). "Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position". In: *Nature Methods* 10, pp. 1213–8. DOI: 10.1038/nmeth.2688.

Buenrostro, J. D., B. Wu, *et al.* (2015). "Single-cell chromatin accessibility reveals principles of regulatory variation". In: *Nature* 523, pp. 486–90. DOI: 10.1038/nature14590.

Castelo-Branco, G. *et al.* (2013). "The non-coding snRNA 7SK controls transcriptional termination, poising, and bidirectionality in embryonic stem cells". In: *Genome Biology* 14, R98. DOI: 10.1186/gb-2013-14-9-r98.

Cusanovich, D. A. *et al.* (2015). "Multiplex single cell profiling of chromatin accessibility by combinatorial cellular indexing". In: *Science* 348, pp. 910–4. DOI: 10.1126/science.aab1601.

Deng, Q. *et al.* (2014). "Single-cell RNA-seq reveals dynamic, random monoallelic gene expression in mammalian cells". In: *Science* 343, pp. 193–6. DOI: 10.1126/science.1245316.

Deuve, J. L. and P. Avner (2011). "The coupling of X-chromosome inactivation to pluripotency". In: *Annual Review of Cell and Developmental Biology* 27, pp. 611–629. DOI: 10.1146/annurev-cellbio-092910-154020.

Dey, S. S. *et al.* (2015). "Integrated genome and transcriptome sequencing of the same cell". In: *Nature Biotechnology* 33, pp. 285–289. DOI: 10.1038/nbt.3129.

Dobin, A. *et al.* (2013). "STAR: ultrafast universal RNA-seq aligner". In: *Bioinformatics* 29, pp. 15–21. DOI: 10.1093/bioinformatics/bts635.

Fan, J. *et al.* (2016). "Characterizing transcriptional heterogeneity through pathway and gene set overdispersion analysis". In: *Nature Methods* 13, pp. 241–4. DOI: 10.1038/nmeth.3734.

Flyamer, I. M. *et al.* (2017). "Single-nucleus Hi-C reveals unique chromatin reorganization at oocyte-to-zygote transition". In: *Nature* 544, pp. 110–114. DOI: 10.1038/nature21711.

Forrest, A. *et al.* (2014). "A promoter-level mammalian expression atlas". In: *Nature* 507, pp. 462–70. DOI: 10.1038/nature13182.

Fort, A. *et al.* (2014). "Deep transcriptome profiling of mammalian stem cells supports a regulatory role for retrotransposons in pluripotency maintenance". In: *Nature Genetics* 46, pp. 558–566. DOI: 10.1038/ng.2965.

Guo, G. *et al.* (2009). "Klf4 reverts developmentally programmed restriction of ground state pluripotency". In: *Development* 136, pp. 1063–9. DOI: 10.1242/dev.030957.

Guo, H. *et al.* (2015). "Profiling DNA methylome landscapes of mammalian cells with single-cell reduced-representation bisulfite sequencing". In: *Nature Protocols* 10, pp. 645–59. DOI: 10.1038/nprot.2015.039.

Halpern, K. B. *et al.* (2017). "Single-cell spatial reconstruction reveals global division of labour in the mammalian liver". In: *Nature* 542, pp. 352–356. DOI: 10.1038/nature21065.

Han, X. *et al.* (2018). "Mapping the Mouse Cell Atlas by Microwell-Seq". In: *Cell* 173, p. 1307. DOI: 10.1016/j.cell.2018.05.012.

Hasegawa, A. *et al.* (2014). "MOIRAI: a compact workflow system for CAGE analysis". In: *BMC Bioinformatics* 15, p. 144. DOI: 10.1186/1471-2105-15-144.

HCA Consortium, A. Regev, and et al. (2017). *THE HUMAN CELL ATLAS*. White Paper.

Heintzman, N. D. *et al.* (2007). "Distinct and predictive chromatin signatures of transcriptional promoters and enhancers in the human genome". In: *Nature Genetics* 39, pp. 311–8. DOI: 10.1038/ng1966.

Hotelling, H. (1933). "Analysis of a complex of statistical variables into principal components". In: *Journal of Educational Psychology* 24, pp. 498–520. DOI: 10.1037/h0071325.

Ji, Z. and H. Ji (2016). "TSCAN: Pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis". In: *Nucleic Acids Research* 44, e117. DOI: 10.1093/nar/gkw430.

Jin, W. *et al.* (2015). "Genome-wide detection of DNase I hypersensitive sites in single cells and FFPE tissue samples". In: *Nature* 528, pp. 142–6. DOI: 10.1038/nature15740.

Karaiskos, N. *et al.* (2017). "The Drosophila embryo at single-cell transcriptome resolution". In: *Science* 358, pp. 194–199. DOI: 10.1126/science.aan3235.

Kharchenko, P. V., L. Silberstein, and D. T. Scadden (2014). "Bayesian approach to single-cell differential expression analysis". In: *Nature Methods* 11, pp. 740–742. DOI: 10.1038/nmeth.2967.

Kim, T. K. *et al.* (2010). "Widespread transcription at neuronal activity-regulated enhancers". In: *Nature* 465, pp. 182–7. DOI: 10.1038/nature09033.

Kind, J. *et al.* (2015). "Genome-wide maps of nuclear lamina interactions in single human cells". In: *Cell* 163, pp. 134–47. DOI: 10.1016/j.cell.2015.08.040.

Klein, A. M. *et al.* (2015). "Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells". In: *Cell* 161, pp. 1187–1201. DOI: 10.1016/j.cell.2015.04.044.

Kouno, T. *et al.* (2019). "C1 CAGE detects transcription start sites and enhancer activity at single-cell resolution". In: *Nature Communications* 10, p. 360. DOI: 10.1038/s41467-018-08126-5.

Lassmann, T. (2015). "TagDust2: a generic method to extract reads from sequencing data". In: *BMC Bioinformatics* 16, p. 24. DOI: 10.1186/s12859-015-0454-y.

Lassmann, T., Y. Hayashizaki, and C. O. Daub (2009). "TagDust—a program to eliminate artifacts from next generation sequencing

data". In: *Bioinformatics* 25, pp. 2839–2840. DOI: 10.1093/bioinformatics/btp527.

Lieberman-Aiden, E. *et al.* (2009). "Comprehensive mapping of long-range interactions reveals folding principles of the human genome". In: *Science* 326, pp. 289–93. DOI: 10.1126/science.1181369.

Maaten, L. van der and G. Hinton (2008). "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9, pp. 2579–2605.

Macaulay, I. C. *et al.* (2015). "G&T-seq: parallel sequencing of single-cell genomes and transcriptomes". In: *Nature Methods* 12, pp. 519–22. DOI: 10.1038/nmeth.3370.

Macosko, E. Z. *et al.* (2015). "Highly Parallel Genome-wide Expression Profiling of Individual Cells Using Nanoliter Droplets". In: *Cell* 161, pp. 1202–1214. DOI: 10.1016/j.cell.2015.05.002.

Marques, S. *et al.* (2016). "Oligodendrocyte heterogeneity in the mouse juvenile and adult central nervous system". In: *Science* 352, pp. 1326–1329. DOI: 10.1126/science.aaf6463.

Mooijman, D. *et al.* (2016). "Single-cell 5hmC sequencing reveals chromosome-wide cell-to-cell variability and enables lineage reconstruction". In: *Nature Biotechnology* 34, pp. 852–6. DOI: 10.1038/nbt.3598.

Munro, S. A. *et al.* (2014). "Assessing technical performance in differential gene expression experiments with external spike-in RNA control ratio mixtures". In: *Nature Communications* 5, p. 5125. DOI: 10.1038/ncomms6125.

Nagano, T., Y. Lubling, T. J. Stevens, *et al.* (2013). "Single-cell Hi-C reveals cell-to-cell variability in chromosome structure". In: *Nature* 502, pp. 59–64. DOI: 10.1038/nature12593.

Nagano, T., Y. Lubling, C. Várnai, *et al.* (2017). "Cell-cycle dynamics of chromosomal organization at single-cell resolution". In: *Nature* 547, pp. 61–67. DOI: 10.1038/nature23001.

Nagano, T., Y. Lubling, E. Yaffe, *et al.* (2015). "Single-cell Hi-C for genome-wide detection of chromatin interactions that occur simultaneously in a single cell". In: *Nature Protocols* 10, pp. 1986–2003. DOI: 10.1038/nprot.2015.127.

Ntranos, V. *et al.* (2016). "Fast and accurate single-cell RNA-seq analysis by clustering of transcript-compatibility counts". In: *Genome Biology* 17, p. 112. DOI: 10.1186/s13059-016-0970-8.

Ohtsuka, S., S. Nishikawa-Torikai, and H. Niwa (2012). "E-cadherin promotes incorporation of mouse epiblast stem cells into normal development". In: *PLoS One* 7, e45220. DOI: 10.1371/journal.pone.0045220.

Payer, B. (2016). "Developmental regulation of X-chromosome inactivation". In: *Seminars in Cell & Developmental Biology* 56, pp. 88–99. DOI: 10.1016/j.semcdb.2016.04.014.

Peng, H. *et al.* (2013). "Expression analysis of Nlrp4a-Nlrp4f during mouse development". In: *Journal of Animal and Veterinary Advances* 12, pp. 754–759. DOI: 10.3923/javaa.2013.754.759.

Peterlin, B. M. and D. H. Price (2006). "Controlling the elongation phase of transcription with P-TEFb". In: *Molecular Cell* 23, pp. 297–305. DOI: 10.1016/j.molcel.2006.06.014.

Petropoulos, S. *et al.* (2016). "Single-Cell RNA-Seq Reveals Lineage and X Chromosome Dynamics in Human Preimplantation Embryos". In: *Cell* 165, pp. 1012–26. DOI: 10.1016/j.cell.2016.03.023.

Pieters, T. and F. van Roy (2014). "Role of cell-cell adhesion complexes in embryonic stem cell biology". In: *Journal of Cell Science* 127, pp. 2603–2613. DOI: 10.1242/jcs.146720.

Prasanth, K. V. *et al.* (2010). "Nuclear Organization and Dynamics of 7SK RNA in Regulating Gene Expression". In: *Molecular Biology of the Cell* 21, pp. 4184–4196. DOI: 10.1091/mbc.E10-02-0105.

Regev, A. *et al.* (2017). "The Human Cell Atlas". In: *eLife* 6, e27041. DOI: 10.7554/eLife.27041.

Reinius, B. *et al.* (2016). "Analysis of allelic expression patterns in clonal somatic cells by single-cell RNA-seq". In: *Nature Genetics* 48, pp. 1430–1435. DOI: 10.1038/ng.3678.

Robert Finestra, T. and J. Gribnau (2017). "X chromosome inactivation: silencing, topology and reactivation". In: *Current Opinion in Cell Biology* 46, pp. 54–61. DOI: 10.1016/j.ceb.2017.01.007.

Robertson, E. J. (1987). *Teratocarcinomas and embryonic stem cells: A practical approach*. The Practical Approach Series. Oxford University Press, pp. 205–224. ISBN: 1852210044.

Rossant, J. (2008). "Stem cells and early lineage development". In: *Cell* 132, pp. 527–31. DOI: 10.1016/j.cell.2008.01.039.

Rostovskaya, M., G. G. Stirparo, and A. Smith (2019). "Capacitation of human naïve pluripotent stem cells for multi-lineage differentiation". In: *Development* 146, dev172916. DOI: 10.1242/dev.172916.

Rotem, A. *et al.* (2015). "Single-cell ChIP-seq reveals cell subpopulations defined by chromatin state". In: *Nature Biotechnology* 33, pp. 1165–72. DOI: 10.1038/nbt.3383.

Royo, H. *et al.* (2010). "Evidence that meiotic sex chromosome inactivation is essential for male fertility". In: *Current Biology* 20, pp. 2117–2123. DOI: 10.1016/j.cub.2010.11.010.

Sado, T., Y. Hoki, and H. Sasaki (2005). "Tsix silences Xist through modification of chromatin structure". In: *Developmental Cell* 9, pp. 159–65. DOI: 10.1016/j.devcel.2005.05.015.

Seki, Y. *et al.* (2007). "Cellular dynamics associated with the genome-wide epigenetic reprogramming in migrating primordial germ cells in mice". In: *Development* 134, pp. 2627–2638. DOI: 10.1242/dev.005611.

Severin, J. *et al.* (2014). "Interactive visualization and analysis of large-scale sequencing datasets using ZENBU". In: *Nature Biotechnology* 32, pp. 217–219. DOI: 10.1038/nbt.2840.

Shalek, A. K. and M. Benson (2017). "Single-cell analyses to tailor treatments". In: *Science Translational Medicine* 9, eaan4730. DOI: 10.1126/scitranslmed.aan4730.

Shiura, H. and K. Abe (2019). "Xist/Tsix expression dynamics during mouse peri-implantation development revealed by whole-mount 3D RNA-FISH". In: *Scientific Reports* 9, p. 3637. DOI: 10.1038/s41598-019-38807-0.

Smallwood, S. A. *et al.* (2014). "Single-cell genome-wide bisulfite sequencing for assessing epigenetic heterogeneity". In: *Nature Methods* 11, pp. 817–820. DOI: 10.1038/nmeth.3035.

Smith, A. (2017). "Formative pluripotency: the executive phase in a developmental continuum". In: *Development* 144, pp. 365–373. DOI: 10.1242/dev.142679.

Stevens, T. J. *et al.* (2017). "3D structures of individual mammalian genomes studied by single-cell Hi-C". In: *Nature* 544, pp. 59–64. DOI: 10.1038/nature21429.

Sugimoto, M. *et al.* (2015). "A simple and robust method for establishing homogeneous mouse epiblast stem cell lines by Wnt inhibition". In: *Stem Cell Reports* 4, pp. 744–757. DOI: 10.1016/j.stemcr.2015.02.014.

Thurman, R. E. *et al.* (2012). "The accessible chromatin landscape of the human genome". In: *Nature* 489, pp. 75–82. DOI: 10.1038/nature11232.

Treutlein, B. *et al.* (2014). "Reconstructing lineage hierarchies of the distal lung epithelium using single-cell RNA-seq". In: *Nature* 509, pp. 371–5. DOI: 10.1038/nature13173.

Tung, P. Y. *et al.* (2017). "Batch effects and the effective design of single-cell gene expression studies". In: *Scientific Reports* 7, p. 39921. DOI: 10.1038/srep39921.

Turner, J. M. A. (2007). "Meiotic sex chromosome inactivation". In: *Development* 134, pp. 1823–1831. DOI: 10.1242/dev.000018.

Weinberger, L. *et al.* (2016). "Dynamic stem cell states: naive to primed pluripotency in rodents and humans". In: *Nature Reviews Molecular Cell Biology* 17, pp. 155–69. DOI: 10.1038/nrm.2015.28.

Whitfield, M. L. *et al.* (2002). "Identification of genes periodically expressed in the human cell cycle and their expression in tumors". In: *Molecular Biology of the Cell* 13, pp. 1977–2000. DOI: 10.1091/mbc.02-02-0030.

Wu, A. R. *et al.* (2014). "Quantitative assessment of single-cell RNA-sequencing methods". In: *Nature Methods* 11, pp. 41–6. DOI: 10.1038/nmeth.2694.

Zeisel, A. *et al.* (2015). "Cell types in the mouse cortex and hippocampus revealed by single-cell RNA-seq". In: *Science* 347, pp. 1138–42. DOI: 10.1126/science.aaa1934.

Online references

https://www.gencodegenes.org/pages/biotypes.html (accessed 26th September 2019)

https://www.fluidigm.com/c1openapp/scripthub/script/2015-07/c1-cage-1436761405138-3 (accessed 26th September 2019)

https://www-s.nist.gov/srmors/certificates/documents/SRM2374_putative_T7_products_NoPolyA_v2.FASTA (accessed 26th September 2019)

https://github.com/Population-Transcriptomics/C1-CAGE-preview/blob/master/OP-WORKFLOW-CAGEscan-short-reads-v2.0.ipynb (accessed 26th September 2019)

https://rdrr.io/bioc/CAGEr/ (accessed 26th September 2019)

http://fantom.gsc.riken.jp/5/data/ (accessed 26th September 2019)

https://epd.epfl.ch/mouse/mouse_database.php?db=mouse (accessed 26th September 2019)

Appendix

Due to the length of used tables and the large quantity of samples a simple attachment of result raw data is not feasible. Instead, some raw files used for analysis can be downloaded here:

http://single-cell.clst.riken.jp/riken_data/mES2EpiSC_summary_view.php

The addition of more downloadable files is pending and will be realized upon publication of a journal paper that is based on this thesis data. These files include among other metadata tables and expression tables for both protocol datasets, tables of all DGE results, GO analysis results, screening plots for cluster specific gene identification and more.

The track views in ZENBU for scRNA-seq and C1-CAGE data are accessible here after free registration and log in:

scRNA-seq:
https://fantom.gsc.riken.jp/zenbu/gLyphs/#config=1qUudPWiDNTgcknv0TJkp

C1-CAGE:
https://fantom.gsc.riken.jp/zenbu/gLyphs/#config=bYYvK4ICElFj8aWmkAJ7z

Declaration

I herewith declare that I have produced this dissertation without the prohibited assistance of third parties and without making use of aids other than those specified. Notions taken over directly or indirectly from other sources have been marked as such.